Universal Agent 6.2.x

Installation, Upgrade, and Applying Maintenance

© 2015 by Stonebranch, Inc. All Rights Reserved.

1. Universal Agent 6.2.x Installation, Upgrade, and Applying Maintenance 4
1.1 Installation, Upgrade, and Applying Maintenance Overview 5
1.2 Installing Universal Agent 6
1.3 Installing Universal Agent - Universal Connector Prerequisite 10
1.4 zOS Installation 12
1.4.1 zOS Installation - Installation Package 14
1.4.2 zOS Installation - Installation Requirements 15
1.4.3 zOS Installation - Distribution File 16
1.4.4 zOS Installation - Transferring Installation Files to zOS 18
1.4.5 zOS Installation - SMPE Installation 32
1.4.5.1 zOS Installation - New Install, New CSI 34
1.4.5.2 zOS Installation - Universal Agent 5.2.0 Upgrade, Existing CSI 36
1.4.5.3 zOS Installation - Workload Automation 5.1.0 Upgrade, Existing CSI 37
1.4.5.4 zOS Installation - Stonebranch Solutions 4.x Upgrade, Existing CSI 38
1.4.5.5 zOS Installation - Universal Products 3.2.0 Upgrade, Existing CSI 39
1.4.6 zOS Installation - Configuration 40
1.4.6.1 zOS Configuration - Started Tasks 41
1.4.6.2 zOS Configuration - Load Library 47
1.4.6.3 zOS Configuration - SMF Exits 50
1.4.6.4 zOS Configuration - JES SYSOUT Processing 54
1.4.6.5 Universal Agent Database Configuration 56
1.4.7 zOS Installation - Customization 60
1.4.8 zOS Installation - Cumulative PTF Maintenance 66
1.4.9 zOS Installation - Time Zone Environment Variable 68
1.4.10 zOS Installation - TCPIP Configuration 69
1.4.11 zOS Installation - SAP RFC DLL 71
1.4.12 zOS Installation - Configuration of zOS System SSL 73
1.4.12.1 Integrated Cryptographic Service Facility (ICSF) 74
1.4.12.2 Universal Broker Digital Certificate (RACF) Set-up 75
1.4.13 zOS Installation - Configuration of Security 77
1.4.13.1 Configuration of Security - RACF Class 78
1.4.13.2 Configuration of Security - ACF2 Class 80
1.4.13.3 Configuration of Security - Universal Command Security 81
1.4.13.4 Configuration of Security - Universal Control Security 84
1.4.13.5 Configuration of Security - Universal Event Monitor Security 85
1.4.14 zOS Installation - Performance Guidelines 86
1.4.15 zOS Installation - Converting STC User Profiles to a Non-Zero UID 88
1.4.16 zOS Installation - Data Set Inventory 91
1.4.17 zOS Installation - Licensing 93
1.5 zOS USS Installation 95
1.5.1 zOS USS Installation - Installation Package 96
1.5.2 zOS USS Installation - Installation Requirements 97
1.5.3 zOS USS Installation - Distribution File 99
1.5.4 zOS USS Installation - SMPE Installation 100
1.5.4.1 zOS USS Installation - New Install, New CSI 101
1.5.4.2 zOS USS Installation - 5.2.0 Upgrade, Existing CSI 102
1.5.4.3 zOS USS Installation - 5.1.0 Upgrade, Existing CSI 103
1.5.4.4 zOS USS Installation - 4.x Upgrade, Existing CSI 104
1.5.4.5 zOS USS Installation - 3.2.0 Upgrade, Existing CSI 105
1.5.5 zOS USS Installation - Customization 106
1.5.6 zOS USS Installation - Data Set Inventory 108
1.5.7 zOS USS Installation - Licensing 109
1.6 Windows Installation 111
1.6.1 Universal Agent for Windows Installation 115
1.6.1.1 Universal Agent for Windows - Installation Package 116
1.6.1.2 Universal Agent for Windows - Installation Requirements 117
1.6.1.3 Universal Agent for Windows - Installation Procedures 120
1.6.1.3.1 Installing Universal Agent via the Graphical Interface 121
1.6.1.3.2 Modifying a Universal Agent Installation via the Graphical Interface 132
1.6.1.3.3 Installing Universal Agent via the Command Line 141
1.6.1.3.4 Modifying a Universal Agent Installation via the Command Line 152
1.6.1.3.5 Migrating between 32- and 64-bit Universal Agent for Windows Installs 154
1.6.1.4 32-Bit Universal Agent for Windows on 64-Bit Windows Systems 159
1.6.1.5 Universal Agent for Windows - File Inventory Lists 163
1.6.2 Universal Enterprise Controller for Windows Installation 172
1.6.2.1 Universal Enterprise Controller for Windows - Installation Package 173
1.6.2.2 Universal Enterprise Controller for Windows - Installation Requirements 174
1.6.2.3 Universal Enterprise Controller for Windows - Installation Procedures 175
1.6.2.3.1 Installing UEC via the Graphical Interface 176
1.6.2.3.2 Modifying a UEC Installation via the Graphical Interface 182
1.6.2.3.3 Installing UEC via the Command Line 192
1.6.2.3.4 Modifying a UEC Installation via the Command Line 195
1.6.2.4 Universal Enterprise Controller for Windows - 64-Bit Windows Editions 197
1.6.2.5 Universal Enterprise Controller for Windows - Database Configuration 198
1.6.2.6 Universal Enterprise Controller for Windows - File Inventory Lists 199

1.6.3 UEC Client Applications Installation . . . 201
1.6.3.1 UEC Client Applications - Installation Package . . . 202
1.6.3.2 UEC Client Applications - Installation Requirements . . . 203
1.6.3.3 UEC Client Applications - Installation Procedures . . . 204
1.6.3.3.1 Installing UEC Client Applications via the Graphical Interface . . . 205
1.6.3.3.2 Modifying a UEC Client Applications Installation via the Graphical Interface . . . 213
1.6.3.3.3 Installing UEC Client Applications via the Command Line . . . 223
1.6.3.3.4 Modifying a UEC Client Applications Installation via the Command Line . . . 228
1.6.3.4 UEC Client Applications - 64-Bit Windows Editions . . . 230
1.6.3.5 UEC Client Applications - File Inventory Lists . . . 231
1.6.4 Universal Agent for SOA for Windows Installation . . . 232
1.6.4.1 Universal Agent for SOA for Windows - Installation Package . . . 233
1.6.4.2 Universal Agent for SOA for Windows - Installation Requirements . . . 234
1.6.4.3 Universal Agent for SOA for Windows - Pre-Installation - Upgrade Backups . . . 235
1.6.4.4 Universal Agent for SOA for Windows - Installation Procedures . . . 236
1.6.4.4.1 Installing SOA for Windows via the Graphical Interface . . . 237
1.6.4.4.2 Modifying a SOA for Windows Installation via the Graphical Interface . . . 242
1.6.4.4.3 Installing SOA for Windows via the Command Line . . . 247
1.6.4.4.4 Modifying a SOA for Windows Installation via the Command Line . . . 250
1.6.4.5 Universal Agent for SOA for Windows - Configuring and Starting UAC Server . . . 251
1.6.4.6 Universal Agent for SOA for Windows - 64-Bit Windows Editions . . . 253
1.6.4.7 Universal Agent for SOA for Windows - File Inventory Lists . . . 254
1.6.5 Windows Installation - Licensing . . . 257
1.7 UNIX Installation . . . 260
1.7.1 Universal Agent for UNIX Installation . . . 262
1.7.1.1 Universal Agent for UNIX - Installation Package . . . 263
1.7.1.2 Universal Agent for UNIX - Installation Requirements . . . 264
1.7.1.3 Universal Agent for UNIX - Distribution File . . . 268
1.7.1.4 Universal Agent for UNIX - Installation Procedures . . . 269
1.7.1.4.1 Universal Agent for AIX Installation . . . 270
1.7.1.4.2 Universal Agent for HP-UX Installation . . . 279
1.7.1.4.3 Universal Agent for Solaris Installation . . . 288
1.7.1.4.4 Universal Agent for Linux Installation . . . 298
1.7.1.5 Universal Agent for UNIX - Customization . . . 309
1.7.1.6 Universal Agent for UNIX - File Inventory Lists . . . 313
1.7.2 Universal Agent for SOA for UNIX Installation . . . 320
1.7.2.1 Universal Agent for SOA for UNIX - Installation Package . . . 321
1.7.2.2 Universal Agent for SOA for UNIX - Installation Requirements . . . 322
1.7.2.3 Universal Agent for SOA for UNIX - Pre-Installation-Upgrade Backups . . . 323
1.7.2.4 Universal Agent for SOA for UNIX - Distribution File . . . 324
1.7.2.5 Universal Agent for SOA for UNIX - Deployment Options . . . 325
1.7.2.6 Universal Agent for SOA for UNIX - Installation Procedures . . . 328
1.7.2.6.1 Universal Agent for SOA for AIX Installation . . . 329
1.7.2.6.2 Universal Agent for SOA for Linux Installation . . . 332
1.7.2.7 Universal Agent for SOA for UNIX - File Inventory Lists . . . 335
1.7.3 UNIX Installation - Licensing . . . 337
1.8 IBM i Installation . . . 339
1.8.1 IBM i Installation - Installation Package . . . 341
1.8.2 IBM i Installation - Installation Requirements . . . 343
1.8.3 IBM i Installation - Distribution File . . . 346
1.8.4 IBM i Installation - Transferring to IBM i . . . 348
1.8.5 IBM i Installation - Installation Procedures . . . 351
1.8.5.1 IBM i Installation - New Installation (Default) . . . 352
1.8.5.2 IBM I Installation - New Installation (Custom) . . . 353
1.8.5.3 IBM I Installation - Re-Installation of Same Release . . . 354
1.8.5.4 IBM i Installation - Upgrade Installation to New Release . . . 355
1.8.5.5 IBM I Installation - Propagating New Release to Additional Systems . . . 357
1.8.5.6 IBM i Installation - Upgrade Installation for Maintenance Release . . . 359
1.8.5.7 IBM i Installation - UCHGRLS (Change Release Tag) Program . . . 360
1.8.5.8 IBM i Installation - Product Removal . . . 362
1.8.6 IBM i Installation - Customization . . . 367
1.8.7 IBM i Installation - Object Inventory Lists . . . 371
1.8.8 IBM i Installation - Licensing . . . 378
1.9 HP NonStop Installation . . . 380
1.9.1 HP NonStop Installation - Installation Package . . . 381
1.9.2 HP NonStop Installation - Installation Requirements . . . 382
1.9.3 HP NonStop Installation - Distribution File . . . 384
1.9.4 HP NonStop Installation - Installation Procedures . . . 386
1.9.5 HP NonStop Installation - Customization . . . 387
1.9.6 HP NonStop Installation - File Inventory Lists . . . 390
1.9.7 HP NonStop Installation - Licensing . . . 392
1.10 Verifying Universal Agent Installation . . . 393
1.11 Upgrading Universal Agent . . . 395
1.12 Applying Maintenance to Universal Agent . . . 398
1.13 Licenses for Third-Party Libraries . . . 399

Universal Agent 6.2.x Installation, Upgrade, and Applying Maintenance

Overview

Universal Agent 6.2.x Installation, Upgrade, and Applying Maintenance provides information on installing, upgrading, and applying maintenance to all Universal Agent packages on all supported operating systems.

(Read the Installation, Upgrade, and Applying Maintenance Overview page to understand the differences between these three procedures.)

Detailed Information

The following pages provide detailed information for installation and administration:

- Installation, Upgrade, and Applying Maintenance Overview
- Installing Universal Agent
- Installing Universal Agent - Universal Connector Prerequisite
- z/OS Installation
- z/OS USS Installation
- Windows Installation
- UNIX Installation
- IBM i Installation
- HP NonStop Installation
- Verifying Universal Agent Installation
- Upgrading Universal Agent
- Applying Maintenance to Universal Agent
- Licenses for Third-Party Libraries

Installation, Upgrade, and Applying Maintenance Overview

- Installation, Upgrade, and Applying Maintenance
  - Installation
  - Upgrade
  - Applying Maintenance

Installation, Upgrade, and Applying Maintenance

There are separate procedures for installing, upgrading, and applying maintenance to Universal Agent.

Installation

Installation refers to the installation of an Agent on a machine with any supported platform that does not already contain an installed Agent.

If you are installing Universal Agent for the first time, see Installing Universal Agent for instructions.

Upgrade

Upgrading refers to the increase of a currently installed Version, Release, or Modification level (see Versioning) of an Agent on a machine to a later Version, Release, or Modification level of that Agent (for example, upgrading Universal Agent 5.2.0.11 to Universal Agent 6.2.0.0).

If you are upgrading from a previous version of Universal Agent, see Upgrading Universal Agent for instructions.

Applying Maintenance

Applying maintenance refers to the increase of a currently installed Maintenance level (see Versioning) of an Agent on a machine to a later Maintenance level of that Agent (for example, applying maintenance to Universal Agent 6.2.0.0 to increase its maintenance level to 6.2.0.1).

If you are applying maintenance to your version of Universal Agent, see Applying Maintenance to Universal Agent.

Installing Universal Agent

- Overview
- Installation Requirements
  - Platform Requirements
  - Space Requirements
  - Network Requirements
  - Additional Requirements
- Ports Configuration
- Product Distribution File Download
- Product Licensing
- Additional Information

Overview

These pages provide information and instructions for the installation of Universal Agent, which differ for each supported operating system.

Note

- If you are upgrading from a previous version of the Agent - for example, from Agent 5.2.0.11 to 6.2.0.0 - see Upgrading Universal Agent for instructions.
- If you are applying maintenance to the Agent - for example, applying maintenance to Agent 6.2.0.0 to increase its maintenance level to Agent 6.2.0.1 - see Applying Maintenance to Universal Agent for instructions.

The following table identifies the different operating systems on which an Agent can be installed, and the Agent packages that can be installed on each system.

Each ✓ is a link to installation information for that package.

Operating System	Universal Agent	Universal Enterprise Controller	UEC Client Applications	Universal Agent for SOA
z/OS [1]	✓			
z/OS USS	✓			
Windows	✓	✓	✓	✓
UNIX - AIX	✓			✓
UNIX - HP-UX	✓			
UNIX - Solaris	✓			
UNIX - Linux	✓			✓
IBM i [2]	✓			
HP NonStop [3]	✓			

[1] Universal Agent for z/OS contains Universal Enterprise Controller.
[2] Workload Automation 5.1.0 is installed on IBM i.
[3] Universal Command 2.1.1 is installed on HP NonStop.

For each Universal Agent installation package, the following information is provided:

- Installation package components
- Product compatibility
- Installation requirements
- Installation upgrades
- System upgrades
- Distribution files
- Installation procedures
- Product customization
    - Configuration
    - Licensing
- File inventory lists

Note
You can install Agents before, during, or after installation of Universal Controller.

Installation Requirements

Platform-specific installation requirements, including system requirements, are located on the Installation Requirements page specific to that platform:

- z/OS Installation Requirements
- Windows Installation Requirements
- UNIX Installation Requirements
- IBM i Installation Requirements
- HP NonStop Installation Requirements

Requirements relevant to all platforms are shown below.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support page to make sure that your platform is supported before performing an installation.

Space Requirements

Space requirements for Universal Agent are driven largely by logging. A minimum 500MB of space is recommended. Each Agent contains a Universal Broker log and an Agent log.

The Universal Broker LOG_FILE_GENERATIONS configuration option lets you specify how many log files to save in the Universal Broker log directory. The default is 5.

The Agent log rolls over at midnight (12:00 a.m.) or whenever the Agent is restarted. The Log File Retention Period in Days Universal Controller system property lets you specify the number of days that an Agent log file (in addition to a Controller log file) is retained before it is purged. The default is 5 days.

Network Requirements

Agent components run on z/OS, Windows, UNIX, IBM i, and HP NonStop operating systems. These systems must be connected with a network that supports TCP/IP. For example, the Universal Command Manager running on a z/OS system must be able to establish a TCP/IP socket connection with the Universal Command Server running on a UNIX or Windows system.

TCP/IP provides a set of commands to verify network connectivity between systems. For example, the ping command can determine if an IP connection is possible between two systems. However, the ping command may not work in all network environments. In addition, the ping command may work, but a firewall may deny all other connection attempts between the two systems. Check with your local network administrator to determine the capabilities and topology of your network.

Universal Agent offers configuration parameters that can facilitate connections through network firewalls. Due to the large variety of firewall configurations, all possibilities cannot be discussed in this document. Check with your local network administrator to determine if a firewall is between the computer systems involved.

Additional Requirements

In order to install any of these Agent components, you must be able to write to the directory from which the installation is launched.

Ports Configuration

Universal Agent is composed of several components that utilize the TCP/IP network for component communications. TCP/IP routes must be available between the distributed components and firewall rules open, if necessary.

The primary components using TCP/IP are Universal Broker, Universal Managers, and Universal Automation Center Agent (UAG).

Universal Broker	Executes on all servers on which workload management services are required. The Universal Broker accepts TCP socket connections on port 7887 from remote clients such as Universal Command Manager and Universal Data Mover Manager. A server running Universal Agent will need to be able to accept TCP connections on port 7887 in order to provide such services.
Universal Manager	Components include Universal Command Manager, Universal Data Mover Manager, and others. Manager components initiate TCP socket connections to remote Universal Brokers. If Universal Manager services are required, the Managers will need to be able to open TCP connections to remote Universal Brokers on port 7887.
UAG	Initiates a TCP socket connection to the Universal Message Service (OMS) deployed on a remote, centrally located server. OMS accepts TCP connections on port 7878. UAG must be able to open a TCP connection to the remote OMS server on port 7878.

The port numbers above are default port numbers. They can be changed in the respective component configuration, if necessary.

Product Distribution File Download

To install, update, or apply maintenance to an Agent package, you first must download the corresponding product distribution file for your specific operating system version and hardware platform.

All Agent distribution files are available for download from the Stonebranch Customer Portal. A customer user name and password - provided by Stonebranch, Inc. - are required to access the Customer Portal.

The operating system-specific pages of this installation guide identify the product distribution files to be downloaded for the different Agent packages.

> **Note**
> Agent installation packages for z/OS, Windows, and UNIX contain the Universal Controller Command Line Interface (CLI).
>
> Agent installation packages for Windows and UNIX contain the Universal Message Service (OMS).

To download a product distribution file:

Step 1	Log in to the Stonebranch Customer Portal.
Step 2	Click the **Software** link.
Step 3	Click the **Universal Agent** link.
Step 4	Click the Universal Agent package link appropriate for your platform.
Step 5	Click **Save File** and browse to your save location. You can then install the software.

Product Licensing

Each Agent installation package includes one or more components that must be licensed before they can be used. Licenses are provided by your Stonebranch, Inc. account representative.

> **Exception**
> UEC Client Applications, a separate, optional installation package of Universal Agent for Windows, does not contain any components that must be licensed.

The following table identifies all licensable components in the Agent package for each supported platform. The platform name is a link to detailed

information about licensing the Agent components for that platform.

Component	z/OS	z/OS USS	Windows	UNIX	IBM i	HP NonStop
Universal Command Manager	✓	✓	✓	✓	✓	✓
Universal Data Mover Manager	✓	✓	✓	✓	✓	
Universal Connector	✓		✓	✓		
Universal Event Monitor Server			✓	✓		
Universal Enterprise Controller	✓		✓			
Universal Application Container Server			✓	✓		

Note
Universal Enterprise Controller is a separate, optional installation package of Universal Agent for Windows.

Universal Application Container Server is a component of Universal Agent for SOA, a separate, optional installation package of Universal Agent for Windows and Universal Agent for UNIX.

Additional Information

In addition to this Installation Guide, Stonebranch, Inc. provides the following information:

Universal Agent 6.2.x Installation Requirements and Summary identifies network requirements for all Agent installations and, for each operating system, the following information:

- System requirements
- Installation summary
- Customization

Universal Agent 6.2.x Installation Quick Start Guides provide the following information:

- System Requirements
- Downloading Universal Agent 6.2.x Distribution File
- Installing Universal Agent 6.2.x
- Licensing your Universal Agent 6.2.x Components
- Encrypting User ID and Password for Job Execution
- Executing a Job
- Running a System Query

Note
There is a separate quick start guide for each Universal Agent and Universal Data Mover supported platform.

Installing Universal Agent - Universal Connector Prerequisite

- Introduction
- Download
- Install
  - Windows
  - UNIX

Introduction

Universal Connector *for Use with SAP® ERP* requires the SAP NW (NetWeaver) RFC libraries to run. Before executing Universal Connector, you must first install the appropriate SAP NW RFC Libraries for the platform on which the connector agent is installed.

Note
Universal Connector for AIX and Universal Connector for Solaris SPARC-based systems do not require the SAP NW (NetWeaver) RFC libraries to run.

Download

To acquire the SAP libraries, access the SAP Service Marketplace and download the appropriate package for your platform:

https://launchpad.support.sap.com/#/softwarecenter/template/products/%20_APP=00200682500000001943&_EVENT=DISPHIER&HEADER=Y&F

Note
In order to extract the downloaded SAP NW RFC package, you must use the SAPCAR executable. That utility program can be found here:

https://support.sap.com/software/patches/a-z-index.html

In order to download products from the SAP Service Marketplace you will need an SAP S-User ID with the "Download Software" authorization, which you can request via your company's SAP user administrator.

Install

Once the appropriate package has been downloaded, it can be extracted using the SAPCAR utility. The command line to extract an SAP NW RFC package is basically the same on all platforms. The only difference is the name of the SAP NW RFC package. The following is a sample command line to extract the SAP NW RFC package and its resulting output:

```
C:\tmp\sap>sapcar -xvf NWRFC_35-20004568.SAR
SAPCAR: processing archive NWRFC_35-20004568.SAR (version 2.01)
x nwrfcsdk
x nwrfcsdk/bin
x nwrfcsdk/bin/rfcexec.exe
x nwrfcsdk/bin/startrfc.exe
x nwrfcsdk/demo
x nwrfcsdk/demo/companyClient.c
x nwrfcsdk/demo/readme.txt
x nwrfcsdk/demo/rfcexec.cpp
x nwrfcsdk/demo/rfcexec.h
x nwrfcsdk/demo/sapnwrfc.ini
x nwrfcsdk/demo/sflightClient.c
x nwrfcsdk/demo/sso2sample.c
x nwrfcsdk/demo/startrfc.cpp
x nwrfcsdk/demo/startrfc.h
x nwrfcsdk/demo/stfcDeepTableServer.c
x nwrfcsdk/doc
x nwrfcsdk/include
x nwrfcsdk/include/sapdecf.h
x nwrfcsdk/include/sapnwrfc.h
x nwrfcsdk/include/sapuc.h
x nwrfcsdk/include/sapucx.h
x nwrfcsdk/lib
x nwrfcsdk/lib/icudt34.dll
x nwrfcsdk/lib/icuin34.dll
x nwrfcsdk/lib/icuuc34.dll
x nwrfcsdk/lib/libicudecnumber.dll
x nwrfcsdk/lib/libsapucum.dll
x nwrfcsdk/lib/libsapucum.lib
x nwrfcsdk/lib/sapdecfICUlib.lib
x nwrfcsdk/lib/sapnwrfc.dll
x nwrfcsdk/lib/sapnwrfc.lib
x SIGNATURE.SMF
SAPCAR: 32 file(s) extracted
```

Windows

From the extracted SAP NW RFC library, the dll files located in `nwrfcsdk/lib/` must be placed in a directory path that is listed in the environment variable `Path`.

This can be done by adjusting the `Path` environment variable to include the `nwrfcsdk/lib/` path or by moving the dll files from the SAP NW RFC SDK archive to a location that already exists in the `Path`.

> **Note**
> It is sufficient to place the SAP NW RFC `*.dll` files in the Universal Connector's `bin` directory. That path for a typical Windows install is `C:\Program Files\Universal\USap\bin`.

UNIX

From the extracted SAP NW RFC library, the shared library files located in `nwrfcsdk/lib/` must be placed in a directory path that is listed in the environment variable `LD_LIBRARY_PATH`.

This can be done by adjusting the `LD_LIBRARY_PATH` environment variable to include the `nwrfcsdk/lib/` path or by moving the shared library files from the SAP NW RFC SDK archive to a location that already exists in `LD_LIBRARY_PATH`.

zOS Installation

- Introduction
- Installation Checklist
- Detailed Information

Introduction

These pages describe the installation of Universal Agent on a z/OS operating system. Unless otherwise specified, all references to Universal Agent for z/OS refer to version 6.2.0.

Universal Agent for z/OS is provided as an easily installed SMP/E package. After the SMP/E installation steps are complete, z/OS must be configured to meet the product requirements. Lastly, the individual product components must be customized to meet local production requirements.

The z/OS package includes the components for both Universal Agent for z/OS and optional components for Universal Agent for z/OS UNIX System Services (USS). For information on installation of Universal Agent for z/OS USS, see z/OS USS Installation.

Note
Starting with the 3.2.0 release of Universal Products, a Universal Broker must run on all systems on which a Universal Agent component is running, including manager components. The Broker maintains product configuration data for all components that have a configuration file.

Installation Checklist

The following installation checklist provides an overview of the z/OS installation steps. The steps refer to pages that describe the procedures in detail. All z/OS installation pages should be read to avoid product installation and configuration problems.

The installation itself consists of running a number of batch jobs. The output of these batch jobs should be kept until a correct installation has been verified.

Step 1	Download the Universal Agent 6.2.0 for z/OS product distribution file to a Windows or UNIX-based workstation.
Step 2	Decompress the distribution file using any utility capable of processing UNIX-compressed files (for example, **compress** and **gzip**).
Step 3	Extract the installation (XMT) files from the decompressed distribution file and transfer them to your z/OS system. See zOS Installation - Transferring Installation Files to zOS for complete details.
Step 4	Edit the INSTALL(#SETUP) member to meet local requirements as described in the #SETUP comment prolog. #SETUP creates customized installation JCL members in the INSTALL library. Execute the installation jobs as required for your SMP/E environment. See zOS Installation - SMPE Installation for complete details.
Step 5	Create started task user and group profiles for Universal Broker and Universal Enterprise Controller. See zOS Configuration - Started Tasks for complete details.
Step 6	Define the SUNVLOAD library to APF, Program Control, and LNKLST. See zOS Configuration - Load Library for complete details.
Step 7	Define SMF exits for Universal Command and Universal Automation Center Agent. See zOS Configuration - SMF Exits for complete details.
Step 8	Configure the UNIX System Services file systems used by the Universal Broker and Universal Enterprise Controller. See Universal Agent Database Configuration for complete details.
Step 9	Perform individual component customization, including JCL procedure edits and adding component licenses. See zOS Installation - Customization for complete details.

Detailed Information

The following pages provide detailed information for z/OS Installation:

- Installation Package
- Installation Requirements

- Distribution File
- Transferring Installation Files to zOS
- SMPE Installation
- Configuration
- Customization
- Cumulative PTF Maintenance
- Time Zone Environment Variable
- TCPIP Configuration
- SAP RFC DLL
- Configuration of zOS System SSL
- Configuration of Security
- Performance Guidelines
- Converting STC User Profiles to a Non-Zero UID
- Data Set Inventory
- Licensing

zOS Installation - Installation Package

- Package Components
- Component Compatibility

Package Components

The Universal Agent for z/OS package contains the following Universal Agent for z/OS components:

- Universal Broker 6.2.x
- Universal Automation Center Agent 6.2.x
- Universal Certificate 6.2.x
- Universal Command Manager and Server 6.2.x
- Universal Connector 5.2.1
- Universal Control Manager and Server 6.2.x
- Universal Controller Command Line Interface (CLI) 6.2.x
- Universal Data Mover Manager and Server 6.2.x
- Universal Encrypt 6.2.x
- Universal Enterprise Controller 6.2.x
- Universal Event Monitor Manager 6.2.x
- Universal Message to Exit Code Translator 6.2.x
- Universal Query 6.2.x

Note
For the list of Universal Agent for z/OS USS components included in the z/OS package, see z/OS USS Installation - Installation Package.

Component Compatibility

The following table identifies the compatibility of Universal Agent for z/OS components with previous component / product versions.

Component	Compatibility
Universal Broker 6.2.x	Workload Automation / Stonebranch Solutions / Universal Products releases 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Command 6.2.x	Universal Command 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Control 6.2.x	Universal Control 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Data Mover 6.2.x	Universal Data Mover 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Encrypt 6.2.x	Universal Encrypt 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Query 6.2.x	Universal Broker 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Enterprise Controller 6.2.x	Not compatible with previous versions of Universal Enterprise Controller Client Applications.
Universal Event Monitor 6.2.x	Universal Event Monitor 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, and 3.1.0.

The component references pertain to all supported platforms for that version.

zOS Installation - Installation Requirements

- System Requirements
- Data Set Space Requirements
- Platform Requirements

System Requirements

Universal Agent for z/OS requires the following software releases:

- z/OS 1.12 or above.
- SMP/E 3.5 or above.
- IBM Communication Server for z/OS 1.12 or above.
- IBM Language Environment (LE) for z/OS 1.12 or above.
- Windows workstation capable of establishing a TCP/IP network connection to the z/OS system.
- TSO user ID with an OMVS segment.
- About 1900 cylinders of DASD.
- Two available TCP/IP ports on z/OS.

All Universal Agent programs use z/OS UNIX System Services. As such, z/OS UNIX requires the user profile with which a program executes to have a properly defined OMVS segment. The OMVS segment should define a unique UID value. The HOME value must specify an existing home directory to which the user ID has read and write access.

Additionally, the group(s) that the user ID is associated with must have an OMVS segment that defines a unique GID value for the group. Refer to IBM's UNIX System Services Planning manual for additional details on defining z/OS UNIX users.

Data Set Space Requirements

As part of the Universal Agent for z/OS package installation, a number of SMP/E and non-SMP/E data sets are allocated and cataloged. The space requirements for these data sets are listed in z/OS Installation - Data Set Inventory.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

zOS Installation - Distribution File

- z/OS Distribution File
- Obtaining the Distribution File
- Distribution File Format
- Distribution File Contents

z/OS Distribution File

The Universal Agent for z/OS product distribution file contains all of the files required for the installation of the Universal Agent for z/OS package.

Obtaining the Distribution File

To obtain the Universal Agent for z/OS package, download the corresponding product distribution file from the Stonebranch Customer Portal.

A customer user name and password — provided by Stonebranch, Inc. — are required to access the Customer Portal.

> **Note**
> If you are installing a Cumulative PTF Maintenance to a Universal Agent for z/OS installation, you must download the z/OS - PTFs Package from the Stonebranch Customer Portal.

Distribution File Format

The name of Universal Agent for z/OS distribution file has the following format:

`sb-Version.Release.Modification Level.Maintenance Level-operating system.tar.z`

For example: **`sb-6.2.0.0-zOS.tar.Z`**

Distribution File Contents

The following table lists the installation files (in XMT file format) included in the Universal Agent for z/OS distribution file.

> **Note**
> These files include the files for z/OS USS.

File Name	Description
z/OS Files	
`INSTALL.XMT`	Universal Agent package installation JCL.
`README.TXT`	Documentation on the package and transfer methods.
`SMPMCS.XMT`	Universal Agent SMP/E MCS statements.
`UAG620F1.XMT`	Universal Automation Center Agent SMP/E FMID TUAG620 relative file 1.
`UAG620F2.XMT`	Universal Automation Center Agent SMP/E FMID TUAG620 relative file 2.
`UAG620F3.XMT`	Universal Automation Center Agent SMP/E FMID TUAG620 relative file 3.
`UBR620F1.XMT`	Universal Broker SMP/E FMID TUBR620 relative file 1.
`UBR620F2.XMT`	Universal Broker SMP/E FMID TUBR620 relative file 2.
`UBR620F3.XMT`	Universal Broker SMP/E FMID TUBR620 relative file 3.
`UCM620F1.XMT`	Universal Command SMP/E FMID TUCM620 relative file 1.

UCM620F2.XMT	Universal Command SMP/E FMID TUCM620 relative file 2.
UCM620F3.XMT	Universal Command SMP/E FMID TUCM620 relative file 3.
UDM620F1.XMT	Universal Data Mover SMP/E FMID TUDM620 relative file 1.
UDM620F2.XMT	Universal Data Mover SMP/E FMID TUDM620 relative file 2.
UDM620F3.XMT	Universal Data Mover SMP/E FMID TUDM620 relative file 3.
UEC620F1.XMT	Universal Enterprise Controller SMP/E FMID TUEC620 relative file 1.
UEC620F2.XMT	Universal Enterprise Controller SMP/E FMID TUEC620 relative file 2.
UEM620F1.XMT	Universal Event Monitor SMP/E FMID TUEM620 relative file 1.
UEM620F2.XMT	Universal Event Monitor SMP/E FMID TUEM620 relative file 2.
UEM620F3.XMT	Universal Event Monitor SMP/E FMID TUEM620 relative file 3.
UNV620F1.XMT	Universal Common SMP/E FMID TUNV620 relative file 1.
USD521F1.XMT	SAP RFC DLL FMID TUSD521 relative file 1.
USP521F1.XMT	Universal Connector SMP/E FMID TUSP521 relative file 1.
USP521F2.XMT	Universal Connector SMP/E FMID TUSP521 relative file 2.
USP521F3.XMT	Universal Connector SMP/E FMID TUSP521 relative file 3.
UTL620F1.XMT	Universal Utilities SMP/E FMID TUTL620 relative file 1.
UTL620F2.XMT	Universal Utilities SMP/E FMID TUTL620 relative file 2.
UXD620F1.XMT	IBM XML Toolkit FMID TUXD620 relative file 1.
z/OS USS Files	
UBR620U1.XMT	USS Universal Broker FMID UUBR620 relative file 1.
UBR620U2.XMT	USS Universal Broker FMID UUBR620 relative file 2.
UCM620U1.XMT	USS Universal Command FMID UUCM620 relative file 1.
UCM620U2.XMT	USS Universal Command FMID UUCM620 relative file 2.
UDM620U1.XMT	USS Universal Data Mover FMID UUDM620 relative file 1.
UDM620U2.XMT	USS Universal Data Mover FMID UUDM620 relative file 2.
UEM620U1.XMT	USS Universal Event Monitor FMID UUEM620 relative file 1.
UEM620U2.XMT	USS Universal Event Monitor FMID UUEM620 relative file 2.
UNV620U1.XMT	USS Universal Common FMID UUNV620 relative file 1.
UTL620U1.XMT	USS Universal Utilities FMID UUTL620 relative file 1.

zOS Installation - Transferring Installation Files to zOS

- Overview
- Transferring the Installation Files
  - Extract the Installation Files
  - Allocate z/OS Data Sets
  - Transfer the Installation Files
  - Unpack the Data Sets

Overview

After downloading the distribution file, you must extract the installation files and transfer them to the z/OS system on which to install Universal Agent for z/OS.

The XMIT (TSO TRANSMIT format) data sets must be unpacked with the TSO RECEIVE command.

Transferring the Installation Files

Transferring the installation files requires each XMT file extracted from the distribution file to be transferred manually from the Windows or UNIX-based workstation to the z/OS system.

1	Extract the installation files from the distribution file.
2	Allocate data sets on z/OS for the installation files.
3	Transfer the Installation Files.
4	Unpack the data sets.

Extract the Installation Files

Issue the following command to extract the installation (XMT) files from the distribution file:

```
zcat sb-6.2.x.x-zos.tar.Z | tar xvf -
```

Allocate z/OS Data Sets

The installation files (with a **.XMT** suffix) have been created using the TSO TRANSMIT command. They must be transferred (in binary mode) to z/OS data sets of the following format:

- Sequential (DSORG=PS)
- Fixed blocked record format (RECFM=FB)
- Record length 80 (LRECL=80)
- Block size 3120 (BLKSIZE=3120)

Use the sample JCL in Figure 1, below, to define a job that creates the TSO XMIT data sets required by the Universal Agent for z/OS installation.

The JCL includes a MODIFICATIONS section at the beginning that describes what JCL modifications are required prior to submitting the job. Read and complete each of the listed modifications.

Submit the job on z/OS to allocate the XMIT data sets. Return code 0 is expected.

Note
The data sets allocated by the USDF1 and USPF1, USPF2, and USPF3 DD statements include **521** instead of **620** in their names; the version of Universal Connector packaged with Universal Agent for z/OS 6.2.0 is release 5.2.1.0, and its FMIDs identify it as such.

Figure 1 Sample JCL

```
//UNVALLOC JOB CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
//             NOTIFY=&SYSUID,COND=(0,NE)
//*********************************************************************
//* (C) COPYRIGHT 2000-2011 STONEBRANCH, INC.  ALL RIGHTS RESERVED.
//*
//* STONEBRANCH, INC.
//* UNIVERSAL PRODUCTS
//*
//* UNVALLOC
//*
//* DESCRIPTION
//* -----------
//* ALLOCATE TSO TRANSMIT (XMIT) DATA SETS FOR THE DISTRIBUTION
//* DATA SETS.
//*
//* MODIFICATIONS
//* -------------
//* 1. MODIFY THE JOB STATEMENT TO MEET LOCAL REQUIREMENTS.
//*
//* 2. CHANGE ALL '#HLQ' TO THE HIGH-LEVEL QUALIFIER OF THE
//*    UNIVERSAL PRODUCTS DISTRIBUTION XMIT DATA SETS.
//*
//* 3. CHANGE ALL '#VOLSER' TO THE VOLUME SERIAL NAME ON WHICH TO
//*    ALLOCATE THE DATA SETS.
//*********************************************************************
//*
//STEP0    EXEC PGM=IEFBR14
//SMPMCS   DD  DSN=#HLQ.UNV.V6R2M0.SMPMCS.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(10,10))
//INSTALL  DD  DSN=#HLQ.UNV.V6R2M0.INSTALL.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UNVF1    DD  DSN=#HLQ.UNV.V6R2M0.TUNV620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UTLF1    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UTLF2    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(70,1))
//UBRF1    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UBRF2    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UBRF3    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(120,1))
//UCMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UCMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UCMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(45,1))
//UDMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UDMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UDMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
```

```
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(55,1))
//UEMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UEMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UEMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(15,1))
//UXDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUXD620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(110,10))
//UAGF1    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(2,1))
//UAGF2    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UAGF3    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(30,1))
//UECF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UECF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(70,1))
//USDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSD521.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(20,1))
//USPF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//USPF2    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//USPF3    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F3.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(15,1))
//UBRU1    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UBRU2    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(50,1))
//UCMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UCMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(30,1))
//UDMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UDMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(30,1))
//UEMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//UEMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F2.XMIT,
//             DISP=(MOD,DELETE,DELETE),
```

```
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(30,1))
//UTLU1    DD  DSN=#HLQ.UNV.V6R2M0.UUTL620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(60,1))
//UNVU1    DD  DSN=#HLQ.UNV.V6R2M0.UUNV620.F1.XMIT,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,SPACE=(CYL,(1,1))
//*
//STEP1    EXEC PGM=IEFBR14
//SMPMCS   DD  DSN=#HLQ.UNV.V6R2M0.SMPMCS.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(10,10))
//INSTALL  DD  DSN=#HLQ.UNV.V6R2M0.INSTALL.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UNVF1    DD  DSN=#HLQ.UNV.V6R2M0.TUNV620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UTLF1    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UTLF2    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(70,1))
//UBRF1    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UBRF2    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UBRF3    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(120,1))
//UCMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UCMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UCMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(45,1))
//UDMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UDMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UDMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(55,1))
//UEMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F1.XMIT,
```

```
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UEMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UEMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(15,1))
//UXDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUXD620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(110,10))
//UAGF1    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(2,1))
//UAGF2    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UAGF3    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(30,1))
//UECF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UECF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(70,1))
//USDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSD521.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(20,1))
//USPF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//USPF2    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//USPF3    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F3.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(5,1))
//UBRU1    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UBRU2    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(50,1))
//UCMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UCMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
```

```
//             SPACE=(CYL,(30,1))
//UDMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UDMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(30,1))
//UEMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
//UEMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F2.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(30,1))
//UTLU1    DD  DSN=#HLQ.UNV.V6R2M0.UUTL620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(60,1))
//UNVU1    DD  DSN=#HLQ.UNV.V6R2M0.UUNV620.F1.XMIT,
//             DISP=(,CATLG),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
```

```
//             UNIT=SYSALLDA,VOL=SER=#VOLSER,
//             SPACE=(CYL,(1,1))
```

Transfer the Installation Files

Transfer the installation files to the allocated z/OS XMIT data sets. A binary transfer must be performed; otherwise, the XMIT data sets will not unpack correctly.

Due to the numerous file transfer products available on the market, it is impractical to provide instructions for them all. FTP is one of the most popular; an example Windows FTP script is provided below. It can be used from a Windows workstation that can establish a TCP/IP connection to the z/OS host.

The following FTP script modifications are required:

Step 1	Change #HOSTNAME to the z/OS host name or IP address.
Step 2	Change #USERNAME to a z/OS user ID that has updated access to the XMIT data sets being updated.
Step 3	Change #PASSWORD to the z/OS user ID's password.
Step 4	Change #HLQ to the data set high level qualifier used to allocate the XMIT data sets.

Note
The file data set names used for Universal Connector components include **521** instead of **620** in their names; the version of Universal Connector packaged with Universal Agent for z/OS 6.2.0 is release 5.2.1.0, and its FMIDs identify it as such.

```
open #HOSTNAME
user #USERNAME #PASSWORD

binary
put INSTALL.XMT '#HLQ.UNV.V6R2M0.INSTALL.XMIT'
put SMPMCS.XMT '#HLQ.UNV.V6R2M0.SMPMCS.XMIT'
put UAG620F1.XMT '#HLQ.UNV.V6R2M0.TUAG620.F1.XMIT'
put UAG620F2.XMT '#HLQ.UNV.V6R2M0.TUAG620.F2.XMIT'
put UAG620F3.XMT '#HLQ.UNV.V6R2M0.TUAG620.F3.XMIT'
put UBR620F1.XMT '#HLQ.UNV.V6R2M0.TUBR620.F1.XMIT'
put UBR620F2.XMT '#HLQ.UNV.V6R2M0.TUBR620.F2.XMIT'
put UBR620F3.XMT '#HLQ.UNV.V6R2M0.TUBR620.F3.XMIT'
put UBR620U1.XMT '#HLQ.UNV.V6R2M0.UUBR620.F1.XMIT'
put UBR620U2.XMT '#HLQ.UNV.V6R2M0.UUBR620.F2.XMIT'
put UCM620F1.XMT '#HLQ.UNV.V6R2M0.TUCM620.F1.XMIT'
put UCM620F2.XMT '#HLQ.UNV.V6R2M0.TUCM620.F2.XMIT'
put UCM620F3.XMT '#HLQ.UNV.V6R2M0.TUCM620.F3.XMIT'
put UCM620U1.XMT '#HLQ.UNV.V6R2M0.UUCM620.F1.XMIT'
put UCM620U2.XMT '#HLQ.UNV.V6R2M0.UUCM620.F2.XMIT'
put UDM620F1.XMT '#HLQ.UNV.V6R2M0.TUDM620.F1.XMIT'
put UDM620F2.XMT '#HLQ.UNV.V6R2M0.TUDM620.F2.XMIT'
put UDM620F3.XMT '#HLQ.UNV.V6R2M0.TUDM620.F3.XMIT'
put UDM620U1.XMT '#HLQ.UNV.V6R2M0.UUDM620.F1.XMIT'
put UDM620U2.XMT '#HLQ.UNV.V6R2M0.UUDM620.F2.XMIT'
put UEC620F1.XMT '#HLQ.UNV.V6R2M0.TUEC620.F1.XMIT'
put UEC620F2.XMT '#HLQ.UNV.V6R2M0.TUEC620.F2.XMIT'
put UEM620F1.XMT '#HLQ.UNV.V6R2M0.TUEM620.F1.XMIT'
put UEM620F2.XMT '#HLQ.UNV.V6R2M0.TUEM620.F2.XMIT'
put UEM620F3.XMT '#HLQ.UNV.V6R2M0.TUEM620.F3.XMIT'
put UEM620U1.XMT '#HLQ.UNV.V6R2M0.UUEM620.F1.XMIT'
put UEM620U2.XMT '#HLQ.UNV.V6R2M0.UUEM620.F2.XMIT'
put UNV620F1.XMT '#HLQ.UNV.V6R2M0.TUNV620.F1.XMIT'
put UNV620U1.XMT '#HLQ.UNV.V6R2M0.UUNV620.F1.XMIT'
put USD521F1.XMT '#HLQ.UNV.V6R2M0.TUSD521.F1.XMIT'
put USP521F1.XMT '#HLQ.UNV.V6R2M0.TUSP521.F1.XMIT'
put USP521F2.XMT '#HLQ.UNV.V6R2M0.TUSP521.F2.XMIT'
put USP521F3.XMT '#HLQ.UNV.V6R2M0.TUSP521.F3.XMIT'
put UTL620F1.XMT '#HLQ.UNV.V6R2M0.TUTL620.F1.XMIT'
put UTL620F2.XMT '#HLQ.UNV.V6R2M0.TUTL620.F2.XMIT'
put UTL620U1.XMT '#HLQ.UNV.V6R2M0.UUTL620.F1.XMIT'
put UXD620F1.XMT '#HLQ.UNV.V6R2M0.TUXD620.F1.XMIT'
```

The FTP script is executed with the FTP `-s` option. Using a Command Prompt window, change to the directory where the installation files were unpacked. The example above used directory name `\upinstall`. Assuming the script is saved in file name `ftp.script`, the FTP script is executed with the following FTP command:

```
C:\upinstall> ftp -s:ftp.script
```

The FTP output must be reviewed to confirm each of the files are transferred successfully.

Unpack the Data Sets

After transferring the installation files to their allocated data sets on z/OS, you must unpack them using the TSO RECEIVE command. You can execute the TSO RECEIVE command interactively or in batch.

Use the sample JCL in Figure 2, below, to define a job that performs a TSO RECEIVE on each of the transmitted data sets provided by the Universal Agent for z/OS distribution file.

The JCL includes a MODIFICATIONS section at the beginning of the file that describes what JCL modifications are required prior to submitting the job. Read and complete each of the listed modifications.

Submit the job on z/OS to unpack the data sets. Return code 0 is expected.

After the data sets are successfully unpacked, they are no longer required and can be deleted. The files then are ready to be installed.

Note
The data sets allocated by the USDF1 and USPF1, USPF2, and USPF3 DD statements include **521** instead of **620** in their names; the version of Universal Connector packaged with Universal Agent for z/OS 6.2.0 is release 5.2.1.0, and its FMIDs identify it as such.

Figure 2 Sample JCL

```
//UNVRECV  JOB CLASS=A,MSGCLASS=X,MSGLEVEL=(1,1),
//             NOTIFY=&SYSUID,COND=(0,NE)
//*****************************************************************
//* (C) COPYRIGHT 2000-2011 STONEBRANCH, INC.  ALL RIGHTS RESERVED.
//*
//* STONEBRANCH, INC.
//* UNIVERSAL PRODUCTS
//*
//* UNVRECV
//*
//* DESCRIPTION
//* -----------
//* PERFORM A TSO RECEIVE ON THE TRANSMITTED DISTRIBUTION DATA SETS.
//* ALL STEPS MUST END WITH RC 0.
//*
//* MODIFICATIONS
//* -------------
//* 1  MODIFY THE JOB STATEMENT TO MEET LOCAL REQUIREMENTS.
//*
//* 2  CHANGE ALL '#HLQ' TO THE HIGH-LEVEL QUALIFIER OF THE
//*    UNIVERSAL PRODUCTS DATA SETS.
//*
//* 3  CHANGE ALL 'VOLSER' TO THE VOLUME SERIAL NAME ON WHICH TO
//*    ALLOCATE THE DISTRIBUTION DATA SETS.
//*****************************************************************
//*
//STEP00 EXEC PGM=IEFBR14
//SMPMCS   DD  DSN=#HLQ.UNV.V6R2M0.SMPMCS,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(100,50),RLSE)
//INSTALL  DD  DSN=#HLQ.UNV.V6R2M0.INSTALL,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(15,10),RLSE)
//UNVF1    DD  DSN=#HLQ.UNV.V6R2M0.TUNV620.F1,
//             DISP=(MOD,DELETE,DELETE),
```

```
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(15,1),RLSE)
//UTLF1    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(15,1),RLSE)
//UTLF2    DD  DSN=#HLQ.UNV.V6R2M0.TUTL620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(1000,10),RLSE)
//UBRF1    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(15,1),RLSE)
//UBRF2    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(15,1),RLSE)
//UBRF3    DD  DSN=#HLQ.UNV.V6R2M0.TUBR620.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(1700,15),RLSE)
//UCMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UCMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UCMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUCM620.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(600,15),RLSE)
//UDMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UDMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UDMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUDM620.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(750,15),RLSE)
//UEMF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UEMF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UEMF3    DD  DSN=#HLQ.UNV.V6R2M0.TUEM620.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(200,15),RLSE)
//UXDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUXD620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(1530,15),RLSE)
//UAGF1    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UAGF2    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UAGF3    DD  DSN=#HLQ.UNV.V6R2M0.TUAG620.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(300,15),RLSE)
//UECF1    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UECF2    DD  DSN=#HLQ.UNV.V6R2M0.TUEC620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(1000,15),RLSE)
//USDF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSD521.F1,
```

```
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(300,15),RLSE)
//USPF1    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//USPF2    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//USPF3    DD  DSN=#HLQ.UNV.V6R2M0.TUSP521.F3,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(200,15),RLSE)
//UBRU1    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UBRU2    DD  DSN=#HLQ.UNV.V6R2M0.UUBR620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(500,50),RLSE)
//UCMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UCMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUCM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(500,50),RLSE)
//UDMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UDMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUDM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(500,50),RLSE)
//UEMU1    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//UEMU2    DD  DSN=#HLQ.UNV.V6R2M0.UUEM620.F2,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(500,50),RLSE)
//UTLU1    DD  DSN=#HLQ.UNV.V6R2M0.UUTL620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(1000,15),RLSE)
//UNVU1    DD  DSN=#HLQ.UNV.V6R2M0.UUNV620.F1,
//             DISP=(MOD,DELETE,DELETE),
//             DCB=(DSORG=PS,RECFM=FB,LRECL=80,BLKSIZE=3120),
//             UNIT=SYSALLDA,SPACE=(TRK,(10,1),RLSE)
//*
//TSO      PROC
//S1       EXEC PGM=IKJEFT01
//SYSLBC   DD  DISP=SHR,DSN=SYS1.BRODCAST
//SYSPROC  DD  DUMMY
//SYSPRINT DD  SYSOUT=*
//SYSTSPRT DD  SYSOUT=*
//SYSTERM  DD  SYSOUT=*
//SYSTSIN  DD  DUMMY
//         PEND
//*
//SMPMCS   EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.SMPMCS.XMIT')
    DA('#HLQ.UNV.V6R2M0.SMPMCS') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(100 50) RELEASE
//INSTALL  EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.INSTALL.XMIT')
    DA('#HLQ.UNV.V6R2M0.INSTALL') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(15 10)
//UNV620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUNV620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUNV620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
```

```
    TRACKS SPACE(15 1) RELEASE
//UTL620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUTL620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUTL620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(15 1) RELEASE
//UTL620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUTL620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUTL620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(1000 10) RELEASE
//UBR620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUBR620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUBR620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(15 1) RELEASE
//UBR620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUBR620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUBR620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(15 1) RELEASE
//UBR620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUBR620.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUBR620.F3') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(1700 15) RELEASE
//UCM620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUCM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUCM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UCM620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUCM620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUCM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UCM620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUCM620.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUCM620.F3') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(600 15) RELEASE
//UDM620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUDM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUDM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UDM620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUDM620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUDM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UDM620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUDM620.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUDM620.F3') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(750 15) RELEASE
//UEM620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUEM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUEM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UEM620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUEM620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUEM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UEM620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUEM620.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUEM620.F3') +
```

```
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(200 15) RELEASE
//UXD620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUXD620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUXD620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(1530 15) RELEASE
//UAG620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUAG620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUAG620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UAG620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUAG620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUAG620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UAG620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUAG620.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUAG620.F3') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(300 15) RELEASE
//UEC620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUEC620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUEC620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UEC620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUEC620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUEC620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(1000 15) RELEASE
//USD521F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUSD521.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUSD521.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(300 15) RELEASE
//USP620F1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUSP521.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUSP521.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//USP620F2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUSP521.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUSP521.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//USP620F3 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.TUSP521.F3.XMIT')
    DA('#HLQ.UNV.V6R2M0.TUSP521.F3') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(200 15) RELEASE
//UBR620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUBR620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUBR620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UBR620U2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUBR620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUBR620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(500 50) RELEASE
//UCM620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUCM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUCM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UCM620U2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUCM620.F2.XMIT')
```

```
    DA('#HLQ.UNV.V6R2M0.UUCM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(500 50) RELEASE
//UDM620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUDM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUDM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UDM620U2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUDM620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUDM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(500 50) RELEASE
//UEM620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUEM620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUEM620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
//UEM620U2 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUEM620.F2.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUEM620.F2') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(500 50) RELEASE
//UTL620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUTL620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUTL620.F1') +
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(1000 15) RELEASE
//UNV620U1 EXEC TSO
//S1.SYSTSIN DD *
  RECEIVE INDA('#HLQ.UNV.V6R2M0.UUNV620.F1.XMIT')
    DA('#HLQ.UNV.V6R2M0.UUNV620.F1') +
```

```
    UNIT(SYSALLDA) VOL(VOLSER) +
    TRACKS SPACE(10 1) RELEASE
```

zOS Installation - SMPE Installation

- SMP/E Installation of Universal Agent for z/OS
- SMP/E
  - SMP/E FMIDs

SMP/E Installation of Universal Agent for z/OS

Universal Agent for z/OS is installed using SMP/E. The pages in this section describe how to perform SMP/E installation of the Universal Agent for z/OS package in a step-by-step process.

Six different installation processes are provided. The installation process that you use depends on the installation environment.

New Install, New CSI	Install any Universal Agent package from Stonebranch, Inc. for the first time, or instal a Universal Agent package in a new SMP/E CSI.
Universal Agent 5.2.0 Upgrade, Existing CSI	Upgrade a Universal Agent 5.2.0 package install. In this case, the Universal Agent package is installed into an SMP/E CSI that contains a Universal Agent 5.2.0 package.
Workload Automation 5.1.0 Upgrade, Existing CSI	Upgrade a Workload Automation 5.1.0 package install. In this case, the Universal Agent package is installed into an SMP/E CSI that contains a Workload Automation 5.1.0 package.
Stonebranch Solutions 4.x Upgrade, Existing CSI	Upgrade a Stonebranch Solutions 4.x package install. In this case, the Universal Agent package is installed into an SMP/E CSI that contains a Stonebranch Solutions 4.x package.
Universal Products 3.2.0 Upgrade, Existing CSI	Upgrade a Universal Products 3.2.0 package install. In this case, the Universal Agent package is installed into an SMP/E CSI that contains a Universal Products 3.2.0 package.
Universal Products 3.1.1 (and earlier) Upgrade, Existing CSI	To upgrade from Universal Products 3.1.1 or earlier, you first must upgrade to Universal Products 3.2.0, and then upgrade to Universal Agent 5.2.0. For information on upgrading to Universal Products 3.2.0, see the Universal Products Installation Guide in the Universal Products 3.2.0 Documentation set.

SMP/E

The Universal Agent for z/OS components are installed in the Universal Agent SMP/E CSI. This CSI should not be shared with any other vendor products; it should be used exclusively for Universal Agent.

Before making any changes to the SMP/E environment, back up the environment per your local procedures. IBM recommends backing up the entire SMP/E pack before any SMP/E installation begins. Two sample JCL members are provide in members **UNVBKUP** and **UNVREST**.

As of Universal Products version 2.2.0, all Universal Agent components share common SMP/E target and distribution libraries:

- Product ID is **UNV**.
- Last qualifier of target libraries start with **SUNV**.
- Last qualifier of distribution libraries start with **AUNV**.

SMP/E FMIDs

The following table identifies the SMP/E FMIDs for the Universal Agent for z/OS components. (For the list of SMP/E FMIDs for the Universal Agent for z/OS USS components, see z/OS USS Installation - Installation Requirements.)

Product	FMID	SMP/E Requisites
Universal Common 6.2.0	TUNV620	Supersedes and deletes FMID TUNV220, TUNV310, TUNV311, TUNV320, TUNV410, TUNV420, TUNV430, TUN510, and TUN520.
Universal Automation Center Agent 6.2.0	TUAG620	TUNV620 is a prerequisite.

Universal Broker 6.2.0	TUBR620	TUNV620 is a prerequisite. Supersedes and deletes FMID TUBR310, TUBR311, TUBR320, TUBR410, TUBR420, TUBR430, TUBR510, and TUBR520.
Universal Command 6.2.0	TUCM620	TUBR620 is a prerequisite. Supersedes and deletes FMID TUCM110, TUCM120, TUCM210, TUCM220, TUCM310, TUCM311, TUCM320, TUCM410, TUCM420, TUCM430, TUCM510, and TUCM520.
Universal Enterprise Controller 6.2.0	TUEC620	TUNV620 is a prerequisite. Supersedes and deletes FMID TUEC110, TUEC310, TUEC320, TUEC410, TUEC420, TUEC430, TUEC510, and TUEC520.
Universal Data Mover 6.2.0	TUDM620	TUBR620 is a prerequisite. Supersedes and deletes FMID TUDM110, TUDM310, TUDM311, TUDM320, TUDM410, TUDM420, TUDM430, TUDM510, and TUDM520.
SAP RFC DLL	TUSD521	TUNV620 is a prerequisite. Supersedes and deletes FMID TUSD310, TUSD311, TUSD320, TUSD410, TUSD420, TUSD430, TUSD510, and TUSD520.
Universal Connector	TUSP521	TUSD521 is a prerequisite. Supersedes and deletes FMID TUSP120, TUSP310, TUSP311, TUSP320, TUSP410, TYSP420, TUSP430, TUSP510, and TUSP520.
Universal Utilities	TUTL620	TUNV620 is a prerequisite. Supersedes and deletes FMID TUEN110, TUEN120, TUEN210, TUEN220, TUEN310, TUEN311, TUTL320, TUTL410, TUTL420, TUTL430, TUTL510, and TUTL520.
Universal Event Monitor	TUEM620	TUBR620 is a prerequisite. Supersedes and deletes FMID TUEM310, TUEM311, TUEM320, TUEM410, TUEM420, TUEM430, TUEM510, and TUEM520.
IBM XML Toolkit	TUXD620	TUNV620 is a prerequisite. Supersedes and deletes TUXD320, TUXD410, TUXD420, TUXD430, TUXD510, and TUXD520.

zOS Installation - New Install, New CSI

New Install, New CSI

The New Install, New CSI installation process describes how to install the Universal Agent package in a newly allocated SMP/E CSI.

Use this installation process for either of these environments:

- Installing a Universal Agent package for the first time.
- Installing a Universal Agent package in a different SMP/E CSI than other Universal Agent components.

The installation JCL referenced by the following installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

Step 1	Edit and submit the JCL in member **#SETUP**. The **#SETUP** JCL creates customized installation JCL used by the following installation steps and for product maintenance and customization. All the JCL is created as members in the **INSTALL** library. If you want to change the customizations after the job is executed, edit the **#SETUP** member with the new customizations and resubmit the job. All **INSTALL** library members will be replaced. • All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVIN01**. The JCL allocates the SMP/E CSI data sets. • All steps must end with a return code 0.
Step 3	Submit the JCL in member **UNVIN02**. The JCL initializes the SMP/E CSI. • All steps must end with a return code 0.
Step 4	Submit the JCL in member **UNVIN03**. The JCL allocates Universal Agent target and distribution data sets and adds SMP/E **DDDEF** definitions to the Universal Agent CSI zones. • All steps must end with a return code 0.
Step 5	Submit the JCL in member **UNVIN04**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. • All steps must end with a return code 0.

Step 6	Submit the JCL in member **UNVIN05**. The JCL performs an SMP/E APPLY of the product FMIDs and any received PTFs.
	
	• Step APYFMID must end with a condition code of 0.
	• Step APYPTFS is considered successful under any of the following conditions:
	  • Step ends with condition code 0.
	  • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**.
	  • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.

Step 7	Submit the JCL in member **UNVIN06**. The JCL member performs an SMP/E ACCEPT of the product FMIDs and any applied PTFs.
	
	• Step ACCFMID must end with a condition code of 0.
	• Step ACCPTFS is considered successful under any of the following conditions:
	  • Step ends with condition code 0 or 4.
	  • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.

Step 8	Submit the JCL in member **UNVIN07**. The JCL member allocates product non-SMP/E data sets and databases, and formats the zFS data sets. • All steps must end with a return code 0.

Step 9	The product databases are allocated as z/OS UNIX zFS file systems. Additional product configuration is required to utilize zFS data sets. See Universal Agent Database Configuration for details. If you prefer to use HFS file systems, HFS data sets must be allocated. If you prefer to use the zFS data sets, skip this step. Submit the JCL in member **UNVINHF**. The JCL renames the zFS data sets and defines their HFS counterparts. • All steps must end with return code 0.
Step 10	Submit the JCL in member **UNVIN08**. The JCL copies sample configuration members to the configuration libraries. • All steps must end with a return code 0.
Step 11	Edit and submit the JCL in member **UNVIN09**. The JCL requires modifications as listed in the MODIFICATIONS section of the comments at the top of the JCL. The JCL copies the Universal Enterprise Controller and Universal Broker started procedure JCL to a system procedure library. • All steps must end with a return code 0.
Step 12	Perform required z/OS configuration steps as described in z/OS Installation - Configuration.

zOS Installation - Universal Agent 5.2.0 Upgrade, Existing CSI

Universal Agent 5.2.0 Upgrade, Existing CSI

The Universal Agent 5.2.0 Upgrade, Existing CSI installation process describes how to upgrade an existing Universal Agent 5.2.0 package in an existing SMP/E CSI.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library. Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

Step 1	Edit and submit the JCL in member **#SETUP**. The **#SETUP** JCL creates customized installation JCL used by the following installation steps and for product maintenance and customization. All the JCL is created as members in the **INSTALL** library. To change customizations after the job is executed, edit the **#SETUP** member with the new customizations and resubmit the job. All **INSTALL** library members will be replaced. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVIN04**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 3	Submit the JCL in member **UNVIN05**. The JCL performs an SMP/E APPLY of the product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVIN06**. The JCL member performs an SMP/E ACCEPT of product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.
Step 5	Submit the JCL in member **UNVIN08**. The JCL copies sample configuration members to the configuration libraries. All steps must end with a return code 0.
Step 6	Edit and submit the JCL in member **UNVIN09**. The JCL requires modifications as listed in the MODIFICATIONS section of the comments at the top of the JCL. The JCL copies the Universal Enterprise Controller and Universal Broker started procedure JCL to a system procedure library. All steps must end with a return code 0.
Step 7	Perform required z/OS configuration steps as described in z/OS Installation - Configuration.

zOS Installation - Workload Automation 5.1.0 Upgrade, Existing CSI

Workload Automation 5.1.0 Upgrade, Existing CSI

The Workload Automation 5.1.0 Upgrade, Existing CSI installation process describes how to upgrade an existing Workload Automation 5.1.0 package in an existing SMP/E CSI.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library. Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

Step 1	Edit and submit the JCL in member **#SETUP**. The **#SETUP** JCL creates customized installation JCL used by the following installation steps and for product maintenance and customization. All the JCL is created as members in the **INSTALL** library. To change customizations after the job is executed, edit the **#SETUP** member with the new customizations and resubmit the job. All **INSTALL** library members will be replaced. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVIN04**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 3	Submit the JCL in member **UNVIN05**. The JCL performs an SMP/E APPLY of the product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVIN06**. The JCL member performs an SMP/E ACCEPT of product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.
Step 5	Submit the JCL in member **UNVIN08**. The JCL copies sample configuration members to the configuration libraries. All steps must end with a return code 0.
Step 6	Edit and submit the JCL in member **UNVIN09**. The JCL requires modifications as listed in the MODIFICATIONS section of the comments at the top of the JCL. The JCL copies the Universal Enterprise Controller and Universal Broker started procedure JCL to a system procedure library. All steps must end with a return code 0.
Step 7	Perform required z/OS configuration steps as described in z/OS Installation - Configuration.

zOS Installation - Stonebranch Solutions 4.x Upgrade, Existing CSI

Stonebranch Solutions 4.x Upgrade, Existing CSI

The Stonebranch Solutions 4.x Upgrade, Existing CSI installation process describes how to upgrade an existing Stonebranch Solutions 4.x package in an existing SMP/E CSI.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library. Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

Step 1	Edit and submit the JCL in member **#SETUP**. The **#SETUP** JCL creates customized installation JCL used by the following installation steps and for product maintenance and customization. All the JCL is created as members in the **INSTALL** library. To change customizations after the job is executed, edit the **#SETUP** member with the new customizations and resubmit the job. All **INSTALL** library members will be replaced. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUG4XX**. The JCL removes the UAR configuration and component definition members in the non-SMP/E UNVCONF and UNVCOMP libraries, respectively, and allocates the UAG model data set. All steps must end with return code 0.
Step 3	Submit the JCL in member **UNVIN04**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 4	Submit the JCL in member **UNVIN05**. The JCL performs an SMP/E APPLY of the product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 5	Submit the JCL in member **UNVIN06**. The JCL member performs an SMP/E ACCEPT of product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.
Step 6	Submit the JCL in member **UNVIN08**. The JCL copies sample configuration members to the configuration libraries. All steps must end with a return code 0.
Step 7	Edit and submit the JCL in member **UNVIN09**. The JCL requires modifications as listed in the MODIFICATIONS section of the comments at the top of the JCL. The JCL copies the Universal Enterprise Controller and Universal Broker started procedure JCL to a system procedure library. All steps must end with a return code 0.
Step 8	Perform required z/OS configuration steps as described in z/OS Installation - Configuration.

zOS Installation - Universal Products 3.2.0 Upgrade, Existing CSI

Universal Products 3.2.0 Upgrade, Existing CSI

The Universal Products 3.2.0 Upgrade, Existing CSI installation process describes how to upgrade an existing Universal Products 3.2.0 package in an existing SMP/E CSI.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library. Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

Step 1	Edit and submit the JCL in member **#SETUP**. The **#SETUP** JCL creates customized installation JCL used by the following installation steps and for product maintenance and customization. All the JCL is created as members in the **INSTALL** library. To change customizations after the job is executed, edit the **#SETUP** member with the new customizations and resubmit the job. All **INSTALL** library members will be replaced. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUG320**. The JCL upgrades the SMP/E Universal Products configuration from 3.2.0 to 5.2.0, and allocates the UAG model data set. The following changes are made: • Allocates a new AUNVHBIN SMP/E distribution data set, if necessary.
Step 3	Submit the JCL in member **UNVIN04**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 4	Submit the JCL in member **UNVIN05**. The JCL performs an SMP/E APPLY of the product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 5	Submit the JCL in member **UNVIN06**. The JCL member performs an SMP/E ACCEPT of product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.
Step 6	Submit the JCL in member **UNVIN08**. The JCL copies sample configuration members to the configuration libraries. All steps must end with a return code 0.
Step 7	Edit and submit the JCL in member **UNVIN09**. The JCL requires modifications as listed in the MODIFICATIONS section of the comments at the top of the JCL. The JCL copies the Universal Enterprise Controller and Universal Broker started procedure JCL to a system procedure library. All steps must end with a return code 0.
Step 8	Perform required z/OS configuration steps as described in z/OS Installation - Configuration.

zOS Installation - Configuration

Overview

The following pages describe the z/OS configuration steps required for the Universal Broker, Universal Enterprise Controller, and Universal Automation Center Agent:

- Started Tasks
- Load Library
- SMF Exits
- JES SYSOUT Processing
- Universal Agent Database Configuration

The Universal Broker component is a required component. Universal Enterprise Controller is optional. Universal Automation Center Agent is required to manage workload with Universal Controller.

The following configuration steps are described:

- Installing the started tasks and setting up the started task user IDs and group IDs.
- APF authorizing the SUNVLOAD library, adding it to program control and adding specific programs to the LNKLST.
- Installation of SMF exits.
- Configuration of Universal Spool and Universal Enterprise Controller HFS or zFS databases.

Member **UNVINRF** in the **INSTALL** library provides a sample JCL that can be used to execute the RACF commands described in the following pages as a TSO batch job.

Carefully read the comments in the member before submitting the JCL. The JCL is provided only as a sample. Follow local procedures as necessary for defining and altering security resources.

zOS Configuration - Started Tasks

- z/OS Configuration - Started Tasks
- Started Task Security
- Universal Broker
  - Universal Broker User and Group Profiles
  - Universal Broker Data Access
- Universal Enterprise Controller
  - Universal Enterprise Controller User and Group Profiles
  - Universal Enterprise Controller Data Access
- Associate Started Tasks with User and Group Profiles
  - STARTED Class Profile
  - Started Procedures Table

z/OS Configuration - Started Tasks

The Universal Agent for z/OS solution consists of two started tasks:

1. Universal Broker is a required started task that provides a number of services for manager and server components.
2. Universal Enterprise Controller is an optional started task that provides monitoring and administration services.

The following started task JCL procedures are provided in the **SUNVSAMP** library:

- **UBROKER** is the JCL procedure for the Universal Broker started task.
- **UECTLR** is the JCL procedure for the Universal Enterprise Controller started task.

The JCL procedures are copied to a system procedure library by installation job **UNVIN09**. If this was not done, copy the JCL members to the appropriate procedure library for your local environment.

The started task programs utilize both z/OS UNIX System Services (USS) and MVS services. As a result of using USS services, the programs execute as USS processes. As do all USS processes, the Universal Broker and Universal Enterprise Controller processes must execute with user profiles that have a properly defined OMVS segments. Additionally, the user profiles must be permitted security access to privileged USS services in order for them to perform specific functions.

This page describes the following configuration tasks:

- Started task security requirements.
- Defining the started task user and group profiles.
- Permitting the started task user profile to required security resources.
- Associating started tasks with the user and group profiles.

Started Task Security

z/OS UNIX System Services (USS) operates in one of two different security modes. If the resource profile BPX.DAEMON is not defined, USS is operating in UNIX-level security mode; otherwise, USS is operating in z/OS UNIX security mode.

UNIX-level security provides few options to control access to USS services. A user account that requires access to privileged services must be defined with a UID value of 0, which is referred to as superuser.

z/OS UNIX security provides better access controls to USS services using a SAF security package, such as RACF. A user account can be defined with a non-zero UID and granted permissions to specific resource profiles that protect USS services. Superuser access is granted not with UID 0 but with READ access to the resource BPX.SUPERUSER in the FACILITY class.

Universal Broker

The Universal Broker started task provides services for local and remote Universal Agent managers, such as Universal Command managers or Universal Data Mover managers. Locally executed managers register with the local Universal Broker for monitoring, configuration data, and event recording. Remotely executed managers communicate with the local Universal Broker over a TCP/IP network connection and request execution of server components in order to process local services, such as execution of commands or transfer of data.

Server components initiated by the Universal Broker started task execute as child processes of the Universal Broker process. As such, the server components inherit the user identifier of the Universal Broker process. Some server components, such as Universal Command server, can switch the user identifier with which the work is executed. Switching a user identifier is a privileged operation. The Universal Broker user profile requires sufficient security access for itself and the server components to perform their services.

Universal Broker User and Group Profiles

The security requirements of the Universal Broker depend upon which services are being utilized. By default, all services are configured to be

used. Some services can be disabled to reduce the amount of authority the Universal Broker user profile requires. The following table lists the USS privileged services for each component and how to disable the service so that security access to the service is not required.

Service	Description	Disabling
Change directory ownership	Universal Broker dynamically mounts its USS file systems. Once the file systems are mounted, the Broker will initialize them. Initialization consists of changing the ownership of the file systems root directories to the Broker user identifier.	Initialize the file system ownership manually. The Broker will not dynamically mount the file systems and initialize them if they are already mounted and initialized.
Create external links owned by UID 0	Universal Broker dynamically creates external links on the USS file system to the server component MVS programs in its STEPLIB ddname allocation. The external links are required for the USS *spawn* function used by the Universal Broker to execute the server components. The external links must be owned by UID 0 when they link to a MVS program that resides in an APF authorized library and are link edited with AC=1. Both the UAGSRV program and UDMSRV programs are link edited with AC=1. Creation of the external links so that they are owned by UID 0 requires superuser access.	Create the external links manually at a permanent location in the USS file system. The external links must be owned by UID 0. Specify the external link absolute path name in the component definition START_COMMAND option for both UAGSRV and UDMSRV, which is located in UNVCOMP(UAGCMP00) and UNVCOMP(UDSCMP00), respectively.
Switch user ID and group associations	Universal Command, Universal Data Mover, and Universal Control switch their user IDs with which a work request is executed. The user ID is first authenticated before switching unless there is a Universal Access Control List (UACL) entry that turns authentication off for the request.	Set the Universal Command, Universal Data Mover, and Universal Control servers configuration SECURITY option to a value of NONE. With SECURITY set to NONE, all work requests are executed with the Universal Broker's user ID.
Change server component job name	Universal Broker will set the job name of child server processes to the appropriate component name. For example, when the Universal Broker starts a Universal Command server component, the job name is set to UCMSRV.	There is no product configuration option to disable this. By simply not permitting the Broker to the resource profile that protects it, all server components will run with the Universal Broker job name with a numeric value appended to it (for example, UBROKER2).

How to configure the Universal Broker started task user profile to meet security requirements depends on whether USS is running with UNIX-level security or z/OS UNIX security. The following sections describe how to configure the Universal Broker user profile to perform the privileged services listed above for both USS UNIX-level security and z/OS UNIX security configurations.

UNIX-level Security

UNIX-level security refers to a USS security environment where the resource profile BPX.DAEMON is not defined to the FACILITY class.

The only method of permitting a user profile access to privileged services is to define the user profile with a UID value of 0 (superuser). The Universal Broker user profile must be defined with UID 0 to perform any privileged service.

The following steps define the Universal Broker user profile for a UNIX-level security environment:

Step 1 Add the Universal Broker group profile **UBRGRP** using the following RACF command:

```
ADDGROUP UBRGRP OWNER(SYS1) OMVS(GID(5001))
```

Change the GID value 5001 to a value suitable for your local USS environment. The GID value must be unique among all group profiles.

Step 2 Add the Universal Broker user profile **UBRUSR** using the following RACF command:

```
ADDUSER UBRUSR DFLTGRP(UBRGRP) OWNER(SYS1) NOPASSWORD OMVS(UID(0))
```

z/OS UNIX Security

z/OS UNIX security refers to a USS security environment where the resource profile BPX.DAEMON is defined to the FACILITY class.

The Universal Broker user profile must be defined with a valid OMVS segment with a non-zero, unique UID value. The user profile security requirements are listed in the following table for each privileged service.

Service	Requirement
Change directory ownership	READ access to BPX.SUPERUSER resource profile in the FACILITY class.
Create external links owned by UID 0	READ access to BPX.SUPERUSER resource profile in the FACILITY class.
Switch user ID and group associations	READ access to BPX.DAEMON and BPX.SUPERUSER resource profiles in the FACILITY class.
Change server component job name	READ access to the BPX.JOBNAME profile in the FACILITY class.

The security requirements can be lifted if the feature that utilizes the service is disabled as described above.

The following steps configure the Universal Broker user profile for a z/OS UNIX security environment:

Step 1 Add the Universal Broker group profile **UBRGRP** using the following RACF command:

```
ADDGROUP UBRGRP OWNER(SYS1) OMVS(GID(5001))
```

Change the GID value 5001 to a value suitable for your local USS environment. The GID value must be unique among all group profiles.

Step 2 Add the Universal Broker user profile **UBRUSR** using the following RACF command:

```
ADDUSER UBRUSR DFLTGRP(UBRGRP) OWNER(SYS1) NOPASSWORD OMVS(UID(5001))
```

Change the UID value 5001 to a value suitable for your local USS environment. The value must be unique among all user profiles.

Step 3 Permit the Universal Broker user profile READ access to the resource profiles required for enabled services. The following RACF commands permit the user profile to the resources required for all privileged services:

```
PE BPX.DAEMON CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
PE BPX.SUPERUSER CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
PE BPX.JOBNAME CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
SETR RACLIST(FACILITY) REFRESH
```

Universal Broker Data Access

The Universal Broker user profile **UBRUSR** requires the following access to the data sets allocated in the Universal Broker started task, **UBROKER**:

Ddname	Access	Description
STEPLIB	READ	Program library
UNVCONF	ALTER	Product configuration data
UNVRFC	READ	Universal Connector SAP RFC file
UNVCOMP	ALTER	Product component definition data
UNVNLS	READ	Product national language support data
UNVCREF	READ	Universal Command command reference definitions
UNVDB	UPDATE	Universal Broker HFS component database
UNVSPOOL	UPDATE	Universal Broker HFS spool database

UNVTMPL	READ	Universal Broker configuration templates
UNVTRACE	UPDATE	Universal Broker application trace PDSE
UNVTRMDL	ALTER	Universal Broker application model trace data set
UNVLOG	UPDATE	Universal Broker log data set
UNVAGMDL	ALTER	Universal Automation Center Agent (UAG) model log data set

Universal Enterprise Controller

The Universal Enterprise Controller started task provides services for monitoring and administering Universal Agents distributed throughout the computer network. GUI clients connect to Universal Enterprise Controller to perform tasks and view component activity and statuses.

Universal Enterprise Controller User and Group Profiles

The security requirements of the Universal Enterprise Controller depend upon which services are being utilized. By default, all services are configured to be used. Some services can be disabled to reduce the amount of authority the Universal Enterprise Controller user profile requires. The following table lists the USS privileged services and how to disable the service so that security access to the service is not required.

Service	Description	Disabling
Mount file system	Universal Enterprise Controller dynamically mounts its USS file system. Mounting a file system requires APF authorization or superuser access. Universal Enterprise Controller is not APF authorized.	Statically mount the Universal Enterprise Controller file system.
Change directory ownership	Universal Enterprise Controller will initialize its file system if it detects initialization has not been completed. Initialization consists of changing the ownership of the file system root directory to the Universal Enterprise Controller user identifier.	Initialize the file system ownership manually.

How to configure the Universal Enterprise Controller started task user profile to meet security requirements depend on whether USS is running with UNIX-level security or z/OS UNIX security. The following sections describe how to configure the Universal Enterprise Controller user profile to perform the privileged services listed above for both USS UNIX-level security and z/OS UNIX security configurations.

UNIX-level Security

UNIX-level security refers to a USS security environment where the resource profile BPX.DAEMON is not defined to the FACILITY class.

The only method of permitting a user profile access to privileged services is to define the user profile with a UID value of 0 (superuser). The Universal Enterprise Controller user profile must be defined with UID 0 to perform any privileged service.

The following steps define the Universal Enterprise Controller user profile for a UNIX-level security environment:

Step 1	Add the Universal Enterprise Controller group profile **UECGRP** using the following RACF command: `ADDGROUP UECGRP OWNER(SYS1) OMVS(GID(5002))` Change the GID value 5002 to a value suitable for your local USS environment. The GID value must be unique among all group profiles.
Step 2	Add the Universal Enterprise Controller user profile **UECUSR** using the following RACF command: `ADDUSER UECUSR DFLTGRP(UECGRP) OWNER(SYS1) NOPASSWORD OMVS(UID(0))`

z/OS UNIX Security

z/OS UNIX security refers to a USS security environment where the resource profile BPX.DAEMON is defined to the FACILITY class.

The Universal Enterprise Controller user profile must be defined with a valid OMVS segment with a non-zero, unique UID value. The user profile security requirements are listed in the following table for each privileged service.

Service	Requirement
Mount file system	READ access to BPX.SUPERUSER resource profile in the FACILITY class.
Change directory ownership	READ access to BPX.SUPERUSER resource profile in the FACILITY class.

The security requirements can be lifted if the feature that utilizes the service is disabled as described above.

The following steps configure the Universal Enterprise Controller user profile for a z/OS UNIX security environment:

Step 1	Add the Universal Enterprise Controller group profile **UECGRP** using the following RACF command: `ADDGROUP UECGRP OWNER(SYS1) OMVS(GID(5002))` Change the GID value 5002 to a value suitable for your local USS environment. The GID value must be unique among all group profiles.
Step 2	Add the Universal Enterprise Controller user profile **UECUSR** using the following RACF command: `ADDUSER UECUSR DFLTGRP(UECGRP) OWNER(SYS1) NOPASSWORD OMVS(UID(5002))` Change the UID value 5002 to a value suitable for your local USS environment. The value must be unique among all user profiles.

Step 3	Permit the Universal Enterprise Controller user profile READ access to the resource profiles required for enabled services. The following RACF commands permit the user profile to the resources required for all privileged services:
	
	`PE BPX.SUPERUSER CLASS(FACILITY) ID(UECUSR) ACCESS(READ)`
	`SETR RACLIST(FACILITY) REFRESH`

Universal Enterprise Controller Data Access

The Universal Enterprise Controller user profile **UECUSR** requires the following access to the data sets allocated in the Universal Enterprise Controller started task, **UECTLR**:

Ddname	Access	Description
STEPLIB	READ	Program library
UNVCONF	READ	Product configuration data
UNVNLS	READ	Product national language support data
UNVDB	UPDATE	Universal Enterprise Controller HFS database
UNVMSGS	UPDATE	Universal Enterprise Controller message trace data
UNVPRSR	UPDATE	Universal Enterprise Controller parser trace data
UNVTRACE	UPDATE	Universal Enterprise Controller application trace data

Associate Started Tasks with User and Group Profiles

The started tasks must be associated with their user and group profiles defined above. IBM provides two different methods to accomplish this using RACF:

1. STARTED Class Profile
2. Started Procedures Table

Both methods are described below. Only one, not both, of the methods is required. They are provided as examples for your reference. Your local

security procedures and processes should be followed.

STARTED Class Profile

The following procedure describes how to associate a user and group profile with the started procedures using the RACF class STARTED method.

Step 1 Define a STARTED class profile for the Universal Enterprise Controller started procedure with the following TSO command:

```
RDEFINE STARTED UECTLR.* STDATA(USER(UECUSR) GROUP(UECGRP))
```

The started procedure member name used in the above command is **UECTLR**. If this has been change, the name in the REDEFINE command must also be changed to match.

Step 2 Define a STARTED class profile for the Universal Broker started procedure with the following TSO command:

```
RDEFINE STARTED UBROKER.* STDATA(USER(UBRUSR) GROUP(UBRGRP))
```

The started procedure member name used in the above command is **UBROKER**. If this has been change, the name in the REDEFINE command must also be changed to match.

Step 3 The STARTED class must be refreshed to recognize the new profile definitions. The following command assumes that the STARTED class is active and RACLIST'ed.

```
SETROPTS RACLIST(STARTED) REFRESH
```

Started Procedures Table

This section describes how to associate a user and group profile with the started procedures using the RACF started procedure table ICHRIN03 method.

The ICHRIN03 table resides in a system LPA library, such as **SYS1.LPALIB**. Changes to the table require a system IPL using the CLPA option for them to take effect. RACF loads the table at IPL.

Step 1 Add the following entry to the **ICHRIN03** table. The table is an assembly language program that is assembled and link edited into a system LPA library.

```
         DC     CL8'UECTLR  '   PROC NAME
         DC     CL8'UECUSR  '   UEC USER PROFILE
         DC     CL8'UECGRP  '   UEC GROUP PROFILE
         DC     XL1'00'
         DC     XL7'00'
         DC     CL8'UBROKER '   PROC NAME
         DC     CL8'UBRUSR  '   UBR USER PROFILE
         DC     CL8'UBRGRP  '   UBR GROUP PROFILE
         DC     XL1'00'
         DC     XL7'00'
```

Step 2 Increment the table count field by two. (The count field is the first 2 bytes of the table.)

Step 3 Assemble and link edit the ICHRIN03 table.
IBM provides a sample ICHRIN03 table and the JCL to assemble and link edit it in **SYS1.SAMPLIB(RACTABLE)**.

Step 4 IPL the system with the CLPA option.

zOS Configuration - Load Library

- Overview
- APF Authorization
- Program Control
- LNKLST

Overview

All Universal Agent programs are installed into the SUNVLOAD PDSE library. The SUNVLOAD library installation requirements are

- Must be APF authorized.
- Must be defined to RACF Program Control.
- Specific programs or SUNVLOAD must be added to the LNKLST only if Universal Automation Center Agent (UAG) is utilized.

The following sections describe the z/OS configuration steps to satisfy these requirements.

APF Authorization

The SUNVLOAD library can be APF authorized using one of the following methods:

- The SETPROG APF console command. The library remains APF authorized only until the next IPL. The SETPROG APF console command is documented in the IBM MVS System Commands manual.
- A PROGxx member of PARMLIB referenced by a SET PROG=xx console command or a PROG=xx statement in IEASYSxx PARMLIB member. The PROGxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual, and the SET console command is documented in the IBM MVS System Commands manual.
- Updating the IEAAPFxx PARMLIB member and perform an IPL. The IEAAPFxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual.

SETPROG APF Command

The SETPROG command temporarily adds a library to the APF list. Use one of the other methods to add the load library permanently to the APF list.

The SETPROG APF console command to add data set #HLQ.SUNVLOAD on volume #SMPVOL to the APF list is illustrated below. Change #HLQ to the appropriate high-level qualifier for your installation and #SMPVOL to the volume on which the library is allocated. SUNVLOAD APF authorization should be permanently established based on local site requirements.

```
SETPROG APF,ADD,DSNAME=#HLQ.SUNVLOAD,VOLUME=#SMPVOL
```

PROGxx PARMLIB Member

The data set can be added permanently to the APF list using the appropriate PROGxx PARMLIB member. The appropriate PROGxx member is referenced in the IEASYSxx PARMLIB member. The PROGxx member can be activated dynamically with the SET PROG=xx console command.

The APF statement of the PROGxx PARMLIB member is illustrated below. The APF statement adds data set #HLQ.SUNVLOAD on volume #SMPVOL to the APF list. Change #HLQ to the appropriate high-level qualifier for your installation and #SMPVOL to the volume on which the library is allocated.

```
APF ADD DSNAME(#HLQ.SUNVLOAD) VOLUME(#SMPVOL)
```

Program Control

Specific services of the z/OS UNIX environment require all programs loaded in the address space to be defined in the security product as controlled. RACF provides program and library control features. By defining a program as controlled, you are implying a certain level of trust, as opposed to a program being run that is not defined as controlled.

The following table identifies the programs in the SUNVLOAD library that must be defined as program controlled.

Program	Description
UCMSRV	Universal Command Server

UDMSRV	Universal Data Mover Server
UCTSRV	Universal Control Server
UECTLR	Universal Enterprise Controller
UAGSRV	Universal Automation Center Agent

The requirement for program control for each of the above programs is the same:

1. Use of the **_passwd** C function to authenticate user ID's
2. Use of the **setuid** C function to switch to the authenticated user ID's

Defining a program or library to RACF program control is accomplished by defining the library to the PROGRAM class.

The following RACF TSO command illustrates defining the library to the PROGRAM class. Change #HLQ to the appropriate high-level qualifier for your installation.

```
RDEF PROGRAM ** ADDMEM('#HLQ.SUNVLOAD'//NOPADCHK) UACC(READ)
```

After defining the library as RACF program controlled, the PROGRAM class must be refreshed. The following RACF TSO command illustrates:

```
SETROPTS WHEN(PROGRAM) REFRESH
```

When all programs executing in an address space are program controlled, the address space is considered a clean environment. All programs loaded dynamically into a clean environment must also be program controlled else the address space is terminated.

Some Universal Agent components may load some IBM modules at runtime to perform requested services. All IBM modules loaded dynamically are loaded from SYS1.LINKLIB. Each member or all members of SYS1.LINKLIB must be set up as program controlled.

The following modules are loaded from SYS1.LINKLIB:

- IGGCSI00 (Catalog Search Interface)
- IEBCOPY (PDS/E copy utility)

SYS1.LINKLIB normally is set up so that all members are program controlled. Verify how your installation has defined SYS1.LINKLIB in regards to the program control facility. If necessary, define the individual members listed above, or all members of SYS1.LINKLIB, as program controlled.

LNKLST

The following Universal Automation Center Agent programs and their aliases must be added to the LNKLST for them to be available to all batch jobs and TSO users:

- UAGRERUN (and its alias OPSRERUN) provides batch job rerun capability. The program must be available to all jobs managed by Automation Center.
- UAGCMDZ (and its alias OPSCMDZ) provides a TSO command interface to Universal Controller. The TSO command interface is optional.

The programs are located in the SUNVLOAD library.

If the Universal Automation Center Agent is not used in your installation, this z/OS configuration step can be skipped.

The UAGRERUN and UAGCMDZ programs and their aliases should be added to the LNKLST using one of the following methods:

- Copy the programs and their aliases to a PDSE library that is already in the LNKLST.
- A PROGxx member of PARMLIB referenced by a SET PROG=xx console command or a PROG=xx statement in IEASYSxx PARMLIB member. The PROGxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual, and the SET console command is documented in the IBM MVS System Commands manual.
- Updating the LNKLSTxx PARMLIB member and perform an IPL. The LNKLSTxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual.

Copying Programs and Aliases

The UAGRERUN and UAGCMDZ and their aliases may be copied to a PDSE data set that is already in the LNKLST.

⚠ The SUNVLOAD library is a PDSE and the programs are program objects using program management features that are not supported in a PDS. For this reason, the programs must be copied to a PDSE in the LNKLST and not to a PDS.

The JCL below executes IEBCOPY to copy the programs and their aliases to a LNKLST library. Change #HLQ to the appropriate high-level qualifier for your installation and change the data set allocated to ddname OUT to the target PDSE in the LNKLST. The job must end with return code 0.

```
//jobname  JOB CLASS=A,MSGCLASS=H,NOTIFY=&SYSUID,COND=(0,NE)
//STEP1    EXEC PGM=IEBCOPY,PARM='RC4NOREP'
//IN       DD  DISP=SHR,DSN=#HLQ.SUNVLOAD
//OUT      DD  DISP=SHR,DSN=lnklst.pdse
//SYSUT3   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSUT4   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))
//SYSPRINT DD  SYSOUT=*
//SYSIN    DD  *
  COPYGRP I=((IN,R)),O=OUT
  S M=UAGRERUN
  S M=UAGCMDZ
/*
```

SETPROG LNKLST Command

The SETPROG LNKLST command dynamically modifies the LNKLST. The LNKLST changes remain in effect until the next IPL. Use one of the other methods to add the load library permanently to the LNKLST.

LNKLST's are managed as sets. A LNKLST set has a name and consists of an order number of data sets or libraries. To dynamically change the current LNKLST, a LNKLST set must be created, updated, and activated. A newly activated LNKLST set will be active for new address spaces, but existing address spaces are not updated.

The SETPROG LNKLST console command is used to create, update, and activate a LNKLST set. The SETPROG commands to add a data set to the LNKLST are illustrated below. The commands define a LNKLST set name *lnklstname* based on the currently active LNKLST set. It then adds the SUNVLOAD library at the top of the data set list. Lastly, it activates the new LNKLST set. Change #HLQ to the appropriate high-level qualifier for your installation and the LNKLST set name to meet your local installation requirements. Ensure each command completes successfully before executing the next.

```
SETPROG LNKLST,DEFINE,NAME=lnklstname,COPYFROM=CURRENT
SETPROG LNKLST,ADD,NAME=lnklstname,DSNAME=#HLQ.SUNVLOAD,ATTOP
SETPROG LNKLST,ACTIVATE,NAME=lnklstname
```

PROGxx PARMLIB Member

The SUNVLOAD library can be added permanently to the LNKLST using the appropriate PROGxx PARMLIB member. The appropriate PROGxx member is referenced in the IEASYSxx PARMLIB member.

The LNKLST statement of the PROGxx PARMLIB member is illustrated below. The LNKLST statement adds data set #HLQ.SUNVLOAD to the LNKLST set *lnklstname*. Change #HLQ to the appropriate high-level qualifier for your installation and *lnklstname* to the appropriate LNKLST set name for your installation.

```
LNKLST ADD NAME(lnklstname) DSNAME(#HLQ.SUNVLOAD)
```

zOS Configuration - SMF Exits

- Overview
- SMF Exit Introduction
- Universal Command Server
  - Configuring SMF
  - Installing SMF Exit Routines
- Universal Automation Center Agent
  - Configuring SMF
  - Installing SMF Exit Routines

Overview

SMF exits are utilized by the following Universal Agent components to provide their z/OS services:

- Universal Command Server uses one SMF exit to provide support for starting Started Tasks (STC) using Universal Command Manager. If Universal Command STC support is not required, the SMF exit does not need to be installed.
- Universal Automation Center Agent uses a number of SMF exits to provide a range of services available in Universal Controller.

The following sections describe how to install the SMF exits for each component. The SMF exits are only required if the component or component services are required.

SMF Exit Introduction

MVS System Management Facilities (SMF) collects and records system events in the form of SMF records. An SMF record is identified with a numeric record type. In addition to recording system events, SMF provides exit points in the control program from which system or application provided exit routines or programs can be called at the time work is being processed. The SMF exit routines can take appropriate actions based on the point at which the exit is called and the information provided in the SMF record. The SMF exits allow for systems and applications to monitor and augment the processing of MVS workload. For additional information on SMF, refer to the IBM manuals *MVS System Management Facilities (SMF)*, *MVS Installation Exits*, and *MVS Initialization and Tuning Reference*.

SMF configuration is performed with PARMLIB member SMFPRMxx. SMF configuration relevant for product installation is the SMF record types being recorded and the SMF exits defined. The SMFPRMxx parameters SYS and SUBSYS define these SMF recording options. The SYS parameter specifies system wide recording options for all subsystems (JES2, TSO, STC, etc.). The SUBSYS parameter specifies subsystem specific recording options. The SUBSYS parameter options override the equivalent options specified on the SYS parameter. Any SUBSYS parameter option not specified will default to the SYS parameter option.

The following example illustrates how SMF SYS and SUBSYS parameters work together.

```
SYS( EXITS(IEFU83,IEFU84) )
SUBSYS( STC, EXITS(IEFU83) )
```

The SYS parameter EXITS option defines the two SMF exits IEFU83 and IEFU84. The SUBSYS parameter for the STC subsystem also specifies the EXITS option, overriding the SYS EXITS option. The SUBSYS parameter EXIT option defines one SMF exit IEFU83 for the STC subsystem. Consequentially, only the IEFU83 exit is called for STC subsystem workload. The IEFU84 exit is not called for STC subsystem workload. For all workload other then the STC subsystem, the SMF exits IEFU83 and IEFU84 are called.

Once SMFPRMxx has defined the appropriate SMF record types to collect and the SMF exits to activate, the product SMF exit routines are installed. IBM provides the dynamic exit facility to add and remove exit routines dynamically and at IPL. IBM has defined all the SMF exits to the dynamic exit facility. The exit routines are added or removed from SMF exits using the PROGxx PARMLIB members. The PROGxx EXIT statements used to add product exit routines are listed in the appropriate sections below.

The exit names used in the dynamic exit facility are formatted as `SYSssn.exitname`, where `ssn` is the subsystem name, or blank if it is a system wide exit, and `exitname` is the name of the exit, such as IEFU83. The following table provides some example dynamic exit facility exit names.

Name	Description
`SYS.IEFU83`	The default SMF exit IEFU83 for all system workload types.
`SYSSTC.IEFU83`	The SMF exit IEFU83 for STC subsystem workload.
`SYSJES2.IEFU83`	The SMF exit IEFU83 for JES2 subsystem workload.

> **Note**
> When using SMF Exit items, please check against conflicting existing SMF definitions, such as `SYSJES2.IEFXXX`.

Universal Command Server

The Universal Broker STC establishes the environment to support STC execution by Universal Command Server. Part of the environment consists of adding SMF exit routine UNVACTRT to SMF exit point SYSSTC.IEFACTRT.

The Broker's UCMD_STC_SUPPORT option specifies whether or not the environment for STC support should be established:

- **yes** (the default) specifies that the environment should be established.
- **no** specifies that the environment should not be established.

Configuring SMF

Configuring SMF consists of defining SMF exit point SYSSTC.IEFACTRT for the STC subsystem.

The exit point is defined in the SMFPRMxx PARMLIB member with the SUBSYS STC EXITS parameter either implicitly, by excluding the EXITS parameter, or explicitly.

The following SUBSYS statement illustrates how to define SYSSTC.IEFACTRT:

```
SUBSYS(STC,EXITS(IEFACTRT,...))
```

Once the SMFPRMxx PARMLIB member has been modified, the SMFPRMxx member can be activated dynamically by restarting SMF with the following command:

```
SET SMF=xx
```

The following console command displays the active SMF options:

```
D SMF,O
```

Installing SMF Exit Routines

The SMF exit routine UNVACTRT is provided as part of the Universal Agent package. UNVACTRT must be added to SMF exit point SYSSTC.IEFACTRT. The exit routine can be added automatically by the Broker STC when it starts or statically in a PARMLIB member.

Automatic Installation

The Broker dynamically adds SMF exit routine UNVACTRT to the SYSSTC.IEFACTRT exit point if the SMF_EXIT_LOAD_LIBRARY configuration option is defined in the Universal Broker configuration file. The option indicates that the Broker dynamically add UNVACTRT and specifies the load library from which UNVACTRT is loaded.

Static Installation

If the Universal Broker configuration option SMF_EXIT_LOAD_LIBRARY is not specified in the Universal Broker configuration file, Universal Broker assumes that exit routine UNVACTRT already is added and will not attempt to add it when it starts.

Exit routine UNVACTRT is permanently added to SYSSTC.IEFACTRT using the appropriate PROGxx PARMLIB member. The EXIT statement of the PROGxx PARMLIB member is illustrated below. The EXIT statement adds exit routine UNVACTRT from the #HLQ.SUNVLOAD load library to the SMF exit point SYSSTC.IEFACTRT. Change #HLQ to the appropriate high-level qualifier for your installation. The exit is no longer given control if it encounters an ABEND.

```
EXIT ADD EXITNAME(SYSSTC.IEFACTRT) MODNAME(UNVACTRT) DSNAME(#HLQ.SUNVLOAD) ABENDNUM(1)
```

The UNVACTRT exit routine is added when the system is IPL'ed and the PROGxx member is processed. The SMF exit can be added dynamically

without an IPL by activating PROGxx member with the SET console command. The following SET command activates PROGxx member:

```
SET PROG=xx
```

The PROGxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual. The SET console command is documented in the IBM MVS System Commands manual.

The following console command displays the exit routines installed for exit point SYSSTC.IEFACTRT:

```
D PROG,EXIT,EN=SYSSTC.IEFACTRT,DIAG
```

Universal Automation Center Agent

The Universal Automation Center Agent (UAG) establishes the environment to support Universal Controller services on z/OS. Part of the environment consists of establishing the following SMF exit routines:

Exit Point	Exit Routine
SYS.IEFUJI	UAGUJI
SYS.IEFUSI	UAGUSI
SYS.IEFUJV	UAGUJV
SYS.IEFU83	UAGU83
SYS.IEFU84	UAGU84
SYSSTC.IEFU83	UAGU83

Note
When using SMF Exit items, please check against conflicting existing SMF definitions, such as `SYSJES2.IEFXXX`.

Configuring SMF

Configuring SMF consists of specifying required SMF record types to collect and the required exit points to define.

The SMF configuration must meet the following requirements:

- SMF record types 14, 15, 16, 17, 30, and 70 must be collected.
- System exit points IEFUJI, IEFUSI, IEFUJV, IEFU83, and IEFU84 must be defined.
- Assuming the Universal Command SMF exit is installed or there is already a SUBSYS EXITS option for STC, the subsystem STC exit point IEFU83 must be defined.

The record types and exit points are defined in the SMFPRMxx PARMLIB member. The following SMFPRMxx statements illustrate how to define the record types to collect and exit points.

The SYS statement uses the NOTYPE parameter to exclude specific record types. This is one of many ways the SYS statement can be specified to meet the SMF record type requirements.

```
SYS(NOTYPE(18:19,62:69,99),EXITS(IEFU83,IEFU84,IEFUJV,IEFUSI,IEFUJI,...))
SUBSYS(STC,EXITS(IEFU83,...))
```

Once the SMFPRMxx PARMLIB member has been modified, the SMFPRMxx member can be activated dynamically by restarting SMF with the following command:

```
SET SMF=xx
```

The following console command displays the active SMF options:

```
D SMF,O
```

Installing SMF Exit Routines

The SMF exit routines are installed using the appropriate PROGxx PARMLIB member. The EXIT statements for the PROGxx PARMLIB member are illustrated below. The EXIT statements add all the exit routines from the #HLQ.SUNVLOAD load library to the appropriate exit points. Change #HLQ to the appropriate high-level qualifier for your installation.

```
EXIT ADD EXITNAME(SYS.IEFUJI) MODNAME(UAGUJI) DSNAME(#HLQ.SUNVLOAD)
EXIT ADD EXITNAME(SYS.IEFUSI) MODNAME(UAGUSI) DSNAME(#HLQ.SUNVLOAD)
EXIT ADD EXITNAME(SYS.IEFUJV) MODNAME(UAGUJV) DSNAME(#HLQ.SUNVLOAD)
EXIT ADD EXITNAME(SYS.IEFU83) MODNAME(UAGU83) DSNAME(#HLQ.SUNVLOAD)
EXIT ADD EXITNAME(SYS.IEFU84) MODNAME(UAGU84) DSNAME(#HLQ.SUNVLOAD)
EXIT ADD EXITNAME(SYSSTC.IEFU83) MODNAME(UAGU83) DSNAME(#HLQ.SUNVLOAD)
```

The exit routines are added when the system is IPL'ed and the PROGxx member is processed. The SMF exit routines can be added dynamically without an IPL by activating PROGxx member with the SET console command. The following SET command activates PROGxx member.

```
SET PROG=xx
```

The PROGxx PARMLIB member is documented in the IBM MVS Initialization and Tuning Reference manual. The SET console command is documented in the IBM MVS System Commands manual.

The following console command displays the exit routines installed for exit point SYS.IEFUJI:

```
D PROG,EXIT,EN=SYS.IEFUJI,DIAG
```

zOS Configuration - JES SYSOUT Processing

- Overview
- Configuration
  - Multiple SYSOUT Applications
  - UAG SYSOUT Management
  - Default Configuration

Note
The information in this section applies to UAG release 5.1.0.12 and above.

Overview

This page describes how to configure Universal Automation Center Agent (UAG) to reliably process JES SYSOUT data sets. If UAG is not used, this information can be skipped.

The batch jobs that UAG submits create JES SYSOUT data sets. UAG processes the following SYSOUT data sets produced by the jobs:

- UAGRERUN report produced by the first step of every batch job UAG submits. The SYSOUT data set is written by step name OPSSTP00 to ddname SYSPRINT.
- JES system data sets produced by every batch job. The JES system data sets include the JESMSGLG, JESJCL and JESYSMSG ddnames.

UAG processes the SYSOUT data sets for the following purposes:

1. Retrieving the UAGRERUN report for storage in the Universal Controller.
2. Analyzing JCL errors that occur during JCL conversion or at run-time.
3. Optionally, retrieving the JES system data sets for storage in the Universal Controller.

JES SYSOUT data sets cannot be processed by more then one application at time. Applications, such as spool archivers and JES offloaders, must process SYSOUT data sets one at a time. If multiple applications process SYSOUT data sets simultaneously, some SYSOUT data sets potentially will not be processed.

For UAG to reliably process SYSOUT data sets, a JES held class must be dedicated exclusively to UAG. A JES class is defined as held with a JES JOBCLASS statement with an OUTDISP parameter value of HOLD,HOLD.

Configuration

UAG provides a flexible SYSOUT processing configuration to accommodate a variety of environments. The best configuration depends on your existing SYSOUT management practices and processes.

There are three UAG configuration options for SYSOUT processing:

- JES_SYSOUT_CLASS specifies the JES held class dedicated to UAG SYSOUT processing.
- JES_SYSOUT_DISP specifies the disposition of the SYSOUT data sets once UAG has completed SYSOUT processing.
- JES_SYSOUT_RETENTION specifies the number of hours that job SYSOUT files are retained in the UAG cache directory.

The following sections discuss some possible UAG SYSOUT configurations. The SYSOUT options allow UAG SYSOUT processing to be customized to meet almost any SYSOUT management requirement. The following configurations are the most common.

Multiple SYSOUT Applications

This configuration addresses environments that have existing SYSOUT applications that process job SYSOUT data sets. These applications are often called - for example - spool archivers, output management systems, or job log archivers. They typically read all SYSOUT data sets from one or more JES held classes and archive the SYSOUT in a database. Once they have finished processing the SYSOUT in the JES spool, the SYSOUT is typically deleted from the spool.

UAG must be properly configured to work with other SYSOUT processing applications. Since SYSOUT data sets cannot be processed simultaneously by multiple SYSOUT applications, each application must process the SYSOUT data sets one at a time.

The first step is to select a JES held class to dedicate to UAG. The class cannot be shared with any other SYSOUT application. Specify the class with the UAG JES_SYSOUT_CLASS configuration option. By specifying a JES_SYSOUT_CLASS value, UAG will modify the JOB statement of the jobs it submits to JES. The JOB statement MSGCLASS parameter is changed to the JES_SYSOUT_CLASS value. If no MSGCLASS parameter is present on the JOB statement, it will be added. The JOB statement MSGCLASS parameter specifies the JES class where the job's JES system data sets are spooled.

Once UAG has completed processing the SYSOUT data sets, it performs final disposition on them as specified by the UAG JES_SYSOUT_DISP

configuration option. The SYSOUT data sets must be moved to the JES class or classes used by the next SYSOUT application to process them.

There are two ways to specify the JES class to which to move the SYSOUT data sets:

- If the original JOB statement MSGCLASS value correctly specified the class, then a JES_SYSOUT_DISP value of **KEEP,*** will keep the SYSOUT data sets and move them to the original MSGCLASS class. This configuration requires no JCL changes nor changes to the other SYSOUT application.
- The class can be specified directly. For example, a JES_SYSOUT_DISP value of **KEEP,H** will keep the SYSOUT data sets and move them to held class H.

UAG SYSOUT Management

UAG z/OS task definitions may be defined to retrieve the job log (that is the job's JES system data sets) once the job completes processing. The job logs are transmitted to the Opwwise Controller and stored in the database. The job logs can be viewed from the Universal Controller web interface. In this configuration, once UAG has completed retrieving the job log, the SYSOUT data sets can be deleted from the JES spool. It is assumed that there are no other SYSOUT applications processing the SYSOUT data sets other then UAG.

The JES_SYSOUT_CLASS option is not required in this configuration. By default, UAG will process a job's SYSOUT data sets from any JES class in which it finds them.

Once UAG has completed processing the SYSOUT data sets, it performs final disposition on them as specified by the UAG JES_SYSOUT_DISP option. Since UAG is the one and only SYSOUT application, the SYSOUT data sets can be deleted. A JES_SYSOUT_DISP value of **DELETE** will remove the SYSOUT data sets from the JES spool.

Default Configuration

The UAG default configuration is backwardly compatible with previous UAG releases. By default UAG will process SYSOUT data sets in the JES spool regardless of their disposition (held or not) or class. Consequentially, if the SYSOUT data sets are processed by another application, the results are unreliable. Once UAG completes processing of the SYSOUT data sets, it leaves them in the JES spool without changing their disposition or class.

Universal Agent Database Configuration

- z/OS UNIX File System Introduction
- zFS Configuration
- HFS Configuration
- Mounting and Unmounting the Databases
  - Dynamic Mounts
  - Manual Mounts
  - TSO Commands
  - Console Commands
  - BPXPRMxx
  - Data Set Initialization
- Memory Management

z/OS UNIX File System Introduction

The z/OS implementation of Universal Agent databases utilizes the z/OS UNIX file system. HFS or zFS data sets are used by Universal Broker and Universal Enterprise Controller started tasks.

The hierarchical file system is a file system used by z/OS UNIX System Services (USS). It is a POSIX conforming hierarchical file system stored in one or more HFS or zFS data sets bound together into one hierarchical directory structure. A single HFS or zFS data set consists of directory tree and files. Refer to the IBM UNIX System Services Planning manual for a complete discussion of the z/OS UNIX file system and its administration.

An HFS or zFS data set must be mounted before a program can access any file or directory within it. A mount operation binds the root directory of the HFS or zFS data set to an existing directory in the hierarchical file system referred to as the mount point. After the mount operation completes, the HFS or zFS data set's directory structure becomes part of the file system hierarchy starting at the mount point. An HFS or zFS data set can only be mounted one at a time.

The mount operation makes the files and directories within the HFS or zFS data set accessible to all users. User access is controlled with directory and file permissions contained within the HFS or zFS data set. Initially, an HFS or zFS data set's root directory is owned by the user that allocated the data set and the directory permissions are set so that only that user has read, write, and execute permissions (permission mode 700). No other users have access.

zFS Configuration

zFS data sets are created by the installation JCL. The zFS data sets are used by Universal Broker and Universal Enterprise Controller.

A zFS data set is referred to as a zFS aggregate. There are two types of aggregates, depending on whether it contains one or more read / write zFS file systems:

1. A zFS aggregate that contains only one file system is a compatibility mode aggregate.
2. A zFS aggregate that contains more than one file system is a multi-file system aggregate.

Universal Agent uses compatibility mode aggregates only.

When a zFS data set is mounted, the zFS address space obtains an exclusive enqueue on the data set, preventing it from being allocated by another address space. For this reason, when using zFS data sets in the Universal Broker or Universal Enterprise Controller started tasks, the data sets cannot be allocated to the **UNVDB** or **UNVSPOOL** ddnames.

The Universal Broker zFS data set names must be specified with the UNIX_DB_DATA_SET and UNIX_SPOOL_DATA_SET Universal Broker configuration options. The Universal Enterprise Controller zFS data set name must be specified with the UNIX_DB_DATA_SET Universal Enterprise Controller configuration option. The configuration options can be used to specify HFS data sets as well if this method is preferred over specifying them as a ddname allocation.

HFS Configuration

An alternative to using zFS data sets is to use HFS data sets. HFS data sets are created optionally as part of the installation steps with JCL member **UNVINHF**.

Universal Broker allocates the HFS data set used for the databases to ddname **UNVDB** and the HFS data set used for the spool to ddname **UNVSPOOL**. Universal Enterprise Controller allocates the HFS data set used for the databases to ddname **UNVDB**. No further HFS configuration is required beyond allocating the appropriate HFS data sets to the ddnames.

When using HFS data sets instead of zFS data sets, uncomment the **UNVDB** and **UNVSPOOL** ddname statements in the Universal Broker started task procedure and uncomment the **UNVDB** ddname statement in the Universal Enterprise Controller started task procedure.

The **DBHLQ** parameter in the Universal Broker started task procedure must also be uncommented.

Alternatively, the HFS data set names can be specified using the UNIX_DB_DATA_SET and UNIX_SPOOL_DATA_SET configuration options. When these configuration options are used, the ddnames **UNVDB** and **UNVSPOOL** are not used. The ddname statements and the **DBHLQ** parameter can be removed from the started task procedures.

Mounting and Unmounting the Databases

When the Universal Broker and Universal Enterprise Controller started tasks are started, they checks if their HFS or zFS data sets have been mounted. If they are mounted, the started tasks will attempt to use them. If they are not mounted, the started tasks will mount the data sets dynamically.

Dynamic Mounts

The started tasks will mount the HFS or zFS data sets if they are not mounted. The data sets are mounted at mount points defined in the directory specified by the Universal Broker MOUNT_POINT configuration option, which defaults to the **`/tmp`** directory. The mount points are subdirectories named after the data set names. For example, if the HFS or zFS data set name is **`UNV.UNVDB`**, the mount point is **`/tmp/UNV.UNVDB`**.

When the started tasks mount a zFS data set, the mount parameter **AGGRGROW** is used to specify that the zFS data set should automatically utilize secondary extents to expand if it runs out of allocated space.

The HFS or zFS data sets are not unmounted when the started tasks are stopped. It is not known whether or not other users are using the mounted data sets.

Manual Mounts

The started tasks will use the existing mounts of the HFS or zFS data sets. Dynamic mounts provide the easiest administration, but you may want to manually mount the data sets to take advantage of several available mount options. For example, the FSFULL PARM value can be used to issue operator messages when a file system reaches a specified percent full.

When mounting zFS data sets, the mount parameter **AGGRGROW** should be used to specify that the zFS data set should automatically utilize secondary extents to expand if it runs out of allocated space.

When the HFS or zFS data sets are manually mounted, the mount point can be any z/OS UNIX directory. The name of the directory does not matter. The started tasks will locate the mount point regardless of location or name.

HFS or zFS data sets can be mounted using the TSO MOUNT command or with PARMLIB member **BPXPRMxx** at IPL. The TSO MOUNT command mounts it for the current IPL only while the **BPXPRMxx** member will mount the data set for each IPL.

HFS or zFS data sets can be unmounted using the TSO UNMOUNT command or with the MODIFY BPXOINIT console command.

TSO Commands

The TSO commands to mount and unmount HFS data set **UNV.UNVDB** at mount point **/opt/unvdb** are illustrated below:

HFS Mount Command

```
MOUNT FILESYSTEM('UNV.UNVDB') MOUNTPOINT('/opt/unvdb') TYPE(HFS)
```

zFS Mount Command

```
MOUNT FILESYSTEM('UNV.UNVDB') MOUNTPOINT('/opt/unvdb') TYPE(ZFS) PARM(AGGRGROW)
```

HFS and zFS Unmount Command

```
UNMOUNT FILESYSTEM('UNV.UNVDB')
```

The user ID that issues the mount or unmount commands must have an OMVS UID of 0 or READ access to the **BPX.SUPERUSER** profile in the FACILITY class.

Console Commands

The console commands to unmount HFS or zFS data **UNV.UNVDB** is illustrated below in addition to the console command to list currently

mounted HFS data sets.

Unmount Command

```
F BPXOINIT,FILESYS=UNMOUNT,FILESYSTEM=UNV.UNVDB
```

> **Note**
> A console reply message will ask for confirmation.

Display Command

```
D OMVS,FILE
```

BPXPRMxx

The BPXPRMxx statement to mount HFS or zFS data set **UNV.UNVDB** at mount point **/opt/unvdb** is illustrated below:

HFS Mount

```
MOUNT FILESYSTEM('UNV.UNVDB') TYPE(HFS) MODE(RDWR) MOUNTPOINT('/opt/unvdb')
```

zFS Mount

```
MOUNT FILESYSTEM('UNV.UNVDB') TYPE(ZFS) MODE(RDWR) MOUNTPOINT('/opt/unvdb') PARM('AGGRGROW')
```

Both of the HFS or zFS data sets must be mounted with mode read/write, which is the default.

Data Set Initialization

When the started tasks start, they find the mount point for their HFS or zFS data sets. Regardless of whether the HFS or zFS data sets were dynamically mounted or statically mounted, the started tasks check for an initialization flag file named **.inited** in the root directory of the mounted data set.

If the file is not found, which is the case when they are first mounted, the started tasks change the owner of the root directory to the user ID with which they are executing and change the permission mode to the MOUNT_POINT_MODE configuration option value, which defaults to 750.

If you want to customize either the owner or permission of the directories, manually create the **.inited** file in the root directory of the HFS or zFS data set to prevent the started tasks from performing the initialization when they start. The USS command **touch .inited** can be used to create an empty file.

Memory Management

Berkeley DB uses a temporary cache in memory to manage its databases. If this cache becomes sufficiently large, it must be written to disk.

Berkeley DB has a default location for storing temporary cache files, but if UEC cannot access that location, or there is no space to write these files in the default location, the following error can occur in UEC, and UEC shuts down:

UNV4301D Database error: 'temporary: write failed for page XXXXX'

To work around this issue, the following steps write the temporary cache files to the UEC database directory:

Step 1	Mount the **UECDB** HFS or zFS data set.
Step 2	Inside the mount point, create a text file named **DB_CONFIG**.

Step 3	Inside the **DB_CONFIG** file, add the following string: **`set_tmp_dir *dbpath*`** Where **dbpath** is the path to the location in which the database files reside.
Step 4	Start / restart UEC.

zOS Installation - Customization

- Overview
- Universal Broker Customization
  - Universal Broker Configuration
  - Universal Broker JCL Procedure
- Universal Automation Center Agent Customization
  - Universal Automation Center Agent Configuration
- Universal Certificate Customization
  - Universal Certificate JCL Procedure
- Universal Command Manager Customization
  - Universal Command Manager Configuration
  - Universal Command Manager JCL Procedure
- Universal Command Server Customization
  - Universal Command Server Configuration
- Universal Connector Customization
  - Universal Connector Configuration
  - Universal Connector SAP RFC Configuration
  - Universal Connector JCL Procedure
- Universal Control Manager Customization
  - Universal Control Manager Configuration
  - Universal Control Manager JCL Procedure
- Universal Control Server Customization
  - Universal Control Server Configuration
- Universal Data Mover Manager Customization
  - Universal Data Mover Manager Configuration
  - Universal Data Mover Manager JCL Procedure
- Universal Data Mover Server Customization
  - Universal Data Mover Server Configuration
- Universal Enterprise Controller Customization
  - Universal Enterprise Controller Configuration
  - Universal Enterprise Controller JCL Procedure
- Universal Event Monitor Manager Customization
  - Universal Event Monitor Manager Configuration
  - Universal Event Monitor Manager JCL Procedure
- Universal Query Customization
  - Universal Query Configuration

Overview

This page provides the following information for the customization of Universal Agent components:

- Configuration
- JCL procedure

(For information on applying product licenses to installed Universal Agent for z/OS components, see z/OS Installation - Licensing.)

Universal Broker Customization

Universal Broker Configuration

Universal Broker for z/OS uses a configuration file for its customizations. The configuration file is member **UBRCFG00** in the **UNVCONF** library allocated to the **UNVCONF** ddname in the started procedure's JCL.

See the Universal Broker 6.2.x Reference Guide for details on configuring Universal Broker.

Universal Broker JCL Procedure

A JCL procedure is provided in member **UBROKER** in library **SUNVSAMP**.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information.

See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

Universal Automation Center Agent Customization

Universal Automation Center Agent Configuration

There are two files used in defining the Universal Automation Center Agent (UAG) configuration:

- UAG component definition file
- UAG configuration file

The execution of UAG is managed by Universal Broker. The component definition file defines UAG attributes to the Broker. The Broker uses the component definition file to manage the execution of UAG. The component definition file is a member of the **UNVCOMP** library. The library is allocated to the **UNVCOMP** ddname of the Universal Broker started task. The member name is **UAGCMP00**.

The UAG configuration file defines system-wide customizations for UAG features and resources. The configuration file is a member of the **UNVCONF** library. The library is allocated to the **UNVCONF** ddname of the Universal Broker started task. The Broker passes the data set name to UAG when it is started by the Broker. The UAG component definition file defines which member of the **UNVCONF** library to use as the configuration file. The default member name is **UAGCFG00**.

See the Universal Automation Center Agent 6.2.x Reference Guide for details on configuring UAG.

Universal Certificate Customization

Universal Certificate JCL Procedure

A JCL procedure is provided in member **UCRPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the **JCLLIB** JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC UCRPRC
```

Universal Command Manager Customization

Universal Command Manager Configuration

Universal Command Manager for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UCMCFG00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Manager.

Universal Command Manager JCL Procedure

A JCL procedure is provided in member **UCMDPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the **JCLLIB** JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC UCMDPRC
```

Universal Command Server Customization

Universal Command Server Configuration

There are two files used in defining the Universal Command Server configuration:

- Server's component definition file
- Server's configuration file

The execution of the Server is managed by Universal Broker. The component definition file defines the Server attributes to the Broker. The Broker uses the component definition file to manage the execution of the server. The component definition file is a member of the **UNVCOMP** library. The library is allocated to the **UNVCOMP** ddname of the Universal Broker started task. The member name is **UCSCMP00**.

The Universal Command Server configuration file defines system-wide customizations for Server features and resources. The configuration file is a member of the **UNVCONF** library. The library is allocated to the **UNVCONF** ddname of the Universal Broker started task. The Broker passes the data set name to the Server when it is started by the Broker. The Server component definition file defines which member of the **UNVCONF** library to use as the configuration file. The default member name is **UCSCFG00**.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Server.

Universal Connector Customization

Universal Connector Configuration

Universal Connector for z/OS uses a configuration file for system-wide customizations. The configuration file is member **USPCFG00** in the **UNVCONF** library allocated to ddname **UNVRFC** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Connector 6.2.x Reference Guide for details on configuring Universal Connector.

Universal Connector SAP RFC Configuration

Universal Connector utilizes SAP's RFC interface. The RFC interface must be configured to meet your local SAP environment. The RFC configuration file is member **USPRFC00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

Universal Connector JCL Procedure

A JCL procedure is provided in member **USPPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the **JCLLIB** JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC USPPRC
```

Universal Control Manager Customization

Universal Control Manager Configuration

Universal Control Manager for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UCTCFG00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Manager.

Universal Control Manager JCL Procedure

A JCL procedure is provided in member **UCTLPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the **JCLLIB** JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC UCTLPRC
```

Universal Control Server Customization

Universal Control Server Configuration

There are two files used in defining the Universal Control Server configuration:

- Server's component definition file.
- Server's configuration file.

The execution of the Server is managed by Universal Broker. The component definition file defines the Server attributes to the Universal Broker, which uses the file to manage the execution of the Server.

The component definition file is a member of the **UNVCOMP** library. The library is allocated to the **UNVCOMP** ddname of the Universal Broker started task. The member name is **UTSCMP00**.

The Universal Control Server configuration file defines system-wide customizations for Server features and resources. The configuration file is a member of the **UNVCONF** library. The library is allocated to the **UNVCONF** ddname of the Universal Broker started task.

The Universal Broker passes the data set name to the Server when it starts the Server. The Server component definition file defines which member of the **UNVCONF** library to use as the configuration file. The default member name is **UTSCFG00**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Server.

Universal Data Mover Manager Customization

Universal Data Mover Manager Configuration

Universal Data Mover Manager for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UDMCFG00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Data Mover.

Universal Data Mover Manager JCL Procedure

A JCL procedure is provided in member **UDMPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the **JCLLIB** JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC UDMPRC
```

Universal Data Mover Server Customization

Universal Data Mover Server Configuration

There are two files used in defining the Universal Data Mover Server configuration:

- Server's component definition file
- Server's configuration file

The execution of the Server is managed by Universal Broker. The component definition file defines the Server attributes to the Universal Broker, which uses the file to manage the execution of the server.

The component definition file is a member of the **UNVCOMP** library. The library is allocated to the **UNVCOMP** ddname of the Universal Broker started task. The member name is **UDSCMP00**.

The Universal Data Mover Server configuration file defines system-wide customizations for Server features and resources. The configuration file is a member of the **UNVCONF** library. The library is allocated to the **UNVCONF** ddname of the Universal Broker started task.

The Universal Broker passes the data set name to the Server when it starts the Server. The Server component definition file defines which member of the **UNVCONF** library to use as the configuration file. The default member name is **UDSCFG00**.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Data Mover Server.

Universal Enterprise Controller Customization

Universal Enterprise Controller Configuration

Universal Enterprise Controller for z/OS uses a configuration file for its customizations.The configuration file is member **UECCFG00** of the **UNVCONF** library allocated to ddname **UNVCONF** in the started procedure's JCL.

See the Universal Enterprise Controller 6.2.x Reference Guide for details on configuring Universal Data Mover.

Universal Enterprise Controller JCL Procedure

A JCL procedure is provided in member **UECTLR** in library **SUNVSAMP**.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

Universal Event Monitor Manager Customization

Universal Event Monitor Manager Configuration

Universal Event Monitor Manager for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UEMCFG00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Event Monitor 6.2.x Reference Guide for details on configuring Universal Event Monitor Manager.

Universal Event Monitor Manager JCL Procedure

A JCL procedure is provided in member **UEMPRC** in library **SUNVSAMP**. Using the procedure simplifies future product upgrades and reduces the amount of JCL statements required in the job JCL.

Edit the JCL procedure to meet local JCL installation requirements. The TZ environment variable should be modified to meet your local time zone information. See z/OS Installation - Time Zone Environment Variable for more information the TZ environment variable.

The JCL procedure can be made available for use by either:

- Copying it to a JES procedure library, such as **SYS1.PROCLIB**
- Using the JCLLIB JCL statement in the job JCL to include the **SUNVSAMP** library in the procedure library search order.

For example:

```
//jobname JOB . . .
//        JCLLIB ORDER=UNV.SUNVSAMP
//STEP1   EXEC UEMPRC
```

Universal Query Customization

Universal Query Configuration

Universal Query for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UQRCFG00** in the **UNVCONF** library allocated to ddname **UNVCONF** in the Universal Broker's started task JCL procedure.

Any changes to the configuration member requires the Broker's configuration cache to be refreshed by either restarting the Broker started task or using the Universal Control utility refresh option.

See the Universal Agent Utilities 6.2.x Reference Guide for details on Universal Query configuration options.

zOS Installation - Cumulative PTF Maintenance

- z/OS Installation - Cumulative PTF Maintenance
- Installation Procedures
- Cumulative PTF Maintenance Distribution File
  - Distribution File Content
- Transferring a Cumulative PTF File to z/OS
- Installing a Cumulative PTF File

z/OS Installation - Cumulative PTF Maintenance

This page describes the procedures for installing a cumulative PTF maintenance for Universal Agent for z/OS.

Installation Procedures

Installation of a cumulative PTF maintenance is comprised of the following procedures:

Step 1	Download the cumulative PTF maintenance distribution file.
Step 2	Decompress the distribution file using any utility capable of processing UNIX-compressed files (for example, **compress** and **gzip**).
Step 3	Transfer the cumulative PTF file from Windows to z/OS. 1. Issue the following command to extract the installation (XMT) files from the distribution file: **`tar -xvf sb-6.2.x.x-zos-PTFs.tar`**. 2. Allocate a data set for the PTF file. 3. Transfer the PTF file to the data set on z/OS. 4. Unpack the data set.
Step 4	Install the cumulative PTF file.

Cumulative PTF Maintenance Distribution File

To obtain a cumulative PTF maintenance, you must download the corresponding distribution file from the Stonebranch Customer Portal.

A customer user ID and password — provided by Stonebranch, Inc. — are required to access this area.

Distribution File Content

The following file is included in a cumulative PTF maintenance distribution file:

- `UNV620xx.XMT` (cumulative PTF file for Universal Agent level xx)

If you do not have this file, contact Stonebranch, Inc. Customer Support for a complete distribution file.

Transferring a Cumulative PTF File to z/OS

A cumulative PTF file must be transferred to the z/OS system on which the maintenance is being installed.

You must extract the PTF file from the distribution file, then transfer it from the Windows or UNIX-based workstation to the z/OS system.

The following steps identify how to transfer of the cumulative PTF file.

Step 1	Allocate an z/OS data set to receive the PTF file as: • DSORG=PS • RECFM=FB • LRECL=80 • BLKSIZE=3120
Step 2	Transfer the PTF file - using a binary transfer from your local workstation - to the XMIT data set allocated on z/OS in Step 1.

Step 3	Enter the following TSO command to RECEIVE the XMT file: **`TSO RECEIVE INDA('cum PTF data set name allocated in step 1')`**
Step 4	At the INMR906A message prompt, enter any desired restore parameters.
Step 5	Continue with the installation of the maintenance (see Installing a Cumulative PTF File, below).

Files ending with the XMT suffix are binary files in a TSO TRANSMIT format.

Installing a Cumulative PTF File

Cumulative PTF Maintenance is installed using members **UNVMTRCV**, **UNVMTAPL**, and **UNVMTACC** in the **INSTALL** product library.

The installation itself consists of running a number of batch jobs. The output of these batch jobs should be kept until a correct installation has been verified.

Step 1	Member **UNVMTRCV** performs an SMP/E RECEIVE of the PTFs. Change the data set allocated on ddname **SMPPTFIN** to the PTF data set. Modify the JCL to meet local requirements, and submit the job.
Step 2	Member **UNVMTAPL** performs an SMP/E APPLY of the PTFs. Modify the JCL to meet local requirements, and submit the job. The job should end with condition code 0. If the job does not end with condition code 0, it is most likely due to a HOLD on one of the PTFs. This is indicated by message GIM30206E printed on ddname **SMPOUT**. In this case, see PTF HOLDs, above.
Step 3	Fully test the maintenance. Only after all testing is completed should you proceed to the next step.
Step 4	Member **UNVMTACC** performs an SMP/E ACCEPT of the PTFs. Modify the JCL to meet local requirements, then submit the job.

The job should end with condition code 0 or 4. If it does not, it is most likely due to a HOLD on one of the PTFs. This is indicated by message GIM30206E printed on ddname **SMPOUT**. In this case, see PTF HOLDs, above.|

zOS Installation - Time Zone Environment Variable

- Time Zone Environment Variable
- North American Values
- European Values

Time Zone Environment Variable

Universal Agent components execute in a z/OS UNIX environment, also known as a POSIX(ON) Language Environment.

The z/OS UNIX environment assumes that the z/OS system time is GMT or UTC format. It then uses the **TZ** environment variable value to determine the adjustments required for local time.

The **TZ** environment variable is an IBM Language Environment (LE) environment variable with a value set to the appropriate time zone and offset information so that time values are properly processed. **TZ** allows you to set the standard and daylight savings (or summer time) time zones and the offset from the local time zone to the UTC time.

LE environment variables, such as **TZ**, are set with the LE option **ENVAR** using the JCL step **PARM** keyword. For example, the following statement sets the standard time zone to Eastern Standard Time (EST) with an offset of 5 hours and the daylight saving time zone to Eastern Daylight Saving Time (EDT).

```
//PS1      EXEC PGM=UCMD,PARM='ENVAR(TZ=EST5EDT)/&UPARM'
```

Note
The forward slash (**/**) character separates the **LE** runtime parameters from the program parameters.

North American Values

TZ environment variables values for North America are listed below.

- EST5EDT: Eastern Standard Time, Eastern Daylight Time
- CST6CDT: Central Standard Time, Central Daylight Time
- MST7MDT: Mountain Standard Time, Mountain Daylight Time
- PST8PDT: Pacific Standard Time, Pacific Daylight Time
- AKST9AKDT: Alaska Standard Time, Alaska Daylight Time

European Values

TZ environment variables values for Europe are listed below.

- GMT0BST: Greenwich Mean Time, British Summer Time
- WET0WEST: Western Europe Time, Western Europe Summer Time
- CET-1CEST: Central Europe Time, Central Europe Summer Time
- EET-2EEST: Eastern Europe Time, Eastern Europe Summer Time

Other common time zone abbreviations may be used. IBM does not document all possible values.

zOS Installation - TCPIP Configuration

- Overview
- Specifying TCP/IP Affinity
- TCP/IP Resource Protection
  - Stack Access Control
  - Port Access Control
  - Network Access Control
  - Socket Option Access Control

Overview

This page describes optional Universal Agent and TCP/IP configuration topics. Whether these steps are required or desirable depends on your local TCP/IP configuration.

Specifying TCP/IP Affinity

Universal Agent programs are considered generic client and server programs in IBM's TCP/IP terminology. They do not have an affinity for a specific transport provider (TCP/IP started task). The programs will utilize the appropriate transport provider based on TCP/IP configuration.

An affinity to a specific transport provider can be established for any Universal Agent program using Language Environment variable **_BPXK_SETIBMOPT_TRANSPORT**. The variable value specifies the TCP/IP started task name the program must use as its transport provider.

The JCL illustration below establishes affinity to TCP/IP started task TCPIPA:

```
//PS1    EXEC PGM=UCMD,
//       PARM='ENVAR("_BPXK_SETIBMOPT_TRANSPORT=TCPIPA")/&UPARM'
```

TCP/IP Resource Protection

The IBM TCP/IP product (Communications Server) offers optional protection to TCP/IP resources using SAF interfaces. The protection is implemented with a set of resource profiles defined in the SERVAUTH class.

If you are using the TCP/IP resource protection, you must permit appropriate privileges to the user profiles with which Universal Agent executes. Refer to the IBM Communications Server: IP Configuration Guide for complete details on TCP/IP resource protection. The TCP/IP resource profiles and the Universal Agent required access are discussed in the following sections.

Stack Access Control

The SAF resource profile **`EZB.STACKACCESS.sysname.tcpname`** in the SERVAUTH class controls which user profiles have access to a TCP/IP stack. All user profiles with which Universal Agent executes require READ access to the appropriate TCP/IP stack access profile.

Port Access Control

The SAF resource profile **`EZB.PORTACCESS.sysname.tcpname.SAF`** keyword in the SERVAUTH class controls access to specific non-ephemeral port ranges.

The Universal Broker binds to a service port (defaults to 7887). The user profile with which the Universal Broker started task executes requires READ access to any resource that protects this port.

The Universal Enterprise Controller binds to a service port (defaults to 8778). The user profile with which the Universal Enterprise Controller started task executes requires READ access to any resource that protects this port.

Network Access Control

The SAF resource profile **`EZB.NETACCESS.sysname.tcpname.zonename`** in the SERVAUTH class controls access to security zones. A security zone defines networks and hosts by IP address. All user profiles with which Universal Agent executes require READ access to the appropriate TCP/IP security zones profile.

Socket Option Access Control

The SAF resource profile **EZB.SOCKOPT.sysname.tcpname.SO_BROADCAST** in the SERVAUTH class controls access to the socket SO_BROADCAST option.

No Universal Agent programs use the SO_BROADCAST socket option, so no user profiles require access.

zOS Installation - SAP RFC DLL

- Overview
- Background
- Output in the USS File System
- Universal Agent Components

Overview

The SAP RFC DLL is a remote function call library provided by SAP AG for applications to interface with the SAP system. A number of Universal Agent components make use of the SAP RFC DLL to provide SAP-related functionality.

The SAP RFC DLL interacts with the z/OS Language Environment (LE) and the z/OS UNIX System Services (USS), resulting in residual files being created by LE in the USS file system.

Background

A z/OS user profile must have a properly defined OMVS segment in order to run a program that utilizes USS services. The OMVS segment specifies, among other attributes, a home directory in the USS file system. If no home directory is specified, it defaults to **/tmp**.

The z/OS Language Environment (LE) produces output under certain conditions. When a LE batch job or started task is executed, the LE MSGFILE option specifies the ddname to which LE output is written. By default, MSGFILE specifies the SYSOUT ddname.

Additionally, for diagnostic purposes, LE can write LE dumps and traces. As a batch job or started task, LE writes the dumps and traces to ddname CEEDUMP. When a LE program is run from the USS environment, the LE MSGFILE option defaults to standard error, and LE dumps and traces are written to a file created in the current working directory with a name starting with CEEDUMP.

The SAP RFC DLL is loaded dynamically at run time when a Universal Agent batch job or started task needs to use SAP RFC functions. When the DLL is loaded, RFC initialization is performed by the DLL. As part of the RFC initialization, a USS **popen** function is called from within the DLL that creates two USS processes that run as child processes of the Universal Agent program. The child processes run for a very brief amount of time (1-2 seconds).

A USS child process inherits a number of attributes from its parent, which in this case is a Universal Agent program. Among the attributes inherited are the user ID profile, including the OMVS segment, as well many of the LE options specified in the batch job or started task. Among the attributes that a child process does not inherit is the ddname allocations of its parent. The child processes run in a separate address space managed by z/OS Workload Manager.

Output in the USS File System

The two child processes created by the **popen** function executed by the SAP DLL can result in LE creating certain files in the USS file system. The LE options that the batch job or started task execute with are inherited by the child processes created by the **popen** function. When the LE options produce output, this output will be created for each of the USS child processes in the USS file system.

As an example, the LE option RPTOPTS(ON) will write a report to the location specified by the LE MSGFILE option. The report lists all the LE option values and the source of the options. The two child processes created by the SAP RFC DLL inherit the LE options, and when both of the child processes end, LE writes a options report to the location specified by the LE MSGFILE option.

Since the child processes run in a USS environment, the LE MSGFILE option defaults to standard error; however, there is no standard error defined in this case, so LE writes its output to a file named SYSOUT in the current working directory. The current working directory will be the home directory specified in the user profile OMVS segment.

There are a number of conditions that cause LE to produce output. A partial list is:

- Certain LE options produce reports to the MSGFILE location. For example, RPTOPTS and RPTSTG.
- LE error messages resulting from invalid options or run-time errors are written to the MSGFILE location.
- LE dumps and traces produced due to run-time exceptions or the LE TRACE option. Dumps and traces are written to either the CEEDUMP ddname or to a USS file name starting with CEEDUMP.

Universal Agent Components

The following Universal Agent components use the SAP RFC DLL:

- Universal Broker
- Universal Enterprise Controller
- Universal Connector

When these components are executed with LE options that product output, a USS file named SYSOUT or a file starting with CEEDUMP will be created in the home directory of the user profile executing the component.

If two or more instances of a component executing with different user profiles share the same home directory, security violations may occur. The first component creates a USS file named SYSOUT owned by its user ID and the second component executing with a different user profile attempts to append to it resulting in a security violation.

To avoid the potential security violation when attempting to append to the LE SYSOUT file on the USS file system, define each z/OS user profile with a unique home directory.

zOS Installation - Configuration of zOS System SSL

- Configuration of z/OS System SSL
  - SSL Benefits
  - Required Conditions for Using SSL
- Additional Information

Configuration of z/OS System SSL

Universal Agent can use the IBM z/OS System SSL library or the OpenSSL SSL library for its SSL network communications. The SSL library selection is made with the Universal Agent SSL_IMPLEMENTATION configuration option.

z/OS System SSL requires the IBM System SSL Cryptographic Services base element. In addition, Universal Agent requires Cryptographic Services Security Level 3 element, which includes the cryptographically strong SSL cipher suites.

SSL Benefits

System SSL provides the following benefits:

- Utilizes any cryptographic hardware features available reducing the amount CPU resources used by Universal Agent.
- Seamless integration with RACF certificate management features.

Note
If RACF digital certificates are new to you or your site, refer to the following documentation for complete details:
- z/OS Security Server RACF Security Administrator's Guide
- z/OS Security Server RACF Command Language Reference

Required Conditions for Using SSL

In order for Universal Agent to use z/OS System SSL, the following conditions must be met:

1. Universal Agent supports z/OS System SSL on z/OS 1.4 and above.
2. Universal Agent component SSL_IMPLEMENTATION configuration values must be set to **system**.
3. User profiles with which the Universal Agent component executes must have READ access to the RACF profile **IRR.DIGTCERT.LISTRING** in the FACILITY class.
4. User profiles with which the Universal Agent component executes must have a certificate key ring associated with them that includes the user's certificate and the CA's certificate.

Additional Information

The following pages provide additional detailed information for Configuration of z/OS System SSL:

- Integrated Cryptographic Service Facility (ICSF)
- Universal Broker Digital Certificate (RACF) Set-up

Integrated Cryptographic Service Facility (ICSF)

Integrated Cryptographic Service Facility (ICSF)

Integrated Cryptographic Service Facility (ICSF)

z/OS System SSL will use ICSF when available. The ICSF started task must be running and ICSF configuration completed.

The user profile with which the System SSL application executes must have access to the following ICSF resources in the CSFSERV class:

- CSFCKI, clear key import
- CSFCKM, multiple clear key import
- CSFDEC, symmetric key decrypt
- CSFDSG, digital signature generate
- CSFDSV, digital signature verify
- CSFENC, symmetric key encrypt
- CSFPKB, PKA key build
- CSFPKD, PKA decrypt
- CSFPKE, PKE encrypt
- CSFPKI, PKA key import

Refer to the IBM z/OS ICSF Administrator's Guide for managing access to ICSF resources.

Universal Broker Digital Certificate (RACF) Set-up

Universal Broker Digital Certificate (RACF) Set-up

Setting up a digital certificate infrastructure in a production environment for the first time requires careful planning by the Security organization and Universal Agent administrator. The instructions provided on this page have been simplified for illustration purposes.

You work with RACF Digital Certificates using the RACF command **RACDCERT**. RACF profiles control access to the functions provided by **RACDCERT**.

The user profile with which the following commands are executed require either:

- SPECIAL attribute
- Appropriate access to the **IRR.DIGTCERT.function** profile in the FACILITY class.
  - READ access is required to **IRR.DIGTCERT.function** to issue **RACDCERT** commands for the executing user.
  - UPDATE access is required to **IRR.DIGTCERT.function** to issue **RACDCERT** commands for other users.
  - CONTROL access is required to **IRR.DIGTCERT.function** to issue **RACDCERT** command for SITE or CERTAUTH certificates.

The member **UNVINDC** in the **INSTALL** library contains the JCL to execute the RACF commands listed in the following steps.

Step 1

Create a Certificate Authority (CA) certificate and private key using the following RACDCERT command:

```
RACDCERT CERTAUTH GENCERT +
   SUBJECTSDN(CN('Certificate Authority') +
      OU('Security') +
      O('Company Name, Inc.') +
      C('US')) +
   NOTAFTER(DATE(2030-01-01)) +
   *KEYUSAGE(HANDSHAKE CERTSIGN) +
   WITHLABEL('Company CA')
```

Change the subject and label names to meet local requirements.

Step 2

Create a certificate for the Universal Broker STC and sign it with the CA certificate created in Step 1 using the following RACDCERT command:

```
RACDCERT ID(UBRUSR) GENCERT +
   SUBJECTSDN(CN('broker.company.com') +
      OU('Operations') +
      O('Company Name, Inc.') +
      C('US')) +
   KEYUSAGE(HANDSHAKE) +
   WITHLABEL('Broker') +
   SIGNWITH(CERTAUTH LABEL('Company CA')
```

Change the subject and label names to meet local requirements. The subject's Common Name (CN) value should uniquely identify this instance of the broker in the enterprise.

Step 3

Create a certificate key ring for the user profile UBRUSR with the following RACDCERT command:

```
RACDCERT ID(UBRUSR) ADDRING(BROKER)
```

Step 4 Connect the CA certificate and the Universal Broker certificate to the key ring with the following RACDCERT command:

```
RACDCERT ID(UBRUSR) CONNECT(CERTAUTH LABEL('Company CA') +
   RING(BROKER)
RACDCERT ID(UBRUSR) CONNECT(LABEL('Broker') RING(BROKER) DEFAULT)
```

Change the labels to match the values used in previous steps.

Step 5 If the resource profile **IRR.DIGTCERT.LISTRING** in the FACILITY class is not defined, define it with the following RDEFINE command:

```
RDEFINE FACILITY (IRR.DIGTCERT.LISTRING) UACC(NONE)
```

Step 6 Permit the Broker user profile UBRUSR READ access to the RACF profile **`IRR.DIGTCERT.LISTRING`** in the FACILITY class using the following PERMIT command:

```
PE IRR.DIGTCERT.LISTRING CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
```

Step 7 Modify the Universal Broker configuration member **UBRCFG00** as follows:

```
ssl_implementation   system
saf_key_ring        BROKER
```

Step 8 The CA certificate must be distributed to the remote systems from which Universal Agent managers are executed. The managers must be configured with the CA certificate in their list of Trusted CA certificates using the CA_CERTIFICATES configuration option.

The CA certificate is exported out of the RACF data base into a data set in a PEM (or base64) format with the following RACDCERT command:

```
RACDCERT CERTAUTH EXPORT (LABEL('Company CA')) +
   DSN(TEST.CA.CERT) FORMAT(CERTB64)
```

Change the label to match the value used in previous steps.

The **tsoprefix.TEST.CA.CERT** data set contains a PEM formatted certificate. The format is a text format that transfers safely across the network in text mode.

Note that the CA private key is not exported. The CA certificate does not contain any private data.

zOS Installation - Configuration of Security

Introduction

Some Universal Agent components utilize z/OS security services to control access to product functions and resources. These pages describe the installation steps to activate product security and define resource controls.

Note
The use of the security resource is optional. If you do not want to use it at this time, this information can be skipped.

Detailed Information

The following pages specify the configuration methods for IBM's RACF and Computer Associate's ACF2 security products:

- RACF Class
- ACF2 Class
- Universal Command Security
- Universal Control Security
- Universal Event Monitor Security

Configuration of Security - RACF Class

RACF Class

All components use the Universal Agent RACF class **$UNV**. The class name can be changed if local requirements require it.

To install the Universal Agent RACF class, perform the following steps:

Step 1	Modify your local RACF class descriptor table by adding the following entry to **ICHRRCDE**:
	
	`ICHERCDE CLASS=$UNV,  +`
	`CASE=ASIS,  +`
	`FIRST=ANY,  +`
	`OTHER=ANY,  +`
	`MAXLNTH=246,  +`
	`DFTUACC=NONE,  +`
	`ID=128,  +`
	`POSIT=128`

Step 2	Modify the ID and POSIT values to meet local requirements. Support for the CASE parameter was added in z/OS 1.2. Case-sensitive profile names are necessary for USS path name support in the standard I/O access profiles. See Universal Command Standard I/O Access Profile for details. Care must be taken when defining case-sensitive profile names so that the profile characters are typed with the desired case. RACF will not automatically upper case the profile names. If the wrong case is used, the desired profile will not be found. If the standard I/O profiles are not necessary or USS path name support is not necessary, the CASE parameter can be removed. Sample assemble and link edit JCL is provided in member **UNVRRCDE** in the **INSTALL** library. IBM provides a sample ICHRRCDE table and the JCL to assemble and link edit it in **SYS1.PARMLIB(RACTABLE)**.

Step 3	Modify your local RACF routing table by adding the following entry to ICHRFR01.
	
	`ICHRFRTB CLASS=$UNV,  +`
	`ACTION=RACF`
	
	Sample assemble and link edit JCL is provided in member **UNVRFR01** in the **INSTALL** library.
	
	IBM provides a sample ICHRRCDE table and the JCL to assemble and link edit it in **`SYS1.PARMLIB(RACTABLE)`**.

Step 4	If you changed the default RACF class name **$UNV** to another name, the UCMD, UCTL, and UEM load modules must be modified to include the same class name. (If you did not change the default class name, skip this step.) 1. Member **UCMCMVS** in the **INSTALL** library contains an SMP/E USERMOD to apply the modification to the UCMD load module. Change the USERMOD to the new class name and submit the job. 2. Member **UCTCMVS** in the **INSTALL** library contains an SMP/E USERMOD to apply the modification to the UCTL load module. Change the USERMOD to the new class name and submit the job. 3. Member **UEMCMVS** in the **INSTALL** library contains an SMP/E USERMOD to apply the modification to the UEM load module. Change the USERMOD to the new class name and submit the job. All steps must end with return code 0.
Step 5	IPL the z/OS system to install the new class descriptor and routing tables.

Step 6	Activate the new class and define it for generic profile checking by entering the following RACF SETROPTS commands:
	
	`SETROPTS CLASSACT($UNV)`
	`SETROPTS GENERIC($UNV)`

Configuration of Security - ACF2 Class

ACF2 CLASS

All components use the Universal Agent ACF2 class **$UNV**. The class name may be changed if local requirements require it.

To install the Universal Agent ACF2 class, perform the following steps:

Step 1 Create a CLASMAP with the following ACF2 command:

```
SET CONTROL(GSO)
INSERT CLASSMAP.$UNV RESOURCE($UNV) RSRCTYPE(UNV) ENTITYLN(246)
```

Step 2 Rebuild and refresh the directories with the following console command:

```
F ACF2,REFRESH(CLASMAP),SYSID(*sysid*),CLASS(C),TYPE(GSO)
```

Step 3 Define the following resource rules for Universal Command and Universal Control with the following commands:

```
SET RESOURCE(OSM)
COMPILE * STORE
   $KEY(UCMD) TYPE(UNV)
   UID(ucmd_admin_id) ALLOW
   - UID(ucmd_admin_id) SERVICE(READ) ALLOW

SET RESOURCE(OSM)
COMPILE * STORE
   $KEY(UCTL) TYPE(UNV)
   UID(uctl_admin_id) ALLOW
   - UID(uctl_admin_id) SERVICE(READ) ALLOW
```

Configuration of Security - Universal Command Security

- Overview
- Universal Command Remote Access Profile
- Universal Command Standard I/O Access Profile
- Universal Command Security Profile Definitions
  - Example 1
  - Example 2
  - Example 3
  - Example 4

Overview

Universal Command access controls consist of the following profile types:

- Remote Access profiles control Universal Command's access to remote systems.
- Stdio Access profiles control Universal Command's access to local data sets used as standard input, output or error.

Universal Command Remote Access Profile

The remote access profile controls Universal Command Manager's access to specific remote systems running Universal Agent. The remote system is identified by the IP address of the remote system, the port number on which the Manager is connecting to the remote system, and the remote system user ID with which the Manager is requesting the command to be executed.

Universal Command Manager identifies a remote system with the REMOTE_HOST and REMOTE_PORT configuration options, and the remote user ID with the USER_ID configuration option.

The remote access profile has the following format:

```
UCMD.Iipaddress.Pport.userid
```

The following table identifies the fields that comprise the profile name.

Field	Description
ipaddress	Numeric dotted-form IP address of the remote host as specified by the REMOTE_HOST option. The format of the IP address is four three-character numeric fields. Each field represents one number of the IP address. For example, IP address **256.10.2.123** is formatted as **256010002123**.
port	Numeric port number on which Universal Broker is listening as specified by the REMOTE_PORT option. The default Universal Broker port number is 7887. The format of the port number is a five-digit number. For example, port **7887** is formatted as **07887**.
userid	Remote user identifier with which Universal Command Manager will execute the remote command as specified by the USER_ID option. Whether or not a user identifier is required depends on the Universal Command Server configuration. If no user identifier is specified for the Manager, the userid value is <BLANK>. The value userid is upper case. Some remote hosts may have case-sensitive user identifiers. All user identifiers specified on the Manager are translated to upper case for building the profile.

Universal Command Standard I/O Access Profile

The standard I/O (**stdio**) access profile controls Universal Command Manager's access to data sets allocated for standard I/O. The standard I/O ddnames are **UNVIN** for standard input, **UNVOUT** for standard output, and **UNVERR** for standard error, unless otherwise changed with the SIO_LOCAL_FILE configuration option.

The standard I/O access profile has the following format:

```
UCMD.stdio.Iipaddress.Pport.allocation
```

The following table identifies the fields that comprise the profile name.

Field	Description

stdio	Standard I/O file which the profile is for. Valid values are: • STDIN for the standard input file. • STDOUT for the standard output file. • STDERR for the standard error file.
ipaddress	Numeric dotted-form IP address of the remote host as specified by the REMOTE_HOST option. The format of the IP address is four three-character numeric fields. Each field represents one number of the IP address. For example, IP address **256.10.2.123** is formatted as **256010002123**.
port	Numeric port number on which Universal Broker is listening as specified by the REMOTE_PORT option. The default Universal Broker port number is 7887. The format of the port number is a five-digit number. For example, port **7887** is formatted as **07887**.

allocation	Entity that is allocated to the standard I/O ddname. There are a number of different items that may be allocated to a ddname. The following formats are available:
	• A data set allocation is represented by **Ddsn**, where **dsn** is the data set name. For example, a ddname allocation of **PROD.APPL.DATA** is formatted as **DPROD.APPL.DATA**.
	  • If allocating a PDS and a member name is specified, do not include the member name in the profile name, only the PDS name.
	  • If allocating a relative generation data set, do not include the relative number in the profile name, only the GDG name.
	• A JES SYSIN, SYSOUT or SUBSYS= allocation is represented by the value SUBSYS.
	• A temporary data set allocation is represented by the value TEMPORARY.
	• A DUMMY or NULLFILE allocation is represented by the value NULLFILE.
	• A USS path name allocation is represented by **Upath**, where **path** is the USS path name. For example, a ddname allocation of **/prod/appl/data** is formatted as **U/prod/appl/data**.
	  • USS path name support starts at z/OS 1.4. Prior to z/OS 1.4, the path name is not available to format the profile name. The value used for path in this case is **UNKNOWNUSSPATHNAME**.
	  • USS path names are case sensitive. The Universal Agent class must be defined with the CASE=ASIS parameter to support case sensitive profile names.
	
	  The maximum length of the profile name is 246 characters. The maximum length of a USS path name allocated to a ddname is 256 characters. It is possible the path name may be truncated in formatting the profile name. Truncation does not result in an error condition.

Universal Command Security Profile Definitions

These examples illustrate sample RACF commands that can be used to define Universal Command security profiles and permit z/OS user identifiers access to those profiles. Refer to the IBM RACF documentation for complete details on RACF commands.

Example 1

Assume that you want to restrict Universal Command Manager for z/OS access to remote host **10.23.90.2**. The following profile would restrict access to only those z/OS users who have read access to the profile **UCMD.I010023090002.*.***.

The following TSO commands define the required profile and permits access to TSO user TSO555.

```
RDEF $UNV (UCMD.I010023090002.*.*) UACC(NONE)
PE UCMD.I010023090002.*.* CLASS($UNV) ID(TSO555) ACCESS(READ)
```

Example 2

Assume that you run all Universal Brokers on privileged port **1000**. To enforce the policy that z/OS Managers connect only to port 1000, define profile **UCMD.*.*.*** with universal access none and define **UCMD.*.P01000.*** with universal access read.

The following TSO commands define the required profiles.

```
RDEF $UNV (UCMD.*.*.*) UACC(NONE)
RDEF $UNV (UCMD.*.P01000.*) UACC(READ)
```

Example 3

Assume that you want to restrict root access to all hosts from Universal Command Manager for z/OS. The following profile would restrict root access to only those z/OS users who have read access to the profile **UCMD.*.*.ROOT**.

The following TSO command defines the required RACF profile.

```
RDEF $UNV (UCMD.*.*.ROOT) UACC(NONE)
```

Example 4

Assume that you want to restrict Universal Command Manager access to data sets **PROD.***. You also don't want Universal Command Manager to use any temporary data sets. The following profiles would restrict access to only those z/OS users who have read access to the profile.

The following TSO command defines the required RACF profile.

```
RDEF $UNV (UCMD.STD*.*.*.DPROD.**) UACC(NONE)
RDEF $UNV (UCMD.STD*.*.*.TEMPORARY) UACC(NONE)
```

Configuration of Security - Universal Control Security

- Overview
- Universal Control Remote Access Profile Format
- Universal Control Security Profile Definition
  - Example 1

Overview

Universal Control access controls consist of the following profile types:

Remote Access profiles control Universal Control's access to remote systems.

Universal Control Remote Access Profile Format

The remote access profile controls Universal Controls Manager's access to specific remote systems running Universal Agent. The remote system is identified by the IP address of the remote system, the port number on which the Manager is connecting to the remote system, and the command to be executed.

Universal Control Manager identifies a remote system with the REMOTE_HOST and REMOTE_PORT configuration options, and the command as one of the command options.

The remote access profile has the following format:

`UCTL.Iipaddress.Pport.command`

The profile name is composed of the following fields.

Field	Description
ipaddress	Numeric dotted-form IP address of the remote host as identified by the REMOTE_HOST option. The format of the IP address is four three-character numeric fields. Each field represents one number of the IP address. For example, IP address **256.10.2.123** is formatted as **256010002123**.
port	Numeric port number on which Universal Broker is listening as identified by the REMOTE_PORT option. The default Universal Broker port number is 7887. The format of the port number is a five-digit number. For example, port **7887** is formatted as **07887**.
command	Universal Control command that the Manager is requesting execution. Possible command values are START, STOP, and REFRESH.

Universal Control Security Profile Definition

This example illustrates sample RACF commands that can be used to define Universal Control security profiles and permit z/OS user identifiers access to those profiles. Refer to the IBM RACF documentation for complete details on RACF commands.

Example 1

Assume you wish to restrict Universal Control Manager for z/OS access to remote host **10.23.90.2**. The following profile would restrict access to only those z/OS users who have read access to the profile **`UCTL.I010023090002.*.*`**.

The following TSO commands define the required profile and permits access to TSO user TSO555.

```
RDEF $UNV (UCTL.I010023090002.*.*) UACC(NONE)
PE UCTL.I010023090002.*.* CLASS($UNV) ID(TSO555) ACCESS(READ)
```

Configuration of Security - Universal Event Monitor Security

- Overview
- Universal Event Monitor Remote Access Profile Format
- Universal Event Monitor Security Profile Definition
  - Example 1

Overview

Universal Event Monitor access controls consist of the following profile types:

Remote Access profiles control Universal Event Monitor's access to remote systems.

Universal Event Monitor Remote Access Profile Format

The remote access profile controls Universal Event Monitor Manager's access to specific remote systems running Universal Agent. The remote system is identified by three elements:

1. System's IP address
2. Port number that the Manager uses to connects to a Universal Broker executing on the system
3. User account specified from the UEM Manager, which is defined on the remote system

The Universal Event Monitor Manager identifies a remote system with the REMOTE_HOST and REMOTE_PORT configuration options, and the remote user account with the USER_ID configuration option.

The remote access profile has the following format:

`UEM.Iipaddress.Pport.userid`

The profile name is composed of the following fields.

Field	Description
ipaddress	IP address of the remote host, in dotted-decimal notation, as identified by the REMOTE_HOST configuration option. The format of the IP address is four three-character numeric fields. Each field represents one number of the IP address. For example, IP address **256.10.2.123** is formatted as **256010002123**.
port	Numeric port number on which Universal Broker is listening as identified by the REMOTE_PORT configuration option. The default Universal Broker port number is 7887. The format of the port number is a five-digit number. For example, port **7887** is formatted as **07887**.
userid	ID of a remote user account, which was specified by the Manager with the USER_ID option. If the Universal Event Monitor Server is configured to not require a user ID, the value for this field is <BLANK>.The value *userid* is upper case. Some remote hosts may have case-sensitive user identifiers. All user identifiers specified on the Manager are translated to upper case for building the profile.

Universal Event Monitor Security Profile Definition

This example illustrates sample RACF commands that can be used to define Universal Event Monitor security profiles and permit local user accounts access to those profiles. Refer to the IBM RACF documentation for complete details on RACF commands.

Example 1

Assume you wish to restrict Universal Event Monitor Manager for z/OS access to remote host **10.23.90.2**. The following profile would restrict access to only those z/OS users who have read access to the profile **`UEM.I010023090002.*.*`**.

The following TSO commands define the required profile and permits access to TSO user TSO555.

```
RDEF $UNV (UEM.I010023090002.*.*) UACC(NONE)PE UEM.I010023090002.*.*
CLASS($UNV) ID(TSO555) ACCESS(READ)
```

zOS Installation - Performance Guidelines

- Overview
- UNIX System Services and Language Environment
- Universal Agent Managers
- Universal Broker and Universal Agent Servers
- Universal Enterprise Controller

Overview

Universal Agent consists of product components distributed throughout the enterprise communicating with each other over the computer network using the TCP/IP communication protocol.

Universal Agent offers reliable, fault tolerant, secure, and efficient communications between its distributed components. In order for product components to effectively meet their communication requirements, z/OS must provide sufficient execution time for the product components.

The execution of the communication protocol is a real-time activity and communication time-outs may be exceeded if product components are not dispatched appropriately while executing the communication protocol.

The following sections provide performance guidelines for Universal Agent for z/OS components.

UNIX System Services and Language Environment

All Universal Agent components are written in C/C++ and utilize z/OS Language Environment (LE) and z/OS UNIX System Services (USS).

The IBM z/OS UNIX System Services Planning manual includes a "Tuning Performance" chapter that should be reviewed to improve USS performance in general.

Universal Agent components do not attempt to use the USS **setpriority**, **chpriority**, or **nice** functions to adjust their performance group or service class.

Universal Agent Managers

Universal Agent Managers consist of Universal Command, Universal Data Mover, and Universal Event Monitor managers. They typically execute in the JES subsystem as a batch job or the OMVS subsystem as a USS shell command.

The managers communicate with remote Universal Brokers and their corresponding Universal Agent Server components on remote systems using the TCP/IP protocol.

In cases where the z/OS workload requires more resources than are available, z/OS will favor workload with a higher dispatch priority over workload with a lower dispatch priority. If a Universal Agent manager is being executed with a lower dispatch priority than other workload competing for resources, it may not be given sufficient execution time to meet its network communication timing requirements. The result will be false network time-out errors in the Universal Agent manager.

The effect of a network time-out condition depends on whether or nor the Universal Agent manager is using the Network Fault Tolerant (NFT) feature. If NFT is used, the manager reestablishes the communication session and continues; otherwise, the manager ends with an error.

False network communication time-out errors can be addressed using one or both of the following options:

1. Increase the manager's NETWORK_DELAY configuration option value (default is 120 seconds). NETWORK_DELAY specifies the maximum amount of time to wait for data on a communication session before considering the session timed out. Increasing the value allows for the manager batch job to be swapped out for a longer period of time before the session will be considered timed out. However, in cases where a condition truly exists in the network that would result in a true network time-out, a larger NETWORK_DELAY value would result in a longer period of time before the manager would detect and respond to the time-out condition.
2. Raise the Universal Agent manager workload dispatch priority by placing it in a higher performance group or service class. Raising the workload dispatch priority will allow z/OS to provide sufficient CPU resources to the manager to meet network timing requirements.

Universal Broker and Universal Agent Servers

The Universal Broker started task is the center of activity on each system on which Universal Agent is installed. Almost all components communicate with a locally installed Universal Broker during their execution, including managers and servers.

Universal Broker is responsible for managing Universal Agent servers that are performing work on z/OS on behalf of remote Universal Agent managers. Universal Agent servers are created by the Universal Broker using the USS spawn function. The servers run as USS child processes

of the Universal Broker started task in the OMVS subsystem.

The Universal Broker started task must execute with a sufficiently high performance group or service class in order to service all manager and server requests in a timely manner to avoid false network time-out conditions.

On heavily loaded systems, it is recommended to make the Universal Broker started task non-swappable to help overall improvement of Universal Broker.

Universal Enterprise Controller

The Universal Enterprise Controller (UEC) started task performs real-time monitoring of Agents distributed throughout the network. UEC communicates with each Agent on a defined polling interval.

UEC is a USS, multi-threaded application written in C/C++ that heavily utilizes TCP / IP. The amount of work that UEC performs depends directly on the number of Agents defined to it. UEC maintains Agent status information and Universal Event Subsystem information in the UEC database. The database is an HFS or zFS database that is mounted in the z/OS UNIX file system.

UEC must execute with a sufficiently high performance group or service class in order to perform its Agent monitoring service effectively. False Agent time-out alerts can result if UEC is not dispatched in a timely manner.

On heavily loaded systems, it is recommended to make the UEC started task non-swappable to help overall performance of UEC.

zOS Installation - Converting STC User Profiles to a Non-Zero UID

- Overview
- Converting Universal Broker User Profile to Non-Zero UID
- Converting Universal Enterprise Controller (UEC) User Profile to Non-Zero UID

Overview

Prior to Stonebranch Solutions 4.2.0, the Universal Broker and Universal Enterprise Controller (UEC) started task user profiles were required to have an OMVS UID value of 0. As of 4.2.0, the products were enhanced to execute with a user profile defined with a non-zero UID value to improve upon the product security features.

A Universal Agent installation that already has a user profile with UID 0 in use can convert the user profile from UID 0 to a non-zero UID value. There are a number of concerns when changing a user profiles UID value. The UID value identifies the user profile in the z/OS UNIX (USS) environment.

The following sections describe how to convert a Universal Broker or Universal Enterprise Controller user profile UID value from 0 to non-zero.

Converting Universal Broker User Profile to Non-Zero UID

The conversion steps assume the following:

- The UID value is being changed from 0 to 5001. If a UID value of 5001 does not work in your local environment, change all references to 5001 in the following steps to a unique, non-zero UID value suitable for your local environment. Note that the UID value must be unique among all user profiles.
- The Universal Broker user profile name is **UBRUSR**. If the Universal Broker STC in your local environment uses a different user profile name, change all references to **UBRUSR** in the following steps to the user profile name used in your local environment.
- The user ID used to execute the commands requires an OMVS segment, and the user ID must have either UID 0 or READ access to the BPX.SUPERUSER profile in the FACILITY class.
- The Universal Broker HFS or zFS data sets must be mounted and their mount point known. The console system command **D OMVS,F** or the USS shell command **df** can be used to display all mounted USS data sets.

Step 1	Stop the Universal Broker STC if it is running.
Step 2	Change the user profile **UBRUSR** UID value to 5001 with the following command: `ALTUSER UBRUSR OMVS(UID(5001))`

Step 3 Permit the user profile **UBRUSR** READ access to the required resource profiles with the following commands:

```
PE BPX.SUPERUSER CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
PE BPX.JOBNAME CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
SETR RACLIST(FACILITY) REFRESH
```

The user profile **UBRUSR** should already have READ access to **BPX.DAEMON** in the FACILITY class based on Universal Broker installation requirements prior to 4.2.0. If **UBRUSR** does not have READ access to **BPX.DAEMON**, the following commands will permit appropriate access:

```
PE BPX.DAEMON CLASS(FACILITY) ID(UBRUSR) ACCESS(READ)
SETR RACLIST(FACILITY) REFRESH
```

For detailed information regarding Universal Broker security requirements, see zOS Configuration - Started Tasks.

Step 4 Universal Broker databases are maintained in USS HFS or zFS data sets. The database files have an owner attribute that is based on the UID value of the Universal Broker STC user profile. The database files, the root directory, and administration files must have their owner attribute changed from UID 0 to the new non-zero UID value 5001.

By default, the Universal Broker STC will dynamically mount the USS data sets in the **/tmp** directory. Assuming the USS data set names are **UNV.UNVDB** and **UNV.UNVSPOOL**, their mount point would be directory **/tmp/UNV.UNVDB** and **/tmp/UNV.UNVSPOOL**. If the mount point is different for your local environment, adjust the commands below to the appropriate directory names.

From the z/OS UNIX shell prompt, execute the following commands:

```
su
cd /tmp/UNV.UNVDB
chown -R 5001 *
chown 5001 .
chown 5001 .inited
cd /tmp/UNV.UNVSPOOL
chown -R 5001 *
chown 5001 .
chown 5001 .inited
exit
```

The first command, **su**, changes to the superuser ID. The user ID used to execute the above commands will need either a UID of 0 or READ access to the **BPX.SUPERUSER** profile in the FACILITY class. If the user ID has UID 0, the **su** command is not necessary.

Step 5 Start the Universal Broker STC.

Converting Universal Enterprise Controller (UEC) User Profile to Non-Zero UID

The conversion steps assume the following:

- The UID value is being changed from 0 to 5002. If a UID value of 5002 does not work in your local environment, change all references to 5002 in the following steps to a unique, non-zero UID value suitable for your local environment. Note that the UID value must be unique among all user profiles.
- The UEC user profile name is **UECUSR**. If the UEC STC in your local environment uses a different user profile name, change all references to **UECUSR** in the following steps to the user profile name used in your local environment.
- The user ID used to execute the commands requires an OMVS segment, and the user ID must have either UID 0 or READ access to the **BPX.SUPERUSER** profile in the FACILITY class.
- The UEC HFS or zFS data set must be mounted and its mount point known. The console system command **D OMVS,F** or the USS shell command **df** can be used to display all mounted USS data sets.

Step 1	Stop the UEC STC if it is running.
Step 2	Change the user profile UECUSR UID value to 5002 with the following command: `ALTUSER UECUSR OMVS(UID(5002))`

Step 3	Permit the user profile **UECUSR** READ access to the required resource profiles with the following commands:
	
	`PE BPX.SUPERUSER CLASS(FACILITY) ID(UECUSR) ACCESS(READ)`
	`SETR RACLIST(FACILITY) REFRESH`

Step 4	UEC databases are maintained in a USS HFS or zFS data set. The database files have an owner attribute that is based on the UID value of the UEC STC user profile. The database files, the root directory, and administration files must have their owner attribute changed from UID 0 to the new non-zero UID value 5002.
	
	By default, the UEC STC will dynamically mount the USS data set in the **`/tmp`** directory. Assuming the USS data set name is **`UNV.UECDB`**, its mount point would be directory **`/tmp/UNV.UECDB`**. If the mount point is different for your local environment, adjust the commands below to the appropriate directory names.
	
	From the z/OS UNIX shell prompt, execute the following commands:
	
	`su`
	`cd /tmp/UNV.UECDB`
	`chown -R 5002 *`
	`chown 5002 .`
	`chown 5002 .inited`
	`exit`
	
	The first command, **su**, changes to the superuser ID. The user ID used to execute the above commands will need either a UID of 0 or READ access to the **BPX.SUPERUSER** profile in the FACILITY class. If the user ID has UID 0, the **su** command is not necessary.

Step 5	Start the UEC STC.

zOS Installation - Data Set Inventory

- Types of Data Sets
- SMP/E Data Sets
- Non-SMP/E Data Sets

Types of Data Sets

As part of the Universal Agent for z/OS package installation, two types of data sets are allocated and cataloged:

- SMP/E data sets
- Non-SMP/E data sets

SMP/E Data Sets

The following table lists the SMP/E data sets — and their space requirements — that are allocated and cataloged as part of the Universal Agent for z/OS installation.

Depending on your installation choices, the data set high-level qualifiers may be different.

Data Set Name	Space (tracks)	Description
UNV.AUNVLOAD	6500	SMP/E distribution library for the product load library.
UNV.AUNVNLS	30	SMP/E distribution library for the product national language support library.
UNV.AUNVSAMP	30	SMP/E distribution library for the product sample library.
UNV.AUNVTMPL	60	SMP/E distribution library for configuration template files.
UNV.SMP.CSI	n/a	SMP/E CSI VSAM cluster for Universal Agent SMP/E zones.
UNV.SMP.CSI.DATA	75	SMP/E CSI VSAM data for Universal Agent SMP/E zones.
UNV.SMP.CSI.INDEX	15	SMP/E CSI VSAM index for Universal Agent SMP/E zones.
UNV.SMP.SMPLOG	30	SMP/E log file.
UNV.SMP.SMPLOGA	30	SMP/E backup log file.
UNV.SMP.SMPLTS	100	SMP/E target library for base versions of load modules using a SYSLIB allocation.
UNV.SMP.SMPMTS	30	SMP/E target library for macros existing only in the distribution libraries.
UNV.SMP.SMPPTS	4500	SMP/E temporary SYSMOD storage.
UNV.SMP.SMPSCDS	30	SMP/E zone backup library.
UNV.SMP.SMPSTS	30	SMP/E target library for source existing only in the distribution libraries.
UNV.SUNVLOAD	6500	SMP/E target library for the product load library.
UNV.SUNVNLS	30	SMP/E target library for product national language support library.
UNV.SUNVSAMP	30	SMP/E target library for the product sample library.
UNV.SUNVTMPL	60	SMP/E target library for configuration template file.

Non-SMP/E Data Sets

The following table lists the non-SMP/E data sets - and their space requirements - that are allocated and cataloged as part of the Universal Agent for z/OS package installation.

Depending on your installation choices, the data set high-level qualifiers may be different.

Data Set Name	Space (tracks)	Description

UNV.MDL	1	Universal Broker sequential trace data set allocation model.
UNV.UAG	1	Universal Automation Center Agent logging data set allocation model.
UNV.UCRDB	15	Universal Certificate database.
UNV.UECDB	4500	Universal Enterprise Controller HFS or zFS databases.
UNV.UNVCOMP	15	Universal Agent component definition library.
UNV.UNVCONF	15	Universal Agent configuration library.
UNV.UNVCREF	75	Universal Command Server command reference library.
UNV.UNVDB	150	Universal Broker HFS or zFS database.
UNV.UNVSPOOL	3000	Universal Agent HFS or zFS spool database.
UNV.UNVTRACE	150	Universal Broker PDS/E trace data set.
UNV.V6R2M0.INSTALL	30	Universal Agent package installation and maintenance JCL.

zOS Installation - Licensing

- Licensing Universal Agent for z/OS Components
- Product License File
  - Format
  - Sample
- Entering License Information
- Restart Universal Broker

Licensing Universal Agent for z/OS Components

After Universal Agent for z/OS has been installed, you must configure the following components with product licenses before they can be used:

- Universal Command Manager
- Universal Data Mover Manager
- Universal Connector

Product License File

For each component, product license information (license parameter keywords and their values) is contained in a separate text file provided by your Stonebranch, Inc. account representative.

Format

The format of the product license file name is: <*component name*>_<*customer name*>_<*operating system*>_<*schedule or solution*>.txt. For example: ***Indesca_Stonebranch_MVS_A1.txt***

- For Universal Command Manager, **Indesca** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.
- For Universal Data Mover Manager, **Infitran** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.

Sample

The following is a sample Universal Agent for z/OS product license file (for Universal Command Manager):

```
License_Product "INDESCA"
License_Customer "STONEBRANCH"
License_OS_Type "MVS"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

Enter each component's product license parameters into its configuration file in the **UNVCONF** product library:

Universal Command Manager configuration file	member **UCMCFG00**
Universal Data Mover Manager configuration file	member **UDMCFG00**
Universal Connector configuration file	member **USPCFG00**

It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration

options.)

If the Universal Broker STC is running, restart the Universal Broker STC to pick up the configuration file changes.

Restart Universal Broker

For Universal Broker to read the license information, you must stop and restart it.

Stop Universal Broker	`STOP UBROKER`
Start Universal Broker	`START UBROKER[,UPARM='options']`

zOS USS Installation

- Introduction
- Installation Procedures
- Detailed Information

Introduction

These pages describe the installation of Stonebranch, Inc.'s Universal Agent for z/OS UNIX System Services (USS) on a z/OS operating system. Unless otherwise specified, all references to Universal Agent for z/OS USS refer to version 6.2.0.

Note
Starting with the 3.2.0 release of Universal Products, a Universal Broker must run on all systems on which a Universal Agent component is running, including manager components. The Broker maintains product configuration data for all components that have a configuration file.

Installation Procedures

All Universal Agent for z/OS USS components are provided in the Universal Agent for z/OS SMP/E package.

The z/OS components in that package must be installed before you can install the z/OS USS components. (See z/OS Installation Procedures for information on completing the z/OS installation.)

After the z/OS components have been installed, you can install the z/OS USS components (see SMPE Installation).

The installation consists of running a number of batch jobs. The output of these batch jobs should be kept until a proper installation has been verified.

Note
Please read z/OS USS Installation - Installation Requirements before installing Universal Agent for z/OS USS.

Detailed Information

The following pages provide detailed information for Configuration of z/OS USS Installation:

- Components
- Installation Requirements
- Distribution File
- SMPE Installation
- Customization
- Data Set Inventory
- Licensing

zOS USS Installation - Installation Package

- Universal Agent for z/OS USS Components
- Component Compatibility

All Universal Agent for z/OS USS components are provided in the Universal Agent for z/OS SMP/E package.

Universal Agent for z/OS USS Components

The Universal Agent for z/OS package contains the following Universal Agent for z/OS USS Components:

- Universal Command Manager 6.2.x
- Universal Control Manager 6.2.x
- Universal Copy 6.2.x
- Universal Data Mover Manager 6.2.x
- Universal Encrypt 6.2.x
- Universal Event Monitor Manager 6.2.x
- Universal Message to Exit Code Translator 6.2.x
- Universal Query 6.2.x
- Universal Certificate 6.2.x
- Universal WTO 6.2.x

Component Compatibility

The following table identifies the compatibility of Universal Agent for z/OS USS components with previous component / product versions.

Component	Compatibility
Universal Command 6.2.x	Universal Command 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Control 6.2.x	Universal Control 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Data Mover 6.2.x	Universal Data Mover 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, and 3.1.0.
Universal Encrypt 6.2.x	Universal Encrypt 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Query 6.2.x	Universal Broker 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Event Monitor 6.2.x	Universal Event Monitor 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, and 3.1.0.

The component references pertain to all supported platforms for that version.

zOS USS Installation - Installation Requirements

- Overview
- Data Set Space Requirements
- SMP/E
- Platform Requirements

Overview

Universal Agent for z/OS USS require the following software releases:

- z/OS 1.12 or above.
- IBM Language Environment for z/OS 1.12 or above.
- IBM Communication Server for z/OS 1.12 or above.
- SMP/E 3.5 or above.
- Minimum 200 cylinders of DASD and 81M bytes on a z/OS UNIX file system.

The user identifier used to execute the installation jobs must meet the following requirements:

1. User ID must have a properly defined OMVS segment.
2. User ID must have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Important
All Universal Agent for z/OS USS components are provided in the Universal Agent for z/OS SMP/E package. The z/OS components must be installed before the z/OS USS components. See z/OS Installation for information on completing the z/OS installation.

Data Set Space Requirements

As part of the Universal Agent for z/OS USS installation, a number of SMP/E data sets are allocated and cataloged and a number of z/OS UNIX directories are created.

The space requirements for these data sets and directories are listed in z/OS USS Installation - Data Set Inventory.

SMP/E

Universal Agent for z/OS USS are installed using SMP/E. The components are installed in the Universal Agent SMP/E CSI.

The following table identifies the SMP/E FMIDs for the Universal Agent for z/OS USS components.

Product	FMID	SMP/E Requisites
USS Universal Common 6.2.x	UUNV620	No prerequisites. Supersedes and deletes FMID UUNV320, UUNV410, UUNV420, UUNV430, UUNV510, and UUNV520.
USS Universal Broker Utilities 6.2.x	UUBR620	UUNV520 and TUBR520 are prerequisites. Supersedes and deletes FMID UUBR320, UUBR410, UUBR420, UUBR430, UUBR510, and UUBR520.
USS Universal Command Manager 6.2.x	UUCM620	UUNV520 and TUBR520 are prerequisites. Supersedes and deletes FMID UUCM320, UUCM410, UUCM420, UUCM430, UUCM510, and UUCM520.
USS Universal Data Mover Manager 6.2.x	UUDM620	UUNV520 and TUBR520 are prerequisites. Supersedes and deletes FMID UUDM320, UUDM410, UUDM420, UUDM430, UUDM510, and UUDM520.
USS Universal Utilities 6.2.x	UUTL520	UUNV620 is a prerequisite. Supersedes and deletes FMID UUTL320, UUTL410, UUTL420, UUTL430, UUTL510, and UUTL520.
USS Universal Event Monitor Manager 6.2.x	UUEM620	UUNV520 and TUBR520 are prerequisites. Supersedes and deletes FMID UUEM320, UUEM410, UUEM420, UUEM430, UUEM510, and UUEM520.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and

Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

zOS USS Installation - Distribution File

Universal Agent for z/OS USS - Product Distribution File

The Universal Agent for z/OS USS installation files are distributed with the Universal Agent for z/OS distribution file.

zOS USS Installation - SMPE Installation

Universal Agent for z/OS USS - SMP/E Installation

The installation steps describe how to perform the SMP/E install of Universal Agent for z/OS USS in a step-by-step process.

> **Note**
> Installation of z/OS USS is dependent upon completion of the z/OS package installation, and must be installed in the same CSI as the z/OS package.

Four different installation processes are provided. Which installation process used depends on the installation environment.

z/OS USS Installation - New Install, New CSI	Install any Universal Agent package from Stonebranch, Inc. for the first time, or installing a Universal Agent for z/OS USS package in a new SMP/E CSI.
z/OS USS Installation - 5.2.0 Upgrade, Existing CSI	Upgrade a Universal Agent 5.2.0 for z/OS USS package install. In this case, the Universal Agent 6.2.0 for z/OS USS package is installed into an SMP/E CSI that contains a Universal Agent 5.2.0 package.
z/OS USS Installation - 5.1.0 Upgrade, Existing CSI	Upgrade a Workload Automation 5.1.0 for z/OS USS package install. In this case, the Universal Agent 6.2.0 for z/OS USS package is installed into an SMP/E CSI that contains a Workload Automation 5.1.0 package.
z/OS USS Installation - 4.x Upgrade, Existing CSI	Upgrade a Stonebranch Solutions 4.x for z/OS USS package install. In this case, the Universal Agent 6.2.0 for z/OS USS package is installed into an SMP/E CSI that contains a Stonebranch Solutions 4.x package.
z/OS USS Installation - 3.2.0 Upgrade, Existing CSI	Upgrade a Universal Products 3.2.0 for z/OS USS package install. In this case, the Universal Agent 6.2.0 for z/OS USS package is installed into an SMP/E CSI that contains a Universal Products 3.2.0 package.

zOS USS Installation - New Install, New CSI

New Install, New CSI

The New Install, New CSI installation process describes how to install the Universal Agent for z/OS USS package in a newly allocated SMP/E CSI.

Use this installation process for either of these environments:

- Installing a Universal Agent for z/OS USS package for the first time.
- Installing a Universal Agent for z/OS USS package in a different SMP/E CSI than other Universal Agent components.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

The user ID used to run the installation jobs must have a valid OMVS segment defined and have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Step 1	Submit the JCL in member **UNVUN01**. The JCL allocates the SMP/E target and distribution data sets, creates the z/OS UNIX directories, and defines the DDDEFs in the SMP/E zones. All steps must end with a return code 0. Step **ALLOCUSS** creates the z/OS UNIX directories required by the USS components. The path prefix where the directories are created was defined when the **#SETUP** member of the **INSTALL** library was modified and run as part of the z/OS installation. IBM recommends the path prefix to be **/usr/lpp**. Review the path prefix used in the **ALLOCUSS** step PARM value to be sure it meets local requirements. The z/OS UNIX directory where the Universal Agent directories are created must be mounted in read/write mode. The user ID used to run this job must have write access to the directory. If the z/OS UNIX directories already have been created, the **ALLOCUSS** step will indicate this in its report and end successfully.
Step 2	Submit the JCL in member **UNVUN02**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 3	Submit the JCL in member **UNVUN03**. The JCL performs a SMP/E APPLY of the USS product FMIDs and any received PTFs. Step **APYFMID** must end with a condition code of 0. Step **APYPTFS** is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname SMPOUT. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT. The user ID used to run the job must have READ access to **BPX.FILEATTR.APF** profile in the **FACILITY** class in order to set the APF attribute on the UDM file when it is created in the z/OS UNIX directory by the APPLY command.
Step 4	Submit the JCL in member **UNVUN04**. The JCL member performs a SMP/E ACCEPT of the USS product FMIDs and any applied PTFs. Step **ACCFMID** must end with a condition code of 0. Step **ACCPTFS** is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname SMPOUT.
Step 5	Submit the JCL in member **UNVUN05**. The JCL copies sample configuration members to the configuration library. All steps must end with a return code 0.

zOS USS Installation - 5.2.0 Upgrade, Existing CSI

Universal Agent 5.2.0 for z/OS USS Upgrade, Existing CSI

The Universal Agent 5.2.0 for z/OS USS Upgrade, Existing CSI installation process describes how to upgrade an existing Universal Agent 5.2.0 package in an existing SMP/E CSI.

> **Note**
> Before the Universal Agent 5.2.0 for z/OS USS package can be upgraded, you first must upgrade the Universal Agent 5.2.0 for z/OS package to Universal Agent 6.2.0.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

The user ID used to run the installation jobs must have a valid OMVS segment defined and have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Step 1	Submit the JCL in member **UNVUN02**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUN03**. The JCL performs a SMP/E APPLY of the USS product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**. The user ID used to run the job must have READ access to **BPX.FILEATTR.APF** profile in the **FACILITY** class in order to set the APF attribute on the UDM file when it is created in the z/OS UNIX directory by the APPLY command.
Step 3	Submit the JCL in member **UNVUN04**. The JCL member performs a SMP/E ACCEPT of the USS product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVUN05**. The JCL copies sample configuration members to the configuration library. All steps must end with a return code 0.

zOS USS Installation - 5.1.0 Upgrade, Existing CSI

Workload Automation 5.1.0 for z/OS USS Upgrade, Existing CSI

The Workload Automation 5.1.0 for z/OS USS Upgrade, Existing CSI installation process describes how to upgrade an existing Workload Automation 5.1.0 package in an existing SMP/E CSI.

> **Note**
> Before the Workload Automation 5.1.0 for z/OS USS package can be upgraded, you first must upgrade the Workload Automation 5.1.0 for z/OS package to Universal Agent 5.2.0.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

The user ID used to run the installation jobs must have a valid OMVS segment defined and have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Step 1	Submit the JCL in member **UNVUN02**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUN03**. The JCL performs a SMP/E APPLY of the USS product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**. The user ID used to run the job must have READ access to **BPX.FILEATTR.APF** profile in the **FACILITY** class in order to set the APF attribute on the UDM file when it is created in the z/OS UNIX directory by the APPLY command.
Step 3	Submit the JCL in member **UNVUN04**. The JCL member performs a SMP/E ACCEPT of the USS product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVUN05**. The JCL copies sample configuration members to the configuration library. All steps must end with a return code 0.

zOS USS Installation - 4.x Upgrade, Existing CSI

Stonebranch Solutions 4.x for z/OS USS Upgrade, Existing CSI

The Stonebranch Solutions 4.x for z/OS USS Upgrade, Existing CSI installation process describes how to upgrade an existing Stonebranch Solutions 4.x package in an existing SMP/E CSI.

> **Note**
> Before the Stonebranch Solutions 4.x for z/OS USS package can be upgraded, you first must upgrade the Stonebranch Solutions 4.x for z/OS package to Universal Agent 5.2.0.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

The user ID used to run the installation jobs must have a valid OMVS segment defined and have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Step 1	Submit the JCL in member **UNVUN02**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUN03**. The JCL performs a SMP/E APPLY of the USS product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**. The user ID used to run the job must have READ access to **BPX.FILEATTR.APF** profile in the **FACILITY** class in order to set the APF attribute on the UDM file when it is created in the z/OS UNIX directory by the APPLY command.
Step 3	Submit the JCL in member **UNVUN04**. The JCL member performs a SMP/E ACCEPT of the USS product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVUN05**. The JCL copies sample configuration members to the configuration library. All steps must end with a return code 0.

zOS USS Installation - 3.2.0 Upgrade, Existing CSI

Universal Products 3.2.0 for z/OS USS Upgrade, Existing CSI

The Universal Products 3.2.0 for z/OS USS Upgrade, Existing CSI installation process describes how to upgrade an existing Universal Products 3.2.0 package in an existing SMP/E CSI.

> **Note**
> Before the Universal Products 3.2.0 for z/OS USS package can be upgraded, you first must upgrade the Universal Products 3.2.0 for z/OS package Universal Agent 5.2.0.

The installation JCL referenced by the installation steps is created by the **#SETUP** member in the Universal Agent **INSTALL** library.

Each step consists of running a batch job. The batch job must end with the appropriate return code before proceeding to the next step.

The user ID used to run the installation jobs must have a valid OMVS segment defined and have READ access to the **BPX.FILEATTR.APF** profile in the **FACILITY** class.

Step 1	Submit the JCL in member **UNVUN02**. The JCL performs an SMP/E RECEIVE of the product FMIDs and available PTFs from the distribution data sets. All steps must end with a return code 0.
Step 2	Submit the JCL in member **UNVUN03**. The JCL performs a SMP/E APPLY of the USS product FMIDs and any received PTFs. Step APYFMID must end with a condition code of 0. Step APYPTFS is considered successful under any of the following conditions: • Step ends with condition code 0. • Step ends with condition code 4, and message GIM42001W is written in ddname **SMPOUT**. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**. The user ID used to run the job must have READ access to **BPX.FILEATTR.APF** profile in the **FACILITY** class in order to set the APF attribute on the UDM file when it is created in the z/OS UNIX directory by the APPLY command.
Step 3	Submit the JCL in member **UNVUN04**. The JCL member performs a SMP/E ACCEPT of the USS product FMIDs and any applied PTFs. Step ACCFMID must end with a condition code of 0. Step ACCPTFS is considered successful under any of the following conditions: • Step ends with condition code 0 or 4. • Step ends with condition code 12, and message GIM24801S is written in ddname **SMPOUT**.
Step 4	Submit the JCL in member **UNVUN05**. The JCL copies sample configuration members to the configuration library. All steps must end with a return code 0.

zOS USS Installation - Customization

- Overview
- Universal Command Manager for z/OS USS Customization
  - Configuration
- Universal Control Manager for z/OS USS Customization
  - Configuration
- Universal Data Mover Manager for z/OS USS Customization
  - Configuration
- Universal Event Monitor Manager for z/OS USS Customization
  - Configuration
- Universal Query for z/OS USS Customization
  - Configuration

Overview

The product executable files intended for command line use are located in the directory **#USSPRE/universal/bin**, where **#USSPRE** is the path prefix where the USS component directories were created. This directory must be added to the PATH environment variable for intended users of the executable files.

(For information on applying product licenses to installed Universal Agent for z/OS USS components, see z/OS USS Installation - Licensing.)

Universal Command Manager for z/OS USS Customization

Configuration

USS Universal Command Manager for z/OS uses a configuration file for system-wide customizations. The configuration file is member **UCMCFGU0** of the **UNVCONF** library.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Manager.

Universal Control Manager for z/OS USS Customization

Configuration

USS Universal Control Manager for z/OS uses a configuration file for system-wide customizations. The default configuration file is member **UCTCFGU0** of the **UNVCONF** library.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Manager.

Universal Data Mover Manager for z/OS USS Customization

Configuration

USS Universal Data Mover Manager for z/OS uses a configuration file for system-wide customizations. The default configuration file is member **UDMCFGU0** of the **UNVCONF** library.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Data Mover.

Universal Event Monitor Manager for z/OS USS Customization

Configuration

USS Universal Event Monitor Manager for z/OS uses a configuration file for system-wide customizations. The default configuration file is member **UEMCFGU0** of the **UNVCONF** library.

See the Universal Event Monitor 6.2.x Reference Guide for details on configuring Universal Event Monitor Manager.

Universal Query for z/OS USS Customization

Configuration

USS Universal Query z/OS uses a configuration file for system-wide customizations. The default configuration file is member **UQRCFGU0** of the **UNVCONF** library.

See Universal Query for details on configuring Universal Query.

zOS USS Installation - Data Set Inventory

SMP/E Data Sets

A number of SMP/E data sets are allocated and cataloged as part of the Universal Agent for z/OS USS installation.

The following table lists the data sets and their space requirements.

Depending on your installation choices, the data set high-level qualifiers may be different.

Data Set Name	Space (tracks)	Description
UNV.AUNVHBIN	3000	SMP/E distribution library for z/OS UNIX executables.
UNV.AUNVHNLS	30	SMP/E distribution library for z/OS UNIX product national language support files.

z/OS UNIX Directories

A number of z/OS UNIX directories are created as part of the Universal Agent for z/OS USS installation.

The following table lists the directories and their space requirements.

Directory Name	Space (MB)	Description
universal/bin/IBM	80	SMP/E target directory for the SUNVHBIN DDDEF.
universal/bin	0	USS Universal Agent program files.
universal/nls/IBM	0.5	SMP/E target directory for the SUNVHNLS DDDEF.
universal/nls	0	USS Universal Agent national language support files.
universal/ucmdmgr	0	Universal Command Manager installation directory.
universal/uctlmgr	0	Universal Control Manager installation directory.
universal/udmmgr	0	Universal Data Mover Manager installation directory.
universal/uemmgr	0	Universal Event Monitor Manager installation directory.
universal/uquery	0	Universal Query installation directory.

zOS USS Installation - Licensing

- Licensing Universal Agent for z/OS USS Components
- Product License File
  - Format
  - Sample
- Entering License Information
- Restart Universal Broker

Licensing Universal Agent for z/OS USS Components

After Universal Agent for z/OS USS has been installed, you must configure the following components with product licenses before they can be used:

- Universal Command Manager
- Universal Data Mover Manager

Product License File

For each component, product license information (license parameter keywords and their values) is contained in a separate text file provided by your Stonebranch, Inc. account representative.

Note
Product license information for Universal Agent for z/OS USS components are contained in the Universal Agent for z/OS product license files.

Format

The format of the product license file name is: <*component name*>_<*customer name*>_<*operating system*>_<*schedule or solution*>.txt. For example: ***Indesca_Stonebranch_MVS_A1.txt***

- For Universal Command Manager, **Indesca** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.
- For Universal Data Mover Manager, **Infitran** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.

Sample

The following is a sample Universal Agent for z/OS product license file (for Universal Command Manager):

```
License_Product "INDESCA"
License_Customer "STONEBRANCH"
License_OS_Type "MVS"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

Enter each component's product license parameters into its configuration file in the **UNVCONF** product library:

- Universal Command Manager configuration file: member **UCMCFGU0**.
- Universal Data Mover Manager configuration file: member **UDMCFGU0**.

It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration options.)

If the Universal Broker STC is running, restart the Universal Broker STC to pick up the configuration file changes.

Restart Universal Broker

For Universal Broker to read the license information, you must stop and restart it.

Stop Universal Broker	`STOP UBROKER`
Start Universal Broker	`START UBROKER[,UPARM='options']`

Windows Installation

- Introduction
- Installation Packages
  - Distribution File Format
- Pre-Installation Guidelines
  - Installation Paths
- Installation Summary
- Windows Installer
  - Windows Installer Interfaces
  - Installing Windows Installer with a Universal Agent Package
  - Installing Windows Installer Separately from a Universal Agent Package
  - Windows Installer Package File Locations
- Detailed Information

Introduction

These pages provide information on the installation of Stonebranch, Inc.'s Universal Agent on Microsoft Windows operating systems. Unless otherwise specified, all references to Universal Agent for Windows refer to version 6.2.0.

Installation Packages

Universal Agent is installed with one required installation package and three optional installation packages.

To obtain a Universal Agent package, you must download the corresponding product distribution file from the Stonebranch Customer Portal.

> **Note**
> A customer user ID and password - provided by Stonebranch, Inc. - are required to access this area.

The following table lists the packages available, their supported platforms, distribution file names, and whether or not the package is required.

Package	Distribution File	Required
Universal Agent	`sb-6.2.0.`*`LEVEL`*`-windows-i386.exe` (32-bit) **OR** `sb-6.2.0.`*`LEVEL`*`-windows-x64.exe` (64-bit)	Yes
Universal Enterprise Controller	`sb-UECtlr-6.2.0.`*`LEVEL`*`-windows-i386.exe`	No
Universal Enterprise Controller Client Applications	`sb-UEClient-6.2.0.`*`LEVEL`*`-windows-i386.exe`	No
Universal Agent for SOA	`sb-soa-6.2.0.`*`LEVEL`*`-windows-i386.exe`	No

The word *`LEVEL`* in each distribution file name is a numeric value representing the product maintenance level contained in the distribution file.

Distribution File Format

The product distribution files are in a Windows self-extracting archive file format.

The names of the distribution files have the following format:

`PROD``-`*`VRML`*`-`*`OS`*`-`*`HW`*`.exe`

- *`PROD`* is the package content
- *`VRML`* is the version, release, modification, and maintenance levels (for example, 6.2.0.1)
- *`OS`* is the name of the operating system supported
- *`HW`* is the hardware type supported

Pre-Installation Guidelines

Before starting any Universal Agent for Windows installation, it is recommended that you close all other Windows programs. Active programs may be using system files that the Universal Agent installation may update.

If a system file is in use during the installation, it is placed in a temporary location. A reboot is required to free the file and move it from the temporary location to its final destination. Closing all programs before the installation reduces the possibility that a file will be in use and that you will have to reboot your computer.

> You must be able to write to the directory from which the installation is launched.

Installation Paths

32-Bit Systems

On 32-bit Windows systems, the default installation path is:

- "`C:\Program Files`"

64-Bit Systems

On 64-bit Windows systems, the default installation paths are:

- "`C:\Program Files (x86)`" (for the 32-bit Universal Agent for Windows package)
- "`C:\Program Files`" (for the 64-bit Universal Agent for Windows package)

Installation Summary

Step 1	Download the distribution file from the Stonebranch Customer Portal.
Step 2	Log on to Windows using an account with the privileges noted above.
Step 3	Exit all running programs to reduce the likelihood of a reboot.
Step 4	Execute the installation file.
Step 5	Follow installation instructions: • Installing Universal Agent via the Graphical Interface • Installing Universal Agent via the Command Line

Windows Installer

The installation of each Universal Agent for Windows package requires Windows Installer (**msiexec.exe**) from Microsoft. Windows Installer is a service application that is a standard component of Windows operating systems. (It also can be obtained from Microsoft's website.)

Each Universal Agent for Windows package provides version **3.1.4000.2435** of Windows Installer.

Windows Installer Interfaces

Windows Installer provides two interfaces that you can use to install Universal Agent:

- Graphical user interface
- Command line

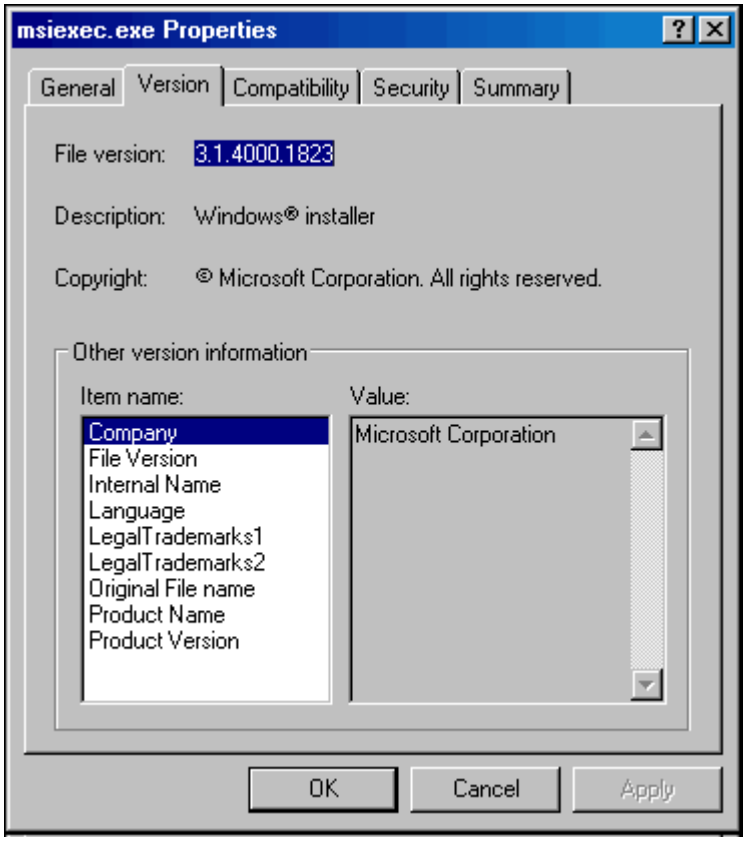
Installing Windows Installer with a Universal Agent Package

If Windows Installer is not present on your Windows operating system, the Universal Agent package being installed will install it before any of its own files are copied.

If Windows Installer is present on your Windows operating system, but it is a version prior to **3.1.4000.2435**, the installation will upgrade it.

Determining if Windows Installer will be Installed or Upgraded

To determine if the installation of a Universal Agent package also will install or upgrade Windows Installer, perform the following steps:

Step 1	From Windows Explorer, go to the 32-bit Windows system directory (for example, **\Windows\System32, \Windows\SysWow64**).
Step 2	Search for file **msiexec.exe** (Windows Installer). • If the file exists, continue to the Step 3. • If the file does not exist, Windows Installer is not installed on your system. The Universal Agent installation will install it.
Step 3	Right-click on the **msiexec.exe** file name to display a pop-up menu.
Step 4	Click **Properties** to display the Properties dialog for **msiexec.exe**.
Step 5	Click the **Version** tab. File Version: identifies the currently installed version of Windows Installer (see the following figure). If the version is prior to **3.1.4000.2435**, the package installation will upgrade it.  (In this figure, the installed Windows Installer is identified as version **3.1.4000.1823**. Since this is the earlier version than the version provided by the package installation, it will be upgraded.)

Rebooting the System

If Windows Installer is installed or upgraded during the installation of the package, a reboot of the system is required.

The installation provides an option to either:

- Reboot after the installation is complete.
- Reboot immediately. The installation will resume automatically after Windows restarts.

If Windows Installer version **3.1.4000.2435** or above already exists on your system, it is not upgraded, and no reboot is required.

Installing Windows Installer Separately from a Universal Agent Package

The Windows Installer version provided with each Universal Agent package also can be installed separately from the package; that is, prior to the

installation of the package.

To install Windows Installer separately, perform the following steps:

Step 1	Execute the package distribution file downloaded from the Stonebranch Customer Portal. Make note of the directory into which the installation files are extracted. After extraction of the files is complete, the installation will begin.
Step 2	Cancel the installation.
Step 3	Go to the directory where the extracted files reside.
Step 4	Execute **WindowsInstaller-KB893803-x86.exe**. • If Windows Installer was upgraded, reboot the system. • If Windows Installer was installed new, no reboot is required.

Windows Installer Package File Locations

When you install a Universal Agent for Windows package, the Windows Installer package file (***.msi**) is extracted to the following locations:

Package	Location
Universal Agent for Windows package (32-bit)	`<LocalAppData>\StonebranchWorkloadAutomation\<packagecode>\Ucmd.msi`
Universal Agent for Windows package (64-bit)	`<LocalAppData>\StonebranchWorkloadAutomationx64\<packagecode>\Ucmdx64.msi`
Universal Enterprise Controller for Windows package	`<LocalAppData>\StonebranchWorkloadAutomation\<packagecode>\UECtlr.msi`
UEC Client Applications package	`<LocalAppData>\StonebranchWorkloadAutomation\<packagecode>\UEClient.msi`
Universal Agent for SOA for Windows Package	`<LocalAppData>\StonebranchWorkloadAutomation\<packagecode>\UPforSOA.msi`

In these paths:

`<LocalAppData>`	User's local Application Data folder. For example: If the installation was performed by the built-in Administrator account, **`<LocalAppData>`** would expand - by default - to: • `C:\Documents and Settings\Administrator\Local Settings\Application Data` on Windows XP and Server 2003. • `C:\Users\Administrator\AppData\Local` on Windows Vista, 7, Server 2008, and Server 2012.
`<packagecode>`	Universally Unique Identifier (UUID) in the format {XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}, where X is a hexadecimal character in the range 0 (zero) - F. For example: • {3B10285A-9602-4DC9-B0A5-4D701BEB5225}.

Detailed Information

The following pages provide detailed information for Windows Installation:

- Universal Agent for Windows Installation
- Universal Enterprise Controller for Windows Installation
- UEC Client Applications Installation
- Universal Agent for SOA for Windows Installation
- Licensing

Universal Agent for Windows Installation

Introduction

The following information is provided for the installation of Universal Agent for Windows:

- Installation Package
- Installation Requirements
- Installation Procedures
- 32-Bit Universal Agent for Windows on 64-Bit Windows Systems
- File Inventory Lists

(For licensing information, see Windows Installation - Licensing.)

Universal Agent for Windows - Installation Package

Components

The Universal Agent for Windows package includes the following product components:

- Universal Broker 6.2.x
- Universal Automation Center Agent 6.2.x
- Universal Certificate 6.2.x
- Universal Command Manager and Server 6.2.x
- Universal Connector 6.2.x
- Universal Control Manager and Server 6.2.x
- Universal Controller Command Line Interface (CLI) 6.2.x
- Universal Data Mover Manager and Server 6.2.x
- Universal Encrypt 6.2.x
- Universal Event Log Dump 6.2.x
- Universal Event Monitor Manager and Server 6.2.x
- Universal Message Service (OMS) 6.2.x
- Universal Message to Exit Code Translator 6.2.x
- Universal Query 6.2.x

Note
Universal Enterprise Controller 6.2.x, Universal Enterprise Controller 6.2.x Client Applications, and Universal Agent for SOA 6.2.x are packaged separately (see Universal Enterprise Controller for Windows - Installation Package, UEC Client Applications - Installation Package, and Universal Agent for SOA for Windows - Installation Package).

Component Compatibility

The following table identifies the compatibility of Universal Agent for Windows components with previous component / product versions.

Component	Compatibility
Universal Broker 6.2.x	Universal Agent / Workload Automation / Stonebranch Solutions / Universal Products releases 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, 2.1.0, and 1.2.0.
Universal Command 6.2.x	Universal Command 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, 2.1.0, and 1.2.0.
Universal Control 6.2.x	Universal Control 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, 2.1.0, and 1.2.0.
Universal Data Mover 6.2.x	Universal Data Mover 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1 and 3.1.0.
Universal Encrypt 6.2.x	Universal Encrypt 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, 2.1.0, and 1.2.0.
Universal Query 6.2.x	Universal Broker 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, 2.1.0, and 1.2.0.
Universal Event Monitor 6.2.x	Universal Event Monitor 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, and 3.1.0.

The component references pertain to all supported platforms for that version.

Universal Agent for Windows - Installation Requirements

- Windows Versions
- Additional Requirements
- Platform Requirements
- Installation Account
- Universal Broker Service
  - UBrokerService: The Default Universal Broker Service Account
  - Using a Windows Domain Account to Execute Universal Broker
- Spool Directory
  - Universal Command Server
  - UAG Cache

Windows Versions

To install Universal Agent for Windows, you must have one of the following versions of Windows:

- Windows Server 2003 SP1 and higher
- Windows Server 2003 R2
- Windows XP SP3 (for 32-bit package only)
- Windows Vista
- Windows 7
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012 (64-bit Windows package only)
- Windows Server 2012 R2 (64-bit Windows package only)

Note
The 64-bit Universal Agent for Windows (x64) package requires a 64-bit edition of the Windows versions listed above (except where noted).

The 32-bit Universal Agent for Windows (x86) package can be installed on any of Windows versions listed above.

Additional Requirements

- An account with administrative privileges.
- Possible reboot: a reboot is required if the Windows Installer service is not installed, a version of the Windows Installer prior to 3.1.4000.1823 is installed, or if required files are in use at the time of the installation.
- TCP/IP.
- About 325 megabytes of disk space.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

Installation Account

The account used to install the package must have administrative privileges.

Universal Broker Service

The Universal Broker service runs either with an Administrative account or with the Local System account.

If you are using an Administrative account, the account must have the following privileges:

- Act as part of the operating system
- Adjust memory quotas for a process
- Bypass traverse checking
- Debug programs
- Deny access to this computer from the network
- Deny log on as batch job
- Deny log on locally
- Deny log on through Terminal Services

- Log on as a service
- Impersonate a client after authentication
- Increase scheduling priority
- Replace a process level token
- Take ownership of files and other objects

UBrokerService: The Default Universal Broker Service Account

The Universal Agent for Windows installation will create an account with the privileges listed above, if the account does not already exist. The name of this account is **`UBrokerService`**. A default password provided for the account can be obtained by contacting Stonebranch support. We strongly suggest that you set this password at install time via the installation dialogs or via the **BROKERPWD** command line option.

> **Note**
> If the UBrokerService account already exists, its password cannot be changed from the install.

See Installing Universal Agent via the Graphical Interface for detailed information about setting up the Universal Broker service account at install time.

Using a Windows Domain Account to Execute Universal Broker

The Universal Broker Service may be configured to execute with an existing Windows domain account, provided that account has been granted the privileges listed in this section, above, on the local system and the account has the file system permissions described in Service Security.

Spool Directory

The spool directory is used to store the following types of information:

- Execution information for Universal Agent components started by Universal Broker.
- Event definitions and event handlers managed by Universal Broker and used by Universal Event Monitor.
- Results of events tracked by Universal Event Monitor.
- Redirected standard I/O files (stdin, stdout, and stderr) captured by Universal Command when run with Manager Fault Tolerance enabled.
- Configuration information for Universal Agent components, when a local Universal Broker is operating in managed mode.

The default location for the spooled standard I/O files is **`.\Universal\spool\ucmdsrv`**.

The default location for the other database files is **`.\Universal\spool\ubroker`**.

Universal Command Server

Location

The spool directory must reside on a local device. It cannot reside on any network device, including network drives that may be mapped to a local drive. By default, the spool files are located in directory **`.\Universal\spool\ucmdsrv`**.

Space

You must have approximately 50 megabytes of disk space for the installation.

The amount of disk space required for the spool directory depend on the following factors:

- Number of spooling user processes that will be executing simultaneously. A user process is created for each command requested by a Universal Command Manager. The default maximum number is 50.
- When a user process ends and a Manager has received all the spool files, the spool files themselves are deleted.
- Average size of the user processes standard input, output, and error files. Keep in mind that spooling is not a feature for file transfer purposes. File transfer-related processes should execute without spooling enabled.

When these numbers have been determined, the average amount of disk space is calculated with the following formula:

`MAX-PROCESSES x AVERAGE-STDIO-SIZE x 2 = required disk space.`

For example, if the maximum number of simultaneous user processes is estimated at 20 and the average size of processes standard I/O files is 100,000 bytes, the average amount of required disk space is 4MB (20 x 100000 x 2).

The Universal Command Server is configured with spooling disabled to prevent unintentional disk utilization. This feature can be enabled through the Universal Configuration Manager.

For more information on the Manager Fault Tolerant feature, spooling of redirected standard I/O files, and Universal Configuration Manager, see the Universal Agent 6.2.x User Guide.

Security

Universal Broker and Universal Command Server require read/write access to the spool directory. No other Universal Agent components access the spool directory directly.

UAG Cache

UAG cache is used by Universal Automation Center Agent (UAG) for storing standard I/O files.

Space

Cache files are located, by default, in directory **`.\Universal\UagSrv\cache`**.

Cache files are created for each job that is run by Universal Agent. They remain in the cache until they are purged by an automated purge process scheduled nightly by Universal Controller. You can configure the number of days that files remain in the cache using the Agent Cache Retention Period in Days Universal Controller system property.

The amount of disk space required for the cache directory depends on:

1. Number of jobs you estimate will run during the cache retention period you specified.
2. These files remain until they are purged by the automated cache purge process scheduled by the Controller daily at midnight.
3. Average size of the user processes standard output and error files.

When these numbers have been determined, the average amount of disk space is calculated with the following formula:

`(RETENTION PERIOD x MAX-JOBS) x (AVERAGE-STDOUT-SIZE + AVERAGE-STDERR-SIZE) = required disk space.`

For example:

If the files are purged every 7 days, and you run 1200 jobs on that agent server daily, and the average size of the STDOUT + STDERR files is 3,000 bytes, the average amount of required disk space is 25MB (7 x 1,200 x 3000).

UAG automatically redirects the standard output and standard error files to the cache directory with no required user input.

Security

Universal Automation Center Agent (UAG) requires read/write access to the UAG cache directory. No other Universal Agent components access the cache directory. No general user access is required.

Universal Agent for Windows - Installation Procedures

Installation Procedures

The following procedures are provided for the installation and modification of Universal Agent for Windows:

- Installing Universal Agent via the Graphical Interface
- Modifying a Universal Agent Installation via the Graphical Interface
- Installing Universal Agent via the Command Line
- Modifying a Universal Agent Installation via the Command Line
- Migrating between 32- and 64-bit Universal Agent for Windows Installs

Note
Modifying an Agent installation refers to the adding / removing of Agent components, repairing a corrupted installation, or removing an installation. To change the installed version of an Agent, see Upgrading Universal Agent and Applying Maintenance to Universal Agent.

Installing Universal Agent via the Graphical Interface

- Installing Universal Agent for Windows via the Windows Installer Graphical Interface
  - Welcome Dialog (32-bit Install)
  - Welcome Dialog (64-bit Install)
- Windows Installer Package File Locations
  - Examples
    - Universal Agent for Windows Package (32-bit)
    - Universal Agent for Windows Package (64-bit)

Installing Universal Agent for Windows via the Windows Installer Graphical Interface

To install Universal Agent for Windows using the Windows Installer graphical interface, perform the following steps:

Step 1

Download the desired Universal Agent for Windows product distribution file to your work station:

- **`sb-6.2.0.<level>-windows-i386.exe`**, the 32-bit Universal Agent for Windows distribution file.
- **`sb-6.2.0.<level>-windows-x64.exe`**, the 64-bit Universal Agent for Windows distribution file (for supported 64-bit versions of Windows only).

Step 2

Execute the distribution file from the command line to begin the installation process, which first determines whether a Windows Installer update is needed (see Windows Installer).

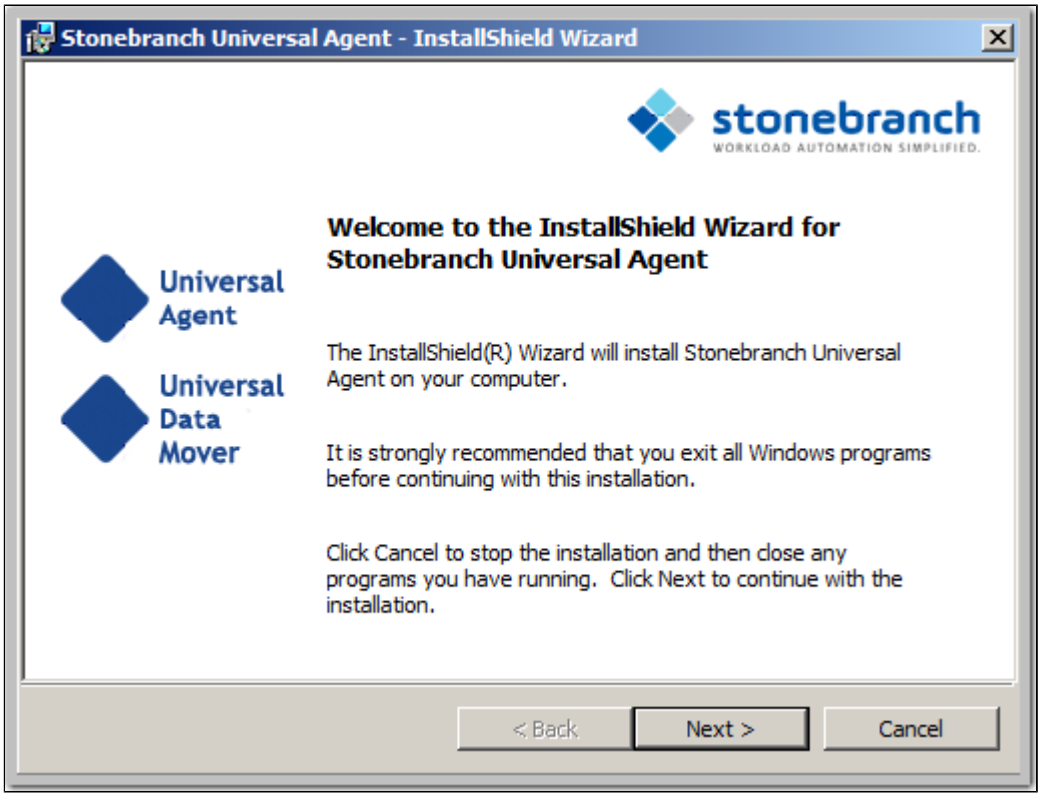
The process then extracts and saves a Windows installer package file (.msi) to one of these locations.

Step 3

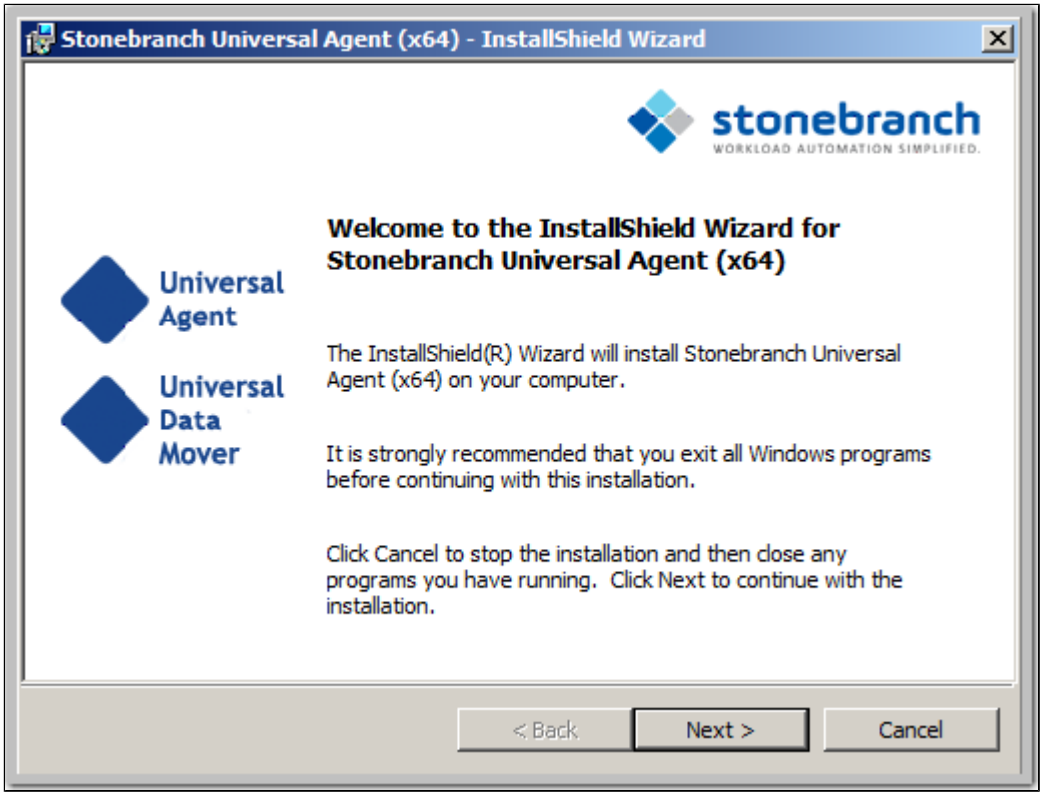
The installation starts after the files are extracted. It first will verify that your machine meets the minimum system requirements (see Installation Requirements and Summary - Windows). If the requirements are met, a Welcome dialog displays.

> **Note**
> The same steps below apply regardless of whether you are performing a 32- or 64-bit Universal Agent for Windows installation. The only difference is the **(x64)** label shown for 64-bit installs.
>
> The Welcome dialog from each install is shown below, but for simplicity, the remaining steps show only the 32-bit install dialogs.

Welcome Dialog (32-bit Install)

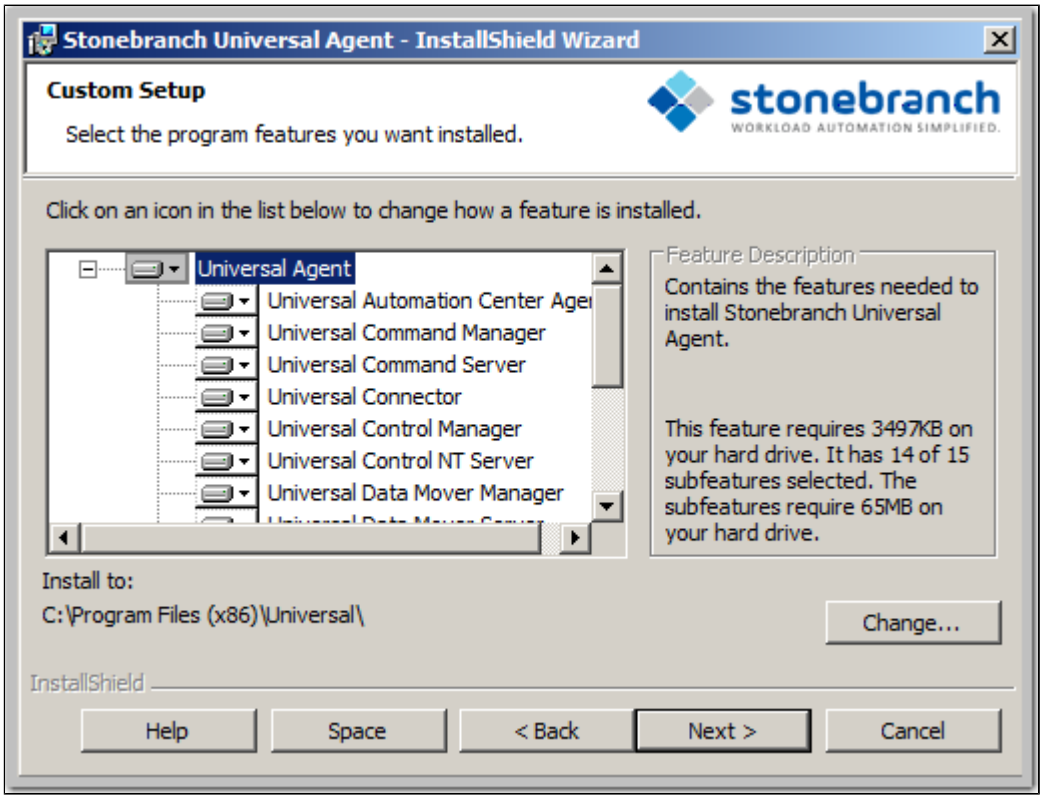


Welcome Dialog (64-bit Install)



Step 4

Click the **Next** button. A list of Universal Agent components included in the installation package then displays. It is from this list that you can select which components to install.



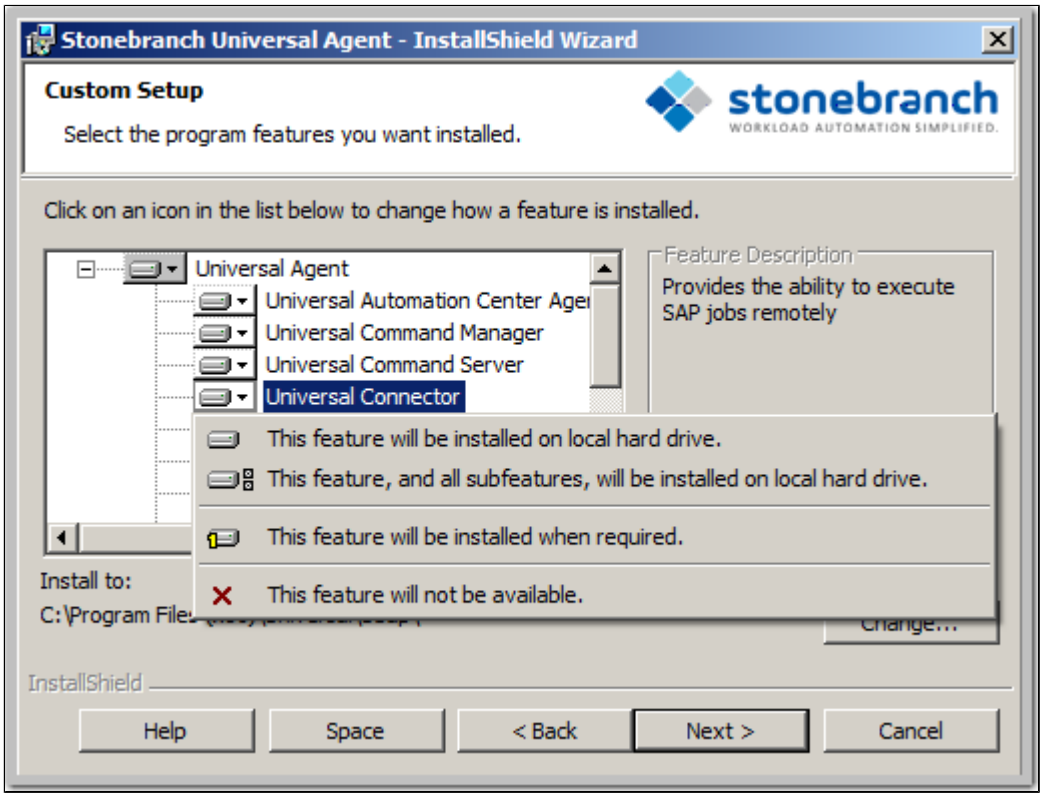
For a new installation, a drive icon displays next to each component in the list, indicating that the component will be installed.

> **Note**
> An **X** icon displays next to the Universal Controller Command Line Interface (CLI) component, indicating that, by default, it will not be installed. If you want to install the CLI, you must click the **X** icon and select the drive icon.

For an upgrade installation, either of the following icons displays next to an item:

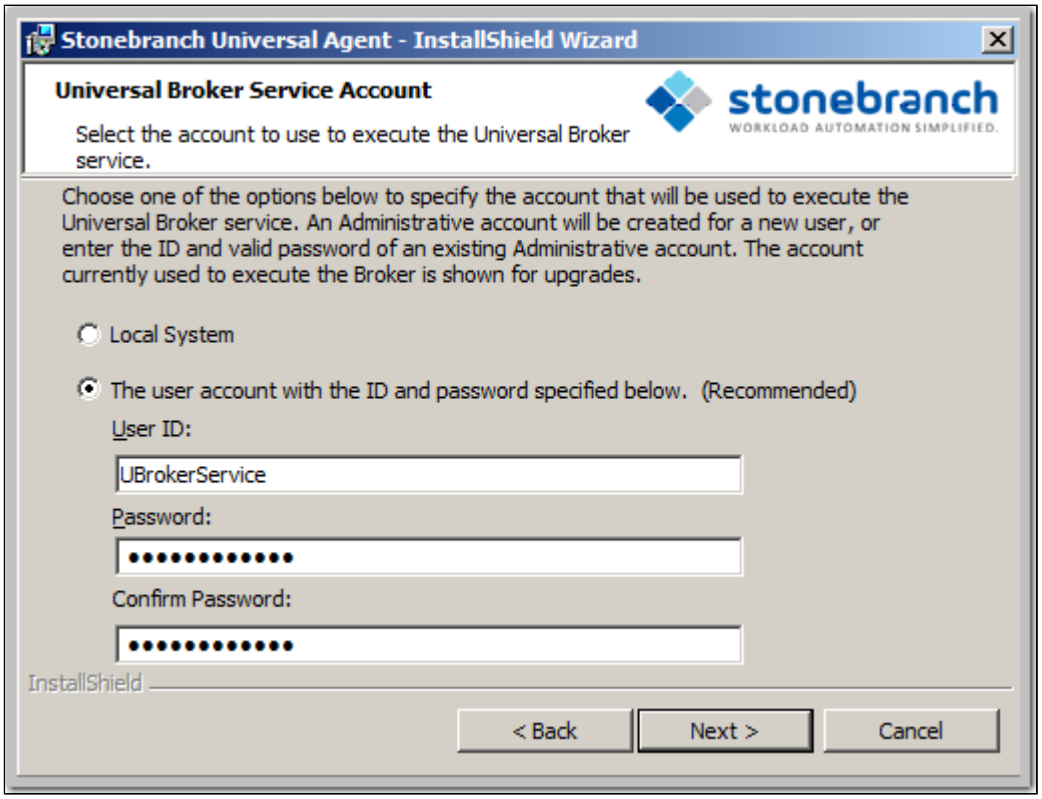
- A drive icon indicates that the component is either:
  - New to the installation and will be installed.
  - Currently is installed and will be upgraded.
- An **X** icon indicates that the component is either:
  - Currently not installed (but previously was available).
  - Previously installed but removed.

> **A Stonebranch Tip**
>
> - If the installation detects an existing Universal Agent installation, currently installed components may be upgraded. (Currently, there is no way to specify that the state of a currently installed component should remain unchanged.)
> - If a component is selected for installation, and the version of the installed component is less than the version of the component being installed, the installed component will be replaced by the component being installed.
> - If a component is not selected for installation (that is, the **X** icon is selected), and it currently is installed, the new installation will remove that component.

Step 5	The previous figure indicates that all Universal Agent components will be installed in their respective directories under the **C:\Program Files\Universal** directory. • If you want to select a different location, click the **Change...** button. • If you want to check the amount of disk space required for the installation, and the amount of available disk space on the selected directory, click the **Space** button.
Step 6	If you do not want to install a component: 1. Click the drive icon next to that component name. 2. From the drop-down list that displays, select the **X** icon to mark the component as one not to be installed. For example, the following figure indicates that Universal Connector has been selected to not be installed. 

Step 7

When you have selected the components (and their installation destinations) that you want to install, click the **Next>** button. The Universal Broker Service Account dialog then displays.



Step 8

Select an account to use to execute the Universal Broker service:

- **Local System** (This is the default for upgrades from releases prior to 4.3.0.1.)
- An Administrative account capable of executing the Universal Broker service. (This is the default for new installs or upgrades where the Universal Broker currently is executing with an account other than **Local System**.)

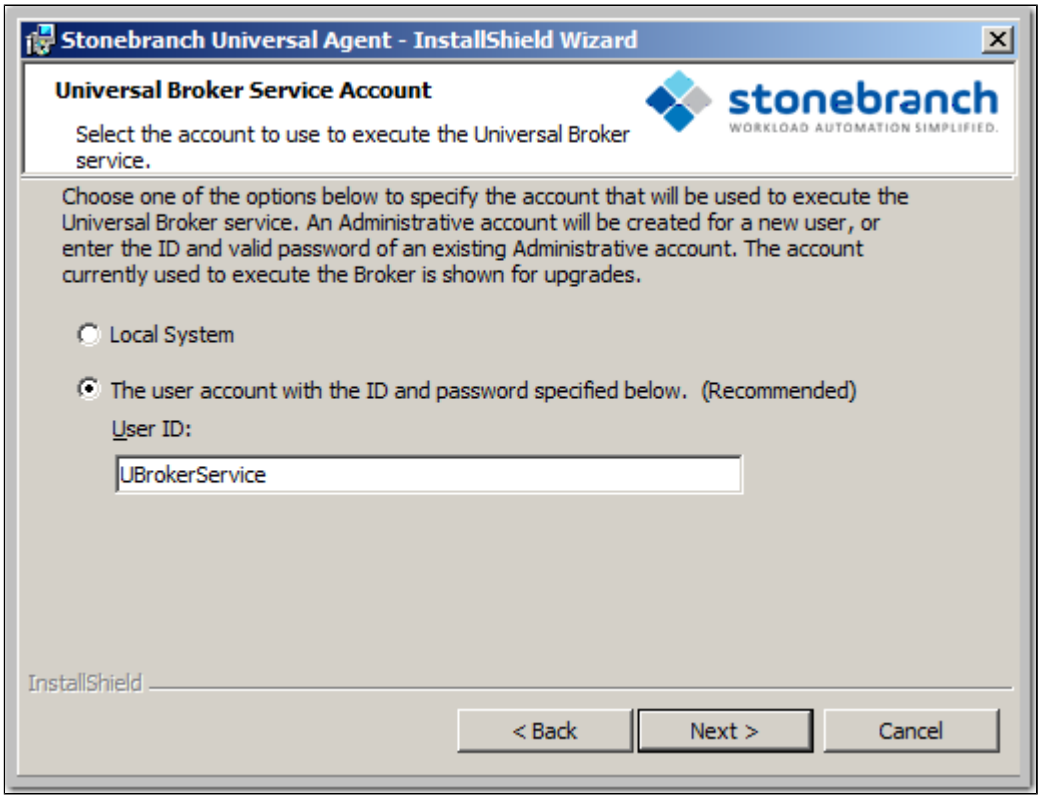
  The user account ID defaults to **UBrokerService**, although any valid user name (up to 20 characters) can be specified.

  If the account does not already exist, the install will create it with the privileges necessary to execute the Universal Broker service. See Starting Universal Broker for the privileges that an account needs to run the Universal Broker service.

  A default password provided for the account can be obtained by contacting Stonebranch support, although any password up to 256 characters can be specified.
  - If the user account does not exist, its password is set to the value entered here.
  - If the user account does exist, the installation process will validate the specified password. You must enter a valid password for the account to continue with the install.

The Broker service may also be run using a Windows domain account. The primary difference between using a local and domain account is that the domain account must already exist, and the installation will not configure it like it does a new local account. Required account privileges and file system access can be found in Service Security.

> **Note**
> For upgrades, if the Universal Broker service is already configured to run with a user account, the install dialog shows only the account name and omits the password fields. The Broker service's configuration in the Windows Service Control Manager is retained across upgrades, making it unnecessary to re-enter account information during an upgrade.



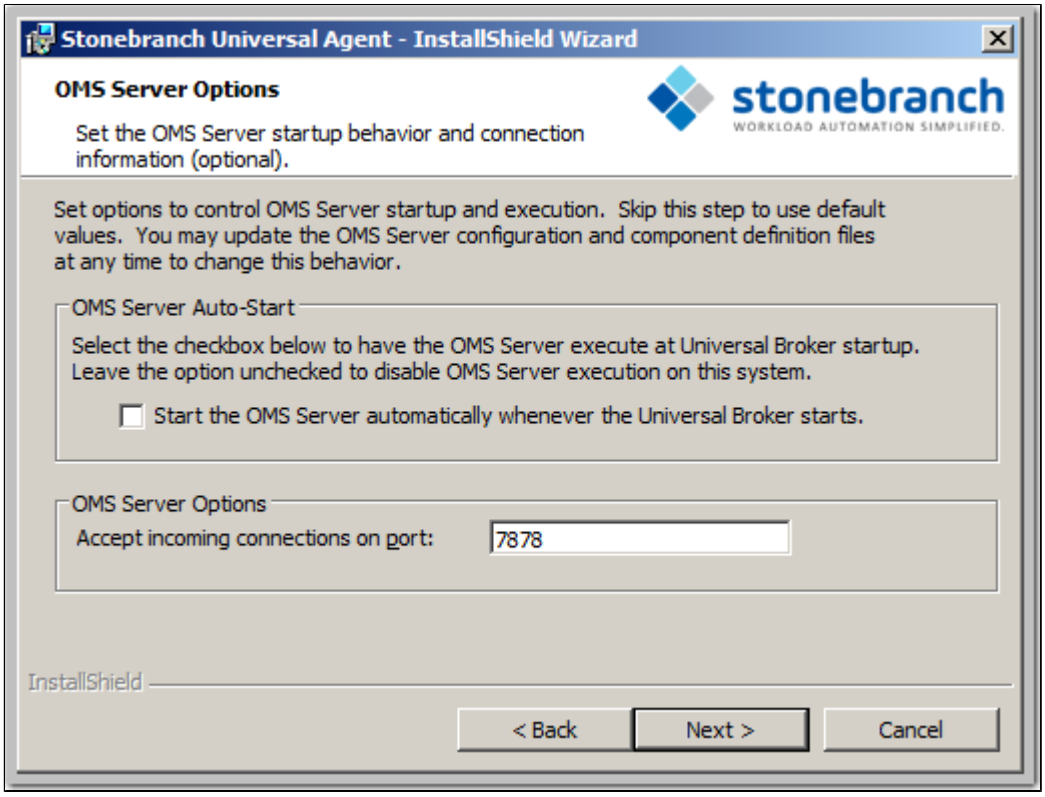
The install provides the RUNBROKERASUSER, BROKERUID, and BROKERPWD command line options to override the default behavior described above.

- Specifying RUNBROKERASUSER=0 causes the install to select Local System, regardless of how the Broker is currently configured to execute.
- Specifying RUNBROKERASUSER=1 causes the install to select the user account option, even if the Broker is currently configured to run as Local System.

When RUNBROKERASUSER=1, the BROKERUID and BROKERPWD command line options are provided to override the default or currently configured user ID and password. The BROKERUID option limits user IDs to 20 characters; the BROKERPWD option limits passwords to 14 characters. (See Windows Installer Command Line Parameters for more information).

Step 9

When you have selected an account, click the **Next>** button. The OMS Server Options dialog then displays.



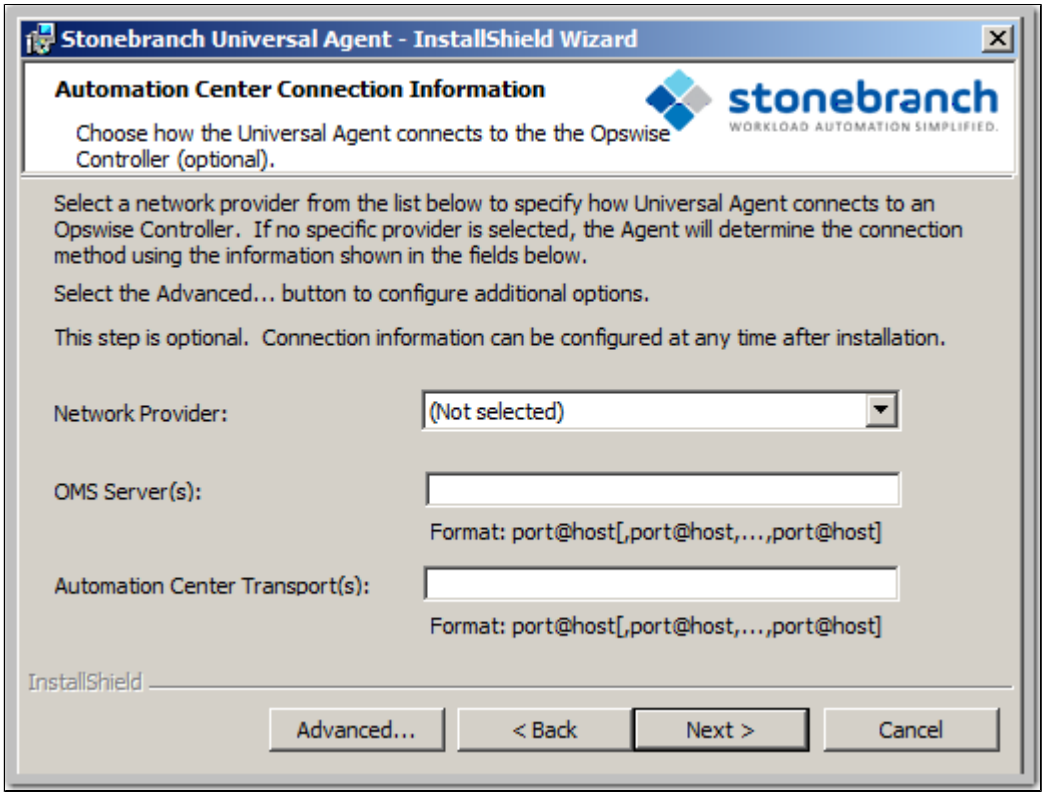
Note
This dialog displays only for new Universal Agent installs, upgrades to Universal Agent from earlier releases, and 32/64-bit Universal Agent crossgrades.

It does not display if you selected, in Step 6, to not install Universal Message Service (OMS), or for upgrades from Universal Agent to a later version.

Step 10

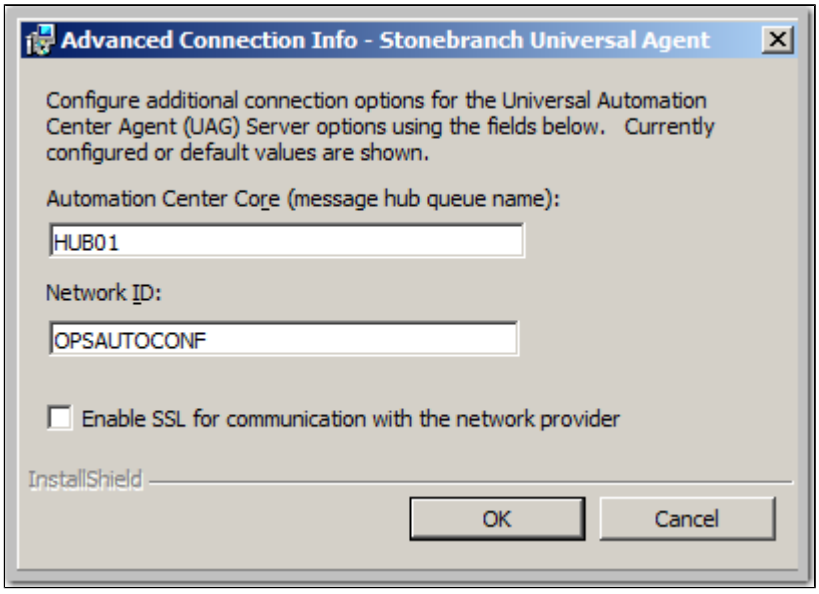
Select whether or not you want the Universal Message Service (OMS) Server to start up when Universal Broker starts up and, optionally, change the OMS Server port connection (the default port, 7878, is pre-selected).

Step 11 Click the **Next>** button. The Automation Center Connection Information dialog displays.



Step 12 Select the type of network communication provider for your Universal Automation Center system - Universal Message Server (OMS) or Opswise Transport and Message Hub - and the host name(s) for the selected provider.

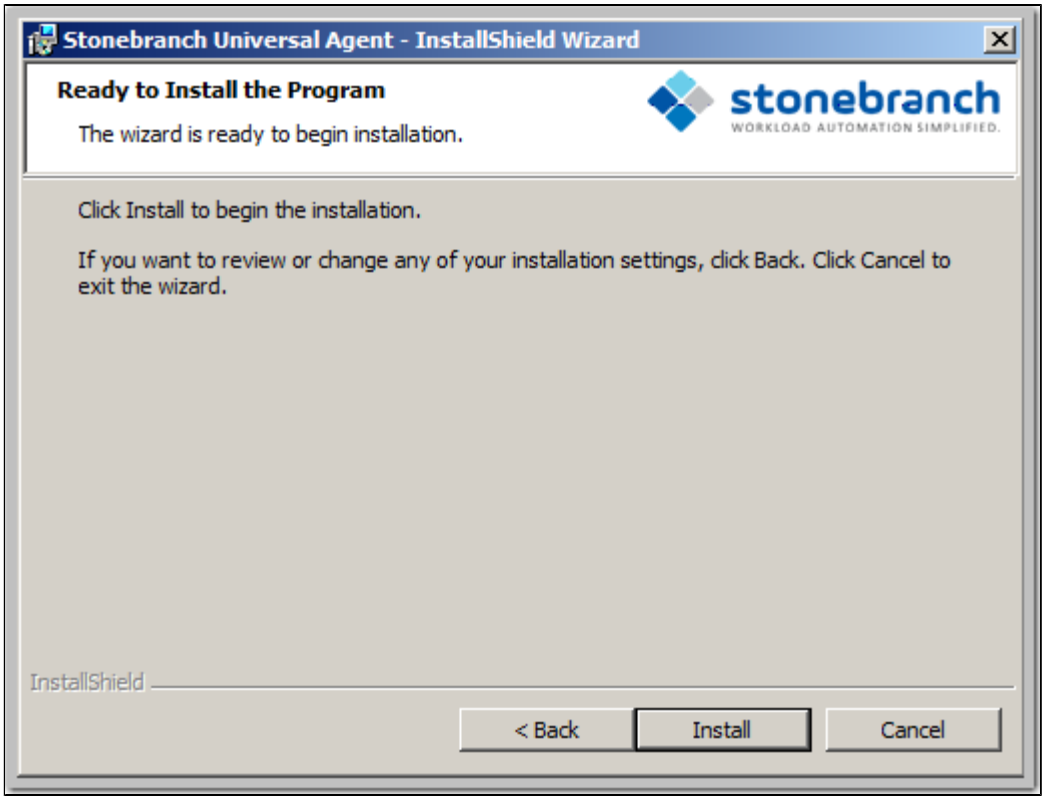
If you want to configure additional connection options for Universal Automation Center Agent (UAG), click the **Advanced** button. The Advanced Connection Info dialog displays.



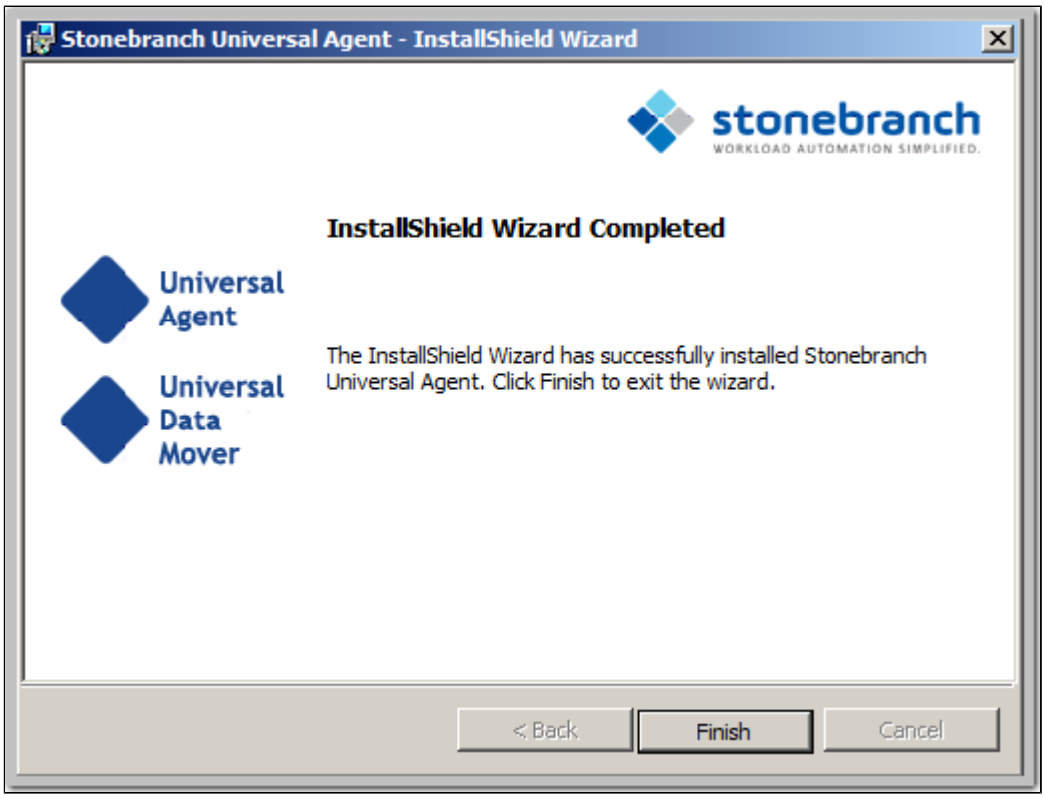
Configure the UAG options as desired and click the **OK** button.

Step 13 Click the **Next>** button to continue the installation process. Depending on the components selected, the installation may prompt for additional values, such as working folders for Server components. Follow the directions provided with each dialog.

When the installation is ready to begin, the Ready to Install dialog displays.



Click the **Install** button to begin the installation or click the **<Back** button to return to change information on any of the previous dialogs.
When the installation completes successfully, the Installation Complete dialog displays.



Step 14	Click the **Finish** button to exit Windows Installation.

Windows Installer Package File Locations

The Windows Installer Package file (**Ucmd.msi** or **Ucmdx64.msi**) is extracted to the following locations:

Universal Agent for Windows package (32-bit)	**`<LocalAppData>\UniversalAgent\<packagecode>\Ucmd.msi`**
Universal Agent for Windows package (64-bit)	**`<LocalAppData>\UniversalAgentx64\<packagecode>\Ucmdx64.msi`**

In these paths:

- **`<LocalAppData>`** represents a particular user's local Application Data folder.
  For example: If the installation was performed by the built-in Administrator account, **`<LocalAppData>`** would expand by default to:
  - **`C:\Documents and Settings\Administrator\Local Settings\Application Data`** on Windows XP and Server 2003.
  - **`C:\Users\Administrator\AppData\Local`** on Windows Vista, 7, Server 2008, and Server 2012.
- **`<packagecode>`** is a Universally Unique Identifier (UUID) in the format **`{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}`**, where 'X' is a hexadecimal character in the range 0 (zero) - F.
  For example: **`{3B10285A-9602-4DC9-B0A5-4D701BEB5225}`**.

Examples

In these examples:

- Package level is 0 (zero).
- Install is executed by the built-in Administrator user.
- Package code for the 32-bit Universal Agent for Windows package is **`{D16BD569-77D7-48D0-8EF0-3B0D143C44D8}`**.
- Package code for the 64-bit Universal Agent for Windows package is **`{3B10285A-9602-4DC9-B0A5-4D701BEB5225}`**.

Universal Agent for Windows Package (32-bit)

Distribution File	sb-6.2.0.0-windows-i386.exe
.msi File Name	Ucmd.msi

Default .msi File Location: Windows Server 2003, XP	**C:\Documents and Settings\Administrator\Local Settings\Application Data\UniversalAgent\{D16BD569-77D7-48D0-8EF0-3B0D143C44D8}**
Default .msi File Location: Windows Vista, 7, Server 2008, and Server 2012	**C:\Users\Administrator\AppData\Local\UniversalAgent\{D16BD569-77D7-48D0-8EF0-3B0D143C44D8}**

Universal Agent for Windows Package (64-bit)

Distribution File	sb-6.2.0.0-windows-x64.exe
.msi File Name	Ucmdx64.msi
Default .msi File Location: Windows XP, Server 2003	**C:\Documents and Settings\Administrator\Local Settings\Application Data\UniversalAgentx64\{3B10285A-9602-4DC9-B0A5-4D701BEB5225}**
Default .msi File Location: Windows Vista, 7, Server 2008, and Server 2012	**C:\Users\Administrator\AppData\Local\UniversalAgentx64\{3B10285A-9602-4DC9-B0A5-4D701BEB5225}**

Modifying a Universal Agent Installation via the Graphical Interface

- Introduction
- Modifying a Universal Agent Installation via the Windows Installer Graphical Interface
- Adding or Removing Universal Agent Components
- Repairing a Corrupted Universal Agent Installation
- Removing a Universal Agent Installation
  - Un-Installed Files

Introduction

The information on this page applies to the 32- and 64-bit Universal Agent for Windows installations.

The appearance of some of the screens shown below may differ between the two packages. The product name displayed for a 32-bit Universal Agent installation is simply "Stonebranch Universal Agent." The "(x64)" label is displayed for a 64-bit Universal Agent installation, making the product name "Stonebranch Universal Agent (x64)."

The sequence of steps to modify, repair, or remove a Universal Agent install is the same for 32- and 64-bit packages.

Modifying a Universal Agent Installation via the Windows Installer Graphical Interface

This page describes how to modify a Universal Agent installation via the Windows Installer graphical interface.

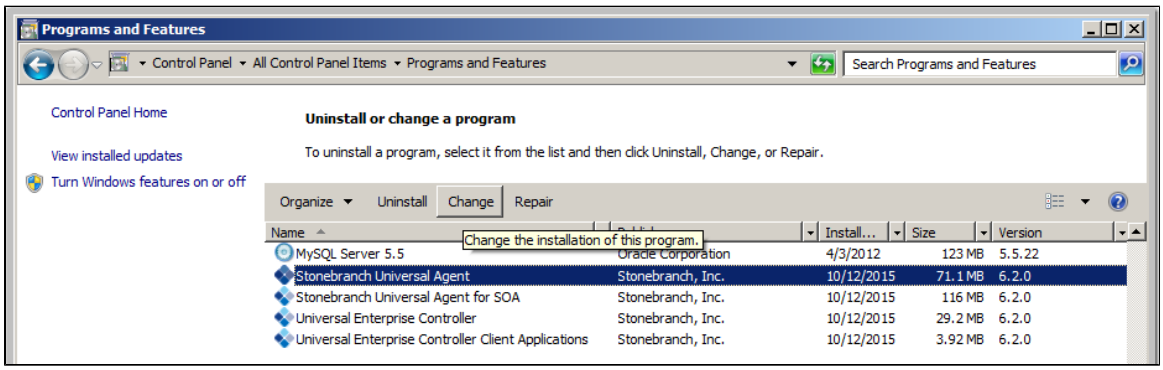
After Universal Agent is installed, the Windows Installer can be run as many times as needed to modify the installation by:

- Adding or Removing Universal Agent Components
- Repairing a Corrupted Universal Agent Installation
- Removing a Universal Agent Installation

Adding or Removing Universal Agent Components

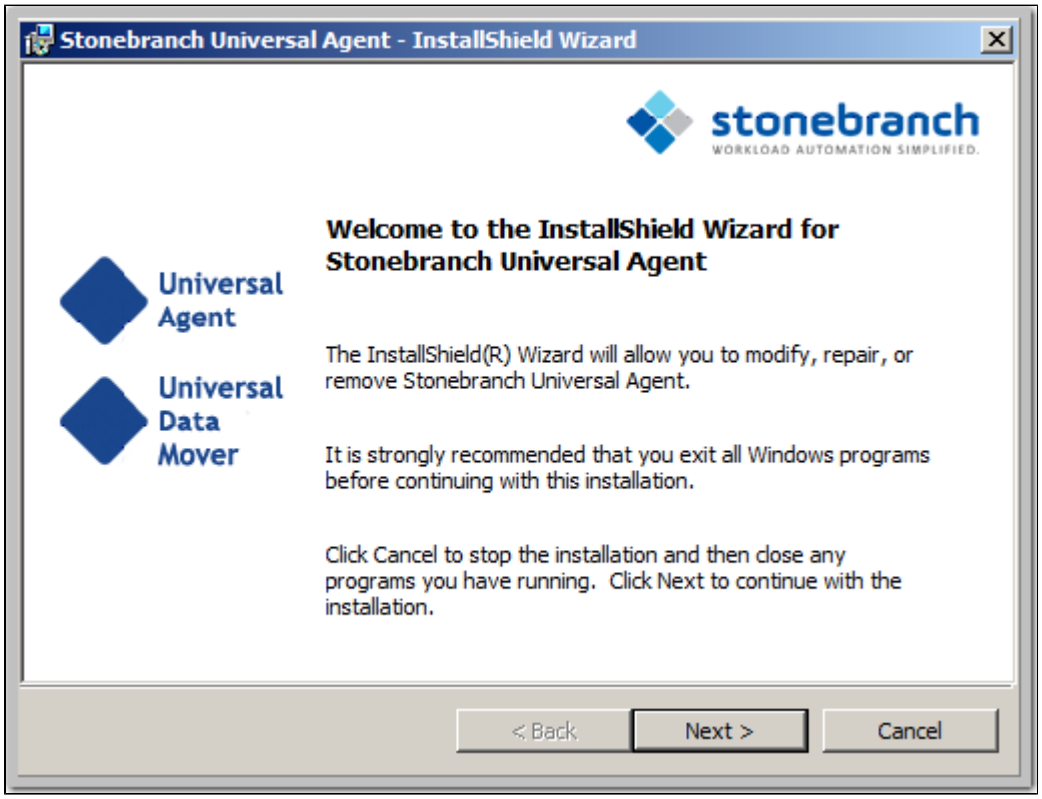
To add or remove components from a Universal Agent installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

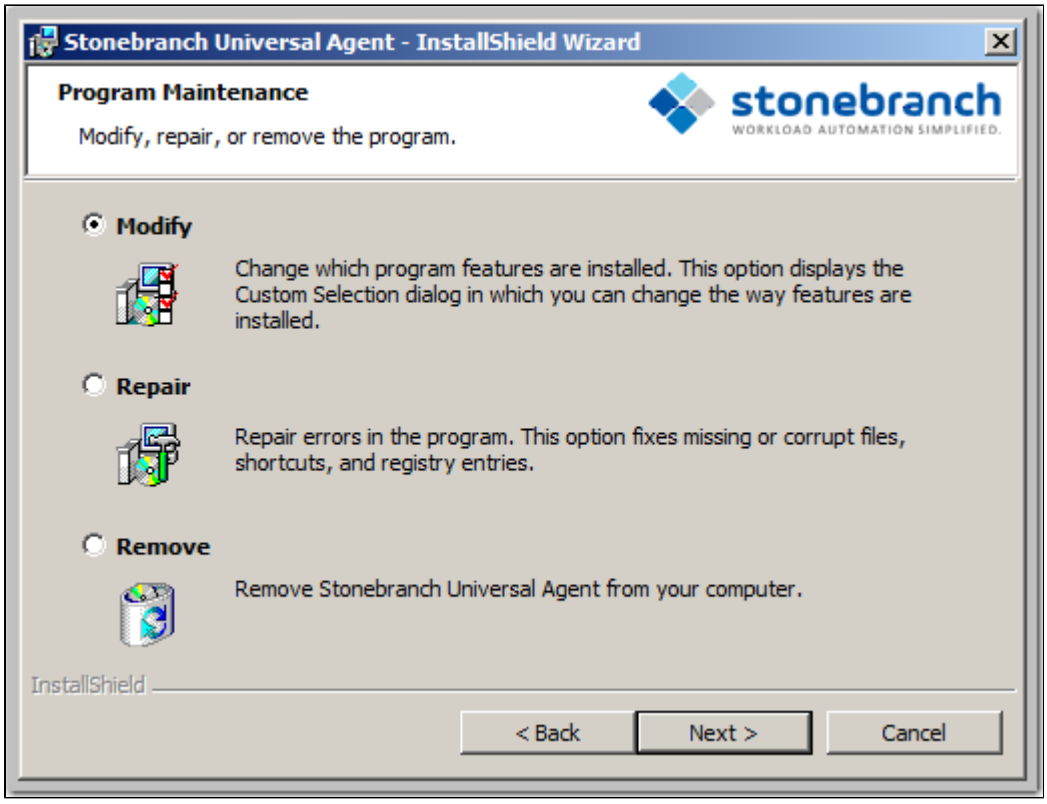


Windows Server 2003
If you are using Windows Server 2003, select **Add or Remove Programs** on the Windows Control Panel.

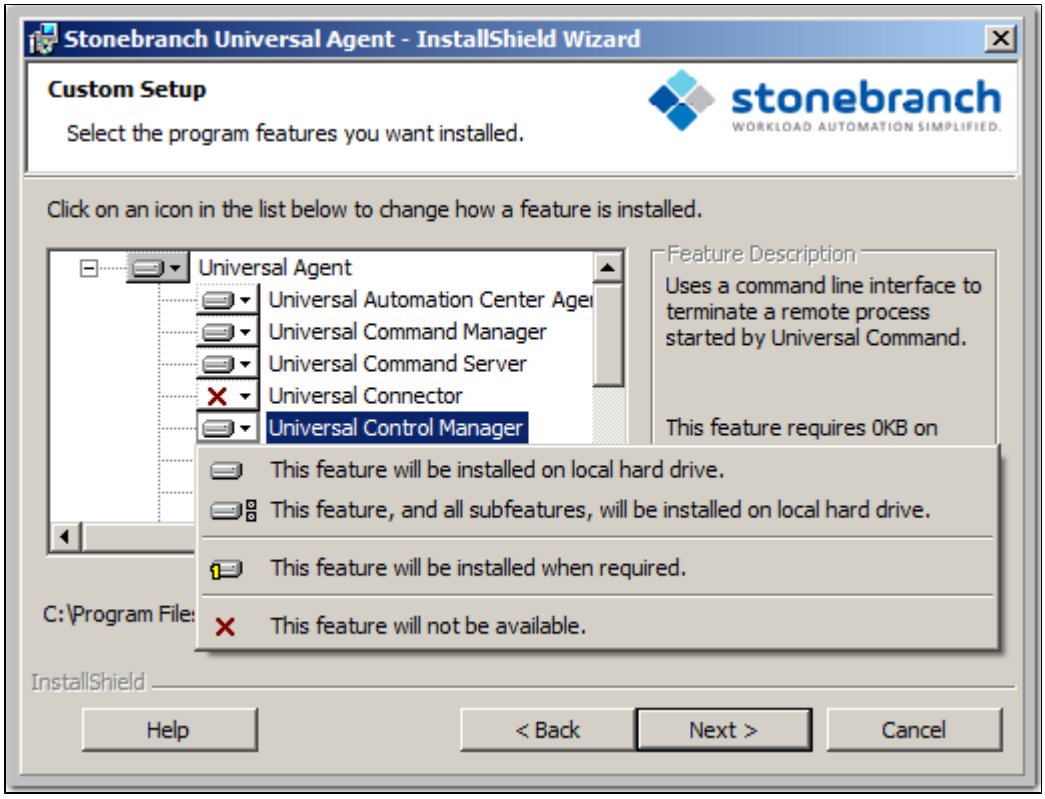
Step 2 From the list of installed programs, select **Stonebranch Universal Agent** and click **Change** to start Windows Installer. The Welcome dialog displays.



Step 3 Click the **Next>** button. The Program Maintenance dialog displays.



Step 4 Click the **Modify** radio button, and then the **Next>** button, to display the Custom Setup dialog.



Currently installed components are identified by a drive icon.

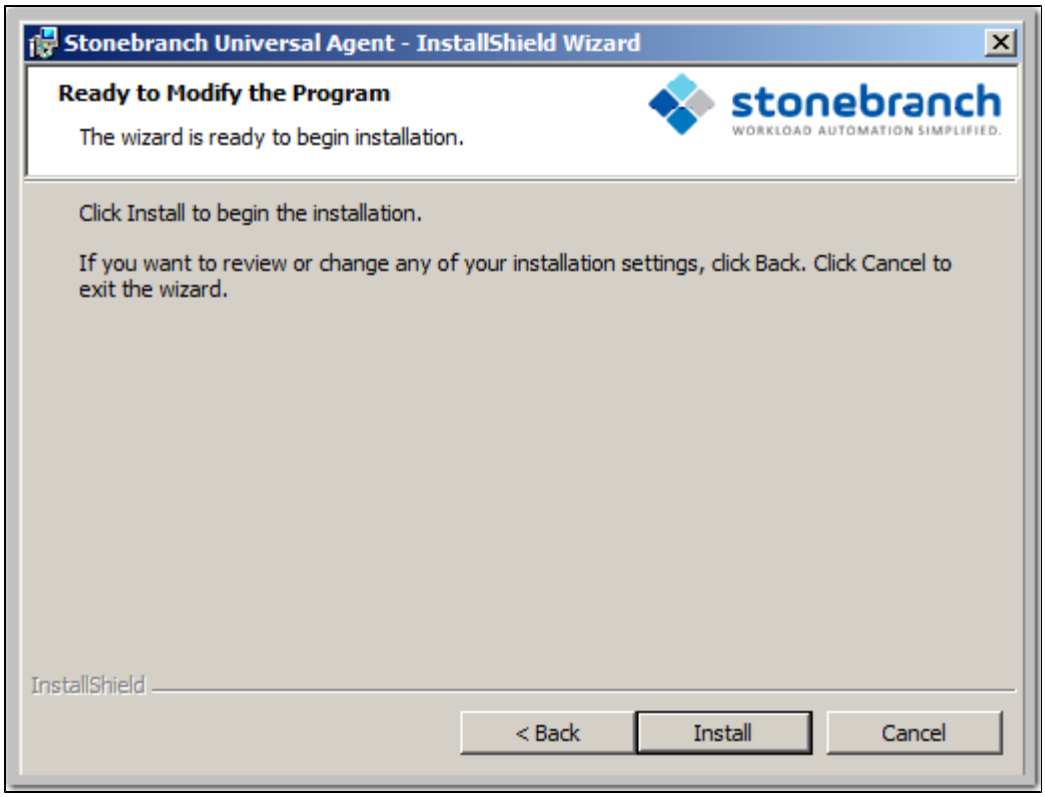
Uninstalled components are identified by an **X** icon.

Step 5 To remove a currently installed component:

1. Click the drive icon next to that component.
2. Select the X icon from the drop-down list to mark the component for removal.

Step 6 To add an uninstalled component:

1. Click the X icon next to that component.
2. Select the drive icon from the drop-down list to mark the component for installation.

Step 7	Click the Next> button to continue the installation process. Depending on the components selected, the installation may prompt for additional values, such as working folders for Server components. Follow the directions provided with each dialog. When the installation is ready to be modified, the Ready to Modify the Program dialog displays. 

Step 8	Click the **Install** button to modify the installation.
	
	When the modifications are complete, the following actions will be taken:
	
	• Components marked with a drive icon will:
	  • Remain installed if they already are installed.
	  • Be installed if they are not already installed.
	• Components marked with an **X** will:
	  • Remain uninstalled if they are not currently installed
	  • Be removed if they currently are installed.

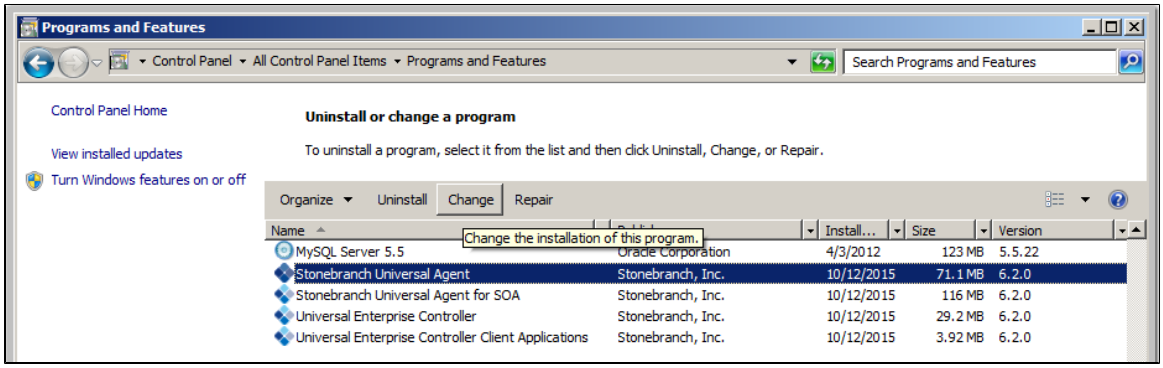
Repairing a Corrupted Universal Agent Installation

Windows Installer has the ability to recover accidentally deleted application files, configuration and component definition files, and registry entries required by Universal Agent. This repair feature will re-install the missing items, making a complete re-install unnecessary.

During a repair, any options stored in configuration and component definition files are preserved. If a component definition or configuration file was deleted, the installation will create a new configuration file with default values.

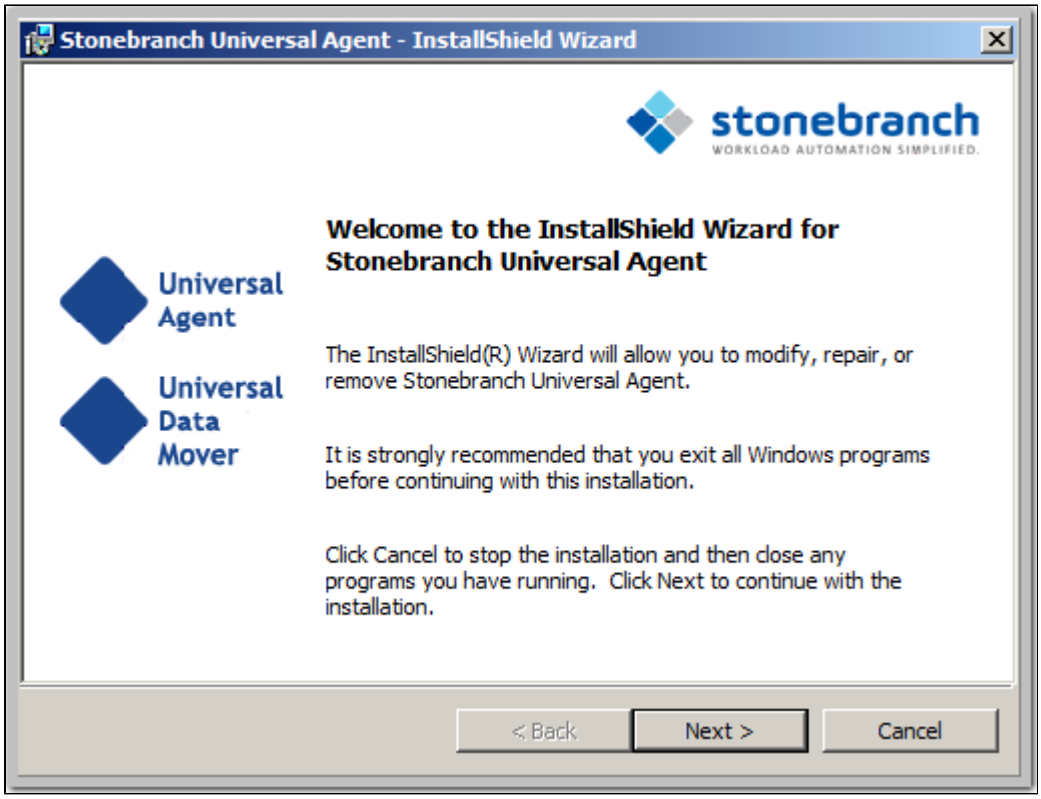
To repair an installation, perform the following steps:

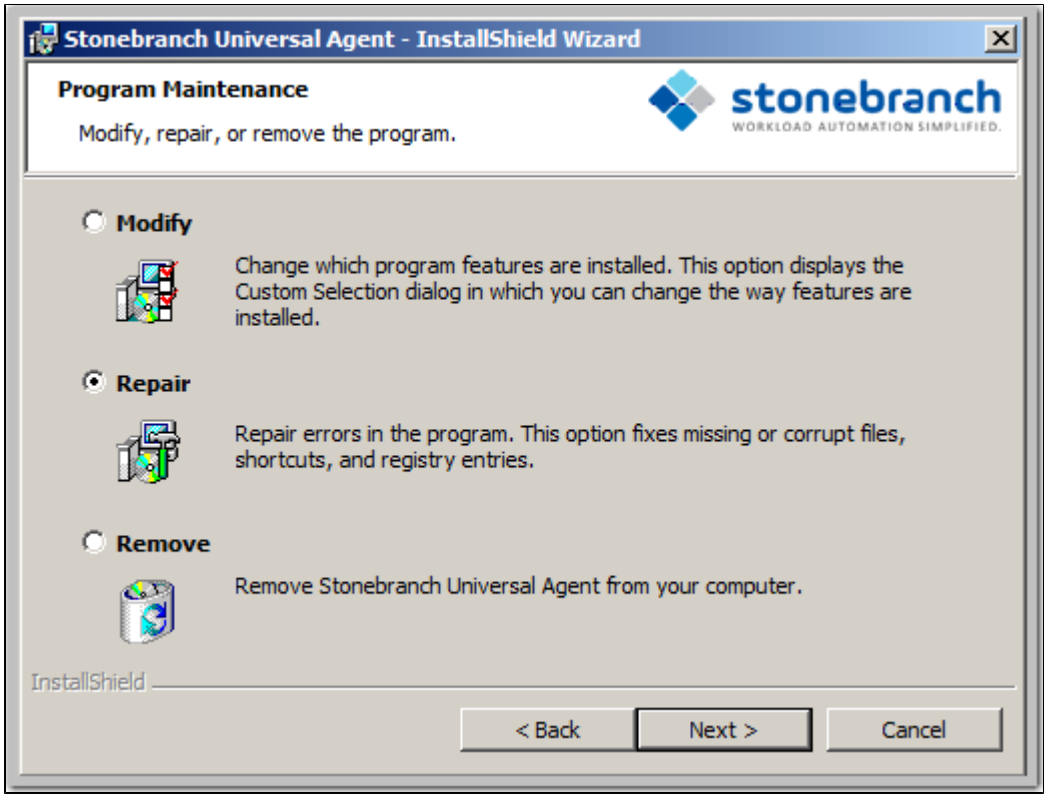
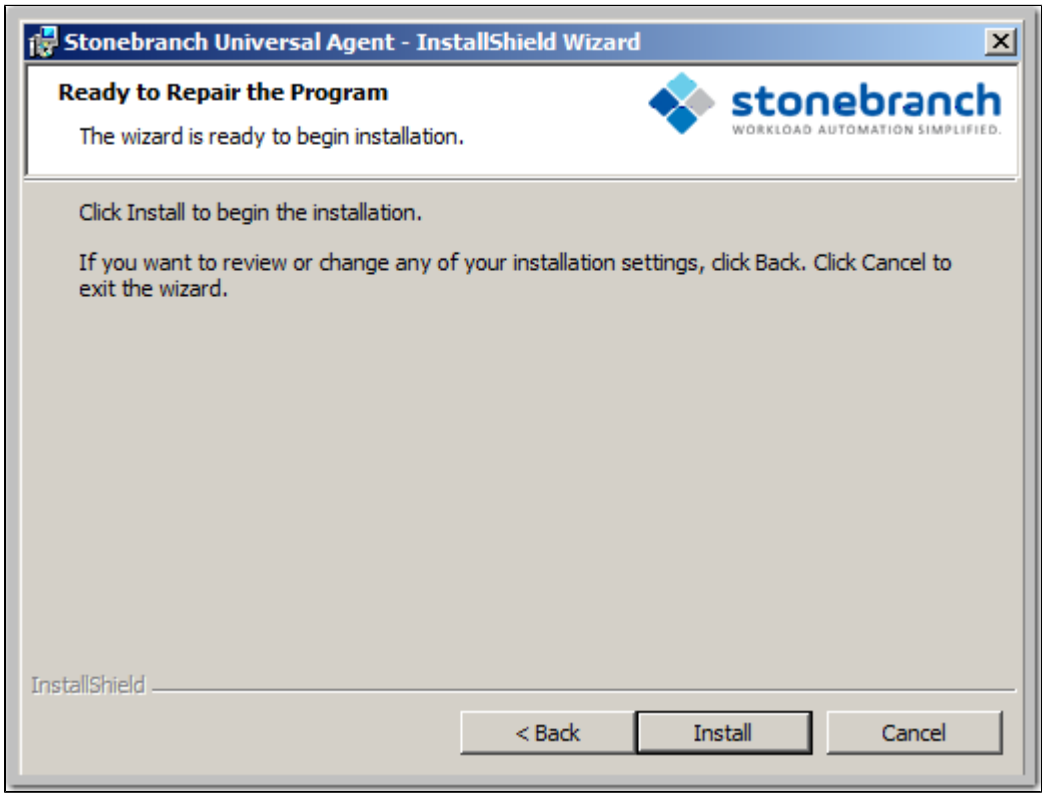
Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.



> **pre-Vista versions of Windows**
> If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Stonebranch Universal Agent** and click **Change** to start Windows Installer. The Welcome dialog displays.

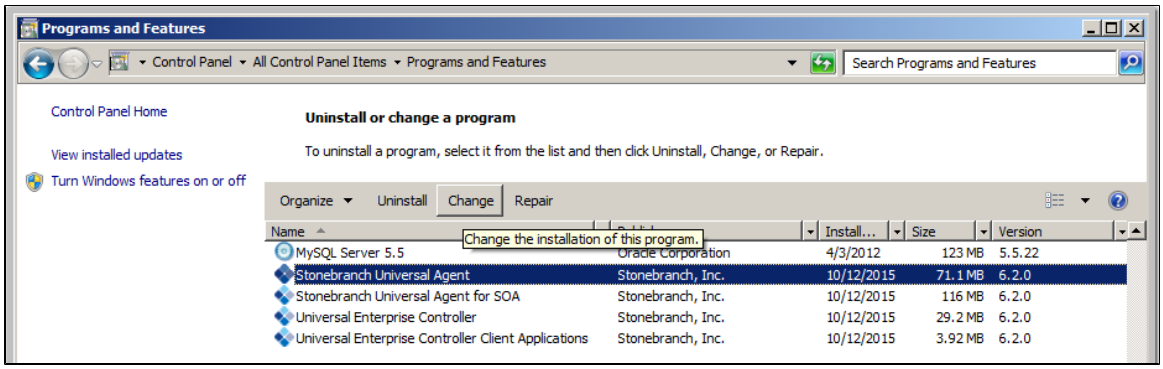


Step 3	Click the **Next>** button. The Program Maintenance dialog displays. 
Step 4	Click the **Repair** radio button, and then the **Next>** button, to display the Ready to Repair the Program dialog. 
Step 5	Click the **Install** button to repair the Stonebranch Universal Agent installation.

Removing a Universal Agent Installation

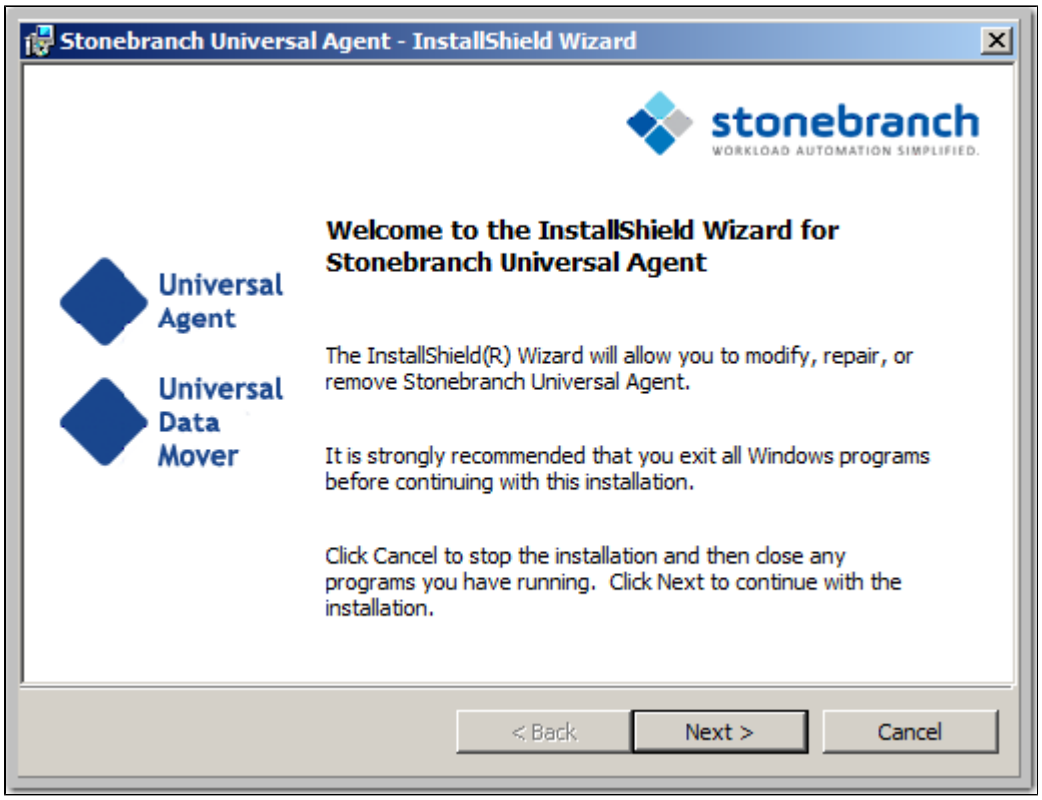
To uninstall a Universal Agent installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

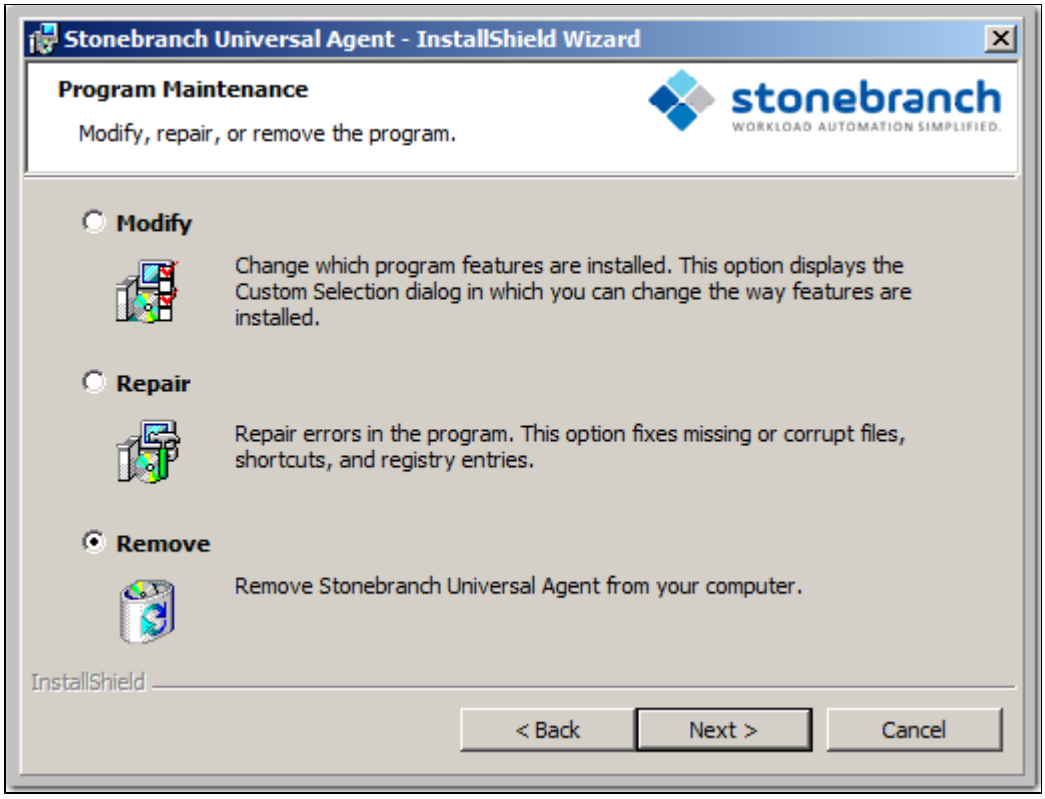


> **pre-Vista versions of Windows**
> If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

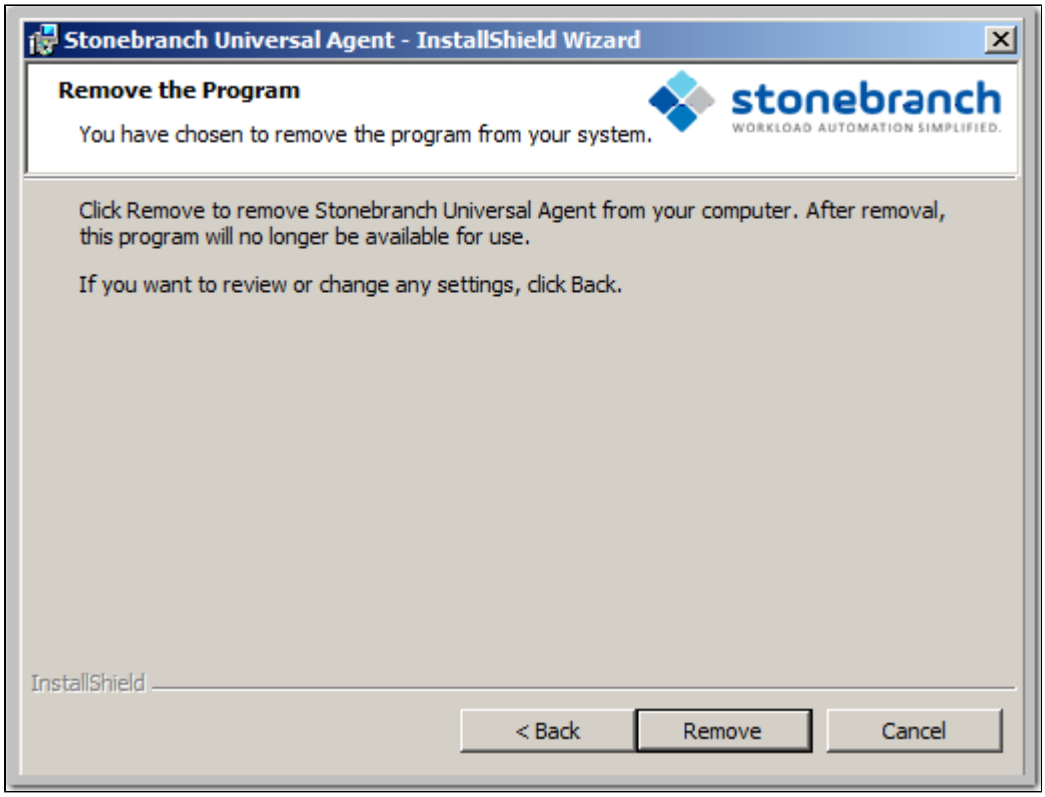
Step 2 From the list of installed programs, select **Stonebranch Universal Agent** and click **Change** to start Windows Installer. The Welcome dialog displays.



Step 3 Click the **Next>** button. The Program Maintenance dialog displays.



Step 4 Click the **Remove** radio button, and then the **Next>** button, to display the Remove the Program dialog.



Step 5	Click the **Remove** button to remove the Stonebranch Universal Agent installation. On the **Add or Remove Programs** screen, click **Change** and follow the Remove instructions in Windows Installer.

Un-Installed Files

The un-install process will remove only those files created during the installation. Some files stored under the **.\Universal** install directory by Universal Agent, such as trace files, files created under the Universal Command Server working directory, and spool files, may be left behind after the un- install. In this situation, those files and/or directories may simply be deleted.

Before deleting the entire **.\Universal** directory, make sure that no other Stonebranch, Inc. products are installed there. (See Universal Agent for Windows - File Inventory Lists for a list of files and directories installed with Universal Agent.)

In addition to those files and directories created by the Universal Agent installation, there may be some shared files (for example, codepage files) and Universal Agent components (for example, Universal Configuration Manager and Universal Encrypt) that may be left behind following an un-install. These components will be removed when the last Stonebranch Inc. product that uses them is un-installed.

Installing Universal Agent via the Command Line

- Introduction
- Installing the Agent
  - Command Line Syntax
  - Command Line Switches
  - Command Line Parameters
  - Command Line Installation Examples
- Detecting the Completion of a Silent Install

Introduction

This page describes how to install Universal Agent using the Windows Installer command line interface. Except where noted, the instructions are the same for the 32- and 64-bit Universal Agent packages.

A command line installation is useful in situations where:

- Several Universal Agent installations must be deployed across many different systems.
- It is not practical or convenient to perform the graphical interface installation.
- It is necessary to generate an installation log file.

Installing the Agent

To install Universal Agent for Windows using the Windows Installer command line interface, perform the following steps:

Step 1	Download the desired Universal Agent for Windows product distribution file to your work station: • **`sb-6.2.0.<level>-windows-i386.exe`**, the 32-bit Universal Agent for Windows distribution file. • **`sb-6.2.0.<level>-windows-x64.exe`**, the 64-bit Universal Agent for Windows distribution file (for supported 64-bit versions of Windows only).
Step 2	Execute the distribution file from the command line, and include all appropriate command line switches and parameters. The installation process determines whether a Windows Installer update is needed. The process then extracts and saves a Windows installer package file (.msi) to one of these locations. After all files (including the .msi) are extracted from the distribution file, the installation process verifies that your machine meets the minimum installation requirements. If the requirements are met, the installation begins.

Command Line Syntax

The following illustrates the command line syntax used to install the Agent:

```
sb-6.2.0.<level>-windows-<platform>.exe [/v"command line parameters"]  [/s] [/w] [/x]
```

In this syntax:

- **`<level>`** is the numeric package level.
- **`<platform>`** is:
  - **`i386`** for the 32-bit Universal Agent distribution file.
  - **`x64`** for the 64-bit Universal Agent distribution file.

The command line switches (/v, /s, /w, and /x) are processed directly by the distribution file to control behavior of the Windows Setup application.

The command line parameters are passed to the Windows Installer (**msiexec**) to control the extracted Windows Installer Package file (**Ucmd.msi** or **Ucmdx64.msi**) behavior during the install process.

Command Line Switches

The following table describes the command line switches available for a command line installation:

/v	Passes parameters to the Windows Installer (**msiexec**). The list of parameters must be enclosed in double (") quotation marks. See Command Line Parameters for available parameters.
/s	Suppresses the initialization and extraction dialogs displayed before the product install Welcome dialog." If you are using the /q command line parameter, use this switch additionally for a completely silent install.
/w	Instructs the Windows Setup application to wait until the installation completes. Use this switch when launching the installation from a script file. Without it, the Setup application may return immediately after launching Windows Installer.
/x	Uninstalls Universal Agent for Windows.

Command Line Parameters

The following table describes the parameters that are available for a command line installation.

The parameters can be specified in any order, with the following exceptions:

- If the Repair (/fom) or Remove (/x) parameter is used, it must be specified ***before*** all other parameters.
- If the Silent install (/q) and/or Log file (/L) parameters are used, they can be specified in any order, but they must be specified ***after*** all other parameters.

These parameters are preceded by the /v command line switch and must be enclosed in double (") quotation marks.

Parameter	Description	Default
/fom	Repairs a Universal Agent for Windows installation. **om** (after the **/f**) are options used by the repair. There are other options available, but for behavior that matches the repair done from the graphical install, the **om** options must be used. **/fom** cannot be used with the ***/x** (remove) parameter.	n/a
/x	Removes a Universal Agent for Windows installation. **/x** cannot be used with the **/fom** (repair) parameter.	n/a
INSTALLDIR=*installdir*	Sets the root installation directory to *installdir*. Each component will be installed under this directory. **INSTALLDIR** is required only if you want to install Universal Agent for Windows under a directory different from the one specified by the **PROGRAMFILES** environment variable (typically **`C:\Program Files\Universal`**). If the directory contains spaces, you must use double (**"**) quotation marks around the path name.	(none)

RUNBROKERASUSER={**0**|**1**}	Controls whether the Universal Broker service executes as an Administrative user account or as **Local System**. • If **RUNBROKERASUSER**=**1**, the install will perform the following steps: 1. If the Universal Broker service is not installed or currently is configured to run as **Local System**, the install will set the service's start-up account to **UBrokerService** or the user ID specified by the BROKERUID command line option. 2. If the Universal Broker service is installed and already is configured to run as a user account, the Broker's start-up account will not be changed (allows customized user accounts to be preserved during upgrades). • If **RUNBROKERASUSER**=**0**, the install will perform the following steps: 1. If the Universal Broker service is not installed or is currently configured to run as an account other than **Local System**, the install will set the service's start-up account to **Local System**. ⚠ **Note** The Universal Broker service's properties will be set to allow Universal Broker to interact with the desktop. This is supported only when Universal Broker runs as Local System. 2. If the Universal Broker service is installed and is currently configured to run as **Local System**, the Universal Broker service's properties will not be changed.	**1**: new installs **0**: upgrades earlier than 4.3.0
BROKERUID=*BrokerAccountId*	Used by the install when RUNBROKERASUSER=**1** to override the default or currently configured user ID associated with the account used to execute the Universal Broker service. The install also uses **BROKERUID** to initialize the User ID shown in the Universal Broker Service Account dialog. Valid values are any 20 characters or less. For domain accounts, a domain name of up to 256 characters is accepted, but the user ID is still limited to 20 characters. **BROKERUID** is ignored if RUNBROKERASUSER=0.	(none)
BROKERPWD=*BrokerAccountPassword*	Used by the install when RUNBROKERASUSER=**1** to specify the password for the account used to execute the Universal Broker service. The install also uses **BROKERPWD** to initialize the Password shown in the Universal Broker Service Account dialog. Valid values are any 256 characters or less. **BROKERPWD** is ignored if RUNBROKERASUSER=**0**.	(none)
CONVERT_OPSAGENT={**yes**|**no**}	Specification for whether or not to convert an existing Opswise Automation Center Agent (1.5, 1.6, or 1.7) to a 6.2.x Agent. The installation process will invoke a script, **`opsmerge.vbs`**, which stops the Agent and converts the configuration options stored in its **`agent.props`** file to corresponding configuration options in the Universal Automation Center Agent (UAG) **`uags.conf`** file. The script also performs several other tasks needed for the conversion. ⚠ **Note** You must include this parameter when upgrading an Opswise Automation Center Agent (1.5, 1.6, or 1.7) to Agent 6.2.x.	**no**

NETWORK_PROVIDER={**oms**|**transport**}	Specifies the network communications provider that the UAG Server uses to connect to the Universal Controller. • If **oms** is specified, the Agent connects to the Controller using the Universal Message Service (OMS). One or more OMS_SERVERS must also be specified. • If **transport** is specified, the Agent connects to the Controller using the traditional Opswise Transporter and Message Hub. Values for AC_TRANSPORTS and AC_CORE also must be specified. If **NETWORK_PROVIDER** is omitted from the command line, the Agent connects to the Connector using the method specified in the Universal Automation Center Agent (UAG) NETWORK_PROVIDER configuration option.	(none)
OMS_SERVERS=*port@ipaddr*[,*port**n**@ipaddr**n***,...]	A list of one or more locations where an OMS Server resides.	(none)
AC_TRANSPORTS	AC_TRANSPORTS=*port@ipaddr*[,*port**n**@ipaddr**n***,...] Specifies a value for the port and network address of the Universal Controller Transporter(s) used for network communication. The install uses this value to set the UAG AUTOMATION_CENTER_TRANSPORTS configuration option. The value specified in **AC_TRANSPORTS** will override any currently configured options, including those imported from **`agent.props`**, if CONVERT_OPSAGENT is set to **yes**.	(none)
AC_CORE=*value*	Specifies a value for the queue name of the Universal Controller Message Hub. The install uses this value to set the UAG AUTOMATION_CENTER_CORE configuration option. The value specified in **AC_CORE** will override any currently configured options, including those imported from **`agent.props`**, if CONVERT_OPSAGENT is set to **yes**.	(none)
AC_ENABLE_SSL={**yes**|**no**}	Specifies whether or not the Agent should use SSL to communicate with the configured network provider. The install uses this value to set the UAG ENABLE_SSL configuration option. The value specified in **AC_ENABLE_SSL** will override any currently configured options. If **AC_ENABLE_SSL** is omitted from the command line, and no other value enables SSL, the default is used.	**no**
AC_NETNAME=*value*	Specifies a value for a network ID that uniquely identifies the Agent on this system to the Universal Controller. The install uses this value to set the UAG NETNAME configuration option. The value specified in **AC_NETNAME** will override any currently configured options, including those imported from **`agent.props`**, if CONVERT_OPSAGENT is set to **yes**. If **AC_NETNAME** is omitted from the command line, and no other value specifies the Agent's network ID, the default is used.	OPSAUTOCONF
UCMDSRVWORKFOLDER=*WorkFolderPath*	Sets the Universal Command Server working folder. This value is used only if the Universal Command Server is being installed. If the folder contains spaces, you must use double (**"**) quotation marks around the path name.	**INSTALLDIR\UcmdHome**
UCTLSRVWORKFOLDER=*WorkFolderPath*	Sets the Universal Control Server working folder. This value is used only if the Universal Control Server is being installed. If the folder contains spaces, you must use double (**"**) quotation marks around the path name.	**INSTALLDIR\UcmdHome**
UDMSRVWORKFOLDER=*WorkFolderPath*	Sets the Universal Data Mover Server working folder. This value is used only if the Universal Data Mover Server is being installed. If the folder contains spaces, you must use double (**"**) quotation marks around the path name.	**INSTALLDIR\UdmHome**
UEMSRVWORKFOLDER=*WorkFolderPath*	Sets the Universal Event Monitor Server working folder. This value is used only if the Universal Event Monitor Server is being installed. If the folder contains spaces, you must use double (**"**) quotation marks around the path name.	**INSTALLDIR\UemHome**

OMS={**yes**|**no**}	Specification for whether or not to install the Universal Message Service (OMS) during new installs, upgrades, or maintenance. • If **yes** is specified, OMS will be installed. • If **no** is specified, OMS will not be installed. If OMS already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **OMS** is not required unless you want to change the current install state. **OMS** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Message Service** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
OMS_AUTOSTART={**yes**|**no**}	Specification for setting the start-up behavior of the Universal Message Service (OMS) Server component. • If **yes** is specified, the AUTOMATICALLY_START OMS component definition option is set to **yes**, which instructs Universal Broker to start the OMS Server automatically when it is started. • If **no** (the default) is specified, or if this option is not included on the command line, the AUTOMATICALLY_START and RESTART OMS component definition options are set to **no**. Universal Broker will not start the OMS Server automatically when it is started. This allows the OMS Server to be started manually any time after the Broker is started. This OMS_AUTOSTART value is used to initialize the state of a check box on the OMS Server Start-up Option dialog, which displays for new installs and upgrades from Universal Agent releases earlier than 5.2.0. Changing the state of this check box will override the value specified from the command line.	**no**
OPSCLI={**yes**|**no**}	Specification (yes or no) for whether or not to install the Universal Controller Command Line Interface (CLI) during new installs, upgrades, or maintenance. • If **yes** is specified, the CLI will be installed. • If **no** is specified, the CLI will not be installed. If the CLI already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **OPSCLI** is not required unless you want to change the current install state. **OPSCLI** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Controller Command Line Interface** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**no**
UAGSRV={**yes**|**no**}	Specification for whether or not to install Universal Automation Center Agent (UAG) during new installs, upgrades, or maintenance. • If **yes** is specified, UAG will be installed. • If **no** is specified, UAG will not be installed. If UAG already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UAGSRV** is not required unless you want to change the current install state. **UAGSRV** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Automation Center Agent** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**

UCMDMGR={**yes**|**no**}	Specification for whether or not to install the Universal Command (UCMD) Manager during new installs, upgrades, or maintenance. • If **yes** is specified, the UCMD Manager will be installed. • If **no** is specified, the UCMD Manager will not be installed. If the UCMD Manager already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UCMDMGR** is not required unless you want to change the current install state. **UCMDMGR** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Command Manager** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UCMDSRV={**yes**|**no**}	Specification for whether or not to install the Universal Command (UCMD) Server during new installs, upgrades, or maintenance. • If **yes** is specified, the UCMD Server will be installed. • If **no** is specified, the UCMD Server will not be installed. If the UCMD Server already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UCMDRSV** is not required unless you want to change the current install state. **UCMDSRV** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Command Server** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UCTLMGR={**yes**|**no**}	Specification (yes or no) for whether or not to install the Universal Control (UCTL) Manager during new installs, upgrades, or maintenance. • If **yes** is specified, the UCTL Manager will be installed. • If **no** is specified, the UCTL Manager will not be installed. If the UCTL Manager already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UCTLMGR** is not required unless you want to change the current install state. **UCTLMGR** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Control Manager** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UCTLSRV={**yes**|**no**}	Specification for whether or not to install the Universal Control (UCTL) Server during new installs, upgrades, or maintenance. • If **yes** is specified, the UCTL Server will be installed. • If **no** is specified, the UCTL Server will not be installed. If the UCTL Server already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UCTLRSV** is not required unless you want to change the current install state. **UCTLSRV** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Control Server** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**

UDMMGR={**yes**|**no**}	Specification for whether or not to install the Universal Data Mover (UDM) Manager during new installs, upgrades, or maintenance. • If **yes** is specified, the UDM Manager will be installed. • If **no** is specified, the UDM Manager will not be installed. If the UDM Manager already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UDMMGR** is not required unless you want to change the current install state. **UDMMGR** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Data Mover Manager** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UDMSRV={**yes**|**no**}	Specification (yes or no) for whether or not to install the Universal Data Mover (UDM) Server during new installs, upgrades, or maintenance. • If **yes** is specified, the UDM Server will be installed. • If **no** is specified, the UDM Server will not be installed. If the UDM Server already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UDMRSV** is not required unless you want to change the current install state. **UDMSRV** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Data Mover Server** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UELD={**yes**|**no**}	Specification for whether or not to install Universal Event Log Dump (UELD) during new installs, upgrades, or maintenance. • If **yes** is specified, UELD will be installed. • If **no** is specified, UELD will not be installed. If UELD already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UELD** is not required unless you want to change the current install state. **UELD** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Event Log Dump** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UEMMGR={**yes**|**no**}	Specification for whether or not to install the Universal Event Monitor (UEM) Manager during new installs, upgrades, or maintenance. • If **yes** is specified, the UEM Manager will be installed. • If **no** is specified, the UEM Manager will not be installed. If the UEM Manager already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UEMMGR** is not required unless you want to change the current install state. **UEMMGR** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Event Monitor Manager** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**

UEMSRV={**yes**|**no**}	Specification for whether or not to install the Universal Event Monitor (UEM) Server during new installs, upgrades, or maintenance. • If **yes** is specified, the UEM Server will be installed. • If **no** is specified, the UEM Server will not be installed. If the UEM Server already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UEMRSV** is not required unless you want to change the current install state. **UEMSRV** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Event Monitor Server** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
UQUERY={**yes**|**no**}	Specification for whether or not to install the Universal Query (UQUERY) during new installs, upgrades, or maintenance. • If **yes** is specified, UQUERY will be installed. • If **no** is specified, UQUERY will not be installed. If UQUERY already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UQUERY** is not required unless you want to change the current install state. **UQUERY** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Query** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
USAP={**yes**|**no**}	Specification for whether or not to install the Universal Connector (USAP) during new installs, upgrades, or maintenance. • If **yes** is specified, USAP will be installed. • If **no** is specified, USAP will not be installed. If USAP already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **USAP** is not required unless you want to change the current install state. **USAP** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Connector** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**
USPOOL={**yes**|**no**}	Specification (yes or no) for whether or not to install the Universal Spool Utilities, Universal Spool List (USLIST) and Universal Spool Remove (USLRM), during new installs, upgrades, or maintenance. • If **yes** is specified, the Universal Spool Utilities will be installed. • If **no** is specified, the Universal Spool Utilities will not be installed. If the Universal Spool Utilities already are present on the system (via a previous installation), they will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **USPOOL** is not required unless you want to change the current install state. **USPOOL** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **Universal Spool Utilities** on the Custom Setup dialog when installing Universal Agent via the graphical interface.	**yes**

/q{**n**|**b**|**r**}	Suppresses the product installation dialogs. Use this parameter in addition to the /s command line switch for a completely silent install. The following flags are available for **/q**: • **n** - Creates no user interface. • **b** - Creates a basic user interface, which displays a simple dialog box to report progress and any errors. • **r** - Creates a reduced user interface, which displays progress and errors in customized windows. See Command Line Switches, Command Line Installation Examples, and Detecting the Completion of a Silent Install for additional information regarding silent installs.	n/a
/L[*v]	Instructs the installation process to create an installation log file named < **`logfilepath`**> (full path name). If <**`logfilepath`**> contains spaces, you must enclose it with double (**"**) quotation marks. ***v** are flags used to specify the level of detail (**verbose**) contained in the log file. To reduce the amount of output generated, ***v** can be omitted. However, using these options is good practice; they can assist Stonebranch Customer Support with problem determination should any errors occur during installation.	n/a

Command Line Installation Examples

The following examples illustrate different ways that Universal Agent for Windows can be installed from the command line.

Graphical User Interface Install, All Components

To install all Universal Agent for Windows components via the graphical interface using the 32-bit distribution file, issue the following command:

```
sb-6.2.0.0-windows-i386.exe
```

Graphical User Interface Install, with Log File

```
sb-6.2.0.0-windows-i386.exe /v"/l*v C:\Temp\install.log"
```

Silent Install, All Components, with Log File

To silently install all 64-bit Universal Agent for Windows components using the 64-bit distribution file and write a log file to **`C:\Temp\windows install.log`**, issue the following command:

```
sb-6.2.0.0-windows-i386.exe /s /v"/qn /l*v C:\Temp\install.log"
```

Silent Install, Different Installation Directory

```
sb-6.2.0.0-windows-i386.exe /s /v"/qn INSTALLDIR=D:\Universal"
```

Silent Install, Quotation Marks for Directory Path with a Space

```
sb-6.2.0.0-windows-i386.exe /s /v"/qn INSTALLDIR=\"C:\Program Files\Stonebranch\""
```

Silent Installation Removal

```
sb-6.2.0.0-windows-i386.exe /x /s /v"/qn"
```

Install Partial Components to a Different Installation Directory

To install only the Universal Broker (which always is installed), Universal Data Mover Manager, Universal Event Monitor Manager, Universal Control Manager, and Universal Query components under **`D:\Universal`** (that is, a directory other than the one specified by the environment variable PROGRAMFILES) using the 32-bit Universal Agent distribution file, issue the following command:

```
sb-6.2.0.0-windows-i386.exe /v"INSTALLDIR=D:\Universal OMS=no UAGSRV=no UCMDMGR=no UCMDSRV=no
UCTLSRV=no UELD=no USPOOL=no UDMSRV=no UEMSRV=no"
```

Install All Components, Set OMS Server Location

To install all Universal Agent components using the 32-bit Universal Agent distribution file, and to set the Universal Automation Center Agent (UAG) OMS_SERVERS configuration option, issue the following command:

```
sb-6.2.0.0-windows-i386.exe /v"OMS_SERVERS=7878@opshost"
```

Install All Components, Set Universal Broker Service

To install all Universal Agent components using the 64-bit Universal Agent distribution file, and to set the Universal Broker service to execute with an administrative account with the specified user ID and password, issue the following command:

```
sb-6.2.0.0-windows-x64.exe /v"RUNBROKERASUSER=1 BROKERUID=ubradmin BROKERPWD=guessme"
```

Detecting the Completion of a Silent Install

If /s and /q are used to perform a silent install, no graphical interface or user interaction is required. One drawback to this is that no feedback is provided that indicates when the Windows Installer process (install, uninstall, or repair) finishes.

One method that can be used to detect when the installation process ends is to execute it using the system's **start** command. Using available command line switches, **start** can be used to initiate the installation process and then wait for it to finish. When **start** returns control to its calling process (for example, the command prompt), the process will have ended.

For example, from the command prompt, issue the following command to perform a silent 64-bit Universal Agent installation using the 64-bit distribution file and wait for it to finish:

```
start /b /wait sb-6.2.0.0-windows-x64.exe /w /s /v"/qn"
```

- The **/b** switch prevents the **start** command from opening a new window.
- The **/wait** parameter causes the **start** command to start the installation and then wait for it to finish.

This also is the recommended way to execute the installation from within a batch file.

For more information on the **start** command, go to the Windows command prompt and enter: **start /?**.

Modifying a Universal Agent Installation via the Command Line

- Modifying a Universal Agent Installation via the Command Line
- Adding or Removing Universal Agent Components
- Repairing a Corrupted Universal Agent Installation
  - Silent Repair, Distribution File Execution, with Log File
  - Silent Repair, 32-bit Installation, Windows Installer Execution, with Log File
  - Silent Repair, 64-bit Installation, Windows Installer Execution, with Log File
- Removing a Universal Agent Installation
  - Interactive Removal, Distribution File Execution
  - Silent Removal, 32-bit Installation, Windows Installer Execution
  - Silent Removal, 64-bit Installation, Windows Installer Execution

Modifying a Universal Agent Installation via the Command Line

This page describes how to modify a Universal Agent installation via the Windows Installer command line interface.

After Universal Agent is installed, Windows Installer can be run as many times as needed to modify the installation by:

- Adding or Removing Universal Agent Components
- Repairing a Corrupted Universal Agent Installation
- Removing a Universal Agent Installation

(For a description of the parameters used in these procedures, see Windows Installer Command Line Parameters.)

Adding or Removing Universal Agent Components

Currently, it only is possible to add or remove individual Universal Agent components using the Windows Installer graphical interface (see Modifying a Universal Agent Installation via the Graphical Interface).

Repairing a Corrupted Universal Agent Installation

To recover accidentally deleted files or registry entries required by Universal Agent using the Windows Installer command line interface, use the ***/f*** switch together with the **om** parameters.

These are the same repair options set internally by the graphical interface installation. They cause Windows Installer to reinstall files that either are missing or older than the version contained in the Universal Agent distribution file.

Silent Repair, Distribution File Execution, with Log File

To repair a Universal Agent installation from the command line using the original distribution file, issue the following command:

```
sb-6.2.0.<level>-windows-<platform>.exe /v"/fom /q /L*v c:\temp\repair.log"
```

In this command, **<level>** is the numeric package level and **<platform>** is either i386 (32-bit distribution file) or x64 (64-bit distribution file).

Silent Repair, 32-bit Installation, Windows Installer Execution, with Log File

To repair a 32-bit Universal Agent installation, without using the Windows Installer graphical interface, using the **.msi** file that was extracted from the distribution file during the initial install, issue the following command:

```
msiexec.exe /fom SetupPath\Ucmd.msi /q /L*v c:\temp\repair.log
```

Silent Repair, 64-bit Installation, Windows Installer Execution, with Log File

To repair a 64-bit Universal Agent installation using the **.msi** file that was extracted from the distribution file during the initial install, issue the following command:

```
msiexec.exe /fom SetupPath\Ucmdx64.msi /q /L*v c:\temp\repair.log
```

In the last two examples, **SetupPath** refers to the location in which the extracted **.msi** file resides. The exact path varies, depending on the user account that performed the install and the Windows version.

For information on how the installation determines **SetupPath**, see Step 2 of the Installing via Command Line procedures.

> **Note**
> All of these commands also use the optional **/L*v** parameter to generate a log file named **C:\temp\repair.log**. If you want to generate a log file, you can substitute the path and file name for one of your choosing. If no log file is desired, simply omit the **/L*v** option and file name.

Removing a Universal Agent Installation

Interactive Removal, Distribution File Execution

To uninstall a Universal Agent installation from the command line using the original distribution file, issue the following command:

```
sb-6.2.0.<level>-windows-<platform>.exe /x
```

In this command, **<level>** is the numeric package level and **<platform>** is i386 for the 32-bit distribution file and x64 for the 64-bit distribution file.

Silent Removal, 32-bit Installation, Windows Installer Execution

To uninstall a 32-bit Universal Agent installation using the **.msi** file that was extracted from the distribution file during the initial install, issue the following command:

```
msiexec.exe /x SetupPath\Ucmd.msi /q
```

Silent Removal, 64-bit Installation, Windows Installer Execution

To uninstall a 64-bit Universal Agent installation using the **.msi** file that was extracted from the distribution file during the initial install, issue the following command:

```
msiexec.exe /x SetupPath\Ucmdx64.msi /q
```

In the last two examples, **SetupPath** refers to the location in which the extracted **.msi** file resides. The exact path varies depending on the user account that performed the install and the Windows version.

For information on how the installation determines **SetupPath**, see Step 2 of the Installing via Command Line procedures.

Migrating between 32- and 64-bit Universal Agent for Windows Installs

- 32- and 64-bit Universal Agent for Windows Crossgrades
- Performing a 32- to 64-bit Universal Agent for Windows Crossgrade
  - Executing a 32- to 64-bit Universal Agent Crossgrade from the Command Line
- Performing a 64- to 32-bit Universal Agent for Windows Crossgrade
  - Executing a 64- to 32-bit Universal Agent Crossgrade from the Command Line

32- and 64-bit Universal Agent for Windows Crossgrades

The Universal Agent for Windows installation supports migrations between 32- and 64-bit versions.

This migration is not an upgrade in the strictest sense, because the installed location of some application files is likely to change. However, because the migration will preserve configuration settings, it does possess some characteristics of an upgrade. Therefore, migrations from a 32-bit to a 64-bit Universal Agent installation (or from a 64-bit to a 32-bit installation) are referred to throughout this document as a 32-bit / 64-bit crossgrade.

The Universal Agent installation supports the following 32-bit / 64-bit crossgrade scenarios:

- From any 32-bit Universal Products (3.2.0.0 and later) or Universal Agent install to a 64-bit Universal Agent install.
- From any 64-bit Universal Agent install to a 32-bit Universal Agent 5.2.0 or later install.

If a 64-bit Universal Agent package currently is installed, and you want to return to a 32-bit install prior to Universal Agent 6.2.x, you first must remove the 64-bit Universal Agent installation and do a clean install of the 32-bit package. No configuration settings are preserved in this scenario.

Note
A crossgrade from a 32-bit Universal Products package prior to 3.2.0.0 is not currently supported. Configuration information for releases prior to 3.2.0.0 was stored in the Windows system registry, and crossgrades to a 64-bit Universal Agent install only preserve options that are stored in text-based configuration files.

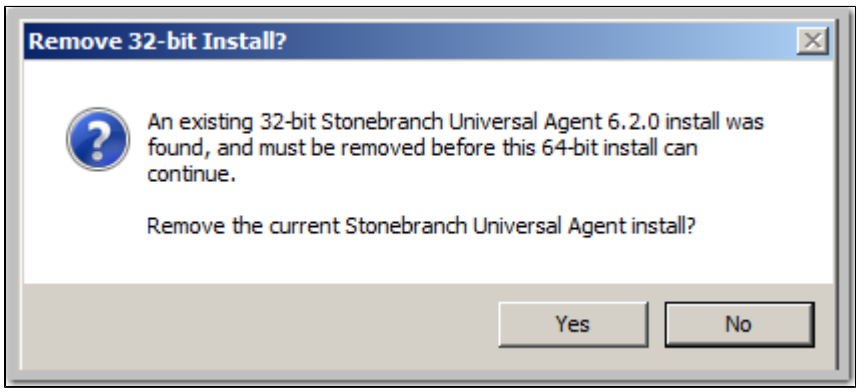
To migrate from a 32-bit Universal Products install that is older than 3.2.0.0, first do an upgrade to the 32-bit Universal Agent package (which will dump configuration options stored in the registry to our text-based .conf files), and then do the 64-bit crossgrade.

Performing a 32- to 64-bit Universal Agent for Windows Crossgrade

To begin a 32- to 64-bit Universal Agent for Windows crossgrade:

Step 1 Execute the 64-bit install package as described in Installing Universal Agent via the Graphical Interface.

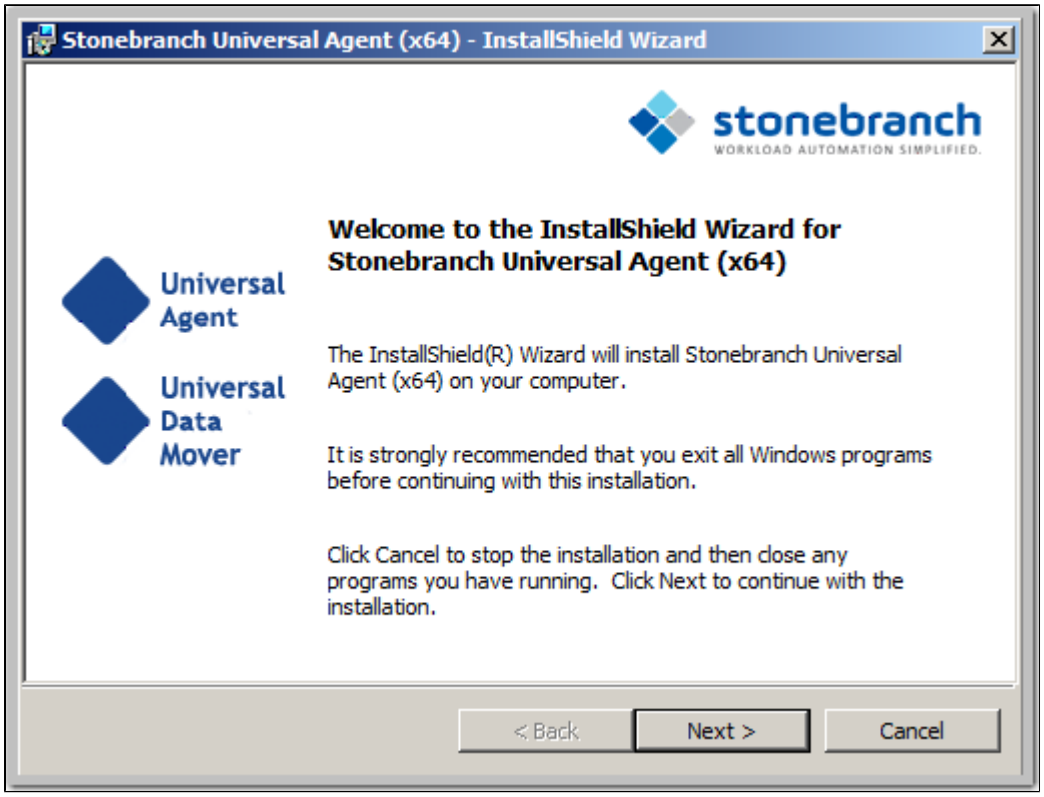
If you are performing an interactive install and the Universal Agent install detects an existing 32-bit Universal Products or Universal Agent installation, the install will display the following message:



Step 2

Click **<Yes>** to save all configured settings and remove the existing 32-bit install.

Once the removal of the 32-bit install is complete, the 64-bit install will resume with the following dialog:



Notice the **(x64)** label, which identifies this as the 64-bit Universal Agent install. Because no Universal Agent install exists on the system at this point, the installation will proceed as would a clean install.

This means some information - the Universal Broker service account and Universal server component working directories, for example - will need to be reentered or specified via command line properties (see Windows Installer Command Line Parameters).

Upon completion of a 32- to 64-bit Universal Agent crossgrade:

- User-configured settings stored in Universal component **.conf** files are retained.
- Some configuration settings that are dependent upon install locations will be updated (for example, trace file directories).
- The location of the Broker spool directory is preserved. The configuration files will still point to the 32-bit install location (for example, **`C:\Program Files (x86)\Universal\spool\ubroker`**) so that no persistent data (for example, Universal Event Subsystem records) is lost during the crossgrade.
- Any Universal Agent components that were installed to the **`%SystemRoot%\SysWow64`** will now reside as native 64-bit applications in the **`%SystemRoot%\System32`** directory.
- The icon for the Universal Configuration Manager Control Panel applet is still compiled as a 32-bit application and will continue to appear in the 32-bit Control Panel items. The Universal Configuration Manager can continue to be used to configure 64-bit Universal Agent components.

Executing a 32- to 64-bit Universal Agent Crossgrade from the Command Line

A 32- to 64-bit Universal Agent crossgrade can be executed from the command line, but it must be run with at least a reduced user interface. A reduced user interface is specified using the **/qr** Windows installer command line option. A reduced user interface still displays some installation dialogs, but allows the install to run unattended (that is, without user interaction).

If a 64-bit install detects an existing 32-bit installation, it automatically will perform a crossgrade without requiring a response to the **Remove 32-bit Install?** dialog displayed for fully interactive installs.

The command to execute a 32- to 64-bit Universal Agent crossgrade might look similar to this:

```
sb-6.2.0.0-windows-x64.exe /s /v"/qr"
```

In this command:

- **/s** option instructs the Windows installer initialization to execute silently.
- **/v** option identifies the start of the Windows installer command line switches.
- **/qr** instructs the installer to execute with a reduced user interface.

Attempts to perform a completely silent crossgrade from the command line using the **/qn** or **/qb** Windows installer switch will fail.

New installs and 64-bit upgrades from older 64-bit installations still can be done silently.

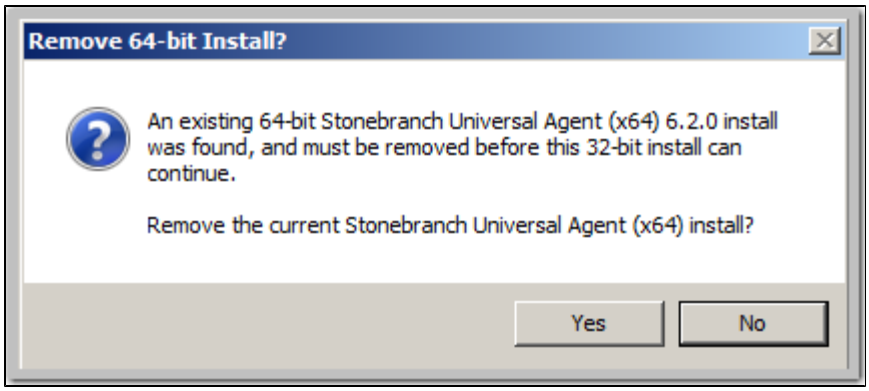
See Installing Universal Agent via the Command Line for more information regarding command line installs.

Performing a 64- to 32-bit Universal Agent for Windows Crossgrade

Support for 64- to 32-bit crossgrades is only available for 32-bit Universal Agent or later install packages. Earlier versions of the 32-bit Universal Products install do not have the ability to detect a 64-bit Universal Agent install.

To return to a 32-bit Universal Products installation (that is, a version prior to 6.2.0.0) when a 64-bit Universal Agent package is currently installed, you must uninstall the 64-bit Universal Agent package first and then do a clean install of the earlier 32-bit package.

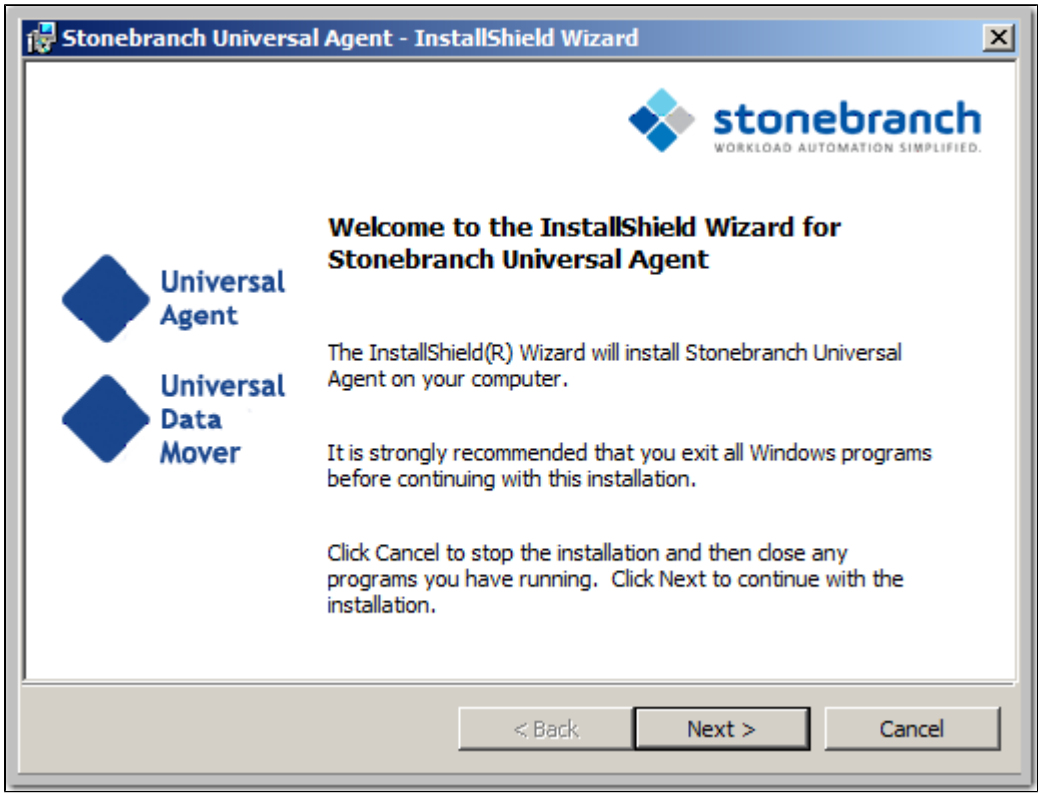
To begin a 64- to 32-bit Universal Agent for Windows crossgrade:

Step 1	Execute the 32-bit Universal Agent install package as described in Universal Agent for Windows - Installation Procedures. If you are performing an interactive install and the Universal Agent install detects an existing 64-bit Universal Products or Universal Agent installation, the install will display the following message: 

Step 2

Select **<Yes>** to save all configured settings and remove the existing 64-bit install.

Once the removal of the 64-bit install is complete, the 32-bit install will resume with the following dialog:



Notice the absence of the **(x64)** label, which identifies this as the 32-bit Universal Agent install.

Because no Universal Agent install exists on the system at this point, the installation will proceed as would a clean install. This means some information - the Universal Broker service account and Universal server component working directories, for example - will need to be reentered or specified via command line properties (see Windows Installer Command Line Parameters).

Upon completion of a 64- to 32-bit Universal Agent crossgrade:

- User-configured settings stored in Universal Agent component **.conf*** files are retained.
- Some configuration settings that are dependent upon install locations will be updated (for example, trace file directories).
- The location of the Broker spool directory is preserved. The configuration files will still point to the 32-bit install location (for example, **`C:\Program Files (x86)\Universal\spool\ubroker`**) so that no persistent data (for example, Universal Event Subsystem records) is lost during the crossgrade.
- Any Universal Agent components that were installed to the **`%SystemRoot%\SysWow64`** will now reside as native 64-bit applications in the **`%SystemRoot%\System32`** directory.
- The icon for the Universal Configuration Manager Control Panel applet is still compiled as a 32-bit application and will continue to appear in the 32-bit Control Panel items. The Universal Configuration Manager can continue to be used to configure 64-bit Universal Agent components.

Executing a 64- to 32-bit Universal Agent Crossgrade from the Command Line

A 64- to 32-bit Universal Agent crossgrade can be executed from the command line, but it must be run with at least a reduced user interface.

A reduced user interface is specified using the **/qr** Windows installer command line option. A reduced user interface still displays some installation dialogs, but allows the install to run unattended (that is, without user interaction).

If a 32-bit install detects an existing 64-bit installation, it automatically will perform a crossgrade without requiring a response to the **Remove 64-bit Install?** dialog displayed for fully interactive installs.

The command to execute a 64- to 32-bit Universal Agent crossgrade might look something like this:

```
sb-6.2.0.0-windows-i386.exe /s /v"/qr"
```

In this command:

- **/s** option instructs the Windows installer initialization to execute silently
- **/v** option identifies the start of the Windows installer command line switches
- **/qr** instructs the installer to execute with a reduced user interface.

Attempts to perform a completely silent crossgrade from the command line using the **/qn** or **/qb** Windows installer switch will fail.

New installs and 32-bit upgrades from older 32-bit installations still can be done silently.

See Installing Universal Agent via the Command Line for more information regarding command line installs.

32-Bit Universal Agent for Windows on 64-Bit Windows Systems

- Overview
- 64-bit Windows Systems
- Modifying the Working Folder for Universal Server Components
- Applications Installed in the Windows System Folder
  - Example 1
  - Example 2
- Identifying 32- and 64-bit Universal Agent Installations

Overview

The information on this page applies only to 32-bit Universal Agent packages installed on 64-bit Windows systems.

A native 64-bit Universal Agent for Windows package, which was made available starting with the 5.2.0 release, requires no special handling on 64-bit Windows systems.

64-bit Windows Systems

All Universal Agent components have been tested and verified on the 64-bit editions of the following Windows systems:

- Windows Server 2003 SP1 and higher
- Windows XP SP3
- Windows Vista
- Windows Server 2008
- Windows 7
- Windows Server 2008 R2

This page describes some modifications that may need to be made to the default installation options to ensure that the installed Universal Agent function correctly.

Modifying the Working Folder for Universal Server Components

On 64-bit Windows editions, the default installation folder for 32-bit applications is **`C:\Program Files (x86)`**.

The default installation folder for 32-bit Windows applications developed by Stonebranch is **`C:\Program Files (x86)\Universal`**. The Universal Agent installation package should have no problems installing its applications to this directory.

Supported Universal Agent releases prior to 5.2.0, installed on 64-bit editions of Windows prior to XP SP2 or Server 2003 SP1, may receive this error - **'C:\Program' is not recognized as an internal or external command, operable program or batch file** - under the following circumstances:

- Universal Command Manager is run using the **-script** option.
- Universal Event Monitor Server invokes an event handler that executes a script.

Note
Universal Agent releases starting with 5.2.0 are only supported on Windows versions where the above error is not expected to occur.

The error above is a result of the way that scripts are prepared for execution and the way that the Windows command shell interpreter reads a quoted string containing special characters (that is, **& < > ( )** @ **^ |**).

Universal Command and Universal Event Monitor prepare scripts for execution by writing the script statements to a temporary **.bat** file in the application's working directory. A command line statement containing that working directory's path is then constructed to execute the **.bat** file. By default, this path includes the **`Program Files (x86)`** directory. Because the path includes one of the special characters listed above, the Windows command shell interpreter incorrectly parses the path, which results in the error above.

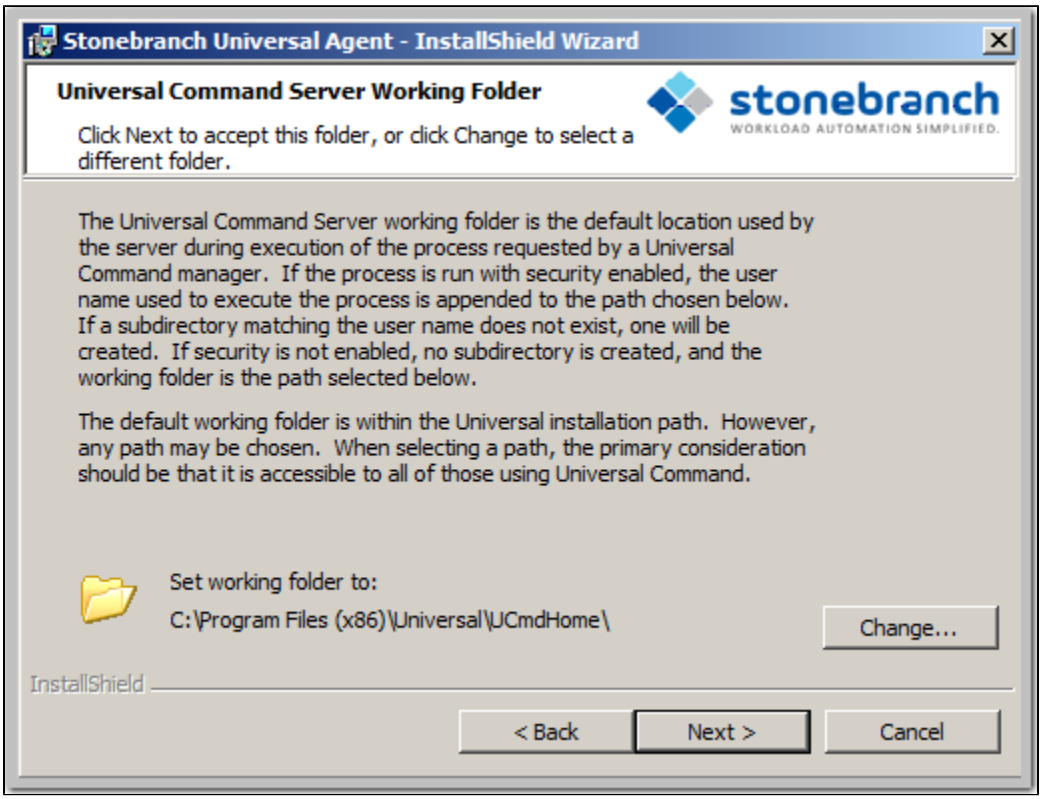
To resolve this issue, it is necessary to change the value (or location) of the Universal Command Server and/or Universal Event Monitor Server working directory.

Either of the following changes can be made:

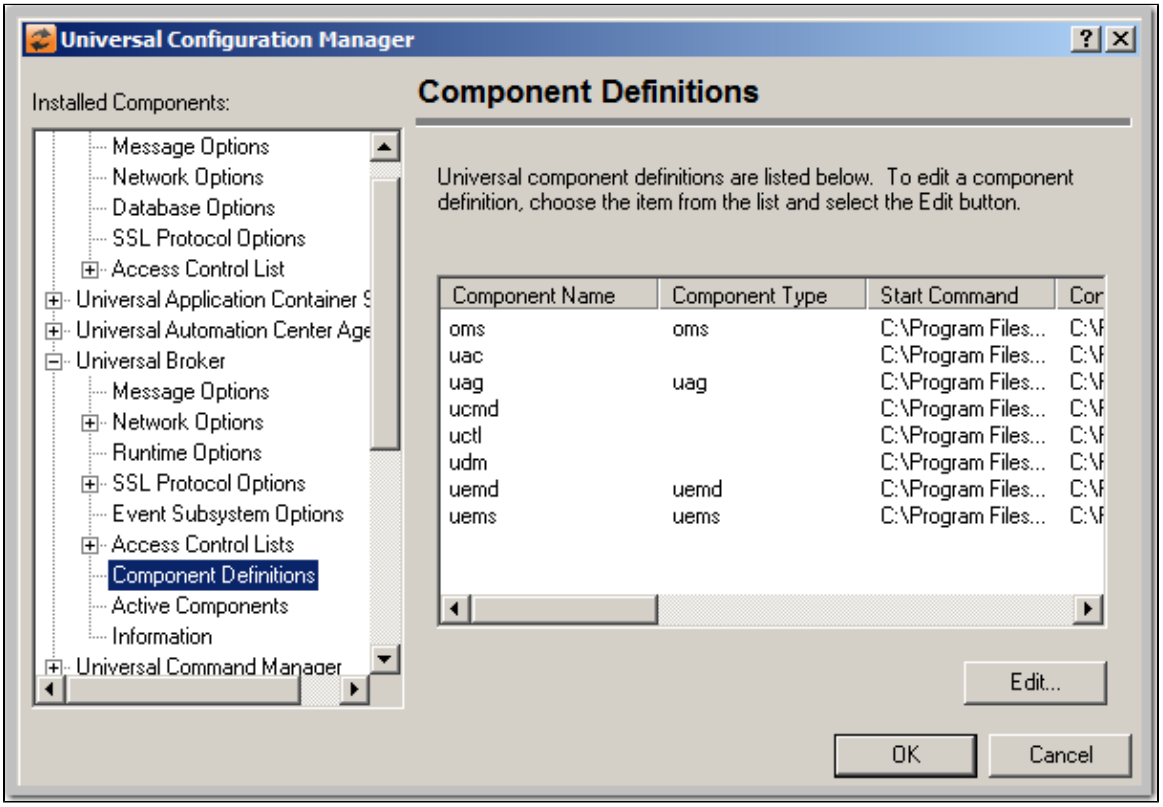
- If short path names are permitted on your system, use the **dir /x** command from the command prompt to find the short name of the **`Program Files (x86)`** directory (for example, **`Progra~2`**). Use this value as part of the working directory's path name.
- Change the working directory to a location outside of the default installation path (for example, **`C:\Universal`**). This new location can contain spaces, but it cannot contain any of the special characters listed above.

The changes can be made either of two ways:

1. During installation.



2. Any time after installation via the Universal Configuration Manager, on the Component Definitions page for Universal Broker.



Applications Installed in the Windows System Folder

The 32-bit Universal Agent package installs several command-line applications in the Windows system folder. The default system folder for 32-bit applications installed on 64-bit Windows editions is the **%SystemRoot%\SysWOW64** directory (for example, **C:\Windows\SysWOW64**).

The following table identifies the affected Universal Agent applications.

File Name	Description
ucert.exe	Universal Certificate
ucmd.exe	Universal Command Manager
ucopy.exe	Universal Copy
uctl.exe	Universal Control Manager
udm.exe	Universal Data Mover Manager
ueld.exe	Universal Event Log Dump Utility
uem.exe	Universal Event Monitor Manager
uemload.exe	Universal Event Monitor Load Utility
uencrypt.exe	Universal Encrypt Utility
umet.exe	Universal Message Translator
uquery.exe	Universal Query
urc.exe	Universal Return Code Utility

These applications can be executed using either the:

- 32-bit command shell (**%SystemRoot%\SysWOW64\cmd.exe**)
- Default 64-bit command shell (**%SystemRoot%\System32\cmd.exe**).

By default, the **%SystemRoot%\SysWOW64** directory is not part of the system path. Therefore, to execute the above command line applications using the 32-bit command shell, either:

- Directory must be added to the **PATH** environment variable.
- Complete path to the application and/or the 32-bit command shell must be specified.

Example 1

To execute UCOPY in the default 64-bit command shell, issue the following command:

```
%SystemRoot%\SysWOW64\ucopy
```

Example 2

To execute UCOPY within the 32-bit command shell, use the following:

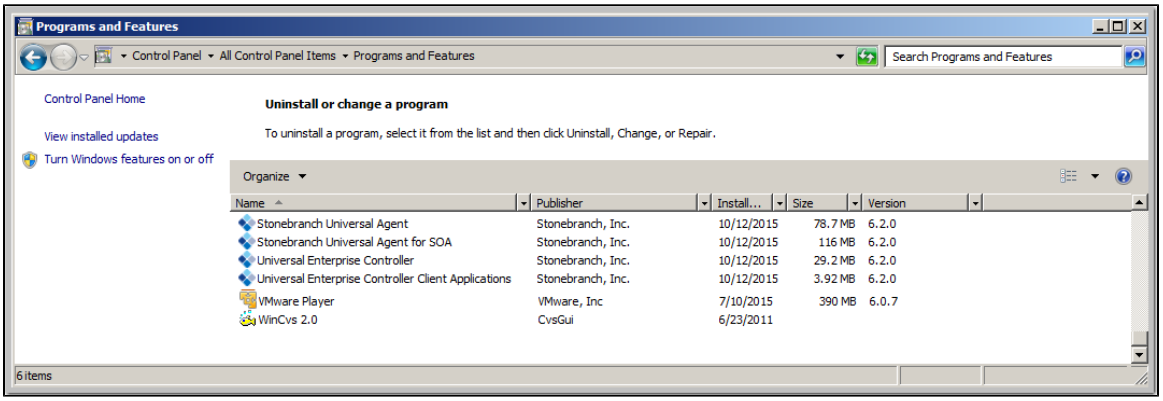
```
%SystemRoot%\SysWOW64\cmd.exe /C %SystemRoot%\SysWOW64\ucopy
```

Identifying 32- and 64-bit Universal Agent Installations

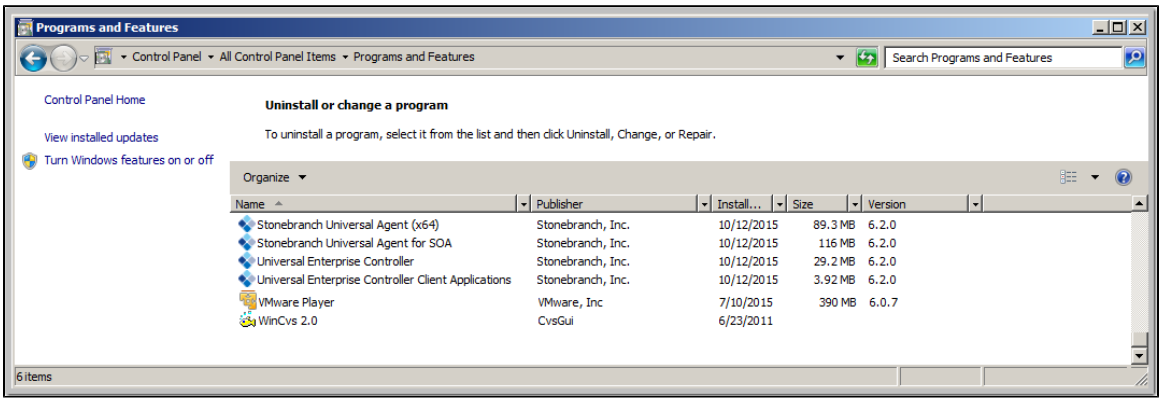
To help identify whether a 32- or 64-bit Universal Agent installation exists on your system, look for the **(x64)** indicator in the package name or file version information. If it is present, a 64-bit version of Universal Agent for Windows package is installed. If it is missing, the 32-bit version of the Universal Agent for Windows package is installed.

For example, the screen shots below show 32- and 64-bit Universal Agent for Windows installs as they appear in Programs and Features on a Windows Server 20008 system.

32-bit Universal Agent Install



64-bit Universal Agent Install (with x64 label)



The **(x64)** label also appears in a component's version information, which is obtained by executing the component with the **-v** command line option.

For example, a 64-bit Universal Command Manager displays version information similar to that shown below when the **-v** option is passed to **`ucmd.exe`**:

```
C:\Users\Administrator>ucmd -v
ucmd 6.2.0 Level 0 Release Build 100 (x64) 06/22/15 17:24:49
xps 6.2.0 Level 0 Build 100 06/22/15 17:18:02
(c) Copyright 2000-2011 Stonebranch, Inc. All rights reserved.
```

The 32-bit version of Universal Command Manager displays the same version information, but the **(x64)** label is not shown.

Universal Agent for Windows - File Inventory Lists

- Universal Agent for Windows - File Inventory Lists
- Universal Automation Center Agent
- Universal Broker
- Universal Command Manager
- Universal Command Server
- Universal Connector
- Universal Control Manager
- Universal Control Server
- Universal Data Mover Manager
- Universal Data Mover Server
- Universal Event Monitor Manager
- Universal Event Monitor Server
- Universal Configuration Manager
- Universal Copy
- Universal Encrypt
- Universal Event Log Dump
- Universal Message Translator
- Universal Products Install Merge
- Universal Query
- Universal Spool Utilities
- Universal Message Service (OMS)
- Universal Controller Command Line Interface (CLI)
- System Files

Universal Agent for Windows - File Inventory Lists

The Universal Agent 6.2.x for Windows package includes the files required for the following components / utilities:

- Universal Automation Center Agent
- Universal Broker
- Universal Command Manager
- Universal Command Server
- Universal Configuration Manager
- Universal Connector
- Universal Control Manager
- Universal Control Server
- Universal Copy
- Universal Data Mover Manager
- Universal Data Mover Server
- Universal Encrypt
- Universal Event Log Dump
- Universal Event Monitor Manager
- Universal Event Monitor Server
- Universal Products Install Merge
- Universal Query
- Universal Spool Utilities
- Universal Message Service (OMS)
- Universal Controller Command Line Interface (CLI)

Universal Broker and Universal Configuration Manager always are installed. Other components are installed as desired. If any of the components already are installed, Windows Installer will upgrade them to the latest version.

This page lists the files installed with each Universal Agent for Windows component. The file paths specified are relative to the root installation directory (for example, **C:\Program Files\Universal**) that was specified during the installation, except **%ALLUSERSPROFILE%**, which resolves as documented.

Items shown with a path of System32 are installed in the Windows system folder. The actual name of this directory depends on the Windows version. It may also depend on whether the 32-bit or the 64-bit Universal Agent package is installed:

- For all supported 32-bit Windows editions, the path is **\Windows\System32**.
- For all supported 64-bit Windows editions when the 64-bit Universal Agent package is installed, the path is **\Windows\System32**.
- For all supported 64-bit Windows editions when the 32-bit Universal Agent package is installed, the path is **\Windows\SysWow64**.

Universal Automation Center Agent

File	Description

UAGSrv\bin\uagsrv.exe	Universal Automation Center Agent Application program.
UAGSrv\bin\uagscfg.dll	Universal Automation Center Agent configuration control.
UAGSrv\bin\uagscfg.hlp	Universal Automation Center Agent bootstrap configuration control help file.
UAGSrv\bin\ops_copyfile.exe	Universal Automation Center Agent copy utility.
UAGSrv\bin\ops_scan.exe	Universal Automation Center Agent scan utility.
UAGSrv\bin\opsmerge.vbs	Visual Basic script used to migrate an existing 1.5, 1.6, or 1.7 Opswise Agent to UAG.
UAGSrv\bin\ftp\bin\cygwin1.dll	Cygwin library, version 1005.25.0.0 (provided to ensure compatibility with the Cygwin FTP client).
UAGSrv\bin\ftp\bin\ftp.exe	Cygwin FTP client, version 1.3.2.
UAGSrv\bin\sftp\bin\cygcrypto-0.9.8.dll	Cygwin encryption library
UAGSrv\bin\sftp\bin\cyggcc_s-1.dll	Cygwin GCC library
UAGSrv\bin\sftp\bin\cygssp-0.dll	Cygwin GCC Static Stack Protection library
UAGSrv\bin\sftp\bin\cygwin1.dll	Cygwin library, version 1007.5.0.0
UAGSrv\bin\sftp\bin\cygz.dll	Cygwin Zlib library
UAGSrv\bin\sftp\bin\expect.exe	Interactive program automation utility
UAGSrv\bin\sftp\bin\sftp.exe	Secure FTP client
UAGSrv\bin\sftp\bin\sftp.exp	SFTP sample script for Expect utility
UAGSrv\bin\sftp\bin\ssh.exe	Secure shell client
UAGSrv\bin\sftp\library\init.tcl	TCL initialization script
UAGSrv\samples	UAG sample directory.
UAGSrv\samples\OPSWISE-MIB.txt	MIB file.
UBroker\tmpl\uagcmp	Template file for the UAG component definition.
UBroker\tmpl\uagcfg	Template file for the UAG configuration.
nls\uagmceng.umf	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uags.conf	UAG configuration file.
%ALLUSERSPROFILE%\Application Data\Universal\comp\uag	UAG component definition file.

Universal Broker

File	Description
UBroker\bin\ubroker.exe	Console application program.
UBroker\bin\ubrsvc.exe	Windows service program.
UBroker\bin\ubrcfg.dll	Used by Universal Configuration Manager to manage Broker properties.
UBroker\bin\ubrbscfg.dll	Used by the Universal Configuration Manager to manage bootstrap options when the local Broker is running in managed mode.
UBroker\bin\ubrbscfg.hlp	Universal Broker bootstrap configuration help file.
UBroker\bin\ubrcfg.hlp	Universal Broker configuration help file.
UBroker\bin\ubrdbrec.bat	Recovers Universal Agent database files.
UBroker\tmpl	XML template files used by I-Management Console for remotely configuring Universal Agent.

nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\ussmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uacl.conf	Universal Access Control List (ACL) configuration file **.%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\conf\ubroker.conf	Universal Broker configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Command Manager

File	Description
System32\ucmd.exe	Application program.
UCmdMgr\bin\ucmccfg.dll	Used by Universal Configuration Manager to manage Universal Command Manager properties.
UCmdMgr\bin\ucmccfg.hlp	Universal Command Manager configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\ucmmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\ucmd.conf	Universal Command Manager configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Command Server

File	Description
UCmdSrv\bin\ucmsrv.exe	Application program.
UCmdSrv\bin\ucmscfg.dll	Used by Universal Configuration Manager to manage Universal Command Server properties.
UCmdSrv\bin\ucmscfg.hlp	Universal Command Server configuration help file.
System32\urc.exe	Universal Return Code utility.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\ucmmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\comp\ucmd	Universal Command Server component definition file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\conf\ucmds.conf	Universal Command Server configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Connector

File	Description
\USap\bin\usap.exe	Universal Connector application program.
\USap\bin\uspcfg.dll	Used by Universal Configuration Manager to manage Universal Connector properties.
\USap\bin\uspcfg.hlp	Universal Connector configuration control context-sensitive help file.
\USap\bin\librfc32.dll	SAP Remote Function Call (RFC) library, v7200. • For 32-bit packages, the file version is 7200.1.120.7137 • For 64-bit packages, the file version is 7200.0.120.2136.
\UBroker\tmpl\uspcfg	Universal Connector remote configuration template.

\nls\uspmceng.umc	Universal Connector English message catalog.
\nls*.stt	Spoollist translation table files.
%ALLUSERSPROFILE%\Application Data\Universal\conf\usap.conf	Universal Connector configuration file.

Universal Control Manager

File	Description
System32\uctl.exe	Application program.
UCtlMgr\bin\uctccfg.dll	Used by Universal Configuration Manager to manage Universal Control Manager properties.
UCtlMgr\bin\uctccfg.hlp	Universal Control Manager configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\uctmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uctl.conf	Universal Control Manager configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Control Server

File	Description
UCtlSrv\bin\uctsrv.exe	Application program.
UCtlSrv\bin\uctscfg.dll	Used by Universal Configuration Manager to manage Universal Control Server properties.
UCtlSrv\bin\uctscfg.hlp	Universal Control Server configuration help file.
Nls*.utt	Code page files used for text translation between different operating systems and platforms.
Nls\uctmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\comp\uctl	Universal Control Server component definition file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uctls.conf	Universal Control Server configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Data Mover Manager

File	Description
System32\udm.exe	Application program.
UdmMgr\bin\udmccfg.dll	Used by Universal Configuration Manager to manage Universal Data Mover Manager properties.
UdmMgr\bin\udmccfg.hlp	Universal Data Mover Manager configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\udmmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\udm.conf	Universal Data Mover Manager configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Data Mover Server

File	Description

UdmSrv\bin\udmsrv.exe	Application program.
UdmSrv\bin\udmscfg.dll	Used by Universal Configuration Manager to manage Universal Data Mover Server properties.
UdmSrv\bin\udmscfg.hlp	Universal Data Mover Server configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\udmmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\comp\udm	Universal Data Mover Server component definition file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\conf\udms.conf	Universal Data Mover Server configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Event Monitor Manager

File	Description
System32\uem.exe	Application program.
UemMgr\bin\uemccfg.dll	Used by Universal Configuration Manager to manage Universal Event Monitor Manager properties.
UemMgr\bin\uemccfg.hlp	Universal Event Monitor Manager configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\uemmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uem.conf	Universal Event Monitor Manager configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Event Monitor Server

File	Description
System32\uemload.exe	Event definition and event handler database load utility.
UemSrv\bin\uemsrv.exe	Application program.
UemSrv\bin\uemscfg.dll	Used by Universal Configuration Manager to manage Universal Event Monitor Server properties.
UemSrv\bin\uemscfg.hlp	Universal Event Monitor Server configuration help file.
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\uemmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\comp\uemd	Component definition file for the Universal Event Monitor Demand-Driven Server. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\comp\uems	Component definition file for the Universal Event Monitor Event-Driven Server. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uems.conf	Universal Event Monitor Server configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Configuration Manager

File	Description
UCfgMgr\bin\ucfgmgr.cpl	Universal Configuration Manager control panel application.
UCfgMgr\bin\ucfgmgr.cpl	Universal Configuration Manager help file.

Universal Copy

File	Description
System32\ucopy.exe	Utility used for binary file copies. Similar to the UNIX cat command. Installed only if Universal Command Server is installed.

Universal Encrypt

File	Description
System32\uencrypt.exe	Application program. Installed only if Universal Command Manager is installed.

Universal Event Log Dump

File	Description
System32\ueld.exe	Application program.
UEld\bin\uelcfg.dll	Used by Universal Configuration Manager to manage Universal Event Log Dump properties.
UEld\bin\uelcfg.hlp	Universal Event Log Dump configuration help file.
Nls\uelmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\ueld.conf	Universal Event Log Dump configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Message Translator

File	Description
System32\umet.exe	Application program, always installed. Used to map application-specific error messages to error codes.

Universal Products Install Merge

File	Description
UPIMerge\bin\upimerge.exe	Application program, always installed. Provides command line access to the same functionality used by the Universal Agent installation to merge options from a new configuration file into an existing file.

Universal Query

File	Description
System32\uquery.exe	Application program.
UQuery\bin\uqrcfg.dll	Used by Universal Configuration Manager to manage Universal Query properties.
UQuery\bin\uqrcfg.hlp	Universal Query configuration help file.
Nls*.utt	Code page files used for text translation between different operating systems and platforms.
Nls\uqrmceng.umc	English message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uquery.conf	Universal Query configuration file. **%ALLUSERSPROFILE%** is a system environment variable that expands to the All Users profile directory, typically **C:\Documents and Settings\All Users**.

Universal Spool Utilities

File	Description
USpool\bin\uslist.exe	Used to list the contents of Universal Spool files.

USpool\bin\uslrm.exe	Used to remove records from Universal Spool files.
• **USpool\bin\udb_archive.exe** • **USpool\bin\udb_dump.exe** • **USpool\bin\udb_load.exe** • **USpool\bin\udb_recover.exe** • **USpool\bin\udb_stat.exe** • **USpool\bin\udb_upgrade.exe** • **USpool\bin\udb_verify.exe**	Miscellaneous spool file utilities. Should be used only for debugging purposes, and only at the request of Stonebranch, Inc. Customer Support.

Universal Message Service (OMS)

File	Description
omssrv\bin\omssrv.exe	OMS Server.
omssrv\bin\omsadm.exe	OMS Administration utility.
nls\omsmceng.umc	OMS message catalog.
UBroker\tmpl\omscfg	OMS Server configuration template.
UBroker\tmpl\omscmp	OMS Server component definition.
%ALLUSERSPROFILE%\Application Data\Universal\conf\omss.conf	OMS configuration file.
%ALLUSERSPROFILE%\Application Data\Universal\comp\oms	OMS component definition file.

Universal Controller Command Line Interface (CLI)

File	Description
OpsCli\bin\uagcmd	CLI executable command.
OpsCli\bin\ops-change-credentials-password.bat	Changes the runtime password for the specified Universal Controller credentials resource.
OpsCli\bin\ops-change-user-password.bat	Changes the password for the specified Universal Controller User account.
OpsCli\bin\ops-connector-status.bat	Lists the status of one or more Connectors.
OpsCli\bin\ops-export-bulk.bat	Performs a bulk export of all Controller database records.
OpsCli\bin\ops-export-trigger.bat	Performs an export of the specified triggers and any associated records.
OpsCli\bin\ops-import-bulk.bat	Imports Controller database records that were exported with ops-export-bulk.
OpsCli\bin\ops-import-trigger.bat	Imports triggers and associated records that were exported using ops-export-trigger.
OpsCli\bin\ops-manual-setcompleted.bat	Sets Manual task status to Success.
OpsCli\bin\ops-manual-setstarted.bat	Changes Manual task status from Action Required to Started.
OpsCli\bin\ops-resume-agent.bat	Allows a suspended Agent to submit tasks once again.
OpsCli\bin\ops-resume-agent-cluster.bat	Allows a suspended Agent Cluster to execute tasks once again.
OpsCli\bin\ops-resume-agent-cluster-membership.bat	Allows a specified Agent to rejoin specified Agent Cluster.
OpsCli\bin\ops-set-agent-cluster-task-execution-limit.bat	Sets the task execution limit for an Agent Cluster.
OpsCli\bin\ops-set-agent-task-execution-limit.bat	Sets the task execution limit for an Agent.
OpsCli\bin\ops-suspend-agent.bat	Temporarily prevents a specified Agent from submitting tasks.
OpsCli\bin\ops-suspend-agent-cluster.bat	Temporarily prevents the Agents in the specified Cluster from submitting tasks.
OpsCli\bin\ops-suspend-agent-cluster-membership.bat	Temporarily removes an Agent from cluster membership.

OpsCli\bin\ops-task-cancel.bat	Cancels a task.
OpsCli\bin\ops-task-forcefinish.bat	Force Finishes a task.
OpsCli\bin\ops-task-hold.bat	Places a task on hold.
OpsCli\bin\ops-task-launch.bat	Executes a task.
OpsCli\bin\ops-task-list.bat	Lists the specified tasks.
OpsCli\bin\ops-task-release.bat	Releases a held task.
OpsCli\bin\ops-task-rerun.bat	Re-executes the specified task.
OpsCli\bin\ops-task-setpriority.bat	Changes the execution priority of a Started task.
OpsCli\bin\ops-task-skip.bat	Skips the specified task instance.
OpsCli\bin\ops-task-status.bat	Displays the status of the task instance(s) associated with a task.
OpsCli\bin\ops-trigger-disable.bat	Disables the specified trigger(s).
OpsCli\bin\ops-trigger-enable.bat	Enables the specified trigger(s).
OpsCli\bin\ops-trigger-now.bat	Satisfies all conditions in the specified trigger and launches its associated tasks.
OpsCli\bin\ops-trigger-status.bat	Displays the status of the specified trigger(s).
OpsCli\bin\ops-update-resource-limit.bat	Sets the resource limit for a resource.
OpsCli\bin\ops-variable-list.bat	Displays the specified variable's value.
OpsCli\bin\ops-variable-set.bat	Sets the specified variable's value.
OpsCli\bin\uagcmd.exe	CLI Executable binary.
%ALLUSERSPROFILE%\Application Data\Universal\conf\cmdtools.props	Optional CLI configuration file.

System Files

The following files will be installed only if they are newer than the existing file.

The directories shown in the following table are relative to the **%SYSTEMROOT%** directory, where **%SYSTEMROOT%** is an environment variable that resolves to **C:\Windows** on all Windows platforms.

File	Description
System32\asycfilt.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\comcat.dll	Version 4.71.1460.1 of the Microsoft Component Category Manager library.
Microsoft C-Runtime v10.0 [1]	Microsoft C runtime side-by-side assembly, version 10.0.40219.325. For 64-bit packages, version 8.0.50727.762 also is provided to support the SAP RFC library, librfc32.dll.
Microsoft Foundation Classes v10.0.40219.325 [2]	Version 10.0.40219.325 of the Microsoft Foundation Class (MFC) side-by-side assembly.
System32\msiexec.exe	Version 3.1.4000.1823 of the Microsoft Windows Installer (see Windows Installer for more information).
System32\oleaut32.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\olepro32.dll	Version 5.0.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\psapi.dll	Version 4.0.1371.1 of the Microsoft process status library.
System32\stdole2.tlb	Version 2.40.4275.1. This file is one of the components of the Microsoft OLE library.

[1] The Microsoft C-Runtime distribution consists of several files, which are subject to change. Refer to Microsoft documentation for a complete list of files delivered with the specified runtime version.
[2] The Microsoft Foundation Classes (MFC) distribution consists of several files, which are subject to change. Refer to Microsoft documentation for a complete list of files delivered with the specified MFC version.

Universal Enterprise Controller for Windows Installation

Overview

The following information is provided for the installation of Universal Enterprise Controller for Windows:

- Installation Package
- Installation Requirements
- Installation Procedures
- 64-Bit Windows Editions
- Database Configuration
- File Inventory Lists

(For licensing information, see Windows Installation - Licensing.)

Universal Enterprise Controller for Windows - Installation Package

- Components
- Component Compatibility

Components

The Universal Enterprise Controller (UEC) 6.2.x for Windows package includes a single component:

- Universal Enterprise Controller 6.2.x

Note
Installations of Universal Enterprise Controller v110-3 and earlier also included desktop application versions of the Universal Enterprise Controller Client Applications: I-Administrator, I-Activity Monitor, and I-Management Console.

As of v110-4, these client applications are contained in their own, separate installation package.

Component Compatibility

The following table identifies the compatibility of Universal Enterprise Controller 6.2.x for Windows with previous component / product versions.

Component	Compatibility
Universal Enterprise Controller 6.2.x	• Universal Broker 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 2.2.0. • Not compatible with previous versions of Universal Enterprise Controller Client Applications.

The component references pertain to all supported platforms for that version.

Universal Enterprise Controller for Windows - Installation Requirements

Windows Versions

To install the Universal Enterprise Controller, you must have one of the following versions of Windows:

- Windows Server 2003 SP1 and higher
- Windows XP SP3
- Windows Vista
- Windows 7
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2

Additional Requirements

- An account with administrative privileges.
- Possible reboot: a reboot is required if the Windows Installer service is not installed, a version of the Windows Installer prior to 3.1.4000.1823 is installed, or if required files are in use at the time of the installation.
- TCP/IP.
- About 35 megabytes of disk space.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

Universal Enterprise Controller for Windows - Installation Procedures

Overview

The following procedures are provided for the installation and modification of Universal Enterprise Controller (UEC) for Windows:

- Installing UEC via the Graphical Interface
- Modifying a UEC Installation via the Graphical Interface
- Installing UEC via the Command Line
- Modifying a UEC Installation via the Command Line

Note
Modifying a UEC installation refers to the adding / removing of UEC components, repairing a corrupted installation, or removing an installation. To change the installed version of UEC, see Upgrading Universal Agent and Applying Maintenance to Universal Agent.

Installing UEC via the Graphical Interface

Installing Universal Enterprise Controller via the Windows Installer Graphical Interface

To install Universal Enterprise Controller using the Windows Installer graphical interface, perform the following steps:

Step 1	Download the Universal Enterprise Controller for Windows product distribution file, **sb-UECtlr-6.2.0.<level>-windows-i386.exe**, to your work station.

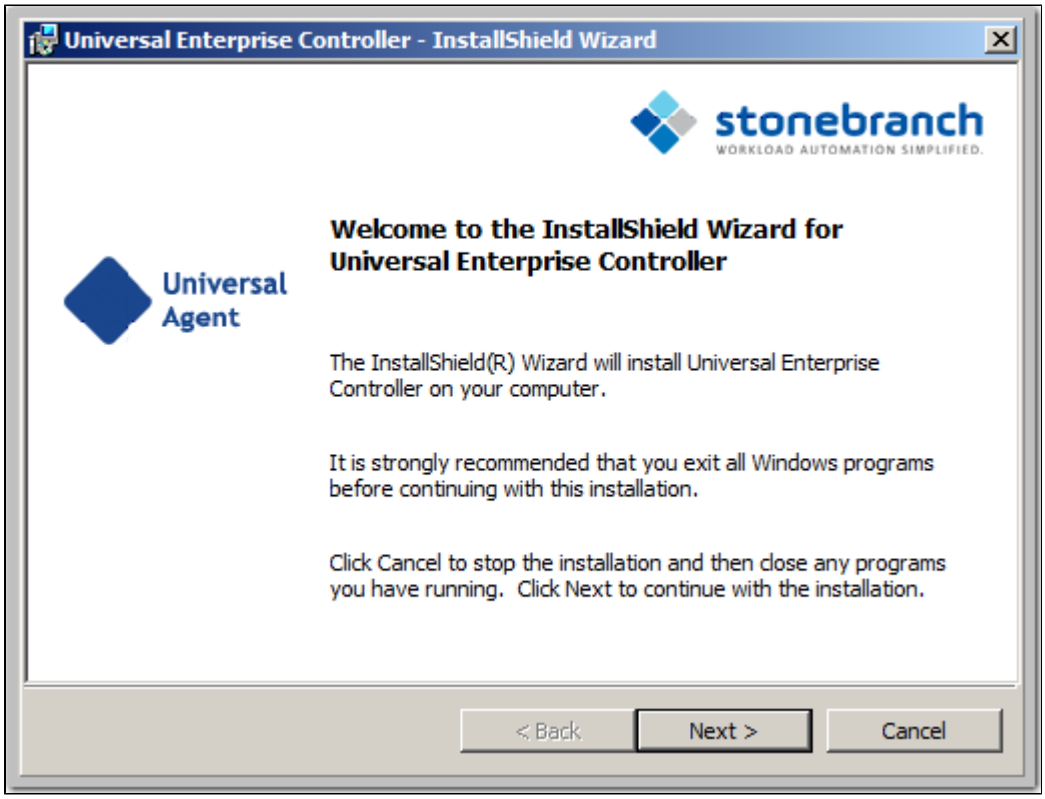
Step 2 Execute the distribution file to extract the files.

> **Note**
> If you already have extracted the files from the distribution file, but cancelled installation in order to separately install Windows Installer, you can simply double-click the extracted Universal Enterprise Controller installation file, **UECtlr.msi**, to begin the installation.

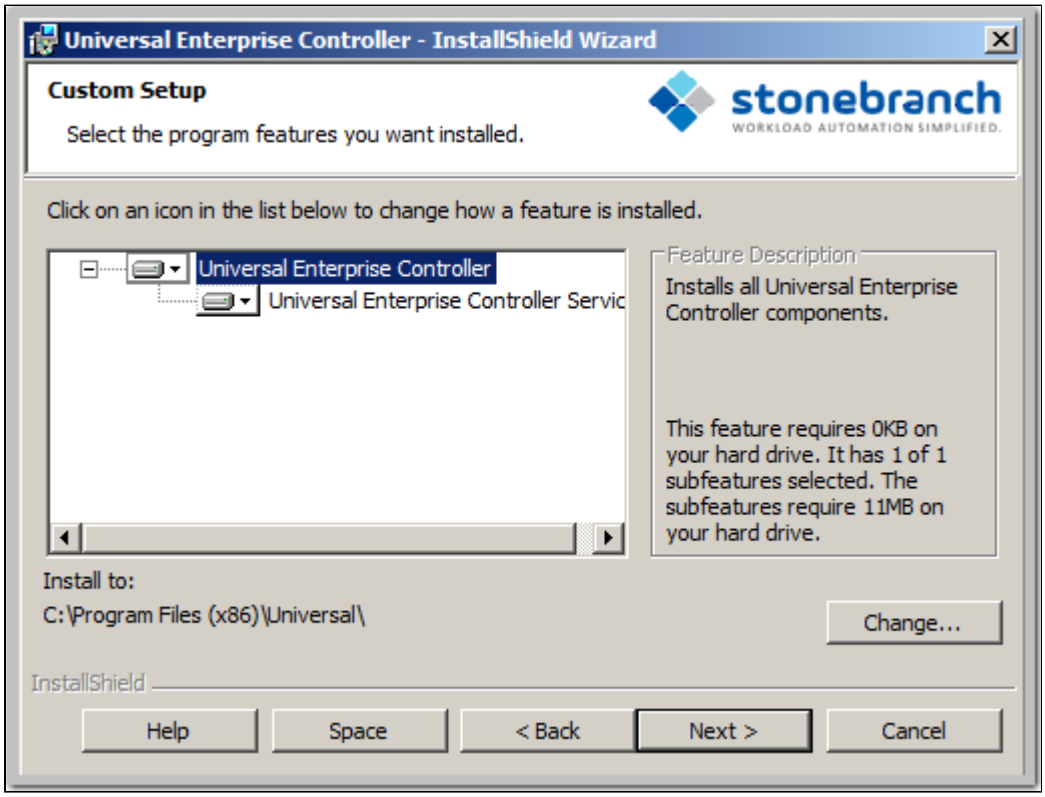
> **Installing over a Remote Desktop Session**
> Starting with Windows Server 2003, Remote Desktop provides distinct session environments for each logged-in user. This means that extraction may use an environment setting that is not available once the Remote Desktop session ends.
>
> For example, the distribution file's default extraction location is based on the **TEMP** environment variable. The location referenced by this variable can change between Remote Desktop sessions, and any files extracted there may not be accessible after the session is closed.
>
> To ensure that extracted files and other required resources are accessible after the initial install, extract the files to a well-known location that is not likely to change between Remote Desktop sessions.
>
> Refer to the Microsoft documentation on the Remote Desktop feature for additional information.

The installation starts after the files are extracted.

It first will verify that your machine meets the minimum system requirements (see Universal Enterprise Controller for Windows - Installation Requirements). If the requirements are met, a Welcome dialog displays.



Step 3 Click the **Next** button. A list of Universal Enterprise Controller components included in the installation package then displays. It is from this list that you can select which components to install.



For a new installation, a drive icon displays next to each item in the list, indicating that the component will be installed.

For an upgrade installation, either of the following icons displays next to an item:

- A drive icon indicates that the component is either:
  - New to the installation and will be installed.
  - Currently is installed and will be upgraded.
- An **X** icon indicates that the component is either:
  - Currently not installed (but previously was available).
  - Previously installed but removed.

A Stonebranch Tip
If the installation detects an existing Universal Enterprise Controller installation, currently installed components may be upgraded.
(Currently, there is no way to specify that the state of a currently installed component remain unchanged.)
If a component is selected for installation, and the version of the installed component is below the version of the component being installed, the installed component will be replaced by the component being installed.
If a component is not selected for installation (that is, the **X** icon is selected), and it currently is installed, the new installation will remove that component.

Step 4 The previous figure shows that all Universal Enterprise Controller components will be installed in their respective directories under the **`C:\Program Files\Universal`** directory.

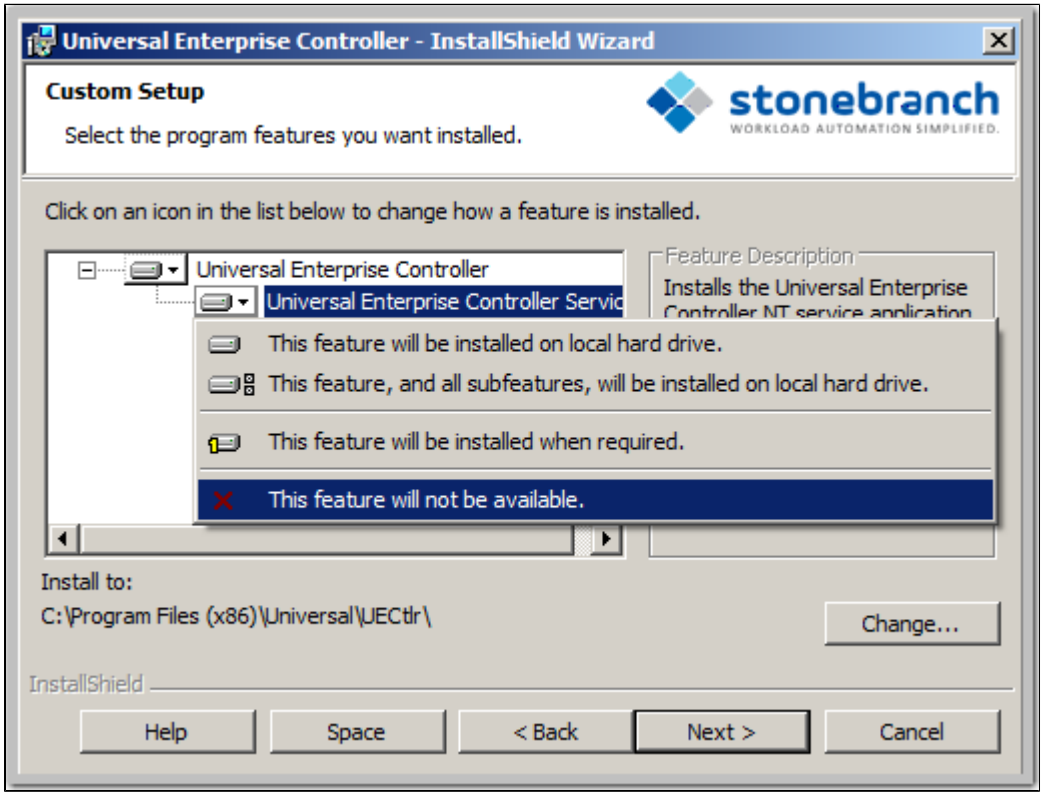
1. If you want to select a different location, click the **Change...** button.
2. If you want to check the amount of disk space required for the installation, and the amount of available disk space on the selected directory, click the **Space** button.

Step 5

If you do not want to install a component:

1. Click the drive icon next to that component name.
2. From the drop\-down list that displays, select the X icon to mark the component as one not to be installed.

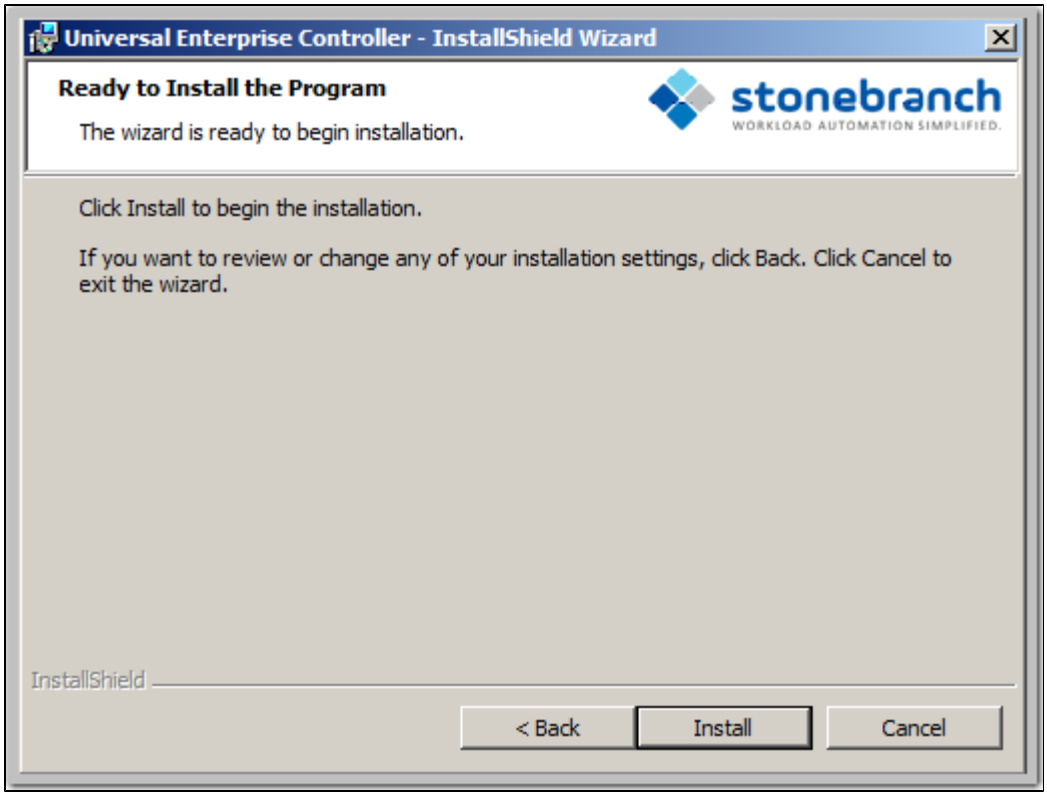
For example, the following figure indicates that the Universal Enterprise Controller Service has been selected to not be installed.



Step 6

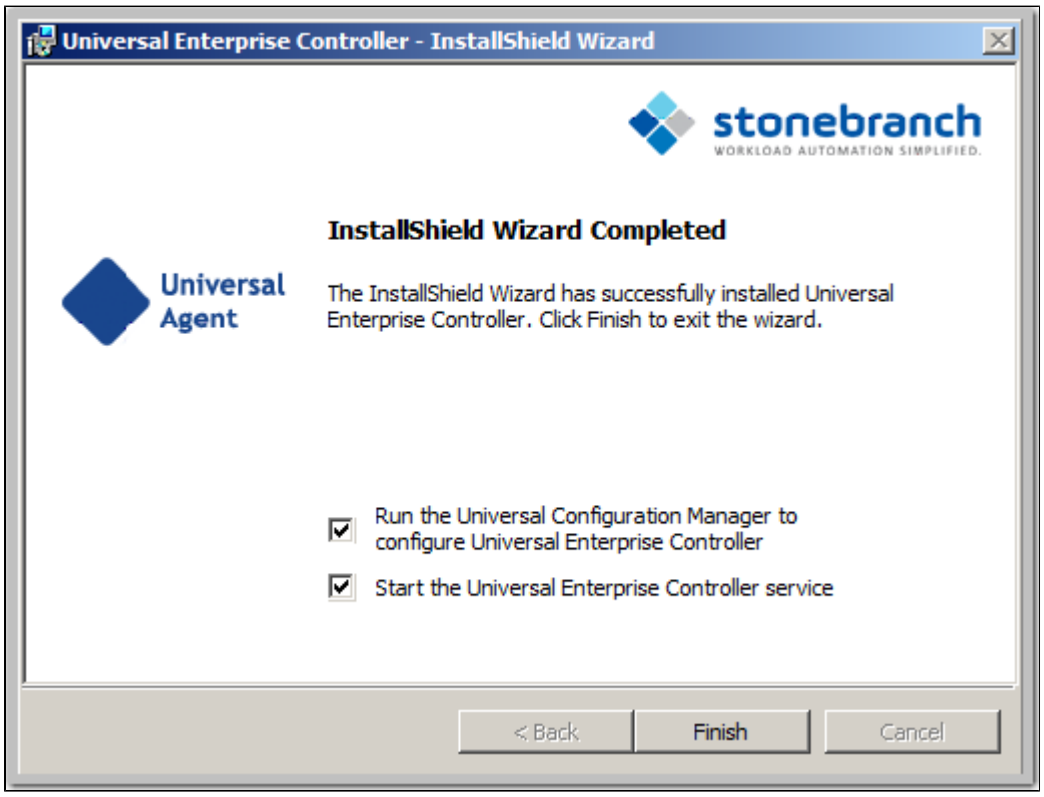
When you have selected the components (and their installation destinations) that you want to install, click the **Next>** button to continue the installation process.

When the installation is ready to begin, the Ready to Install dialog displays.



Click the **Install** button to begin the installation or click the **<Back** button to return to change information on any of the previous dialogs.

When the installation completes successfully, the Installation Complete dialog displays.



Step 7	If the Universal Enterprise Controller service was installed, the following options display on this dialog: • **Run the Universal Configuration Manager** • **Start the Universal Enterprise Controller service** License information must be entered into the Universal Enterprise Controller service's configuration before the service application will run. Select both of these options to enter the license information and then start the Universal Enterprise Controller service.
Step 8	Click the **Finish** button to exit Windows Installation.

Modifying a UEC Installation via the Graphical Interface

- Modifying a Universal Enterprise Controller Installation via the Windows Installer Graphical Interface
- Adding or Removing Universal Enterprise Controller Components
- Repairing a Corrupted Universal Enterprise Controller Installation
- Removing a Universal Enterprise Controller Installation
  - Un-Installed Files

Modifying a Universal Enterprise Controller Installation via the Windows Installer Graphical Interface

After Universal Enterprise Controller is installed, Windows Installer can be run as many times as needed to modify the installation.

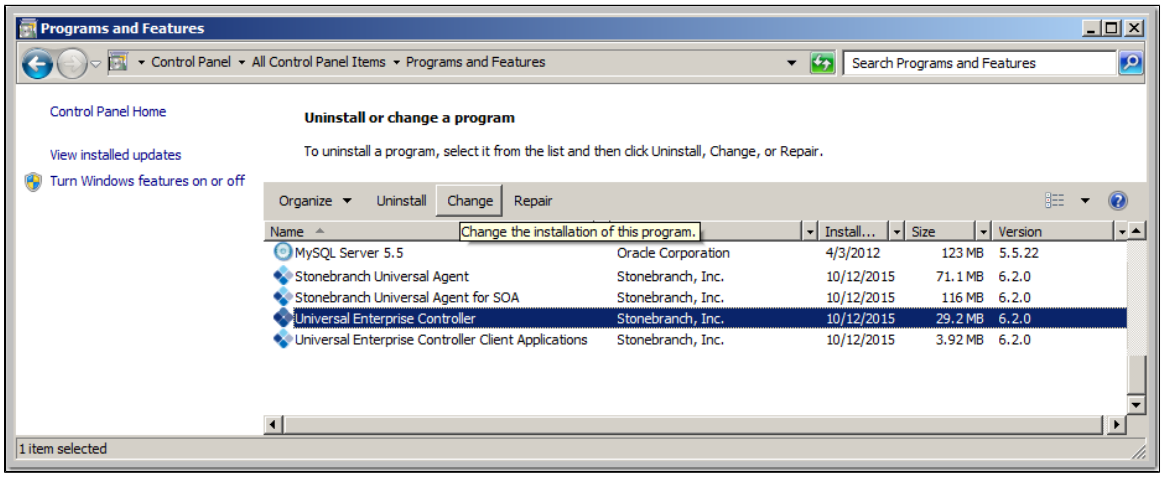
These installation modifications are:

- Adding or Removing Universal Enterprise Controller Components
- Repairing a Corrupted Universal Enterprise Controller Installation
- Removing a Universal Enterprise Controller Installation

Adding or Removing Universal Enterprise Controller Components

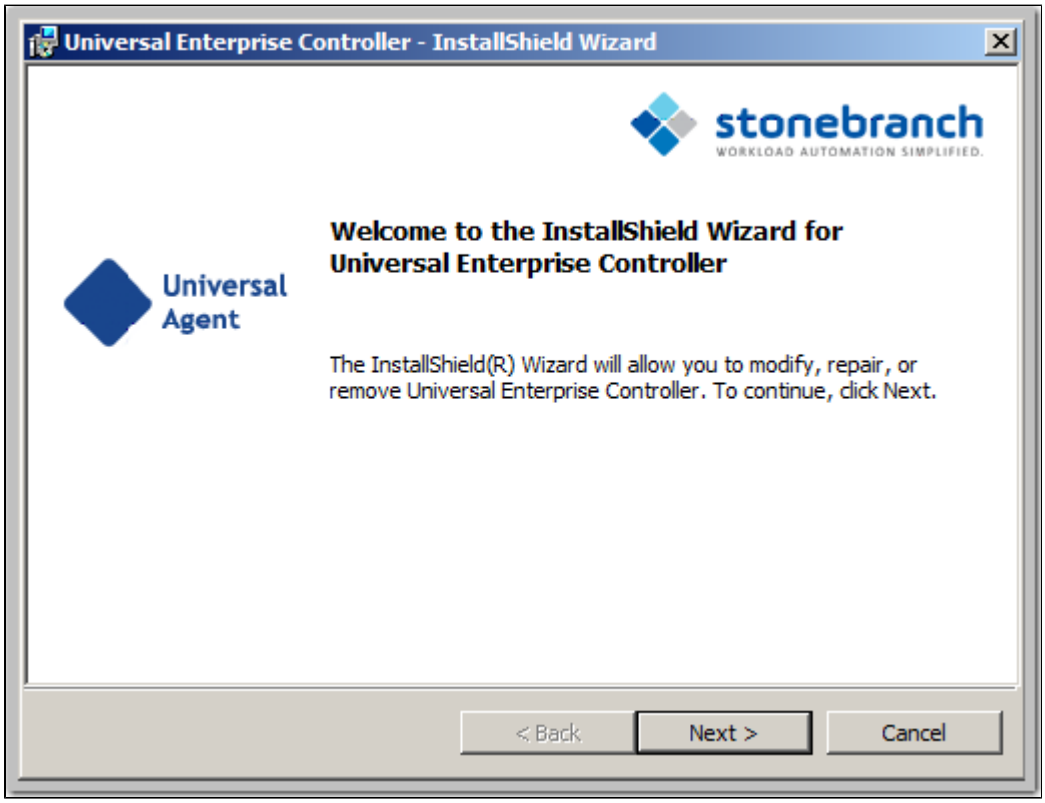
To add or remove components from a Universal Enterprise Controller installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

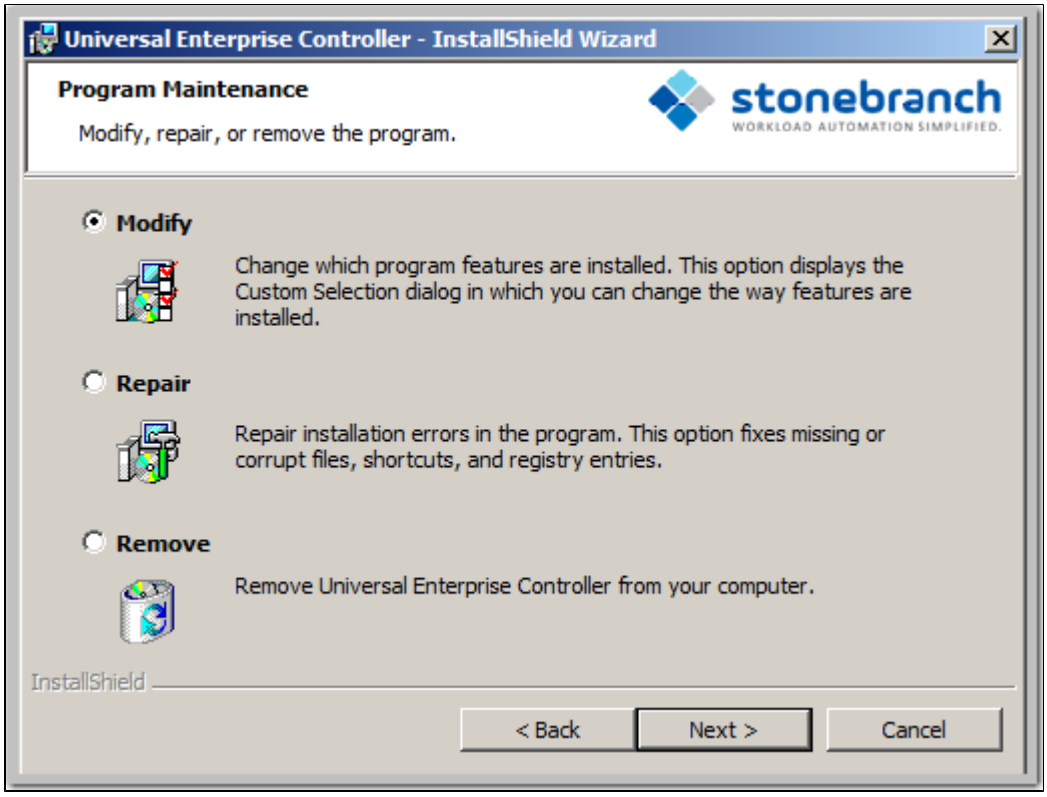


Windows Server 2003
If you are using Windows Server 2003, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller** and click **Change** to start Windows Installer. The Welcome dialog displays.

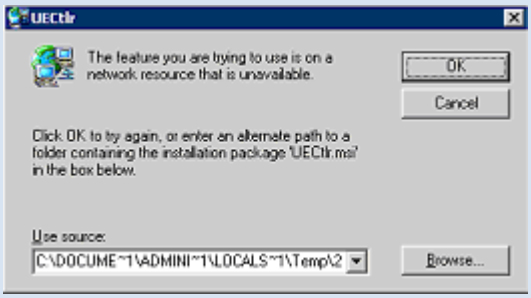


Step 4 Click the **Next>** button. The Program Maintenance dialog displays.



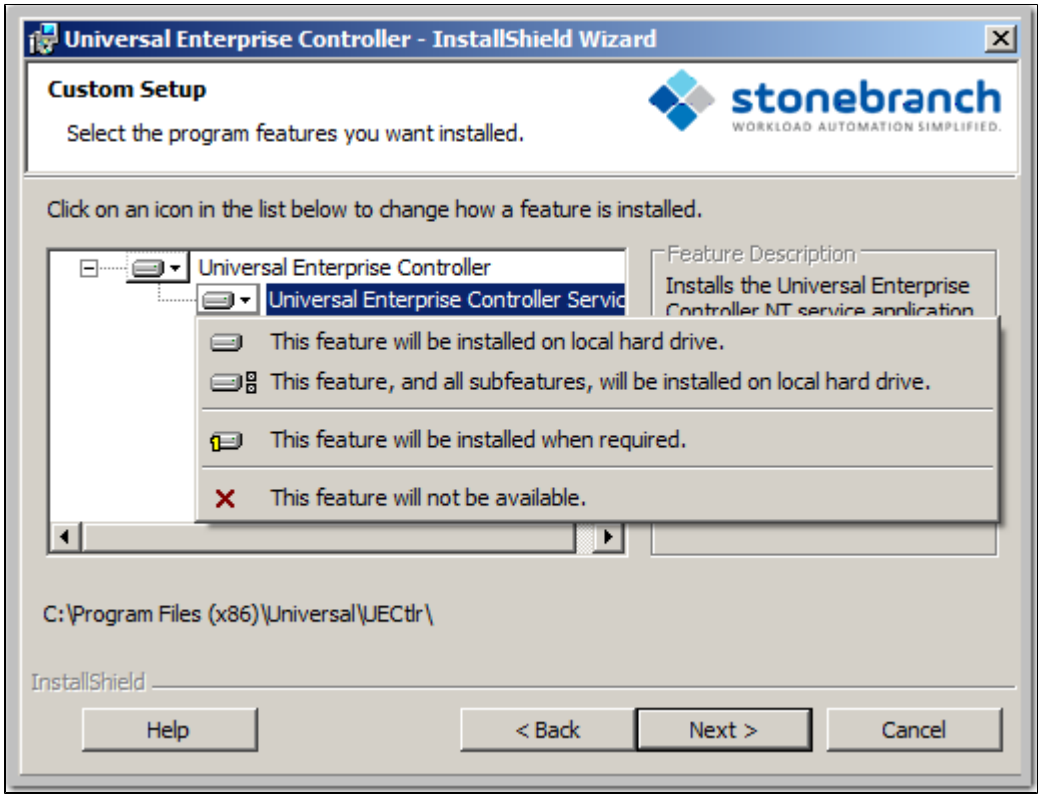
Installing over a Remote Desktop Session
If Universal Enterprise Controller was installed via a Remote Desktop connection, the environment settings used during that session may no longer be available. Starting with Windows Server 2003, Remote Desktop provides distinct session environments for each logged-in user.

The distribution file's default extraction location is based on the **TEMP** environment variable. The location referenced by this variable can change between Remote Desktop sessions, and any files extracted there may not be accessible after the session is closed. Consequently, any attempts to modify the installation may fail because the Windows Installer can't locate the installation's source files (a dialog similar to the one shown below may be displayed).



To resolve this issue, re-extract the distribution files to a location that is independent of a Remote Desktop environment and specify that location in the dialog above. Keep in mind, however, that the extracted files must come from the same distribution package used to do the initial install. If matching distribution files can't be found, Universal Enterprise Controller must be uninstalled and then reinstalled with the desired modifications.

Step 5 Click the **Modify** radio button, and then the **Next>** button, to display the Custom Setup dialog.



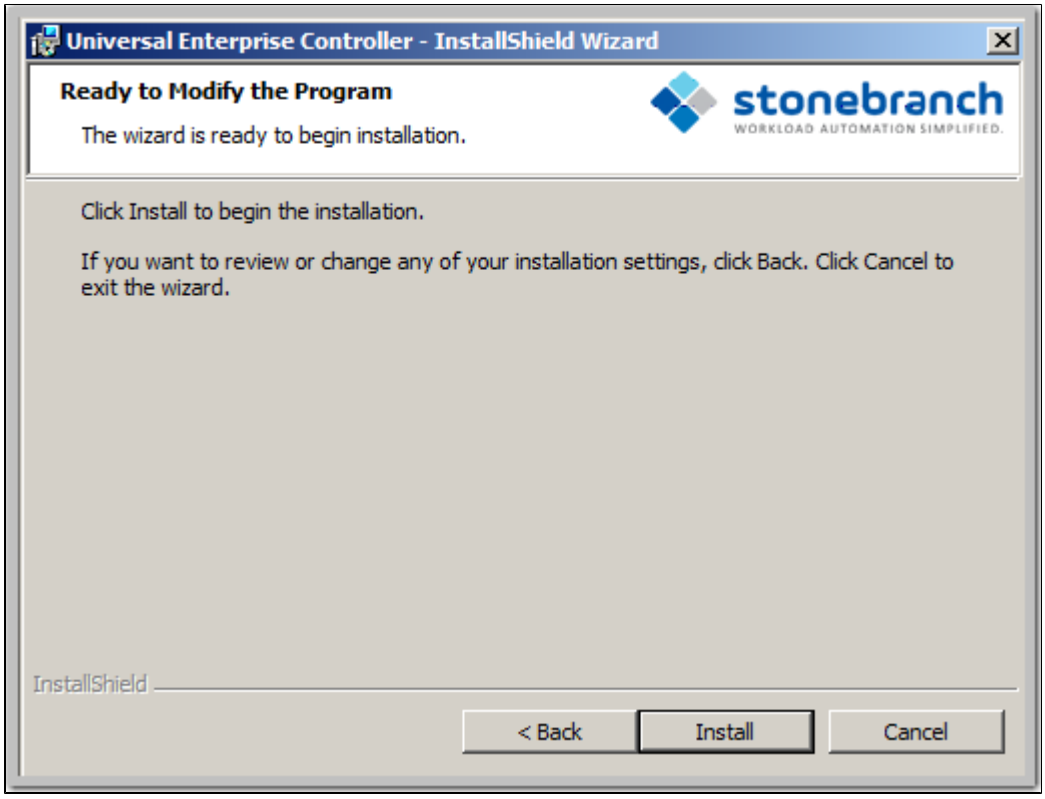
Currently installed components are identified by a drive icon.
Uninstalled components are identified by an **X** icon.

Step 6 To remove a currently installed component:

1. Click the drive icon next to that component.
2. Select the X icon from the drop-down list to mark the component for removal.

Step 7 To add an uninstalled component:

1. Click the X icon next to that component.
2. Select the drive icon from the drop-down list to mark the component for installation.

Step 8	Click the **Next>** button to display the Ready to Modify the Program dialog. 

Step 9	Click the **Install** button to modify the installation.
	
	When the modifications are complete, the following actions will be taken:
	
	• Components marked with a drive icon will:
	  • Remain installed if they already are installed.
	  • Be installed if they are not already installed.
	• Components marked with an **X** will:
	  • Remain uninstalled if they are not currently installed
	  • Be removed if they currently are installed.

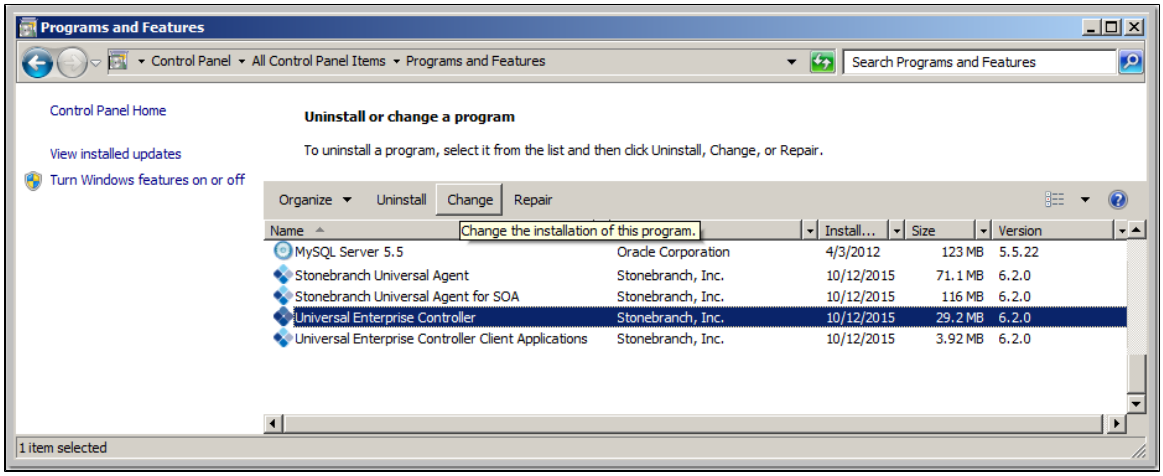
Repairing a Corrupted Universal Enterprise Controller Installation

Windows Installer has the ability to recover accidentally deleted application files, configuration files, or registry entries required by Universal Enterprise Controller. This repair feature will re-install the missing items, making a complete re-install unnecessary.

During a repair, any options stored in the Universal Enterprise Controller configuration file is preserved. If the UEC configuration file was deleted, the installation will create a new configuration file with default values.

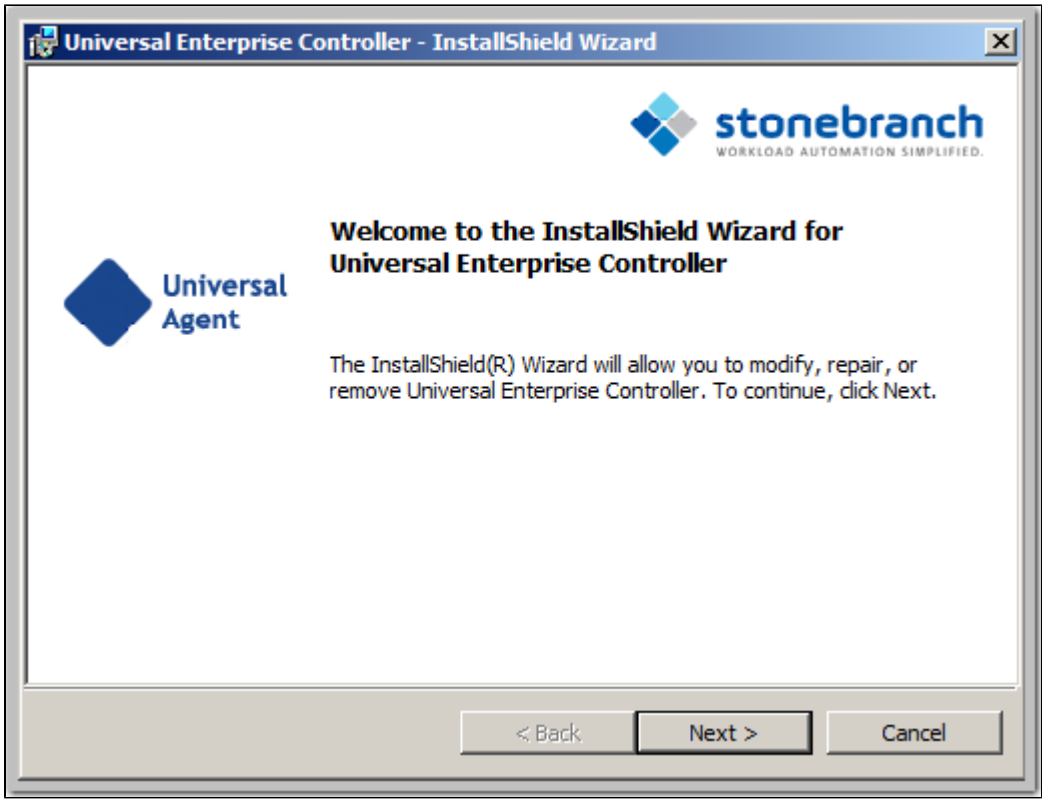
To repair an installation, perform the following steps:

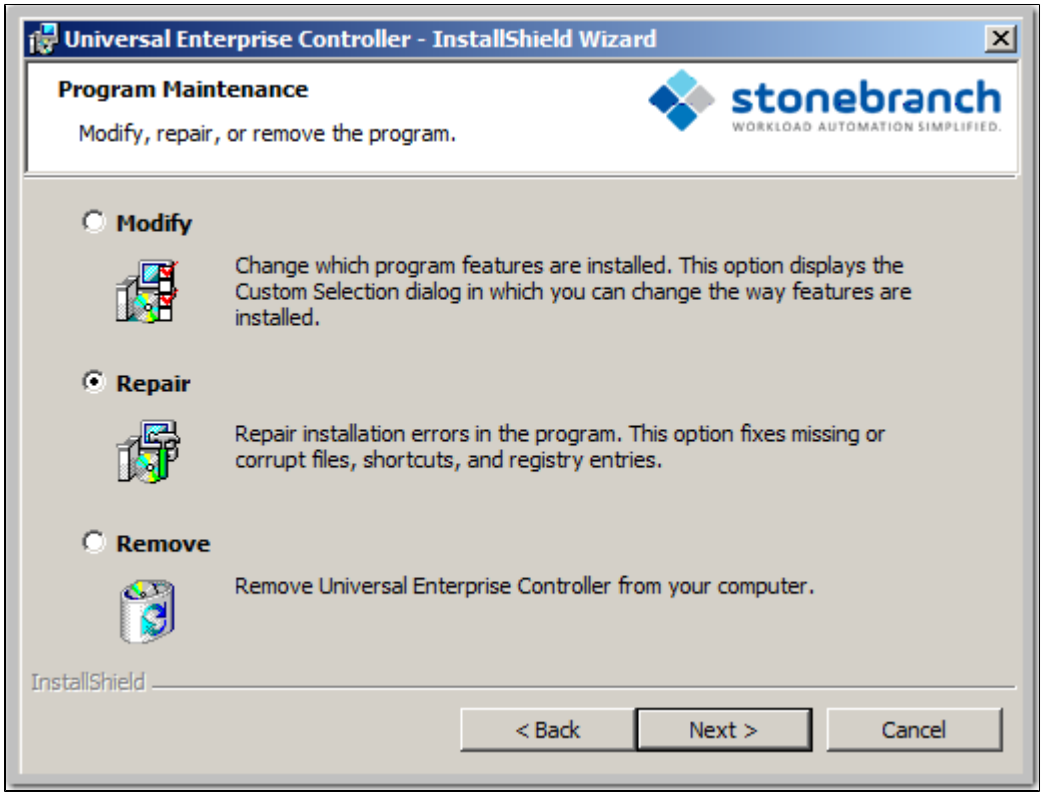
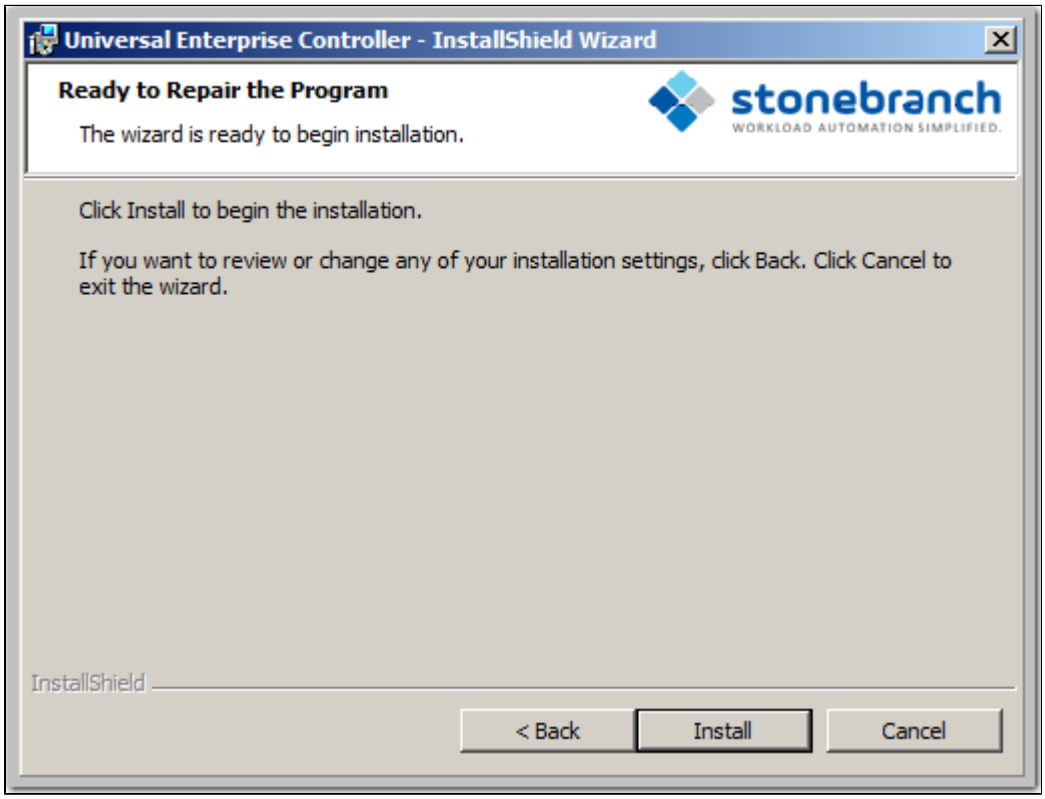
Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.



pre-Vista versions of Windows
If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller** and click **Change** to start Windows Installer. The Welcome dialog displays.

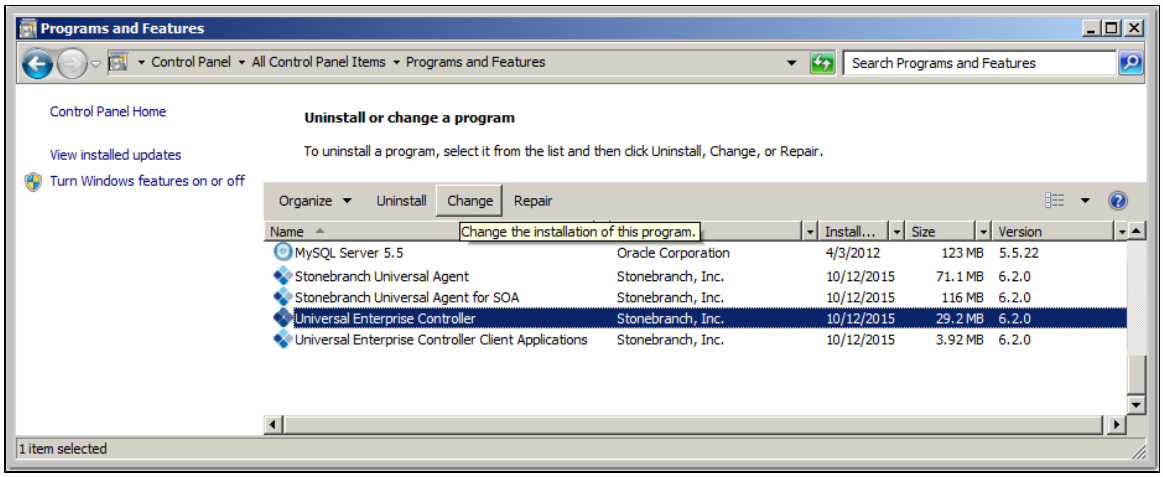


Step 3	Click the **Next>** button. The Program Maintenance dialog displays. 
Step 4	Click the **Repair** radio button, and then the **Next>** button, to display the Ready to Repair the Program dialog. 
Step 5	Click the **Install** button to repair the installation.

Removing a Universal Enterprise Controller Installation

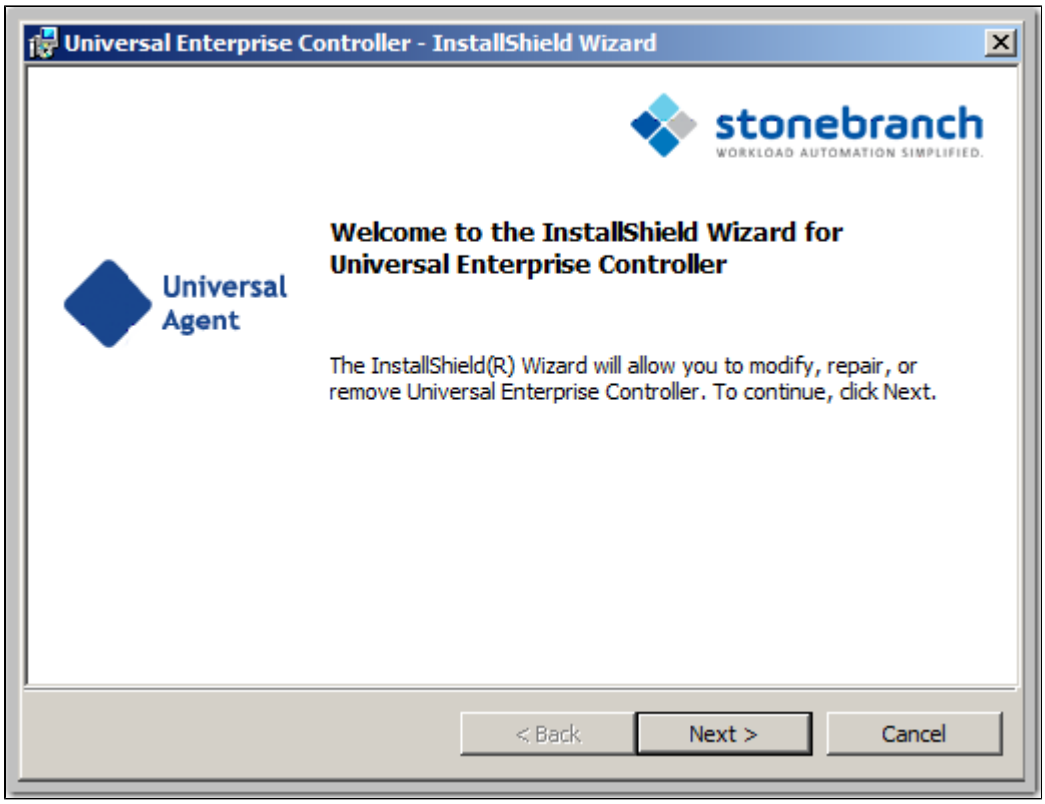
To uninstall a Universal Enterprise Controller installation, perform the following steps:

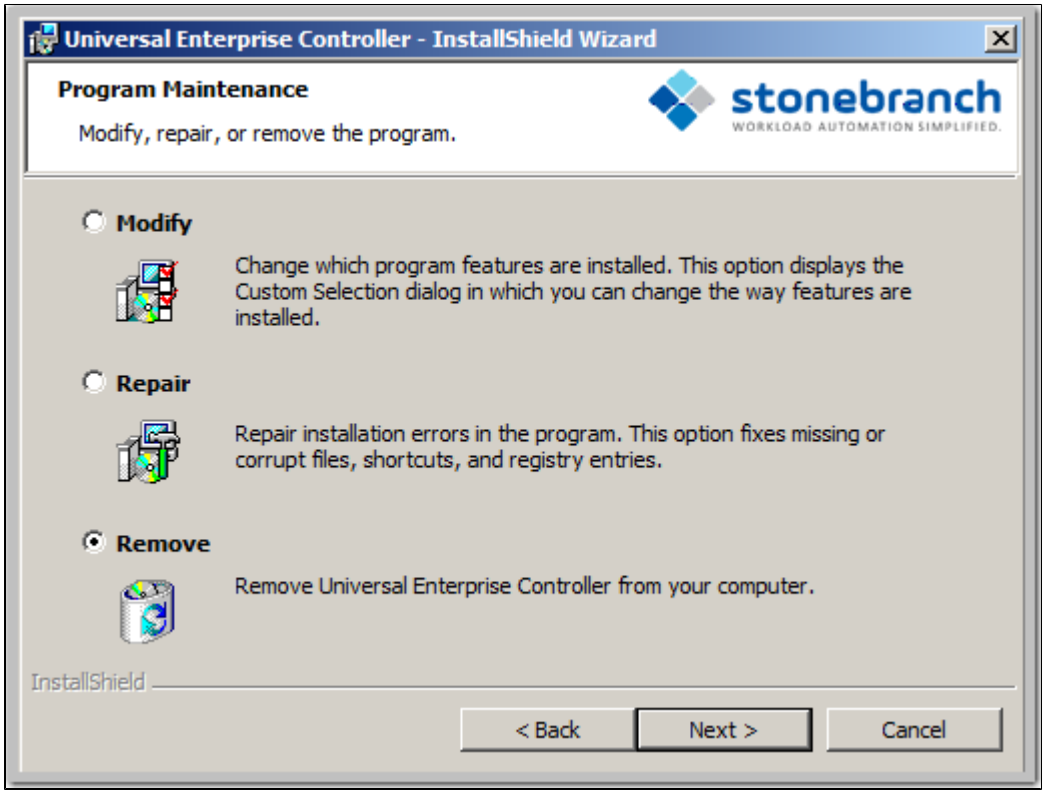
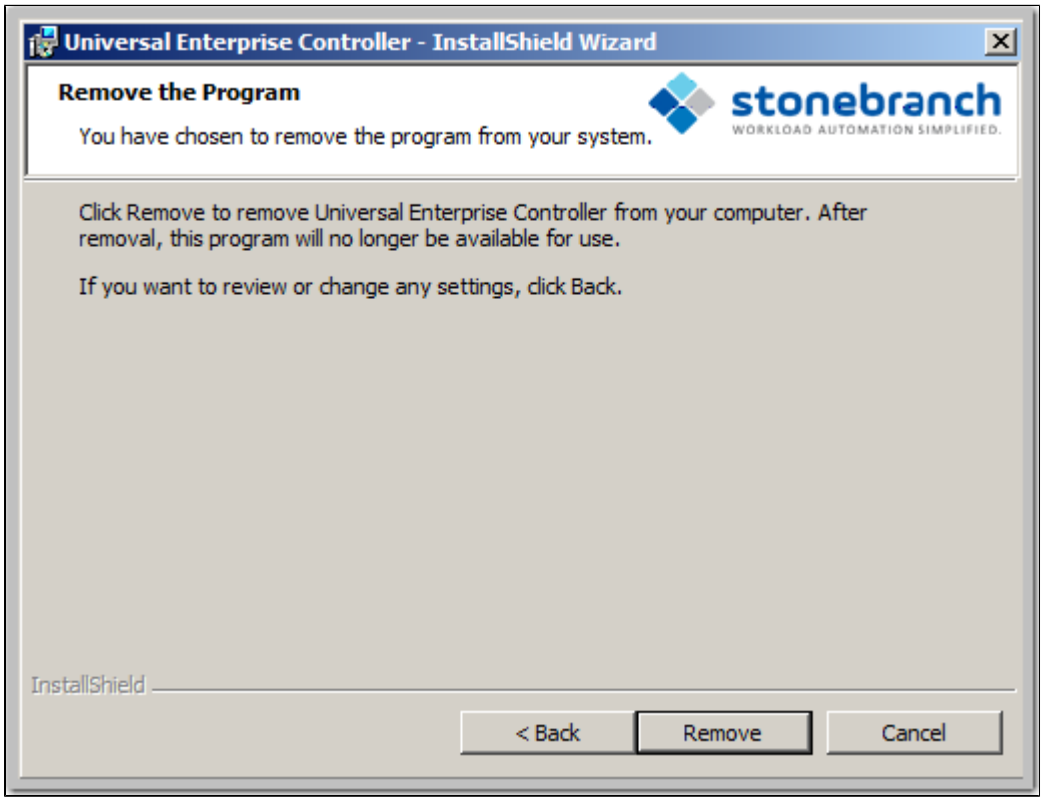
Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.



> **pre-Vista versions of Windows**
> If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller** and click the **Change** button to start Windows Installer. The Welcome dialog displays.



Step 3	Click the **Next>** button. The Program Maintenance dialog displays. 
Step 4	Click the **Remove** radio button, and then the **Next>** button, to display the Remove the Program dialog. 
Step 5	Click the **Remove** button to remove the installation.

Un-Installed Files

The un-install process will remove only those files created during the installation. Some files stored under the **.\Universal** install directory by Universal Enterprise Controller, such as trace files, may be left behind after the uninstall. In this situation, those files and/or directories may simply be deleted.

Before deleting the entire **.\Universal** directory, make sure that no other Stonebranch, Inc. products are installed there. (See Universal Enterprise Controller for Windows - File Inventory Lists for a list of files and directories installed with Universal Enterprise Controller.)

In addition to those files and directories created by the Universal Enterprise Controller installation, there may be some shared files left behind following an uninstall. These components will be removed when the last Stonebranch Inc. product that uses them is uninstalled.

Installing UEC via the Command Line

- Introduction
- Installing UEC
  - Command Line Syntax
  - Command Line Switches
  - Command Line Parameters
  - Command Line Installation Examples
- Detecting the Completion of Silent Installs

Introduction

This page describes how to install Universal Enterprise Controller (UEC) using the Windows Installer command line interface.

A command line installation is useful in situations where:

- Several Universal Enterprise Controller installations must be deployed.
- It is not practical or convenient to perform the graphical interface installation.
- It is necessary to generate an installation log file.

Installing UEC

Step 1	Download the Universal Enterprise Controller for Windows product distribution file, **sb-UECtlr-6.2.0.<level>-windows-i386.exe**, to your work station.
Step 2	Execute the distribution file from the command line, and include all appropriate command line switches and parameters. The installation process determines whether a Windows Installer update is needed. The process then extracts and saves a Windows installer package file (.msi) to this location. After all files (including the .msi) are extracted from the distribution file, the installation process verifies that your machine meets the minimum installation requirements. If the requirements are met, the installation begins.

Command Line Syntax

The following illustrates the command line syntax used to install UEC:

```
sb-UECtlr-6.2.0.<level>-windows-i386.exe [/v"command line parameters"] [/s] [/w] [/x]
```

In this syntax:

- **<level>** is the numeric package level.

The command line switches (/v, /s, /w, and /x) are processed directly by the distribution file to control behavior of the Windows Setup application.

The command line parameters are passed to the Windows Installer (**msiexec**) to control the extracted Windows installer package file (.msi) behavior during the install process.

Command Line Switches

The following table describes the command line switches available for a command line installation:

/v	Passes parameters to the Windows Installer (**msiexec**). The list of parameters must be enclosed in double (") quotation marks. See Command Line Parameters for available parameters.
/s	Suppresses the initialization and extraction dialogs displayed before the product install Welcome dialog." If you are using the /q command line parameter, use this switch additionally for a completely silent install.
/w	Instructs the Windows Setup application to wait until the installation completes. Use this switch when launching the installation from a script file. Without it, the Setup application may return immediately after launching Windows Installer.

/x	Uninstalls UEC.

Command Line Parameters

The following table describes the parameters that are available for a command line installation.

The parameters can be specified in any order, with the following exceptions:

- If the Repair (/fom) or Remove (/x) parameter is used, it must be specified ***before*** all other parameters.
- If the Silent install (/q) and/or Log file (/L) parameters are used, they can be specified in any order, but they must be specified ***after*** all other parameters.

These parameters are preceded by the /v command line switch and must be enclosed in double (") quotation marks.

Parameter	Description	Default
/fom	Repairs a Universal Enterprise Controller installation. **om** (after the **/f**) are options used by the repair. There are other options available, but for behavior that matches the repair done from the graphical install, the **om** options must be used. **/fom** cannot be used with the **/x** (remove) parameter.	n/a
/x	Removes Universal Enterprise Controller. **/x** cannot be used with the **/fom** (repair) parameter.	n/a
INSTALLDIR *=installdir*	Sets the root installation directory to <**`installdir`**>. Each component will be installed under this directory. **INSTALLDIR** is required if you want to install UEC under a directory different from the one specified by the **PROGRAMFILES** environment variable (typically **`C:\Program Files\Universal`**). If the directory contains spaces, you must use double (**"**) quotation marks around the path name.	(none)
UECTLR={ **yes**|**no**}	Specification for whether or not to install the UEC Service component during new installs, upgrades, or maintenance. • If **yes** is specified, UEC Service will be installed. • If **no** is specified, UEC Service will not be installed. If UEC Service already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UEC Service** is not required unless you want to change the current install state. **UEC Service** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **UEC Service** on the Custom Setup dialog when installing UEC via the graphical interface.	**yes**
/q	Suppresses the product installation dialogs. Use this parameter in addition to the /s command line switch for a completely silent install. See Command Line Switches, Command Line Installation Examples, and Detecting the Completion of Silent Installs for additional information regarding silent installs.	n/a
/L*v	Instructs the installation process to create an installation log file named <**`logfilepath`**> (full path name). If <**`logfilepath`**> contains spaces, you must enclose it with double (**"**) quotation marks. ***v** are flags used to specify the level of detail (**verbose**) contained in the log file. To reduce the amount of output generated, ***v** can be omitted. However, using these options is good practice; they can assist Stonebranch Customer Support with problem determination should any errors occur during installation.	n/a

Command Line Installation Examples

The following examples illustrate different ways that UEC can be installed from the command line.

Graphical User Interface Install, All Components

To install all UEC components via the graphical user interface, issue the following command:

```
sb-UECtlr-5.2.0.0-windows-i386.exe
```

Graphical User Interface Install, All Components, with Log File

To install all UEC components using the Windows Installer graphical user interface and write a log file to **`C:\temp\install.log`** during the installation, issue the following command:

```
sb-UECtlr-5.2.0.0-windows-i386.exe /v"/l*v c:\temp\install.log"
```

Silent Install, Different Installation Directory

To install UEC under **D:\Universal**, which is a directory other than the one specified by the environment variable **PROGRAMFILES**, issue the following command:

```
sb-UECtlr-5.2.0.0-windows-i386.exe /s /v"/qn INSTALLDIR=D:\Universal"
```

Silent Install, All Components

To install all UEC components without using a graphical interface — that is, a silent install — issue the following command:

```
sb-UECtlr-5.2.0.0-windows-i386.exe /s /v"/qn"
```

Detecting the Completion of Silent Installs

If the /q switch is used to perform a silent install, no graphical interface or user interaction is required. One drawback to this is that no feedback is provided indicating when the Windows Installer process (install, uninstall, or repair) finishes.

One method that may be used to detect when the Windows Installer process ends is to execute it using the system's **start** command. Using available command line switches, the **start** command can be used to initiate the Windows Installer process and then wait for it to finish. When the **start** command returns control to its calling process (for example, the command prompt), the process will have ended.

For example, from the command prompt, issue the following command to start the Universal Enterprise Controller installation and wait for it to finish.

```
start /b /wait sb-UECtlr-5.2.0.0-windows-i386.exe /w /s /v"/qn"
```

- The **/b** switch prevents the **start** command from opening a new window.
- The **/wait** parameter causes the **start** command to start the application, **sb-UECtlr-5.2.0.0-windows-i386.exe**, and then wait for it to finish.

The syntax above can also be used to execute the **start** command from within a script, such as a .**bat** file.

For more information on the **start** command, go to the Windows command prompt and enter: **start /?**.

Modifying a UEC Installation via the Command Line

- Modifying a Universal Enterprise Controller Installation via the Windows Installer Command Line Interface
- Adding or Removing Universal Enterprise Controller Components
- Repairing a Corrupted Universal Enterprise Controller Installation
  - Silent Repair
  - Interactive Repair, with Log File
- Removing a Universal Enterprise Controller Installation
  - Silent Removal

Modifying a Universal Enterprise Controller Installation via the Windows Installer Command Line Interface

This page describes how to modify a Universal Enterprise Controller installation via the Windows Installer command line.

After Universal Enterprise Controller is installed, Windows Installer can be run as many times as needed to modify the installation by:

- Adding or Removing Universal Enterprise Controller Components
- Repairing a Corrupted Universal Enterprise Controller Installation
- Removing a Universal Enterprise Controller Installation

(For a description of the parameters used in these procedures, see Windows Installer Command Line Parameters.)

Adding or Removing Universal Enterprise Controller Components

Currently, it only is possible to add or remove Universal Enterprise Controller components using the Windows Installer graphical interface (see Modifying a UEC Installation via the Graphical Interface).

Repairing a Corrupted Universal Enterprise Controller Installation

To recover accidentally deleted files or registry entries required by Universal Enterprise Controller using the Windows Installer command line interface, use the **/f** switch together with the **om** parameters.

These are the same repair options set internally by the graphical interface installation. They cause Windows Installer to reinstall files that are missing or older than the version contained in the Universal Enterprise Controller distribution file.

Silent Repair

To repair a Universal Enterprise Controller installation from the command line, issue the following command:

```
msiexec.exe /fom SetupPath\UECtlr.msi /q
```

Interactive Repair, with Log File

To repair a Universal Enterprise Controller installation using the Windows Installer graphical interface, and instruct Windows Installer to write a log file to **`C:\Temp\repair.log`** while running the repair, issue the following command:

```
msiexec.exe /fom SetupPath\UECtlr.msi /L C:\Temp\repair.log
```

Removing a Universal Enterprise Controller Installation

To uninstall Universal Enterprise Controller using the Windows Installer command line interface, use the **/x** switch.

Silent Removal

To uninstall Universal Enterprise Controller without using the Windows Installer graphical interface, issue the following command:

```
msiexec.exe /x SetupPath\UECtlr.msi /q
```

Universal Enterprise Controller for Windows - 64-Bit Windows Editions

- Universal Enterprise Controller - Installing on 64-bit Windows Editions
- Applications Installed in the Windows System Folder
  - Example 1
  - Example 2

Universal Enterprise Controller - Installing on 64-bit Windows Editions

All Universal Agent components have been tested and verified on the 64-bit editions of the following Windows systems:

- Windows XP
- Windows Server 2003
- Windows Vista
- Windows Server 2008
- Windows 7
- Windows Server 2008 R2

This page describes some modifications that may need to be made to the default installation options to ensure that the installed Universal Enterprise Controller components function correctly.

Applications Installed in the Windows System Folder

The Universal Enterprise Controller package installs several command-line applications in the Windows system folder. The default system folder for 32-bit applications installed on 64-bit Windows editions is the **`%SystemRoot%\SysWOW64`** directory (for example, **`C:\Windows\SysWOW64`**).

The following table identifies the affected Universal Agent applications.

File Name	Description
ucert.exe	Universal Certificate
uecload.exe	UECLoad
uencrypt.exe	Universal Encrypt

These applications can be executed using either the:

- 32-bit command shell (**%SystemRoot%\SysWOW64\cmd.exe**)
- Default 64-bit command shell (**%SystemRoot%\System32\cmd.exe**).

By default, the **`%SystemRoot%\SysWOW64`** directory is not part of the system path. Therefore, to execute **uecload.exe** using the 32-bit command shell, either:

- Directory must be added to the **PATH** environment variable.
- Complete path to the application and/or the 32-bit command shell must be specified.

Example 1

To execute UECLoad in the default 64-bit command shell, issue the following command:

```
%SystemRoot%\SysWOW64\uecload
```

Example 2

To execute UECLoad within the 32-bit command shell, issue the following command:

```
%SystemRoot%\SysWOW64\cmd.exe /C %SystemRoot%\SysWOW64\uecload
```

Universal Enterprise Controller for Windows - Database Configuration

Universal Enterprise Controller - Database Configuration

Berkeley DB uses a temporary cache in memory to manage its databases. If this cache becomes sufficiently large, it must be written to disk.

Berkeley DB has a default location for storing temporary cache files, but if UEC cannot access that location, or there is no space to write these files in the default location, the following error can occur in UEC, and UEC shuts down:

UNV4301D Database error: 'temporary: write failed for page XXXXX'

To work around this issue, perform the following steps to write the temporary cache files to the UEC database directory:

Step 1	Inside the UEC database directory, create a text file named **DB_CONFIG**.
Step 2	Inside the **DB_CONFIG** file, add the following string: **`set_tmp_dir *dbpath*`** (**`dbpath`** is the path to the location in which the database files reside.)
Step 3	Start / restart UEC.

Universal Enterprise Controller for Windows - File Inventory Lists

- Universal Enterprise Controller - File Inventory Lists
- Universal Enterprise Controller
- Universal Products Install Merge Utility
- System Files

Universal Enterprise Controller - File Inventory Lists

The Universal Enterprise Controller installation includes the files required for the following components / utilities:

- Universal Enterprise Controller
- Universal Configuration Manager
- Universal Products Install Merge

The Universal Configuration Manager is installed whenever Universal Enterprise Controller is installed. It is used to maintain the configuration options for the Universal Enterprise Controller service. If any of the components already are installed, Windows Installer will upgrade them to the latest version.

The files installed with each Universal Enterprise Controller component / utility are listed in the following tables. The file paths specified are relative to the root installation directory (for example, **`C:\Program Files\Universal`**) that was specified during the installation.

Items shown with a path of System32 are installed in the 32-bit system folder. The actual name of this directory depends on the Windows version:

- For all supported 32-bit Windows editions, the path is **`\Windows\System32`**.
- For all supported 64-bit Windows editions, the path is **`\Windows\SysWow64`**.

Universal Enterprise Controller

File	Description
nls*.utt	Code page files used for text translation between different operating systems and platforms.
nls\README.TXT	Information about the contents of the **.\nls** directory.
nls\uecmceng.umc	English message catalog.
UCfgMgr\bin\ucfgmgr.cpl	Universal Configuration Manager control panel application.
UCfgMgr\bin\ucfgmgr.hlp	Universal Configuration Manager help file.
UECtlr\bin\acl.xml	Used to store Access Control List entries for Universal Enterprise Controller.
UECtlr\bin\brokers.xml	Used to store a list of Universal Brokers that will be monitored by the Universal Enterprise Controller.
UECtlr\bin\groups.xml	Used to store defined groups of Universal Brokers.
UECtlr\bin\librfc32.dll	Support file for Universal Enterprise Controller.
UECtlr\bin\ueccfg.dll	Used by Universal Configuration Manager to manage Universal Enterprise Controller configuration options.
UECtlr\bin\ueccfg.hlp	Universal Enterprise Controller configuration help file.
UECtlr\bin\uecdbrec.bat	Recovers a Universal Enterprise Controller database, specified from the command line. The default database is **uec_evm.db**.
UECtlr\bin\uecmsgnt.dll	Used to write error messages to the Windows Application event log.
UECtlr\bin\uectlrs.exe	Universal Enterprise Controller installation file.
UECtlr\bin\users.xml	Used to store a list of user accounts authorized to use Universal Enterprise Controller and its associated Client Applications: I-Administrator, I-Activity Monitor, and I-Management Console.
UPIMerge\bin\upimerge.exe	Command line interface to configuration file merge routines).
USpool\bin\udb_dump.exe	UEC database utility; to be used only upon request of Stonebranch, Inc. Customer Support.
USpool\bin\udb_load.exe	UEC database utility; to be used only upon request of Stonebranch, Inc. Customer Support.
Universal\conf\ uec.conf [1]	Universal Enterprise Controller configuration file.

ucert.exe [2]	Universal Certificate utility, used to generate X.509 Certificates.
uecload.exe [2]	Universal Enterprise Controller load utility.
uencrypt.exe [2]	Universal Encrypt utility, used to encrypt sensitive Universal application command line options.

[1] This file is installed under **%ALLUSERSPROFILE%\ Application Data**, which, by default, resolves to:

- **C:\Documents and Settings\All Users\Application Data** on 2003 and XP.
- **C:\ProgramData** on Vista and Windows 2008 Server.

[2] This file is installed under environment variable **%SystemRoot%**, which, by default, resolves to:

- **C:\Windows** on all platforms.
  - On x86 platforms, these files are installed under **%SystemRoot%\system32**.
  - On x64 platforms, these files are installed under **%SystemRoot%\SysWow64**.

Universal Products Install Merge Utility

File	Description
UPIMerge\bin\upimerge.exe	Application program, always installed. Provides command line access to the same functionality used by the Universal Agent installation to merge options from a new configuration file into an existing file.

System Files

The following files will be installed only if they are newer than the existing files.

The directories shown in this table are relative to the **%SYSTEMROOT%** directory, where **%SYSTEMROOT%** is an environment variable that resolves to **C:\Windows** on all Windows platforms.

File	Description
System32\asycfilt.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\comcat.dll	Version 4.71.1460.1 of the Microsoft Component Category Manager library.
Microsoft C-Runtime v8.0.50727.762 [1]	Version 8.0.50727.762 of the Microsoft C runtime side-by-side assembly.
System32\msiexec.exe	Version 3.1.4000.1823 of the Microsoft Windows Installer (see Windows Installer for more information).
System32\oleaut32.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\olepro32.dll	Version 5.0.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\psapi.dll	Version 4.0.1371.1 of the Microsoft process status library.
System32\stdole2.tlb	Version 2.40.4275.1. This file is one of the components of the Microsoft OLE library.

[1] The Microsoft C-Runtime distribution consists of several files, which are subject to change. Refer to Microsoft documentation for a complete list of files delivered with the specified runtime version.

UEC Client Applications Installation

UEC Client Applications

The following information is provided for the installation of UEC Client Applications:

- Installation Package
- Installation Requirements
- Installation Procedures
- 64-Bit Windows Editions
- File Inventory Lists

UEC Client Applications - Installation Package

Components

The Universal Enterprise Controller (UEC) 6.2.x Client Applications for Windows package includes the desktop application versions of the following components:

- I-Administrator 6.2.x
- I-Activity Monitor 6.2.x
- I-Management Console 6.2.x

Component Compatibility

The following table identifies the compatibility of Universal Enterprise Controller Client Applications 6.2.x with previous component / product versions.

Component	Compatibility
Universal Enterprise Controller Client Applications 6.2.x	Not compatible with previous versions of Universal Enterprise Controller for Windows.

The component references pertain to all supported platforms for that version.

UEC Client Applications - Installation Requirements

- Windows Versions
- Additional Requirements
- Platform Requirements
- Java Runtime Environment

Windows Versions

To install the UEC Client Applications, you must have one of the following versions of Windows:

- Windows Server 2003 SP1 and higher
- Windows Server 2003 R2
- Windows XP SP3
- Windows Vista
- Windows 7
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2

Note
Itanium not supported for Windows Server 2003/2003 R2 and Windows Server 2008/2008 R2.

Additional Requirements

- For a per-machine install, Administrator access is required. For a per-user install, any account with the authority to install software can be used.
- The following conditions must be met before the UEC Client Applications can be successfully installed:
  - Account used for the installation must have write access to the desired destination folder.
  - No system policies (defined in Windows by your system administrator) may be in place that prohibit non-Administrative accounts from installing software.
- Possible reboot: a reboot is required if the Windows Installer service is not installed, a version of the Windows Installer prior to 3.1.4000.1823 is installed, or if required files are in use at the time of the installation.
- TCP/IP.
- About 5 megabytes of disk space. This value does not include space required for the Java Runtime Environment (JRE). See the JRE installation requirements, available from Sun, for more information.
- Sun Java Runtime Environment (JRE 1.5 or above).

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

Java Runtime Environment

To run the UEC Client Applications, you also must have the Java Runtime Environment (JRE) version 1.5 or above. You can download the latest JRE installation package directly from Sun's website, www.java.com.

A Stonebranch Tip
Changes in Sun's versioning scheme for Java has created some confusion.

With version 1.2 of the Java environment, Sun began referring to Java as Java 2. The formal name was actually Java 2 with SDK 1.2.

As of January 2008, version 1.6 is latest version of the Java environment - a newer version than the original Java 2.

UEC Client Applications - Installation Procedures

UEC Client Applications - Installation Procedures

The following procedures are provided for the installation and modification of UEC Client Applications:

- Installing UEC Client Applications via the Graphical Interface
- Modifying a UEC Client Applications Installation via the Graphical Interface
- Installing UEC Client Applications via the Command Line
- Modifying a UEC Client Applications Installation via the Command Line

Note
Modifying a UEC Client Applications installation refers to the adding / removing of UEC Client Applications components, repairing a corrupted installation, or removing an installation. To change the installed version of UEC Client Applications, see Upgrading Universal Agent and Applying Maintenance to Universal Agent.

Installing UEC Client Applications via the Graphical Interface

Installing UEC Client Applications via the Windows Installer Graphical Interface

To install UEC Client Applications for Windows using the Windows Installer graphical interface, perform the following steps:

Step 1	Download the UEC Client Applications for Windows product distribution file, **`sb-UEClient-6.2.0.<level>-windows-i386.exe`**, to your work station.

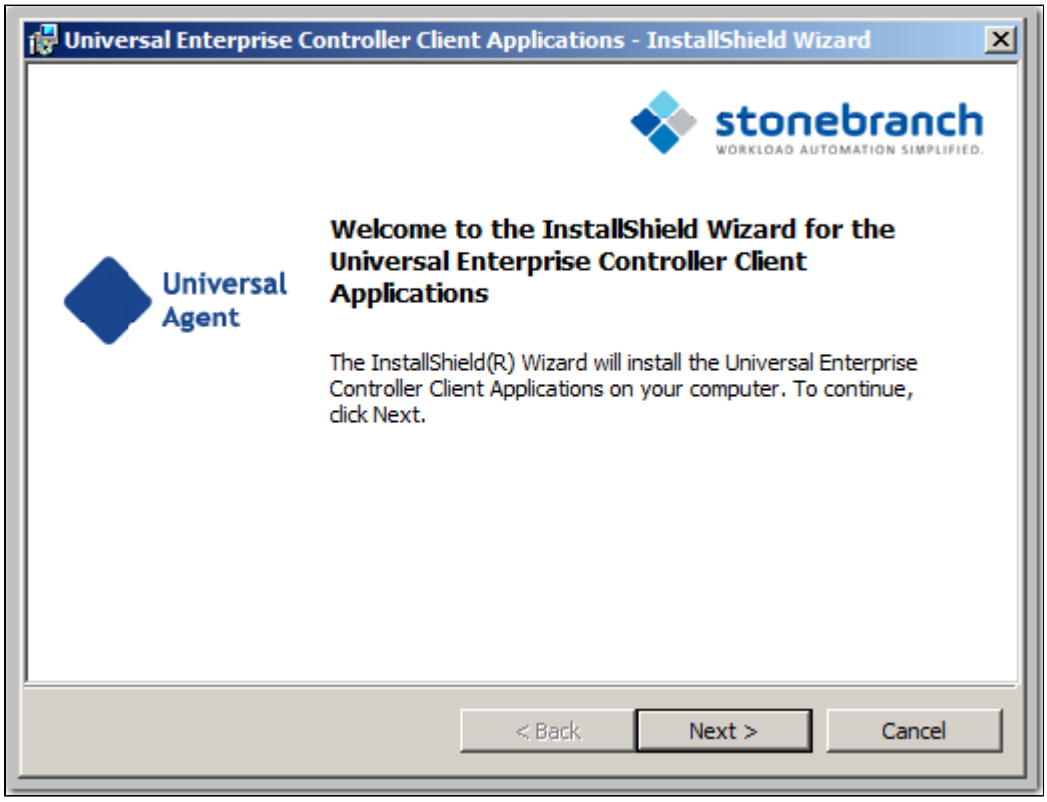
Step 2 Execute the distribution file to extract the files.

> **Note**
> If you already have extracted the files from the distribution file, but cancelled installation in order to separately install Windows Installer (see Windows Installer), you can simply double-click the extracted Client Applications installation file, **`UEClient.msi`**, to begin the installation.

> **Installing over a Remote Desktop Session**
> Starting with Windows Server 2003, Remote Desktop provides distinct session environments for each logged-in user. This means extraction may use an environment setting that is not available once the Remote Desktop session ends.
>
> For example, the distribution file's default extraction location is based on the TEMP environment variable. The location referenced by this variable can change between Remote Desktop sessions, and any files extracted there may not be accessible after the session is closed.
>
> To ensure that extracted files and other required resources are accessible after the initial install, extract the files to a well-known location that is not likely to change between Remote Desktop sessions.
>
> Refer to the Microsoft documentation on the Remote Desktop feature for additional information.

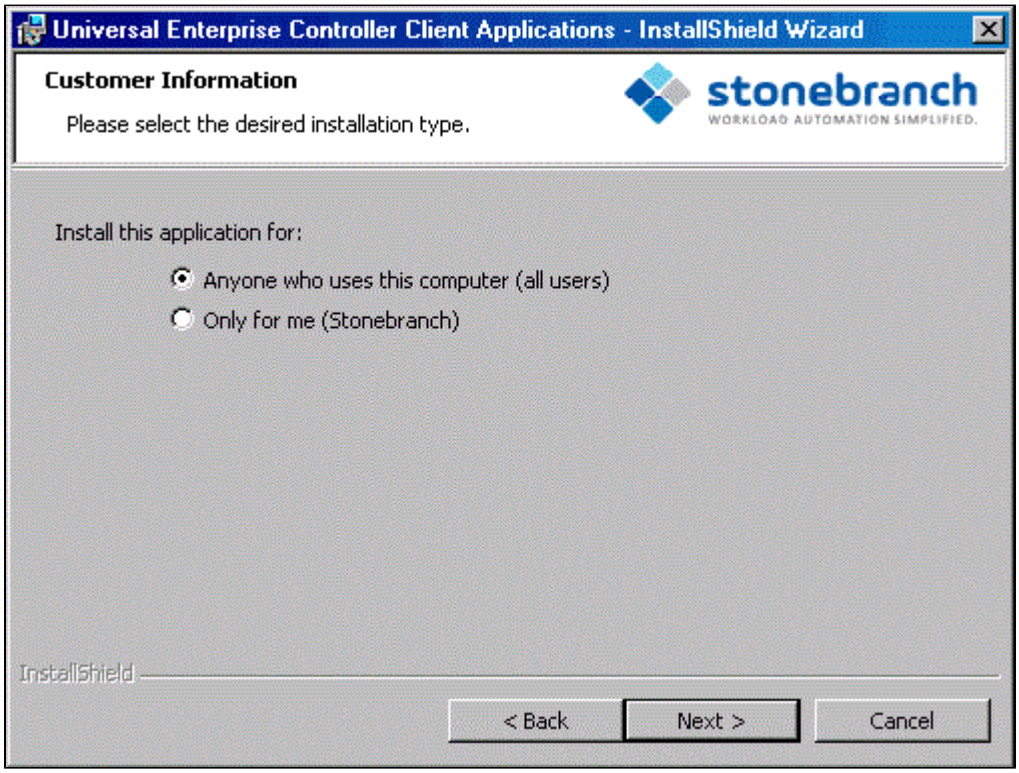
The installation automatically will begin after the files are extracted.

It first will verify that your machine meets the minimum system requirements (see UEC Client Applications - Installation Requirements). If the requirements are met, a Welcome dialog displays.



Step 3 Click the **Next** button.

- If the installation is being performed by a user account that is a member of the Administrators group, the Customer Information dialog, below, displays.
- If the installation is being performed by a user account that is NOT a member of the Administrators group, the Custom Setup dialog displays (see Step 5).



Step 4 Click the appropriate radio button on this dialog:

- If you want to perform an installation for all users with access to a given workstation, click **Anyone who uses this computer**. This is referred to as a *per-machine* installation. When this type of installation is performed, shortcuts added to the **Start** menu, and any configuration settings stored in the Windows registry, are placed in a location accessible to anyone who logs on to that particular machine. This type of installation also creates an entry for UEC Client Applications in the **Add or Remove Programs** list, accessible via the Windows Control Panel. However, only accounts with administrative privileges will be able to:
  - Modify the installation (see Modifying a UEC Client Applications Installation via the Graphical Interface)
  - Uninstall the product (see Removing a UEC Client Applications Installation).

- If you want to perform an installation for the user identified on this dialog, click **Only for me**. (This is the account performing the installation.) This is referred to as a *per-user* installation. When this type of installation is performed, the Start menu shortcuts, the configuration options stored in the Windows registry, and the application itself (depending on where it is installed) will be accessible only by this user. This type of installation also creates an entry for UEC Client Applications in the **Add or Remove Programs** list, accessible via the Windows Control Panel. However, it will be visible only by this user.

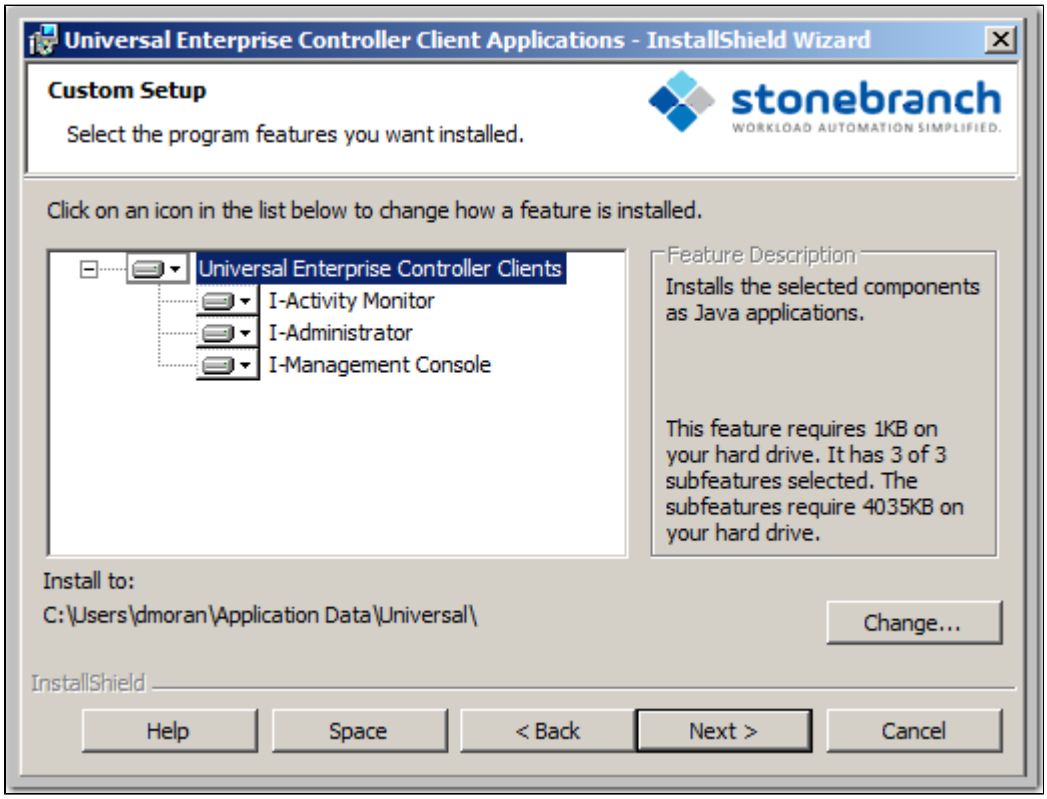
  In this case, the user also will be able to:
  - Modify the installation (see Modifying a UEC Client Applications Installation via the Graphical Interface).
  - Uninstall the product (see Removing a UEC Client Applications Installation). For the UEC Client Applications to be available to another user on this machine, that user also must perform a _per\-user_ installation.

Note
It is possible for a *per-machine* and a *per-user* installations to be performed on the same machine, although there is no benefit in doing this. In that situation, when any user that has done a *per-user* installation is logged on, that installation takes precedence. If the user removes the Client Applications, the *per-machine* installation will remain in effect. Conversely, if the *per-machine* installation is removed, users that have performed a *per-user* installation will still have access to Client Applications.

A Stonebranch Tip
For Non-Administrative Users:
Because *per-machine* installations require access to certain system resources to which most non-Administrative accounts do not have access, all installations performed by non-Administrative users are *per-user* installations.

Step 5

Click the **Next>** button. A list of UEC Client Applications components included in the installation package then displays. It is from this list that you can select which components to install.



For a new installation, a drive icon displays next to each item in the list, indicating that the component will be installed.
For an upgrade installation, either of the following icons displays next to an item:

- A drive icon indicates that the component is either:
  - New to the installation and will be installed.
  - Currently is installed and will be upgraded.
- An *X* icon indicates that the component is either:
  - Currently not installed (but previously was available).
  - Previously installed but removed.

A Stonebranch Tip
The directory identified in the figure above is typical for a new, *per-machine* installation. Depending on the type of installation being performed, the directory may be different.

For a new, *per-user* installation, the dialog will identify a directory located within the user's profile directory (for example, **`C:\Documents and Settings\username`**).

For installation upgrades, whether *per-machine* or *per-user*, the UEC Client Applications' current location is displayed.

Step 6

The figure above shows that all UEC Client Applications will be installed in their respective directories under the **`C:\Program Files\Universal`** directory.

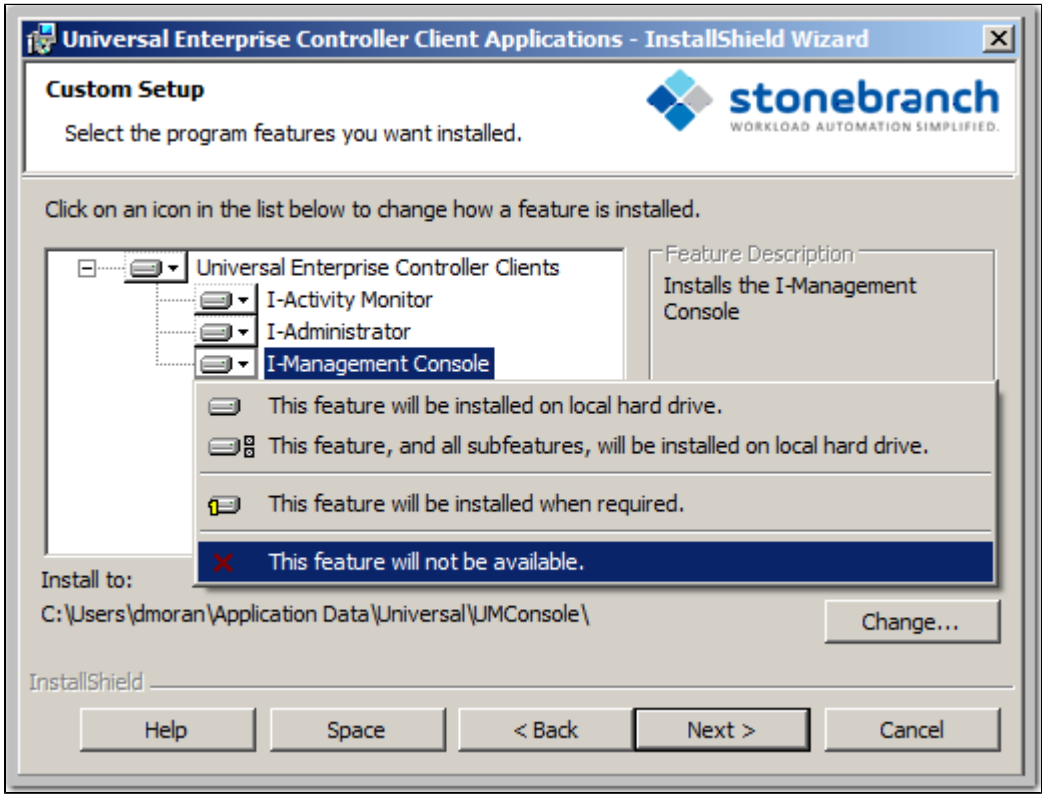
1. If you want to select a different location, click the **Change...** button.
2. If you want to check the amount of disk space required for the installation, and the amount of available disk space on the selected directory, click the **Space** button.

Step 7

If you do not want to install a component:

1. Click the drive icon next to that component name.
2. From the drop\-down list that displays, select the X icon to mark the component as one not to be installed.

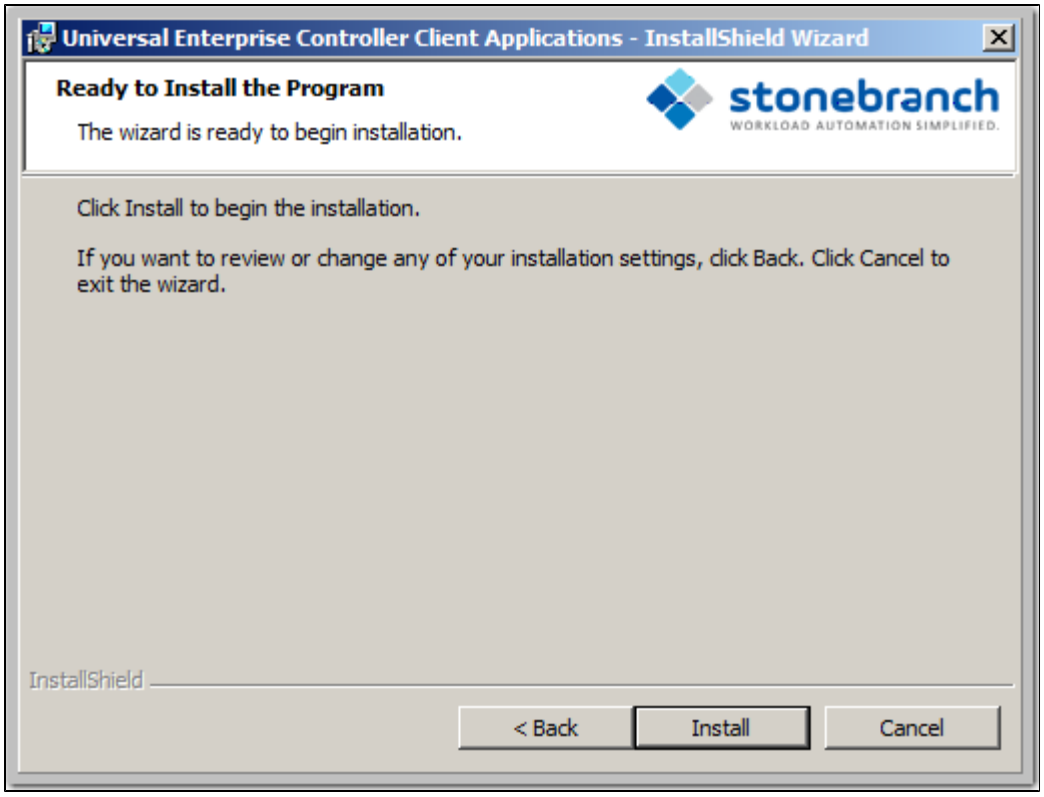
For example, the figure below indicates that I-Management Console has been selected to not be installed.



Step 8

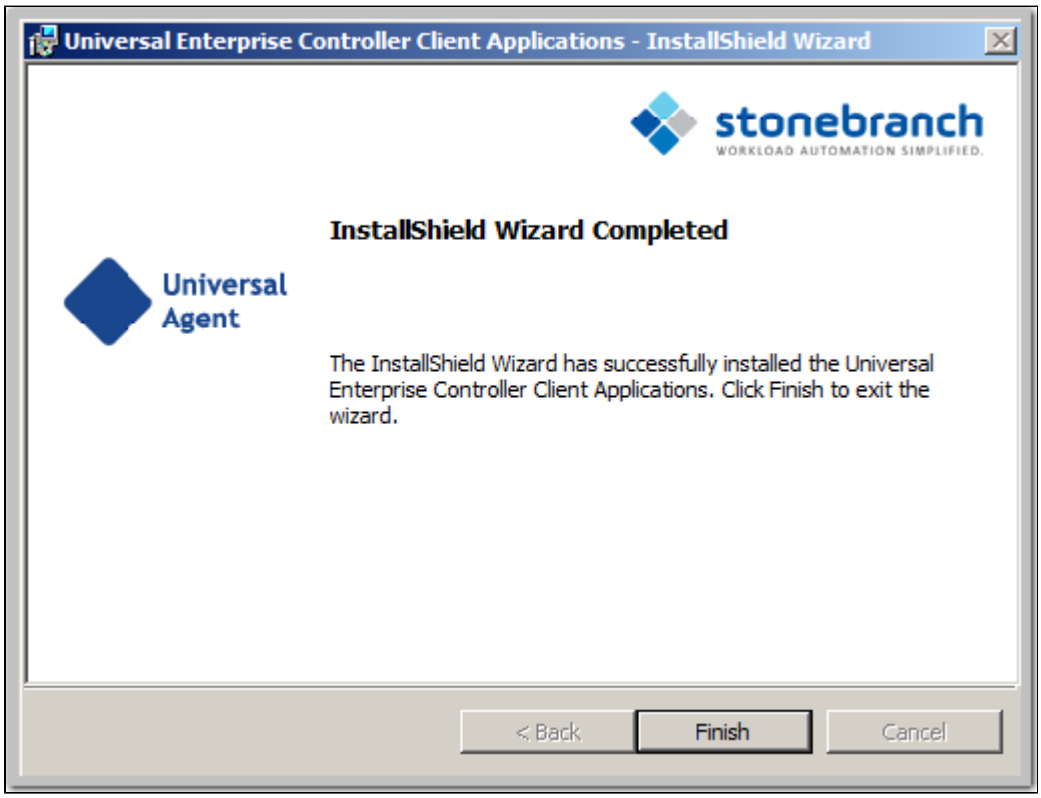
When you have selected the components (and their installation destinations) that you want to install, click the **Next** button to continue the installation process.

When the installation is ready to begin, the Ready to Install dialog displays.



Click the **Install** button to begin the installation or click the **<Back** button to return to change information on any of the previous dialogs.

When the installation completes successfully, the Installation Complete dialog displays.



Note
During the install, you may have been notified that no Java Runtime Environment (JRE) was detected on your system. The UEC Client Applications require Java Runtime Environment (JRE) Version 1.5 or greater (see UEC Client Applications - Installation Requirements).

Step 9	Click the **Finish** button to exit Windows Installation.

Modifying a UEC Client Applications Installation via the Graphical Interface

- Modifying a UEC Client Applications Installation via the Windows Installer Graphical Interface
- Adding or Removing UEC Client Applications Components
- Repairing a Corrupted UEC Client Applications Installation
- Removing a UEC Client Applications Installation
  - Un-Installed Files

Modifying a UEC Client Applications Installation via the Windows Installer Graphical Interface

After the UEC Client Applications are installed, Windows Installer can be run as many times as needed to modify the installation.

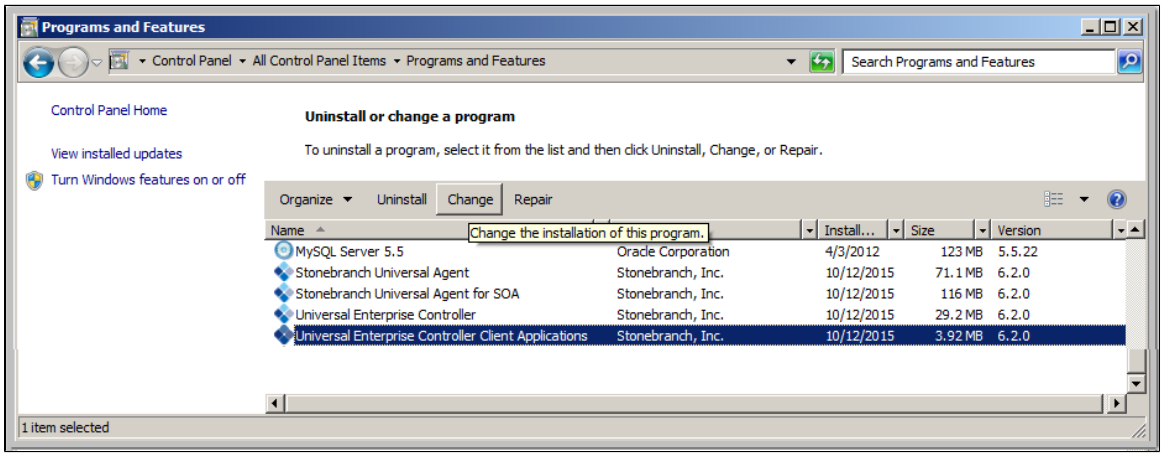
These installation modifications are:

- Adding or Removing UEC Client Applications Components
- Repairing a Corrupted UEC Client Applications Installation
- Removing a UEC Client Applications Installation

Adding or Removing UEC Client Applications Components

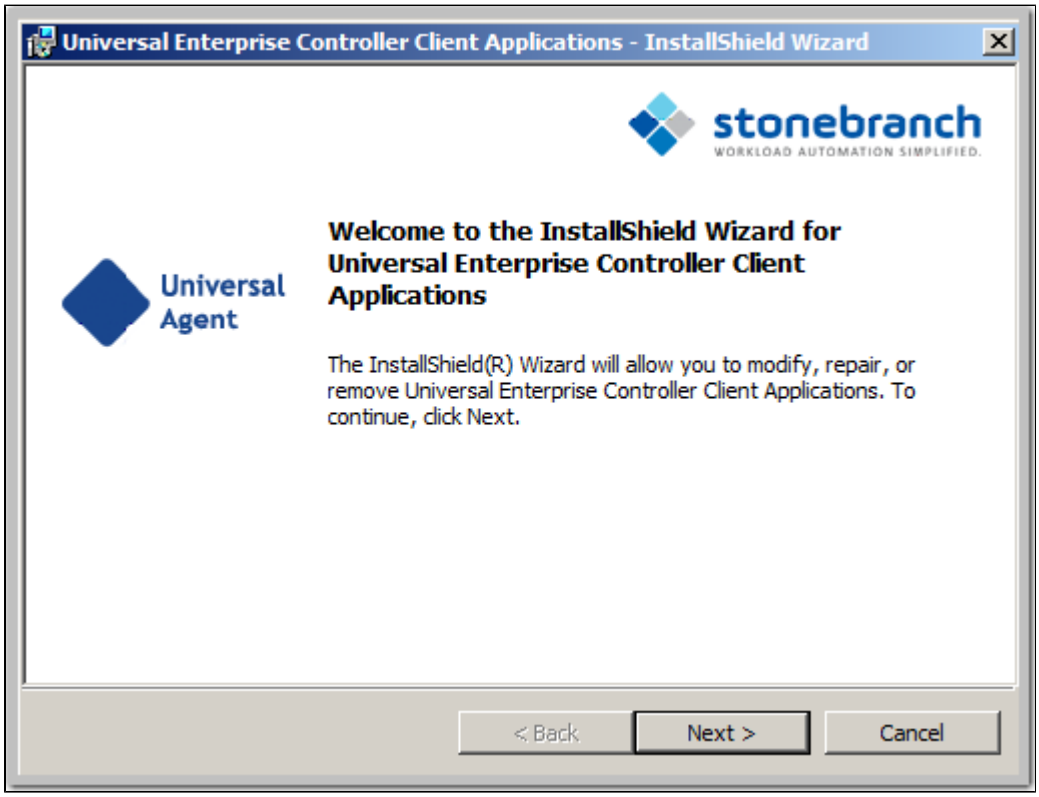
To add or remove components from a UEC Client Applications installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

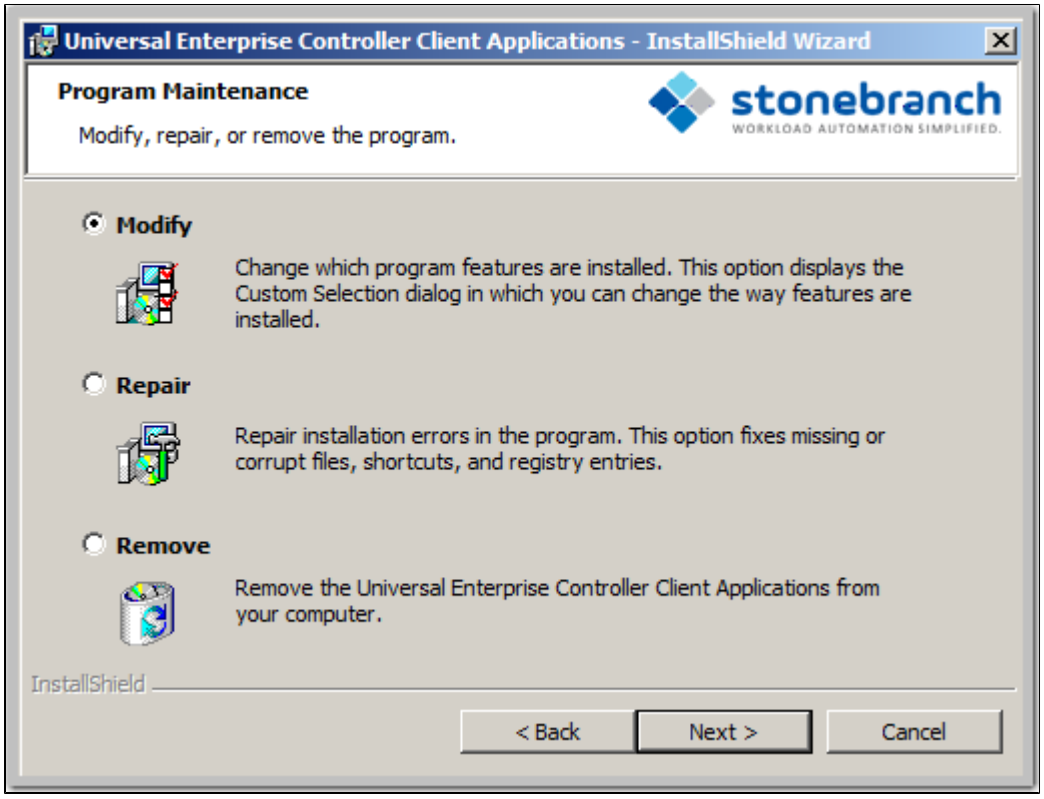


Windows Server 2003
If you are using Windows Server 2003, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller Client Applications** and click the **Change** button to start Windows Installer. The Welcome dialog displays.

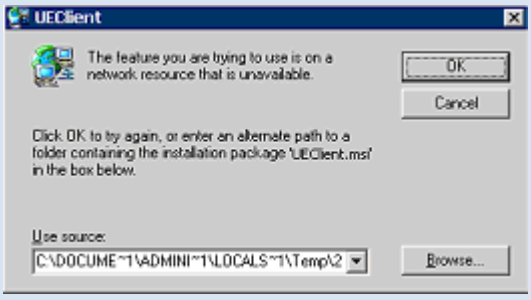


Step 3 Click the **Next>** button. The Program Maintenance dialog displays.

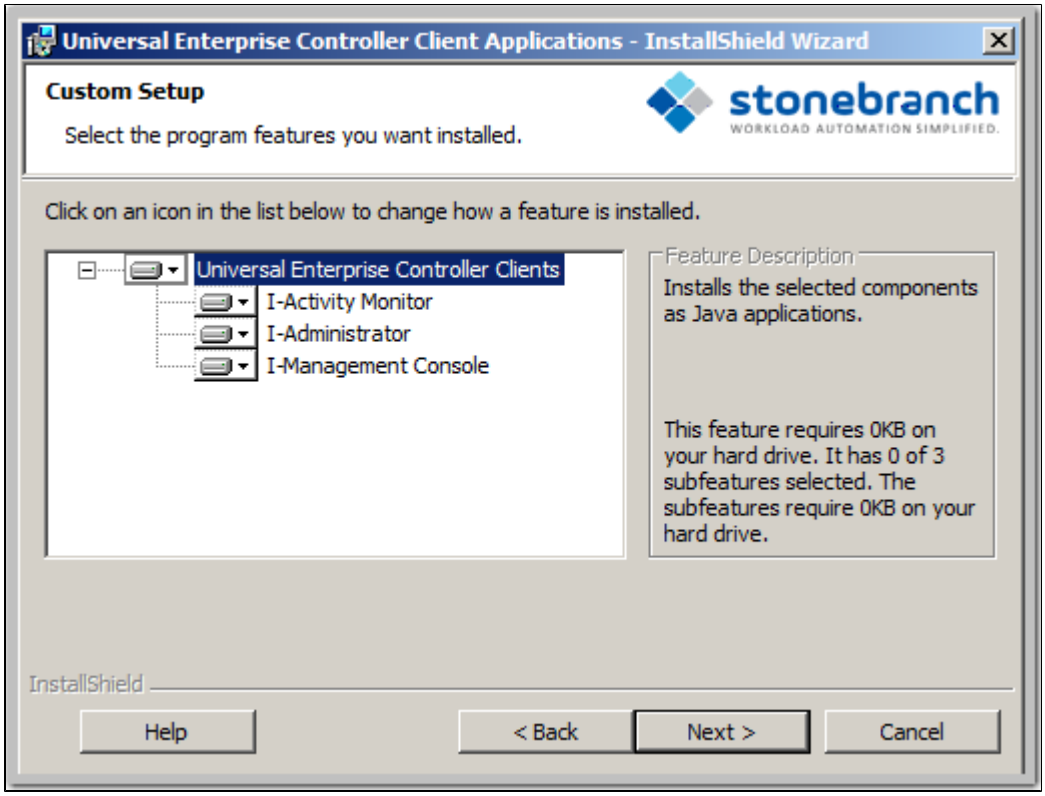


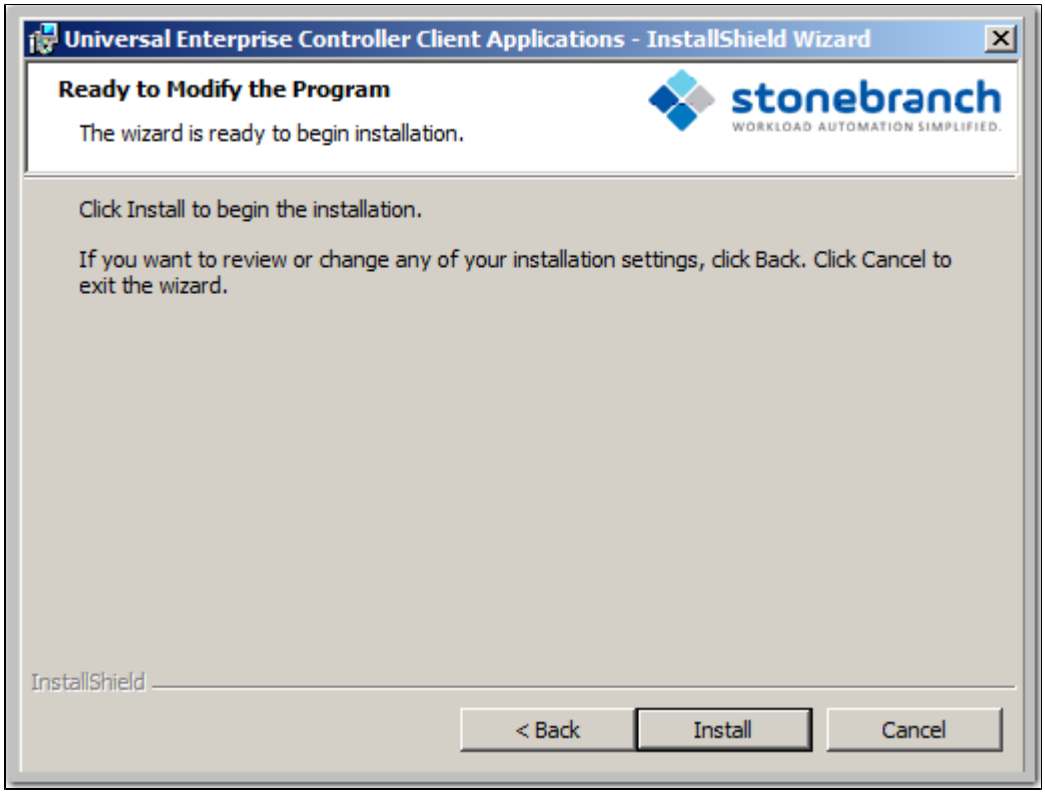
Windows Server 2003
If the UEC Client Applications were installed via a Remote Desktop connection, the environment settings used during that session may no longer be available. Windows Server 2003 allows multiple Remote Desktop sessions for multiple users, and each session has its own environment. Depending on the way the Remote Desktop session for the UEC Client Applications installation was established, some problems may arise whenever an attempt is made to modify that installation.

The distribution file's default extraction location is based on the **TEMP** environment variable. The location referenced by this variable can change between Remote Desktop sessions, and any files extracted there may not be accessible after the session is closed. Consequently, any attempts to modify the installation may fail because the Windows Installer cannot locate the installation's source files (a dialog similar to the one shown below may be displayed).



To resolve this issue, re-extract the distribution files to a location that is independent of a Remote Desktop environment and specify that location in the dialog above. Keep in mind, however, that the extracted files must come from the same distribution package used to do the initial install. If matching distribution files can't be found, the UEC Client Applications must be uninstalled and then reinstalled with the desired modifications.

Step 5	Click the **Modify** radio button, and then the **Next>** button, to display the Custom Setup dialog.  Currently installed components are identified by a drive icon. Uninstalled components are identified by an **X** icon.
Step 6	To remove a currently installed component: 1. Click the drive icon next to that component. 2. Select the X icon from the drop-down list to mark the component for removal.
Step 7	To add an uninstalled component: 1. Click the X icon next to that component. 2. Select the drive icon from the drop-down list to mark the component for installation.

Step 8	Click the **Next>** button to display the Ready to Modify the Program dialog. 

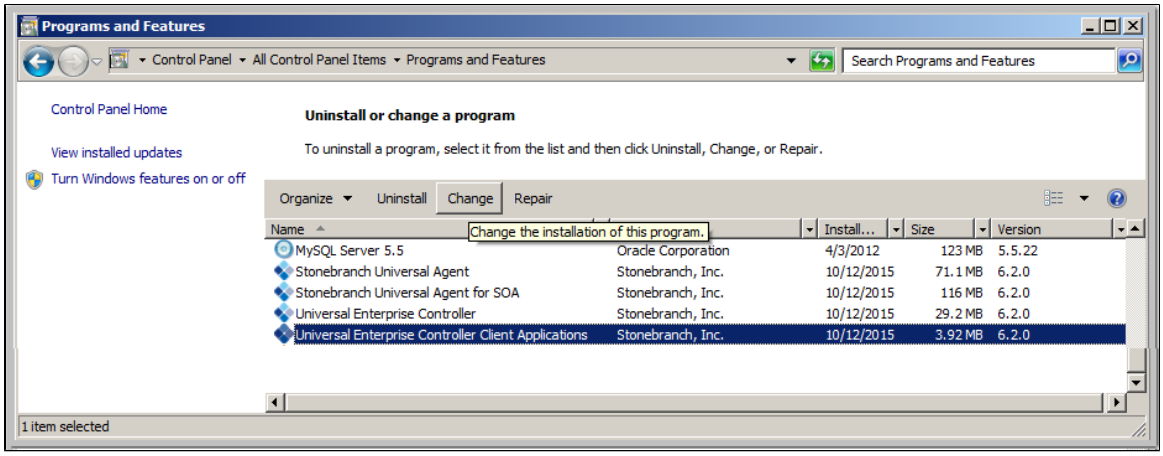
Step 9	Click the **Install** button to modify the installation.
	
	When the modifications are complete, the following actions will be taken:
	
	• Components marked with a drive icon will:
	  • Remain installed if they already are installed.
	  • Be installed if they are not already installed.
	• Components marked with an **X** will:
	  • Remain uninstalled if they are not currently installed
	  • Be removed if they currently are installed.

Repairing a Corrupted UEC Client Applications Installation

Windows Installer has the ability to recover accidentally deleted application files or registry entries required by the UEC Client Applications. This repair feature will re-install the missing items, making a complete re-install unnecessary.

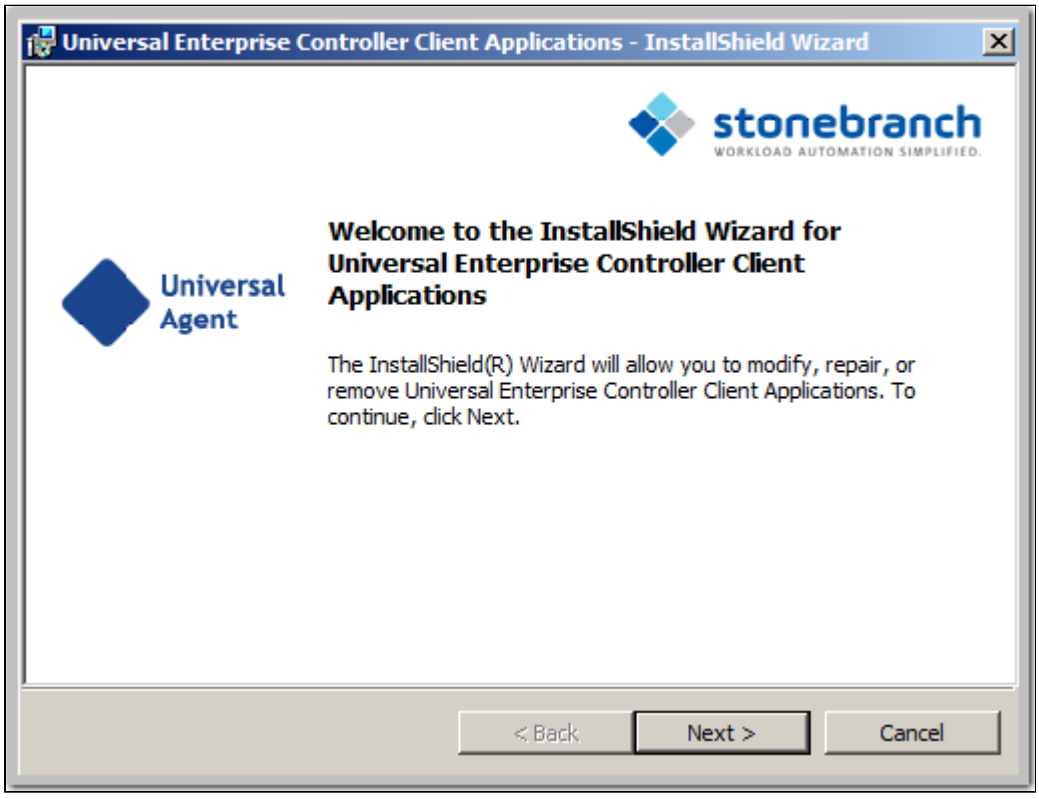
To repair an installation, perform the following steps:

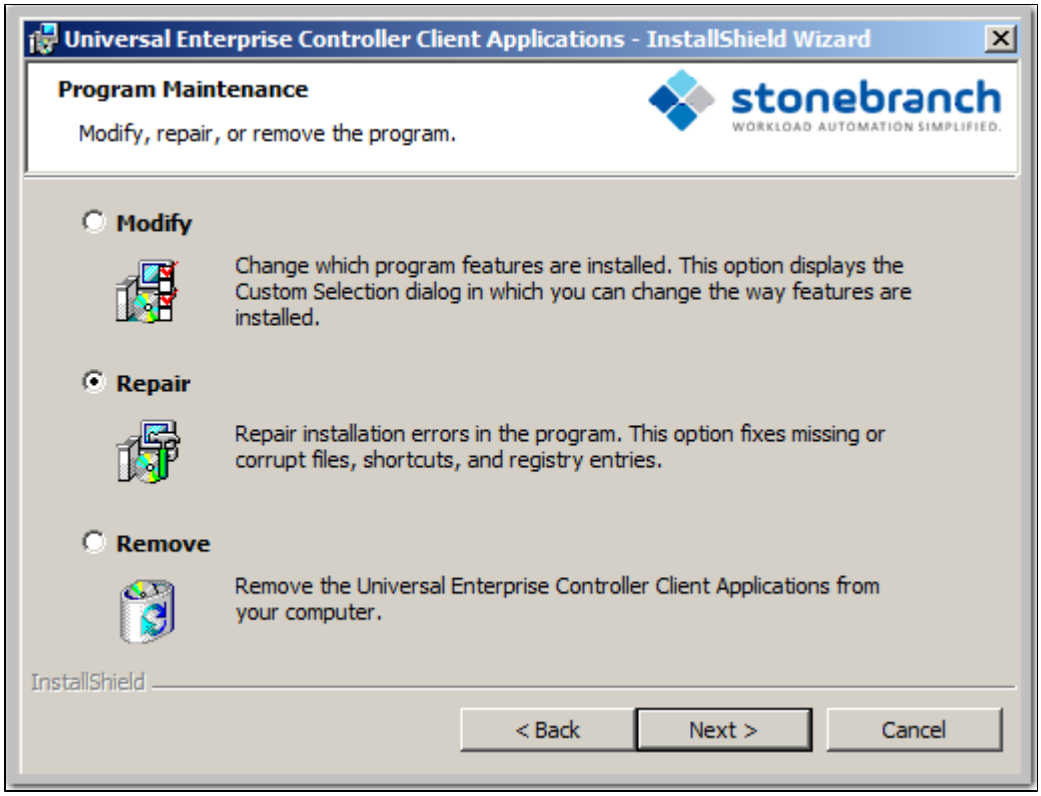
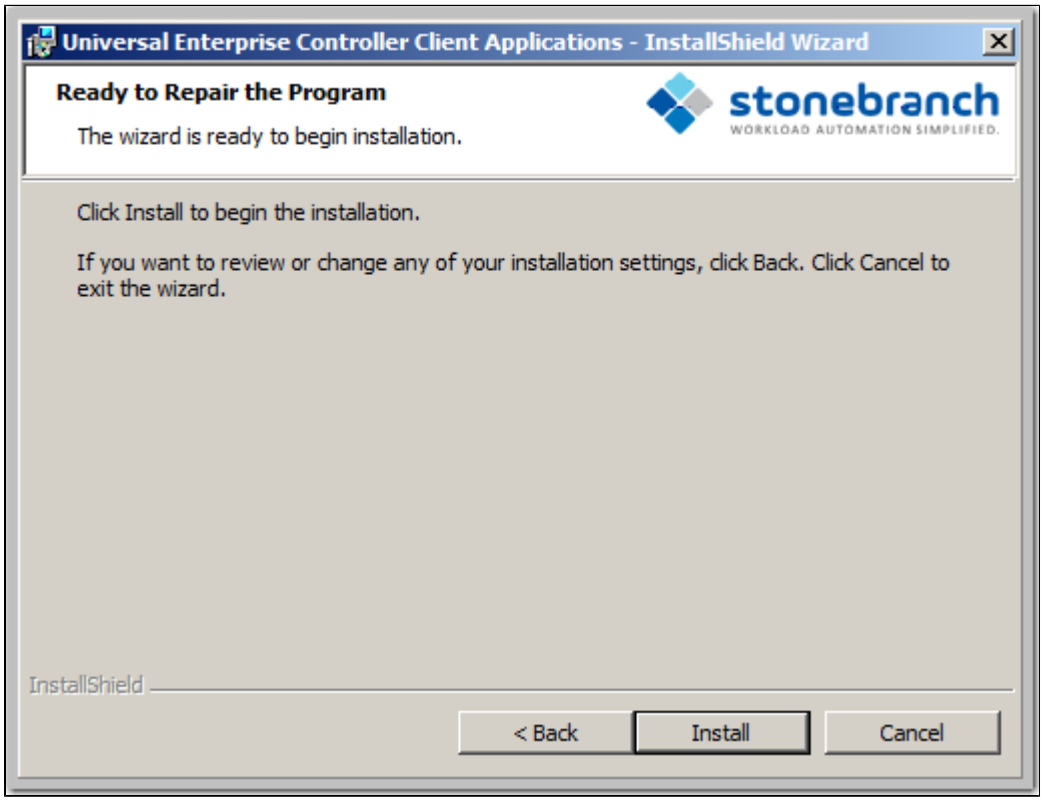
Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.



pre-Vista versions of Windows
If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller Client Applications** and click the **Change** button to start Windows Installer. The Welcome dialog displays.

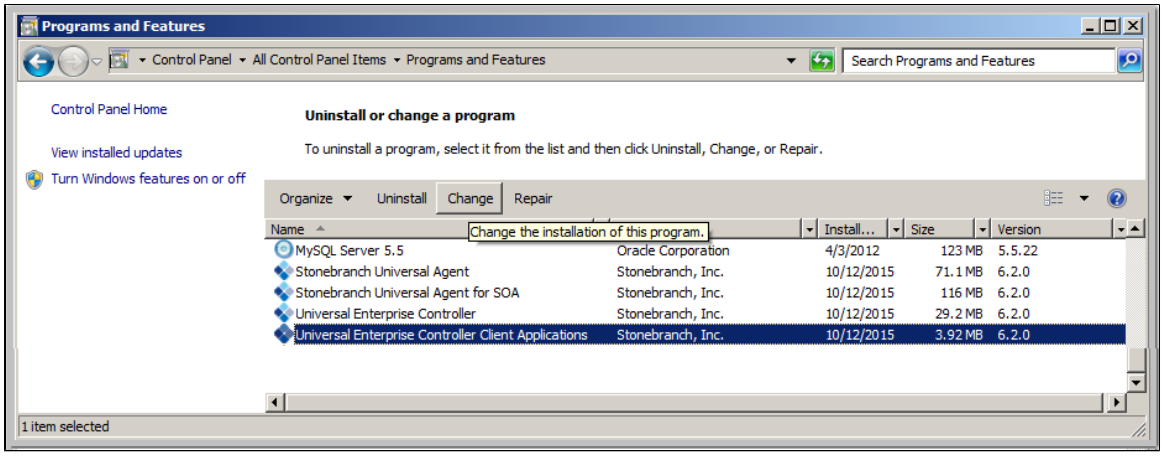


Step 3	Click the **Next>** button. The Program Maintenance dialog displays. 
Step 4	Click the **Repair** radio button, and then the **Next>** button, to display the Ready to Repair the Program dialog. 
Step 5	Click the **Install** button to repair the installation.

Removing a UEC Client Applications Installation

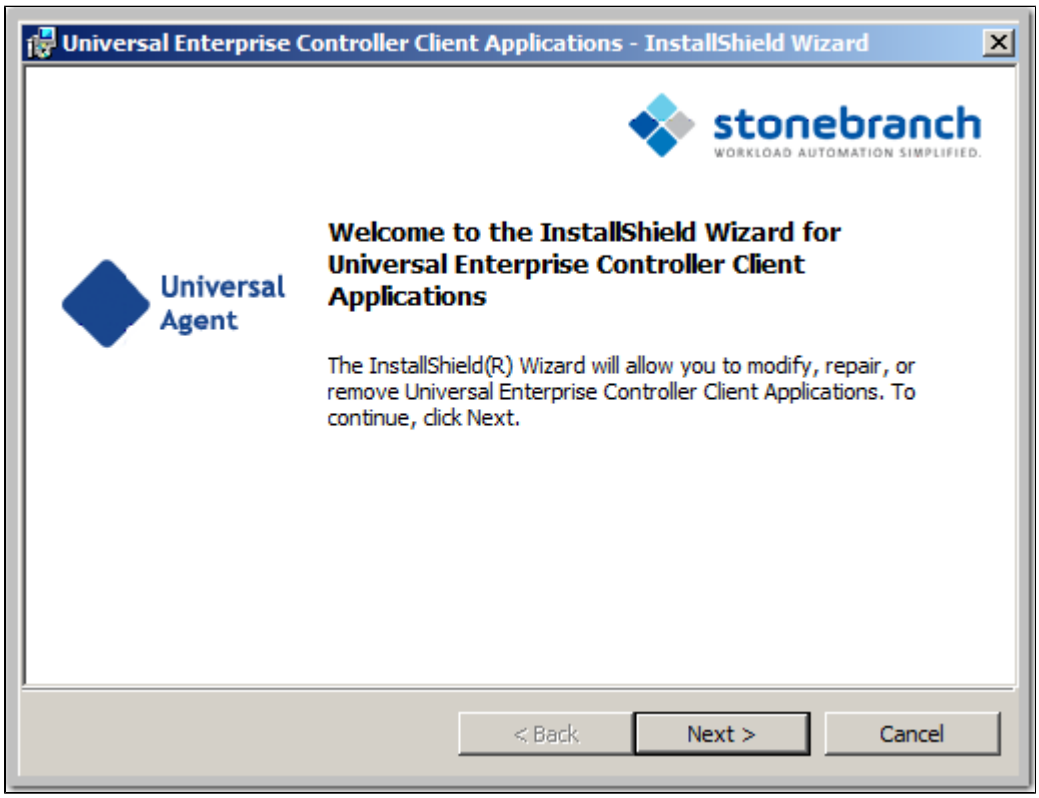
To uninstall a UEC Client Applications installation, perform the following steps:

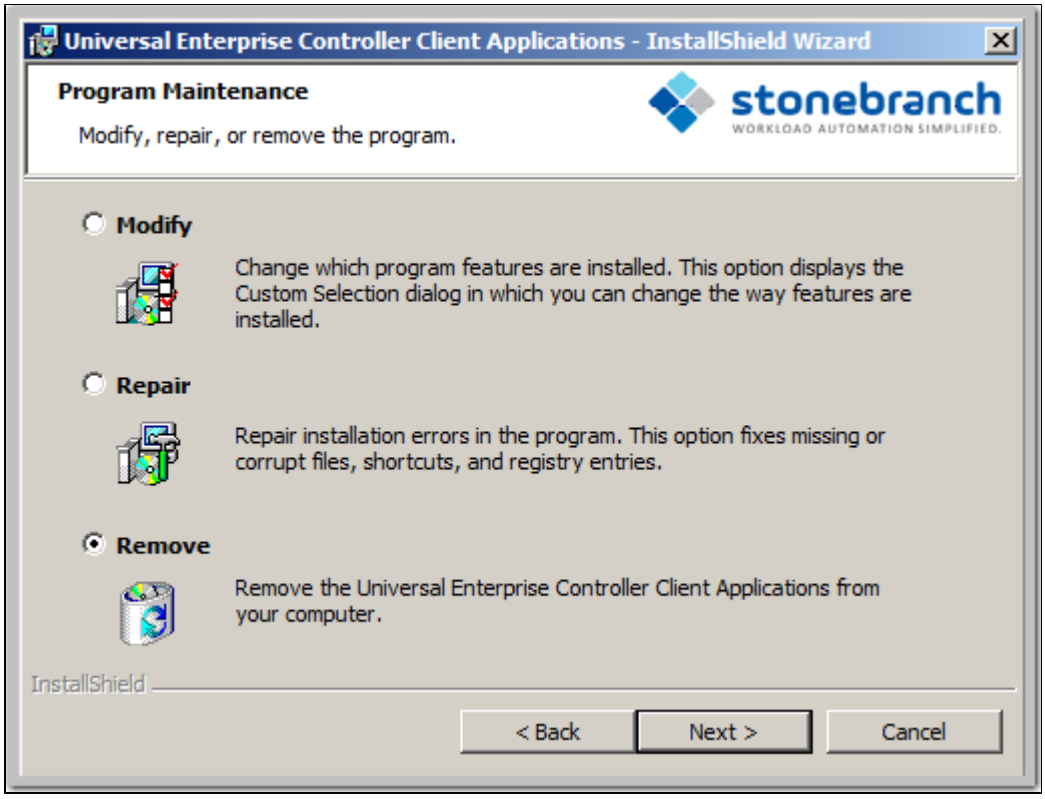
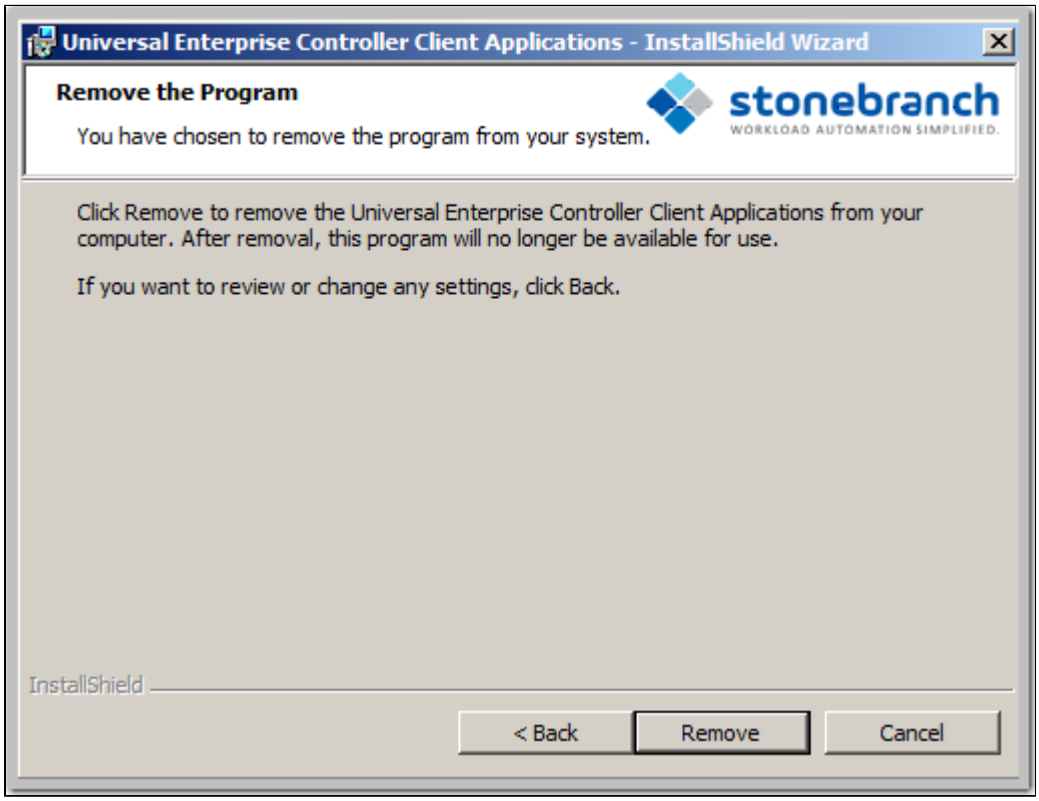
Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.



pre-Vista versions of Windows
If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Universal Enterprise Controller Client Applications** and click the **Change** button to start Windows Installer. The Welcome dialog displays.



Step 3	Click the **Next>** button. The Program Maintenance dialog displays. 
Step 4	Click the **Remove** radio button, and then the **Next>** button, to display the Remove the Program dialog. 
Step 5	Click the **Remove** button to remove the installation.

Un-Installed Files

The uninstall process will remove only those files created during the installation. Some files stored under the **.\Universal** install directory may be left behind after the uninstall. In this situation, those files and/or directories may simply be deleted.

Before deleting the entire **.\Universal** directory, make sure that no other Stonebranch, Inc. products are installed there. (See UEC Client Applications - File Inventory Lists for a list of files and directories installed with UEC Client Applications.)

Installing UEC Client Applications via the Command Line

- Introduction
- Installing UEC Client Applications
  - Command Line Syntax
  - Command Line Switches
  - Command Line Parameters
  - Command Line Installation Examples
- Detecting the Completion of Silent Installs

Introduction

This page describes how to install UEC Client Applications using the Windows Installer command line interface.

A command line installation is useful in situations where:

- Several UEC Client Applications installations must be deployed across many different systems.
- It is not practical or convenient to perform the graphical interface installation.
- It is necessary to generate an installation log file.

Installing UEC Client Applications

Step 1	Download the UEC Client Applications for Windows product distribution file, **`sb-UEClient-6.2.0.<level>-windows-i386.exe`**, to your work station.
Step 2	Execute the distribution file from the command line, and include all appropriate command line switches and parameters. The installation process determines whether a Windows Installer update is needed. The process then extracts and saves a Windows installer package file (.msi) to this location. After all files (including the .msi) are extracted from the distribution file, the installation process verifies that your machine meets the minimum installation requirements. If the requirements are met, the installation begins.

Command Line Syntax

The following illustrates the command line syntax used to install UEC Client Applications:

```
sb-UEClient-6.2.0.<level>-windows-i386.exe [/v"command line parameters"] [/s] [/w] [/x]
```

In this syntax:

- **`<level>`** is the numeric package level.

The command line switches (/v, /s, /w, and /x) are processed directly by the distribution file to control behavior of the Windows Setup application.

The command line parameters are passed to the Windows Installer (**msiexec**) to control the extracted Windows installer package file (.msi) behavior during the install process.

Command Line Switches

The following table identifies the command line switches available for a command line installation:

/v	Passes parameters to the Windows Installer (**msiexec**). The list of parameters must be enclosed in double (") quotation marks. See Command Line Parameters for available parameters.
/s	Suppresses the initialization and extraction dialogs displayed before the product install Welcome dialog." If you are using the /q command line parameter, use this switch additionally for a completely silent install.
/w	Instructs the Windows Setup application to wait until the installation completes. Use this switch when launching the installation from a script file. Without it, the Setup application may return immediately after launching Windows Installer.

/x	Uninstalls UEC Client Applications.

Command Line Parameters

The following table describes the parameters that are available for a command line installation.

The parameters can be specified in any order, with the following exceptions:

- If the Repair (/fom) or Remove (/x) parameter is used, it must be specified ***before*** all other parameters.
- If the Silent install (/q) and/or Log file (/L) parameters are used, they can be specified in any order, but they must be specified ***after*** all other parameters.

These parameters are preceded by the /v command line switch and must be enclosed in double (") quotation marks.

Parameter	Description	Default
/fom	Repairs a UEC Client Applications installation. **om** (after the **/f**) are options used by the repair. There are other options available, but for behavior that matches the repair done from the graphical install, the **om** options must be used. **/fom** cannot be used with the **/x** (remove) parameter.	n/a
/x	Removes the UEC Client Applications. **/x** cannot be used with the **/fom** (repair) parameter.	n/a
INSTALLTO =*installdir*	Sets the root installation directory to <**`installdir`**>. Each component will be installed under this directory. **INSTALLTO** is required if you want to install the UEC Client Applications in a directory that is different from the default, which varies depending on the type of installation being done (see PERUSER parameter). If the directory contains spaces, you must use double (**"**) quotation marks around the path name.	**per-machine installation**: Directory specified by the **PROGRAMFILES** environment variable (typically **`C:\Program Files\Universal`**). **per-user installation**: Location under the directory specified by the **USERPROFILE** environment variable. For example, if the **USERPROFILE** directory is set to **`C:\Documents and Settings\username`**, the default target directory will be **`C:\Documents and Settings\username\Application Data\Universal`**. If the UEC Client Applications already are installed, the installation directory will default to its existing location.

PERUSER={**0**|**1**}	Performs an installation for all users or a specific user account: • **0** specifies a per-machine installation; it performs an install for all users of a given machine. This means that any UEC Client Applications configuration options stored in the Windows registry, Start menu short cuts that launch the UEC Client Applications, and the applications files themselves will be visible to all accounts on the machine where the UEC Client Applications was installed. • **1** specifies a per-user installation; it performs an install that is specific to the user account doing the installation. This means that any UEC Client Applications configuration options stored in the Windows registry, Start menu short cuts that access the UEC Client Applications, and the application files themselves will be visible only to the user account with which the installation was performed. It will appear to all other users of a given machine as though the UEC Client Applications is not actually installed. **PERUSER** is required only under the following situations: • For per-user installs, when the installation is being done with a Windows account that is a member of the Administrators group. • For uninstalls, where a Windows account that is a member of the Administrators group is removing a per-user installation.	**0**, if UEC Client Applications is installed using an Administrative account. **1**, if a regular user account (that is, a non-Administrative user) is executing the installation.
UECADMIN ={**yes**|**no**}	Specification for whether or not to install the I-Administrator component during new installs, upgrades, or maintenance. • If **yes** is specified, I-Administrator will be installed. • If **no** is specified, I-Administrator will not be installed. If I-Administrator already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UECADMIN** is not required unless you want to change the current install state. **UECADMIN** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **I-Administrator** on the Custom Setup dialog when installing UEC Client Applications via the graphical interface.	**yes**
UAMONITOR ={**yes**|**no**}	Specification for whether or not to install the I-Activity Monitor component during new installs, upgrades, or maintenance. • If **yes** is specified, I-Activity Monitor will be installed. • If **no** is specified, I-Activity Monitor will not be installed. If I-Activity Monitor already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UAMONITOR** is not required unless you want to change the current install state. **UAMONITOR** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **I-Activity Monitor** on the Custom Setup dialog when installing UEC Client Applications via the graphical interface.	**yes**

UMGMTCON ={**yes**|**no**}	Specification for whether or not to install the I-Management Console component during new installs, upgrades, or maintenance. • If **yes** is specified, I-Management Console will be installed. • If **no** is specified, I-Management Console will not be installed. If I-Management Console already is present on the system (via a previous installation), it will be removed. Since, by default, each component's install state is preserved during an upgrade or maintenance, **UMGMTCON** is not required unless you want to change the current install state. **UMGMTCON** is ignored during an uninstall. Setting this parameter has the same effect as selecting whether or not to install **I-Management Console** on the Custom Setup dialog when installing UEC Client Applications via the graphical interface.	**yes**
/q	Suppresses the product installation dialogs. Use this parameter in addition to the /s command line switch for a completely silent install. See Command Line Switches, Command Line Installation Examples, and Detecting the Completion of Silent Installs for additional information regarding silent installs.	n/a
/L*v	Instructs the installation process to create an installation log file named <**logfilepath**> (full path name). If <**logfilepath**> contains spaces, you must enclose it with double (**"**) quotation marks around the path name. ***v** are flags used to specify the level of detail (**verbose**) contained in the log file. To reduce the amount of output generated, ***v** can be omitted. However, using these options is good practice; they can assist Stonebranch Customer Support with problem determination should any errors occur during installation.	n/a

Command Line Installation Examples

The following examples illustrate different ways that UEC Client Applications can be installed from the command line.

Graphical User Interface Install of All Components

To install all UEC Client Applications components via the graphical user interface, issue the following command:

```
sb-UEClient-6.2.0.x-windows-i386.exe
```

Graphical User Interface Install, All Components, with Log File

To install all UEC Client Applications components using the Windows Installer graphical user interface and write a log file to **C:\temp\install.log** during the installation, issue the following command:

```
sb-UEClient-6.2.0.x-windows-i386.exe /v"/l*v c:\temp\install.log"
```

Silent Install of All Components

To silently install all UEC Client Applications components, issue the following command:

```
sb-UEClient-6.2.0.x-windows-i386.exe /s /v"/qn"
```

Silent Install of All Components, Per-User Installation, Different Installation Directory

To silently install all UEC Client Applications components as a *per-user* installation (that is, one in which registry entries, Start menu shortcuts, and so on are visible only to the user performing the installation), and to override the default installation directory, issue the following command:

```
sb-UEClient-6.2.0.x-windows-i386.exe /s /v"/qn PERUSER=1 INSTALLTO=C:\UEClientApps\Universal"
```

Silent Install of All Components, with Log File

To silently install all UEC Client Applications components and write a log file to **C:\Temp\install.log** during the installation, issue the following command:

```
sb-UEClient-6.2.0.x-windows-i386.exe /s /v"/qn /l*v c:\temp\install.log"
```

Detecting the Completion of Silent Installs

If the **/q** switch is used to perform a silent install, no graphical interface or user interaction is required. One drawback to this is that no feedback is provided indicating when the Windows Installer process (install, uninstall, or repair) finishes.

One method that can be used to detect when the Installer process (**msiexec.exe**) ends is to execute it using the system's **start** command. Using available command line switches, the **start** command can be used to kick off the Installer process, and then wait for it to finish. When the **start** command returns control to its calling process (for example, the command prompt), the Installer process will have ended.

For example, from the command prompt, the following command can be issued to start the UEC Client Applications installation, and wait for it to finish.

```
start /b /wait sb-UEClient-6.2.0.x-windows-i386.exe /w /s /v"/qn"
```

- The **/b** switch prevents the **start** command from opening a new window.
- The **/wait** parameter causes the **start** command to start the application, **sb-UEClient-6.2.0.x-windows-i386.exe**, and then wait for it to finish.

The syntax above also can be used to execute the **start** command from within a script, such as a .**bat** file.

For more information on the **start** command, go to the Windows command prompt and enter: **start /?**.

Modifying a UEC Client Applications Installation via the Command Line

- Modifying a UEC Client Applications Installation via the Windows Installer Command Line Interface
- Adding or Removing UEC Client Applications Components
- Repairing a Corrupted UEC Client Applications Installation
- Removing UEC Client Applications from the Command Line

Modifying a UEC Client Applications Installation via the Windows Installer Command Line Interface

This page describes how to modify a UEC Client Applications installation via the Windows Installer command line interface.

After UEC Clients are is installed, Windows Installer can be run as many times as needed to modify the installation by:

- Adding or Removing UEC Client Applications Components
- Repairing a Corrupted UEC Client Applications Installation
- Removing a UEC Client Applications Installation

(For a description of the parameters used in these procedures, see Windows Installer Command Line Parameters.)

Adding or Removing UEC Client Applications Components

Currently, it only is possible to add or remove UEC Client Applications components using the Windows Installer graphical interface. (see **Adding or Removing Components** in Modifying a UEC Client Applications Installation via the Graphical Interface).

Repairing a Corrupted UEC Client Applications Installation

To recover accidentally deleted files or registry entries required by the UEC Client Applications using the Windows Installer command line interface, use the **/f** switch together with the **om** parameters.

These are the same repair options set internally by the graphical interface installation. They cause Windows Installer to reinstall files that are missing or that are older than the version contained in the UEC Client Applications distribution file.

Silent Repair

To repair a UEC Client Applications installation from the command line, without using the Windows Installer graphical interface, issue the following command:

```
msiexec.exe /fom SetupPath\UEClient.msi /q
```

Interactive Repair, with Log File

To repair a UEC Client Applications installation using the Windows Installer graphical interface, and instruct Windows Installer to write a log file to **C:\Temp\repair.log** while running the repair, issue the following command:

```
msiexec.exe /fom SetupPath\UEClient.msi /L C:\Temp\repair.log
```

Removing UEC Client Applications from the Command Line

To uninstall UEC Client Applications using the Windows Installer command line interface, use the **/x** command line switch.

Silent Removal

To uninstall the UEC Client Applications without using a graphical interface, issue the following command:

```
msiexec.exe /x SetupPath\UEClient.msi /q
```

Silent Removal, Per-User Installation

To remove a *per-user* installation that was installed using an Administrator account, issue the following command:

```
msiexec.exe /x SetupPath\UEClient.msi PERUSER=1 /q
```

A Stonebranch Tip
If you know that the UEC Client Applications were installed using an account that is a member of the Administrators group, but are not sure if a *per-user* or *per-machine* installation was done, it might be best to uninstall the Client Applications using the graphical interface (see Removing a UEC Client Applications Installation).

This will ensure complete removal of the UEC Client Applications.

If you are removing a UEC Client Applications installation that was installed using a non-administrative account, the **PERUSER** parameter is not necessary.

UEC Client Applications - 64-Bit Windows Editions

UEC Client Applications - Installing on 64-bit Windows Editions

All Universal Agent components have been tested and verified on the 64-bit editions of the following Windows systems:

- Windows XP
- Windows Server 2003
- Windows Vista
- Windows Server 2008
- Windows 7
- Windows Server 2008 R2

The installation defaults for the UEC Client Applications should not require any modification when installing on 64-bit Windows editions.

UEC Client Applications - File Inventory Lists

- UEC Client Applications - File Inventory List
  - System 32 Path
- I-Administrator
- I-Activity Monitor
- I-Management Console
- System Files

UEC Client Applications - File Inventory List

The UEC Client Applications installation includes the files required for the following components:

- I-Administrator
- I-Activity Monitor
- I-Management Console

If any of the components already are installed, Windows Installer will upgrade them to the latest version.

This page lists the files installed with each UEC Client Applications component. The file paths specified are relative to the root installation directory that was specified during the installation.

System 32 Path

Items shown with a path of System32 are installed in the 32-bit system folder. The actual name of this directory depends on the Windows version:

- For all supported 32-bit Windows editions, the path is **\Windows\System32**.
- For all supported 64-bit Windows editions, the path is **\Windows\SysWow64**.

I-Administrator

File	Description
UECAdmin\uecadmin.jar	I-Administrator Java archive (JAR) file.
UECAdmin\lib\ueccommon.jar	Common routines shared between I-Administrator and I-Management Console.
UECAdmin\lib\uecumc.jar	I-Management Console resources used by I-Administrator.

I-Activity Monitor

File	Description
UAMonitor\uam.jar	I-Activity Monitor Java archive (JAR) file.

I-Management Console

File	Description
UMConsole\uecumc.jar	I-Management Console Java archive (JAR) file.
UMConsole\lib\ueccommon.jar	Common routines shared between I-Management Console and I-Administrator.

System Files

The following files will be installed only if they are newer than the existing file.

File	Description
System32\msiexec.exe	Version 3.1.4000.1823 of the Microsoft Windows Installer (see Windows Installer for more information).

Universal Agent for SOA for Windows Installation

Overview

The following information is provided for the installation of Universal Agent for SOA for Windows:

- Installation Package
- Installation Requirements
- Pre-Installation - Upgrade Backups
- Installation Procedures
- Configuring and Starting UAC Server
- 64-Bit Windows Editions
- File Inventory Lists

(For licensing information, see Windows Installation - Licensing.)

Universal Agent for SOA for Windows - Installation Package

Components

The Universal Agent for SOA 6.2.x for Windows package includes the following components:

- Universal Agent for SOA
  - Universal Application Container Server
  - Universal Application Container
  - Universal Application Interface

Component Compatibility

The following table identifies the compatibility of Universal Agent for SOA 6.2.x for Windows with previous component / product versions.

Component	Compatibility
Universal Automation Center 5 for SOA 6.2.x	Universal Command Manager 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, and 3.2.0.

The component references pertain to all supported platforms for that version.

Universal Agent for SOA for Windows - Installation Requirements

Windows Systems

To install Universal Agent for SOA for Windows, you must have one of the following versions of Windows:

- Windows Server 2003 SP1 and higher
- Windows XP SP3
- Windows Vista
- Windows 7
- Windows Server 2008
- Windows Server 2008 R2
- Windows Server 2012
- Windows Server 2012 R2

Additional Requirements

In addition, you must have:

- An account with administrative privileges.
- Possible reboot. A reboot is required if the Windows Installer service is not installed, a version of the Windows Installer prior to 3.1.4000.1283 is installed, or required files are in use at the time of the installation.
- TCP/IP.
- About 110 megabytes of disk space.
- Universal Agent 3.2.0 or later (32-bit packages only).

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

Universal Agent for SOA for Windows - Pre-Installation - Upgrade Backups

Universal Agent for SOA for Windows - Pre-Installation / Upgrade Backups

Before upgrading to the latest release of Universal Agent for SOA, we recommend stopping any active Universal Application Container (UAC) Server instances (via the Universal Control utility or by stopping the local Universal Broker). If the UAC Server is active during the upgrade, the Windows Installer will require a reboot of the system. Also, stopping the UAC Server before backing up the files listed below will ensure the latest copies of each are captured.

The installation process overwrites the current files (exception: see the Note for Log4jConfiguration.xml in the following table), effectively removing your modifications. Backing up these files will optimize the time it takes you to get up and running after installing or upgrading.

The following table identifies the files - and their locations - that should be backed up or copied before you install a new release or upgrade a current release.

File	Location
UAC.xml File	**`%ALLUSERSPROFILE%\Application Data\Universal\uac`**
Log4jConfiguration.xml File	**`Program Files\Universal\uac`** (UAC) **`Program Files\Universal\uai`** (UAI) **Note** The installation process does not overwrite Log4jConfiguration.xml files.
JMS Provider Client Jar Files	**`Program Files\Universal\uac\container\webapps\axis2 \WEB-INF\lib`** **Note** The names of the jar files differ depending on which JMS Provider you are using.
JMS Provider Client Properties Files	**`Program Files\Universal\uai\xml`** **Note** These are suggested locations; you can place these files anywhere on the files system. If you have located these files under the **uai** directory, you should back them up.
Payload Files	Normally, payload files should be located on the host system where Universal Command Manager is installed. If you have chosen to store them elsewhere, the suggested location is: **`Program Files\Universal\uai\xml`** **Note** You can place these files anywhere on the files system. If you have located these files under the **uai** directory, you should back them up.

Universal Agent for SOA for Windows - Installation Procedures

Universal Agent for SOA for Windows - Installation Procedures

The following procedures are provided for the installation and modification of Universal Agent for SOA for Windows:

- Installing SOA for Windows via the Graphical Interface
- Modifying a SOA for Windows Installation via the Graphical Interface
- Installing SOA for Windows via the Command Line
- Modifying a SOA for Windows Installation via the Command Line

Note
Modifying a Universal Agent for SOA installation refers to the adding / removing of Universal Agent for SOA components, repairing a corrupted installation, or removing an installation. To change the installed version of Universal Agent for SOA, see Upgrading Universal Agent and Applying Maintenance to Universal Agent.

Installing SOA for Windows via the Graphical Interface

Installing Universal Agent for SOA via the Windows Installer Graphical Interface

To install Universal Agent for SOA for Windows using the Windows Installer graphical interface, perform the following steps:

Step 1	Download the Universal Agent for SOA for Windows product distribution file, **sb-soa-6.2.0.<level>-windows-i386.exe**, to your work station.

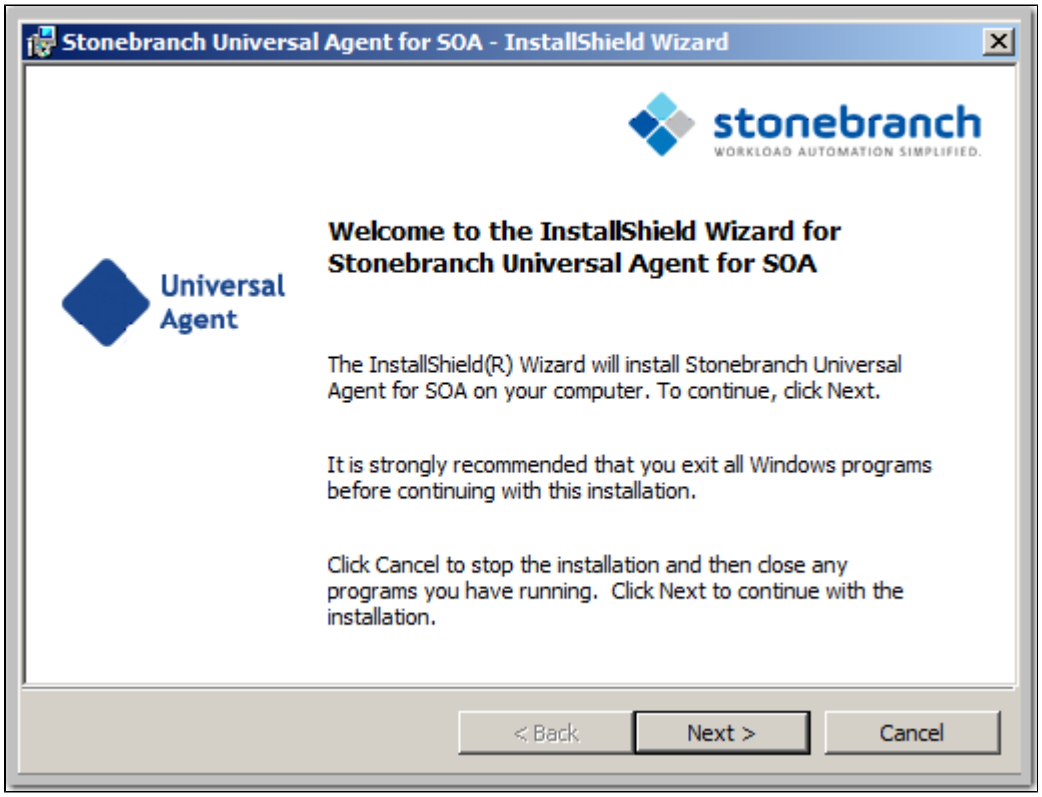
Step 2 Execute the distribution file to extract the files.

> **Note**
> If you already have extracted the files from the distribution file, but cancelled the installation process in order to install Windows Installer separately (see Windows Installer), you can simply double-click the extracted Universal Agent for SOA installation file, **UPforSOA.msi**, to begin the installation.

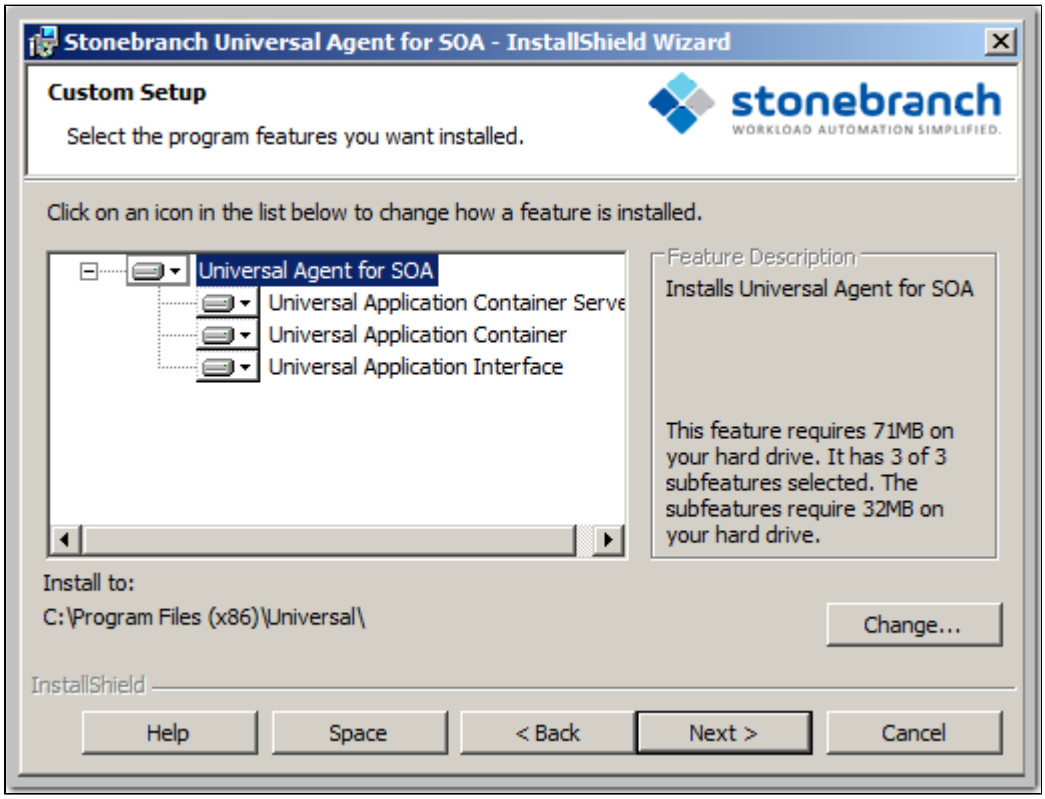
> **Installing Over a Remote Desktop Session**
> Starting with Windows Server 2003, Remote Desktop provides distinct session environments for each logged-in user. This means the file extraction location may rely on an environment setting that is not available after the Remote Desktop session ends.
>
> For example, the distribution file's default extraction location is based on the **TEMP** environment variable. The location referenced by this variable can change between Remote Desktop sessions, and any files extracted there may not be accessible after the session is closed.
>
> To ensure that extracted files and other required resources are accessible after the initial install, extract the files to a well-known location that is not likely to change between Remote Desktop sessions.
>
> (Refer to the Microsoft documentation on the Remote Desktop feature for additional information.)

The installation automatically begins after the file extraction completes.

It first will verify that your machine meets the minimum system requirements (see Installation Requirements). If the requirements are met, a Welcome dialog displays.



Step 3 Click the **Next** button. A list of Universal Agent for SOA components included in the installation package then displays. It is from this list that you can select which components to install.



For a new installation, a drive icon displays next to each item in the list, indicating that the component will be installed.

For an upgrade installation, the drive icon next to each item indicates that the component is either:

- New to the installation and will be installed.
- Currently installed and will be upgraded.

A Stonebranch Tip

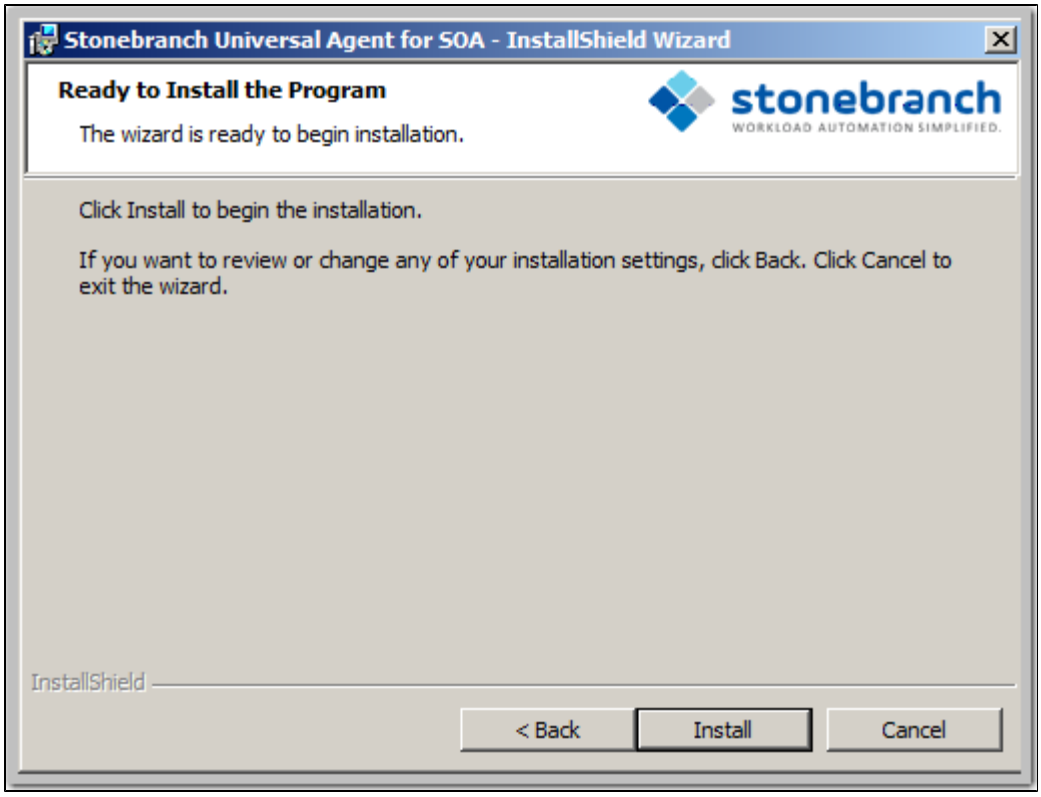
- If the installation detects an existing Universal Agent for SOA installation, currently installed components may be upgraded. (Currently, there is no way to specify that the state of a currently installed component remain unchanged.)
- If a component is selected for installation, and the version of the installed component is below the version of the component being installed, the installed component will be replaced by the component being installed.

Step 4 The previous figure shows that all Universal Agent for SOA components will be installed in their respective directories under the **C:\Program Files\Universal** directory.

1. If you want to select a different location, click the **Change...** button.
2. If you want to check the amount of disk space required for the installation, and the amount of available disk space on the selected directory, click the **Space** button.

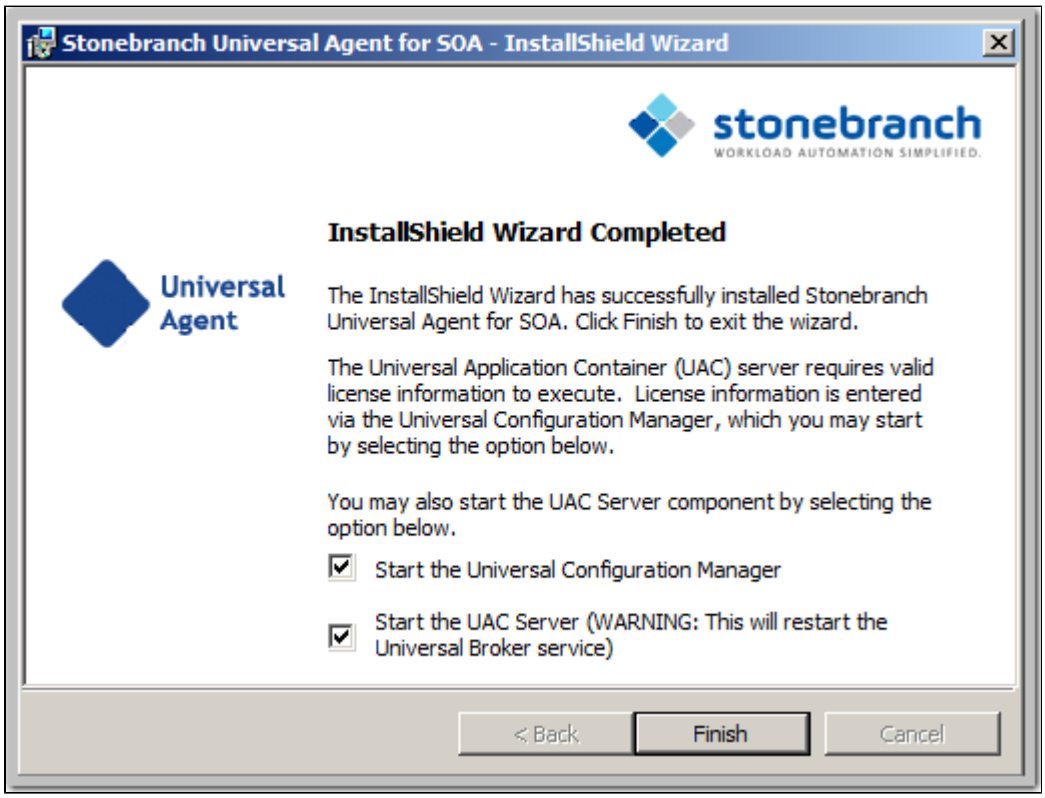
Step 5 After verifying the install location, click the **Next>** button to continue the installation process.

When the installation is ready to begin, the Ready to Install dialog displays.



Click the **Install** button to begin the installation or click the **<Back** button to return to change information on any of the previous dialogs.

When the installation completes successfully, the Installation Complete dialog displays.



Step 6	If the Universal Application Container (UAC) Server was installed, the following options display on this dialog: • **Start the Universal Configuration Manager** • **Start the UAC Server** License information must be entered into the Universal Application Container Server's configuration before the server will run. Select both of these options to enter the license information and then start the UAC server.
Step 7	Click the **Finish** button to exit the Windows installation.

Modifying a SOA for Windows Installation via the Graphical Interface

- Modifying a Universal Agent for SOA Installation via the Windows Installer Graphical Interface
- Repairing a Corrupted Universal Agent for SOA Installation
- Removing a Universal Agent for SOA Installation
  - Un-Installed Files

Modifying a Universal Agent for SOA Installation via the Windows Installer Graphical Interface

This section describes how to modify a Universal Agent for SOA installation via the Windows Installer graphical interface.

After installing the Universal Agent for SOA, run the installation programs as needed to modify the installation by:

- Repairing a corrupted Universal Agent for SOA Installation.
- Removing a Universal Agent for SOA Installation.

Note
Although Windows Installer provides a Modify selection for Universal Agent for SOA, it cannot be modified; that is, individual components cannot be added or removed.

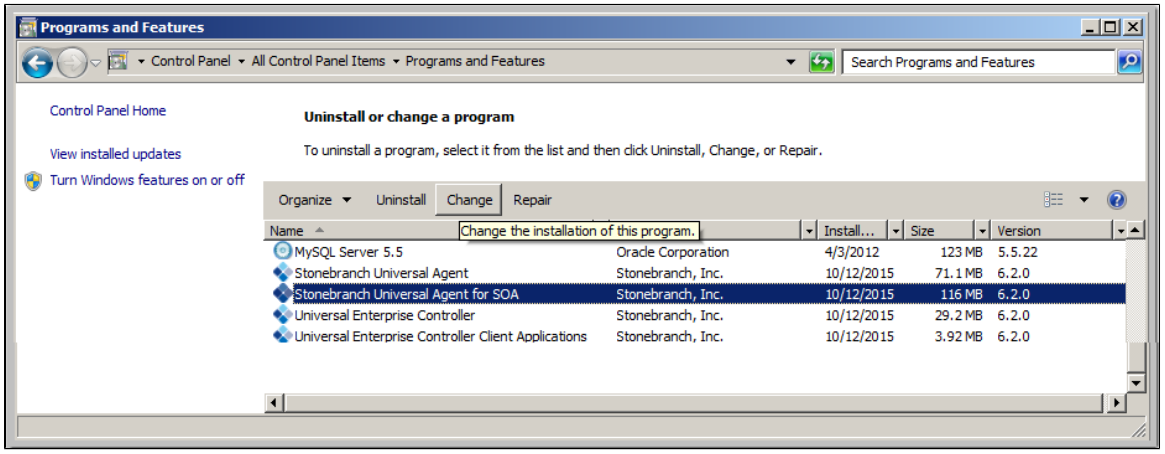
Repairing a Corrupted Universal Agent for SOA Installation

The installation program has the ability to recover accidentally deleted application files, configuration and component definition files, or registry entries required by Universal Agent for SOA. This repair feature will re-install the missing items, making a complete re-install unnecessary.

During a repair, any options stored in configuration and component definition files are preserved. If a configuration or component definition file was deleted, the installation will create a new file with default values.

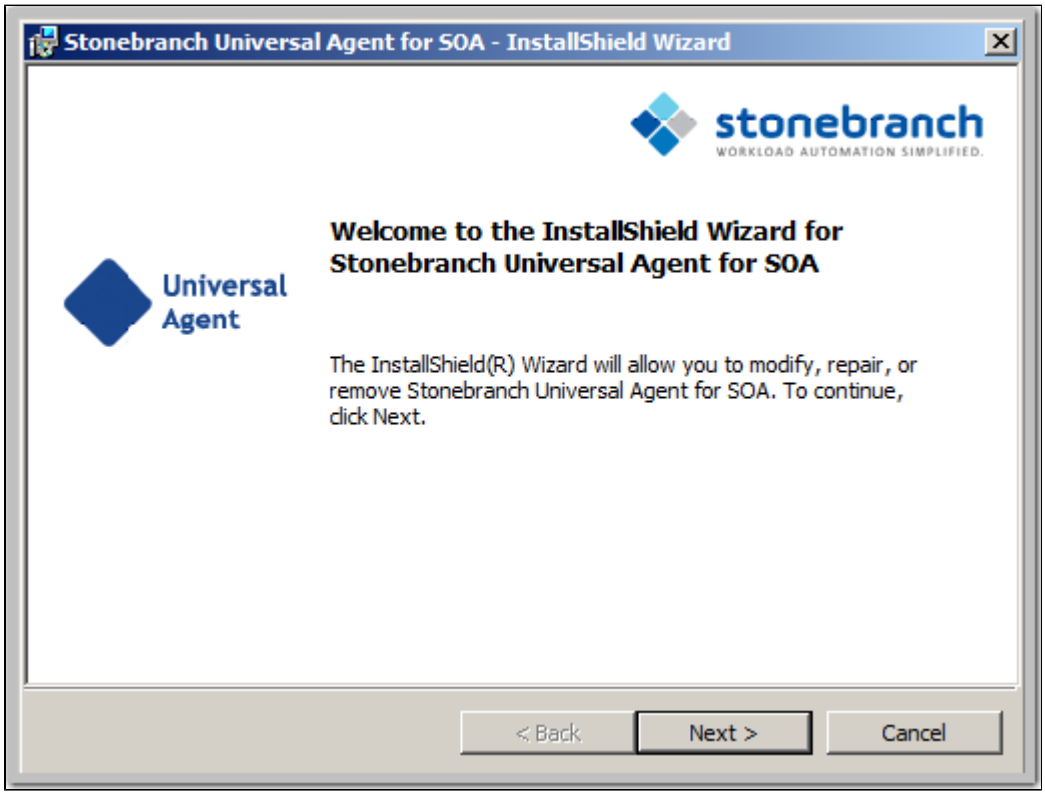
To repair an installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

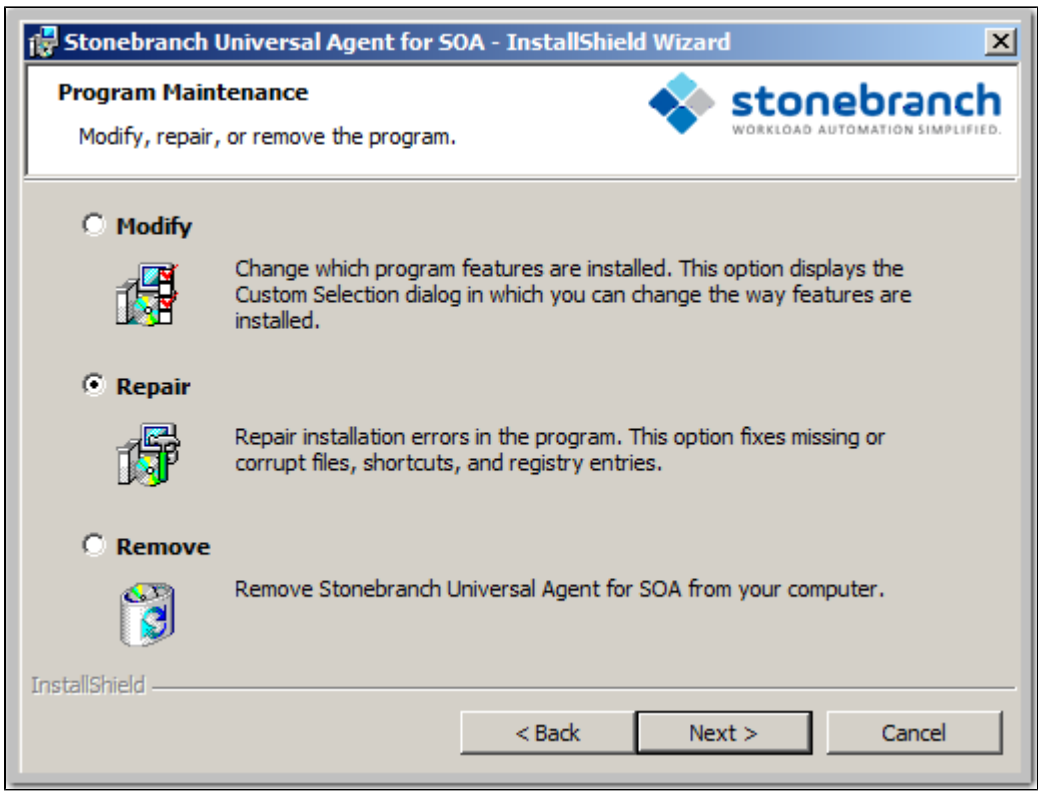


Windows Server 2003
If you are using Windows Server 2003, select **Add or Remove Programs** on the Windows Control Panel.

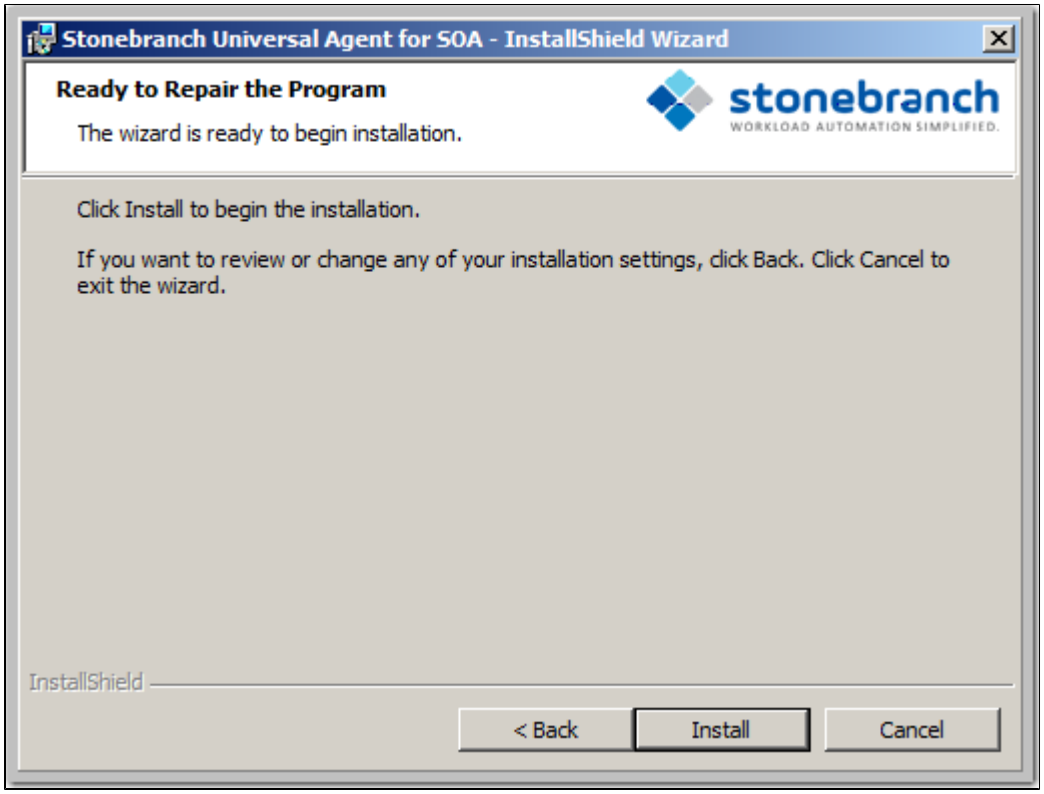
Step 2 From the list of installed programs, select **Stonebranch Workload Automation for SOA** and click the **Change** button to start Windows Installer. The Welcome dialog displays.



Step 3 Click the **Next>** button. The Program Maintenance dialog displays.



Step 4 Click the **Repair** radio button, and then the **Next>** button, to display the Ready to Repair the Program dialog.

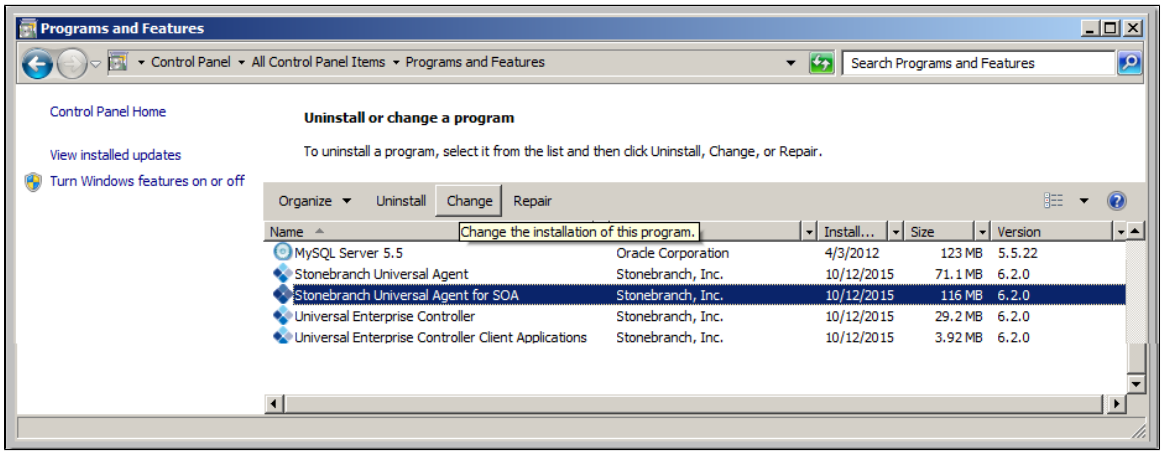


Step 5 Click the **Install** button to repair the installation.

Removing a Universal Agent for SOA Installation

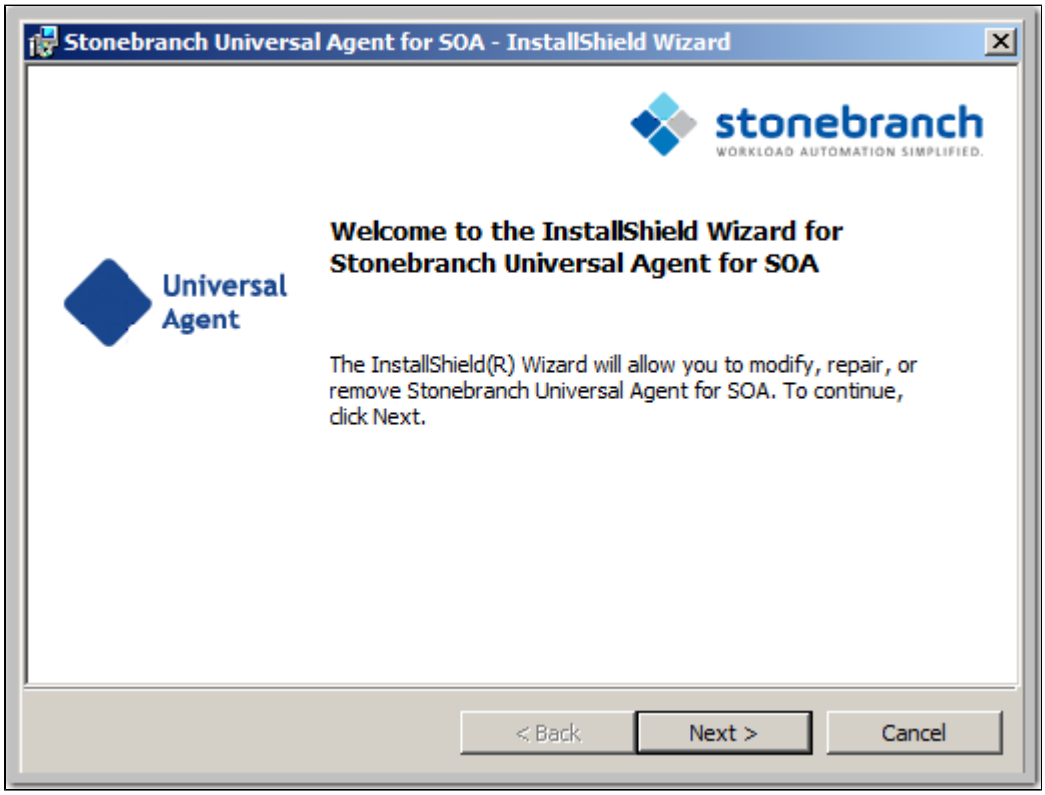
To uninstall a Universal Agent for SOA installation, perform the following steps:

Step 1 On the Windows Control Panel, select **Programs and Features**. The Programs and Features dialog displays.

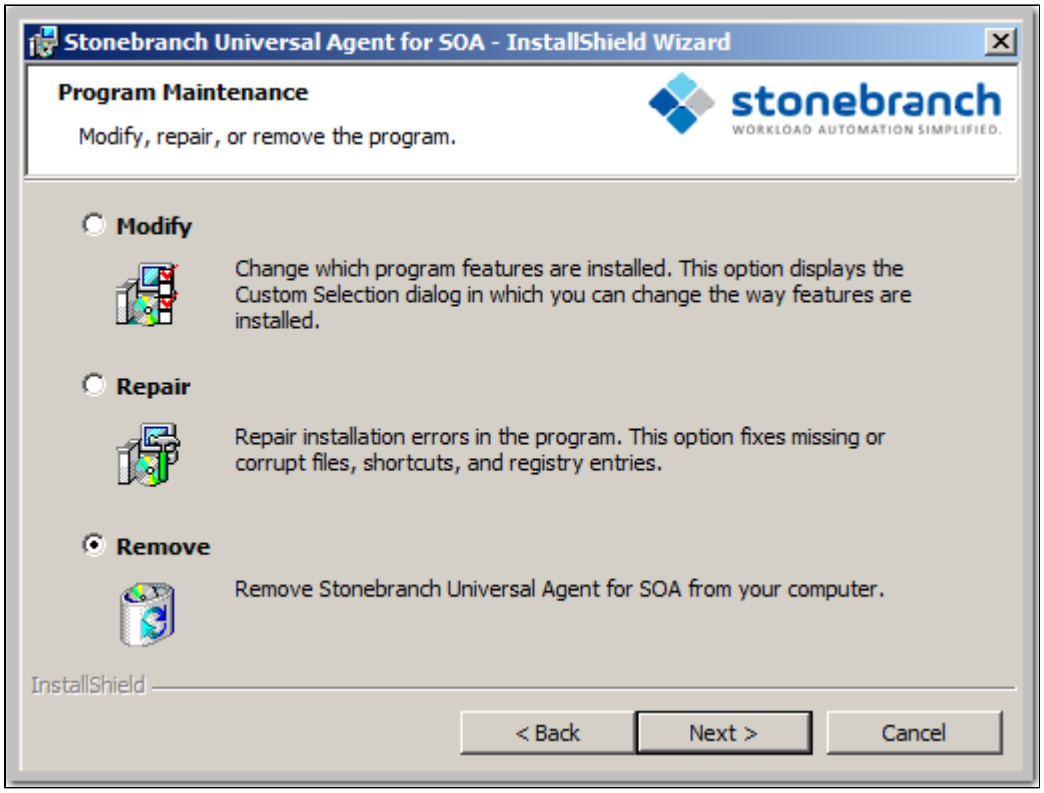


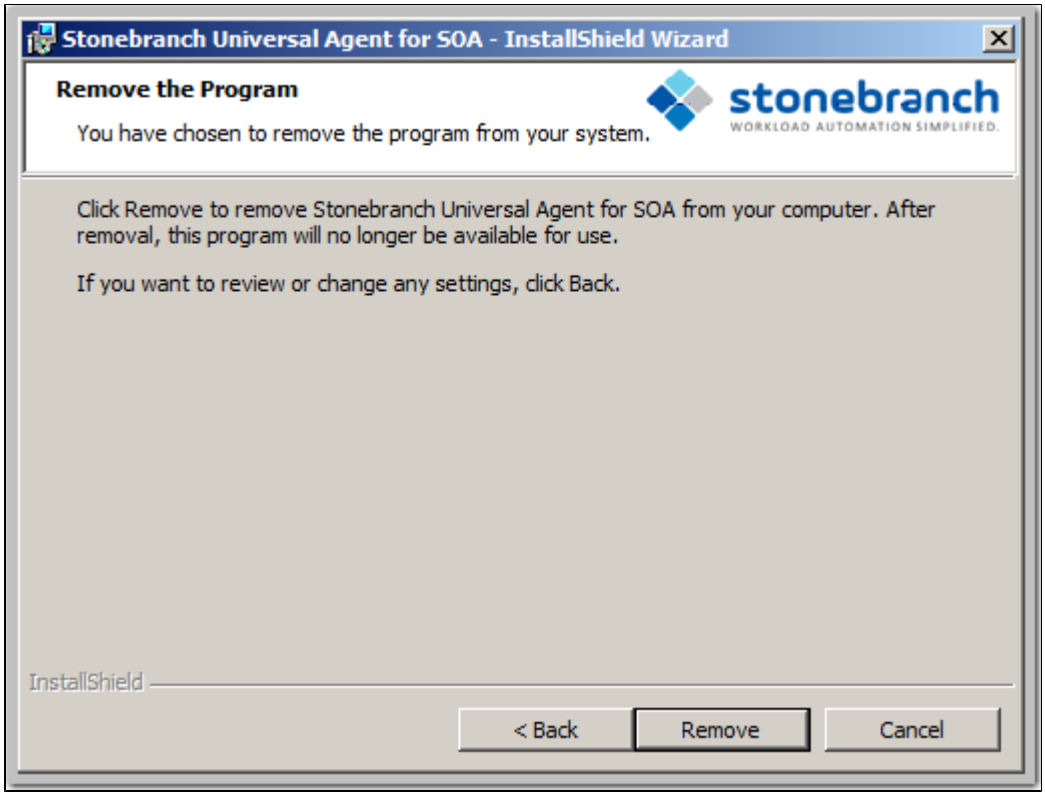
pre-Vista versions of Windows
If you are using an earlier version of Windows than Windows Vista, select **Add or Remove Programs** on the Windows Control Panel.

Step 2 From the list of installed programs, select **Stonebranch Workload Automation for SOA** and click the **Change** button to start Windows Installer. The Welcome dialog displays.



Step 3 Click the **Next>** button. The Program Maintenance dialog displays.



Step 4	Click the **Remove** radio button, and then the **Next>** button, to display the Remove the Program dialog. 
Step 5	Click the **Remove** button to remove the installation.

Un-Installed Files

The uninstall process will remove only those files created during the installation. Some files stored under the **.\Universal** install directory by Universal Agent for SOA, such as trace files, may be left behind after the uninstall. In this situation, simply delete the uninstalled files and/or directories.

Before deleting the entire **.\Universal** directory, make sure it does not contain application files for any other installed Stonebranch, Inc. product. (See Universal Agent for SOA for Windows - File Inventory Lists for a list of files and directories installed with Universal Agent for SOA.)

Installing SOA for Windows via the Command Line

- Introduction
- Installing Universal Agent for SOA
  - Command Line Syntax
  - Command Line Switches
  - Command Line Parameters
  - Command Line Installation Examples
- Detecting the Completion of a Silent Install

Introduction

This page describes how to install Universal Agent for SOA using the Windows Installer command line interface.

A command line installation is useful in situations where:

- Several Universal Agent for SOA installations must be deployed across many different systems.
- It is not practical or convenient to perform a graphical interface installation.
- It is necessary to generate an installation log file.

Installing Universal Agent for SOA

Step 1	Download the Universal Agent for SOA for Windows product distribution file, **`sb-soa-6.2.0.<level>-windows-i386.exe`**, to your work station.
Step 2	Execute the distribution file from the command line, and include all appropriate command line switches and parameters. The installation process determines whether a Windows Installer update is needed. The process then extracts and saves a Windows installer package file (.msi) to this location. After all files (including the .msi) are extracted from the distribution file, the installation process verifies that your machine meets the minimum installation requirements. If the requirements are met, the installation begins.

Command Line Syntax

The following illustrates the command line syntax used to install Universal Agent for SOA:

```
sb-soa-6.2.0.-windows-i386.exe [/v"command line parameters"] [/s] [/w] [/x]
```

In this syntax:

- **`<level>`** is the numeric package level.

The command line switches (/v, /s, /w, and /x) are processed directly by the distribution file to control behavior of the Windows Setup application.

The command line parameters are passed to the Windows Installer (**msiexec**) to control the extracted Windows installer package file (.msi) behavior during the install process.

Command Line Switches

The following table describes the command line switches available for a command line installation:

/v	Passes parameters to the Windows Installer (**msiexec**). The list of parameters must be enclosed in double (") quotation marks. See Command Line Parameters for available parameters.
/s	Suppresses the initialization and extraction dialogs displayed before the product install Welcome dialog." If you are using the /q command line parameter, use this switch additionally for a completely silent install.
/w	Instructs the Windows Setup application to wait until the installation completes. Use this switch when launching the installation from a script file. Without it, the Setup application may return immediately after launching Windows Installer.

/x	Uninstalls Universal Agent for SOA for Windows.

Command Line Parameters

The following table describes the parameters that are available for a command line installation.

The parameters can be specified in any order, with the following exceptions:

- If the Repair (/fom) or Remove (/x) parameter is used, it must be specified ***before*** all other parameters.
- If the Silent install (/q) and/or Log file (/L) parameters are used, they can be specified in any order, but they must be specified ***after*** all other parameters.

These parameters are preceded by the /v command line switch and must be enclosed in double (") quotation marks.

Parameter	Description	Default
/fom	Repairs a Universal Agent for SOA for Windows installation. **om** (after the **/f**) are options used by the repair. There are other options available, but for behavior that matches the repair done from the graphical install, the **om** options must be used. **/fom** cannot be used with the **/x** (remove) parameter.	n/a
/x	Removes a Universal Agent for SOA for Windows installation. **/x** cannot be used with the **/fom** (repair) parameter.	n/a
INSTALLDIR *=installdir*	Sets the root installation directory to <**`installdir`**>. All components are installed in this directory. **INSTALLDIR** is required if you want to install Universal Agent for SOA for Windows under a directory different from the one specified by the **PROGRAMFILES** environment variable (typically **`C:\Program Files\Universal`**). If the directory contains spaces, you must use double (**"**) quotation marks around the path name.	(none)
/q	Suppresses the product installation dialogs. Use this parameter in addition to the /s command line switch for a completely silent install. See Command Line Switches, Command Line Installation Examples, and Detecting the Completion of a Silent Install for additional information regarding silent installs.	n/a
/L*v	Instructs the installation process to create an installation log file named <**logfilepath**> (full path name). If <**logfilepath** > contains spaces, you must enclose it with double (") quotation marks. ***v** are flags used to specify the level of detail (**verbose**) contained in the log file. To reduce the amount of output generated, ***v** can be omitted. However, using these options is good practice; they can assist Stonebranch Customer Support with problem determination should any errors occur during installation.	n/a

Command Line Installation Examples

The following examples illustrate different ways that Universal Agent for SOA for Windows can be installed from the command line.

Graphical User Interface Install, All Components

To install all Universal Agent for SOA components via the graphical user interface, issue the following command:

```
sb-soa-6.2.0.x-windows-i386.exe
```

Silent Install

To silently install all Universal Agent for SOA components, issue the following command:

```
sb-soa-6.2.0.x-windows-i386.exe /s /v"/qn"
```

Graphical User Interface Install, All Components, with Log File

To install all Universal Agent for SOA components using the Windows Installer graphical user interface and write a log file to `C:\temp\install.log` during the installation, issue the following command:

```
sb-soa-6.2.0.x-windows-i386.exe /v"/l*v C:\Temp\install.log"
```

Silent Install, Different Installation Directory

To silently install all Universal Agent for SOA components to a non-default location, issue the following command:

```
sb-soa-6.2.0.x-windows-i386.exe /s /v"/qn INSTALLDIR=D:\Universal"
```

Detecting the Completion of a Silent Install

If the **/q** switch is used to perform a silent install, no graphical interface or user interaction is required. One drawback to this is that no feedback is provided indicating when the Windows Installer process (install, uninstall, or repair) finishes.

One method that can be used to detect when the Windows Installer process ends is to execute it using the system's **start** command. Using available command line switches, **start** can initiate the Windows Installer process and then wait for it to finish. When **start** returns control to its calling process (for example, the command prompt), the process will have ended.

For example, from the command prompt, issue the following command to start the Universal Agent for SOA installation and wait for it to finish:

```
start /b /wait sb-soa-6.2.0.x-windows-i386.exe /w /s /v"/qn"
```

- The **/b** switch prevents the start command from opening a new window.
- The **/wait** parameter causes the start command to start the application, sb-soa-6.2.0.x-windows-i386.exe, and then wait for it to finish.

You also can use the syntax above to execute the **start** command from within a script, such as a **.bat** file.

For more information on the **start** command, open a Windows command prompt and enter: **start /?**.

Modifying a SOA for Windows Installation via the Command Line

- Modifying a Universal Agent for SOA Installation via the Windows Installer Command Line Interface
- Repairing a Corrupted Universal Agent for SOA Installation
  - Silent Repair
  - Interactive Repair, with Log File
- Removing a Universal Agent for SOA Installation
  - Silent Removal

Modifying a Universal Agent for SOA Installation via the Windows Installer Command Line Interface

This page describes how to modify a Universal Agent for SOA installation via the Windows Installer command line interface.

After installing Universal Agent for SOA, you may rerun the installation program as needed in order to:

- Repair a Corrupted Universal Agent for SOA Installation.
- Remove a Universal Agent for SOA Installation.

(For a description of the parameters used in these procedures, see Windows Installer Command Line Parameters.)

Repairing a Corrupted Universal Agent for SOA Installation

To recover accidentally deleted application files, configuration and component definition files, or registry entries required by Universal Agent for SOA using the Windows Installer command line interface, use the **/f** switch together with the **om** parameters.

These are the same repair options set internally by the graphical interface installation. They cause the installation program to reinstall files that are missing or are older than the version contained in the Universal Agent for SOA distribution file.

Silent Repair

To repair a Universal Agent for SOA installation from the command line, issue the following command:

```
msiexec.exe /fom SetupPath\UPforSOA.msi /q
```

Interactive Repair, with Log File

To repair a Universal Agent for SOA installation interactively while writing a log file to **C:\Temp\repair.log**, issue the following command:

```
msiexec.exe /fom SetupPath\UPforSOA.msi /L C:\Temp\repair.log
```

Removing a Universal Agent for SOA Installation

To uninstall a Universal Agent for SOA installation using the Windows Installer command line interface, use the **/x** switch. Issue the following command:

Silent Removal

To uninstall a Universal Agent for SOA installation without using the Windows Installer command line interface, issue the following command:

```
msiexec.exe /x SetupPath\UPforSOA.msi /q
```

Universal Agent for SOA for Windows - Configuring and Starting UAC Server

Universal Agent for SOA for Windows - Configuring and Starting the UAC Server

After installing Universal Agent for SOA for Windows, perform the following steps before attempting to configure and start the Universal Application Container Server.

(For more information, see the UAC Server section in the Universal Command Agent for SOA 6.2.x Reference Guide.)

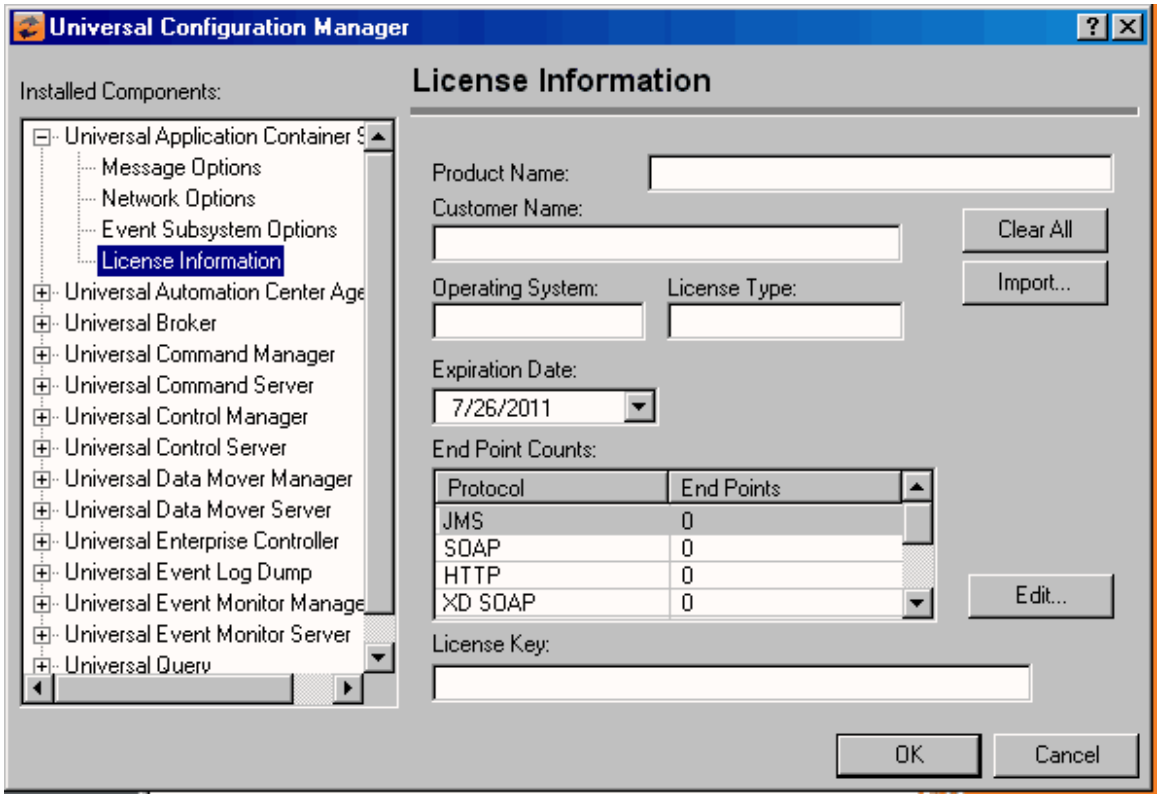
Step 1	Refresh the local Universal Broker's cached configuration. Starting with version 3.2.0.0, Universal Broker caches configuration and component definition information for all Universal Agent applications. Before UAC Server will run, its configuration and component definition information must reside in the local Broker's cache. To refresh the local Broker's cache without stopping and restarting the Broker, issue a REFRESH command via Universal Control (For more information, see the Universal Agent 6.2.x User Guide.)

Step 2 Add Universal Application Container (UAC) Server license information to the UAC Server configuration file.

Stonebranch, Inc. provides UAC Server license information in a formatted text file. This file contains the required license parameter keywords and values. You can simply append this license information to the UAC Server configuration file, **uacs.conf**, or import it using the Universal Configuration Manager.

To import license information using the Universal Configuration Manager.

1. Open the Windows Control Panel to start the Universal Configuration Manager.
2. In the Installed Components tree, click on the **+** symbol next to Universal Application Container Server to display its property pages.
3. Select License Information to display the License Information property page.



4. If you want to enter the license information manually on this page, make sure that you enter the information exactly as it appears in the license information file.
   - All information is case\-sensitive.
   - You must enter all punctuation and spacing as provided in the file.

   Instead of entering the information manually, you can import the license file by clicking the **Import** button, and then opening the file from the Select License File dialog.
5. After entering or importing the license information, click the **OK** button to save the information in the UAC Server configuration file.

Step 3 Recycle the local Universal Broker to start the UAC Server.

By default, the UAC Server is set to start automatically whenever its local Universal Broker starts. Stopping and restarting the Universal Broker will cause UAC Server to start.

Universal Agent for SOA for Windows - 64-Bit Windows Editions

Universal Agent for SOA for Windows - Installing on 64-bit Windows Editions

All Universal Agent for SOA components have been tested and verified on the 64-bit editions of the following Windows systems:

- Windows XP
- Windows Server 2003
- Windows Vista
- Windows Server 2008
- Windows 7
- Windows Server 2008 R2

The installation defaults for Universal Agent for SOA should not require any modification when installing on 64-bit Windows editions.

Universal Agent for SOA for Windows - File Inventory Lists

- Universal Agent for SOA for Windows - File Inventory Lists
- Universal Application Container Server
- Universal Application Container
- Universal Application Interface
- Sun Java Runtime Environment,1.5.0_14
- System Files

Universal Agent for SOA for Windows - File Inventory Lists

The Universal Agent for SOA 6.2.x for Windows package includes the files required for the following components / utilities:

- Universal Application Container Server
- Universal Application Container
- Universal Application Interface
- Sun Java Runtime Environment, 1.5.0_14

This page lists the files installed with each Universal Agent for SOA component. The file paths specified are relative to the root installation directory (for example, **C:\Program Files\Universal**) that was specified during the installation.

Universal Application Container Server

File	Description
UAC\startUAC.bat	Starts the Universal Application Container.
UAC\shutdownUAC.bat	Stops the Universal Application Container.
UAC\UAC.xsd	File used by Universal Application Container to validate content of the **UAC.xml** inbound configuration file. It requires no user interaction.
UAC\derby.log	Universal Application Container database log file.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uac_log4jConfiguration.xml	Universal Application Container logging configuration file that you may need to modify if you want to change the logging level of the UAC component.
UAC\INTEventLogAppender.dll	Universal Application Container Windows logging DLL.
UAC\uacValidateInbound.bat	Script that validates the contents of the **UAC.xml** file using the schema definition contained in **UAC.xsd**.
UAC\bin\uacsrv.exe	Universal Application Container Server program file.
UAC\bin\uacscfg.dll	Used by Universal Configuration Manager to manage Universal Application Container Server configuration options.
UAC\bin\uacscfg.hlp	Universal Application Container Server configuration help file.
UAC\lib\uacsrv.jar	Universal Application Container Server class file.
UBroker\tmpl\uaccfg	Universal Application Container Server configuration template file.
UBroker\tmpl\uaccmp	Universal Application Container Server component definition template file.
nls\uacmceng.umc	Universal Application Container Server message catalog.
%ALLUSERSPROFILE%\Application Data\Universal\conf\UAC.xml	Inbound operation configuration file that you may need to modify to support your inbound workload operations.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uacs.conf	Universal Application Container Server configuration file.

%ALLUSERSPROFILE%\Application Data\Universal\comp\uac	Universal Application Container Server component definition file. **%ALLUSERSPROFILE%\ Application Data**, by default, resolves to: • **`C:\Documents and Settings\All Users\Application Data`** on Windows 2003 and XP. • **`C:\ProgramData`** on Windows Vista / Windows 2008 Server and later.

Universal Application Container

File	Description
UAC\container	Sub-directory containing the libraries and other deployable objects needed for UAC operation.

Universal Application Interface

File	Description
UAI\lib	Sub-directory containing the **uacsrv.jar** file needed for communication between the **uacsrv** and **uac** components. It requires no user interaction.
UAI\.keystore	File that contains the certificates for secure communication between UAC and UAI.
%ALLUSERSPROFILE%\Application Data\Universal\conf\uai_log4jConfiguration.xml	Universal Application Interface logging configuration file.
UAI\uai.bat	Universal Application Interface start-up script.

Sun Java Runtime Environment,1.5.0_14

File	Description
jre*.*	Files required for redistribution of the 1.5.0_14 JRE.

System Files

The following files will be installed only if they are newer than the existing file.

The directories shown in this table are relative to the **%SYSTEMROOT%** directory, where **%SYSTEMROOT%** is an environment variable that resolves to **C:\Windows** on all Windows platforms.

File	Description
System32\asycfilt.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\comcat.dll	Version 4.71.1460.1 of the Microsoft Component Category Manager library.
Microsoft C-Runtime v8.0.50727.762 [1]	Version 8.0.50727.762 of the Microsoft C runtime side-by-side assembly.
Microsoft Foundation Classes v8.0.50727.762 [2]	Version 8.0.50727.762 of the Microsoft Foundation Class (MFC) side-by-side assembly.
System32\msiexec.exe	Version 3.1.4000.1823 of the Microsoft Windows Installer (see Windows Installer for more information).
System32\oleaut32.dll	Version 2.40.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\olepro32.dll	Version 5.0.4275.1. This DLL is one of the components of the Microsoft OLE library.
System32\psapi.dll	Version 4.0.1371.1 of the Microsoft process status library.
System32\stdole2.tlb	Version 2.40.4275.1. This file is one of the components of the Microsoft OLE library.

[1] The Microsoft C-Runtime distribution consists of several files, which are subject to change. Refer to Microsoft documentation for a complete list of files delivered with the specified runtime version.

[2] The Microsoft Foundation Classes (MFC) distribution consists of several files, which are subject to change. Refer to Microsoft documentation for

a complete list of files delivered with the specified MFC version.

Windows Installation - Licensing

- Licensing Universal Agent for Windows Components
- Product License File
  - Format
  - Sample
- Entering License Information
  - Licensing via Universal Configuration Manager
- Restart Universal Broker
  - Via Windows Services
  - Via Universal Configuration Manager

Licensing Universal Agent for Windows Components

After Universal Agent for Windows has been installed, you must configure the following components with product licenses before they can be used:

- Universal Command Manager
- Universal Data Mover Manager
- Universal Connector
- Universal Application Container Server

Note
Universal Application Container Server (as a component of Universal Command Agent for SOA 6.2.x for Windows) is packaged, and licensed, separately.

Product License File

For each component, product license information (license parameter keywords and their values) is contained in a separate text file provided by your Stonebranch, Inc. account representative.

Format

The format of the product license file name is: <*component name*>_<*customer name*>_<*operating system*>_<*schedule or solution*>.txt. For example: **Indesca_Stonebranch_NT_A1.txt**

- For Universal Command Manager, **Indesca** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.
- For Universal Data Mover Manager, **Infitran** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.

Sample

The following is a sample Universal Command Manager for Windows product license file:

```
License_Product "INDESCA"
License_Customer "STONEBRANCH"
License_OS_Type "NT"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

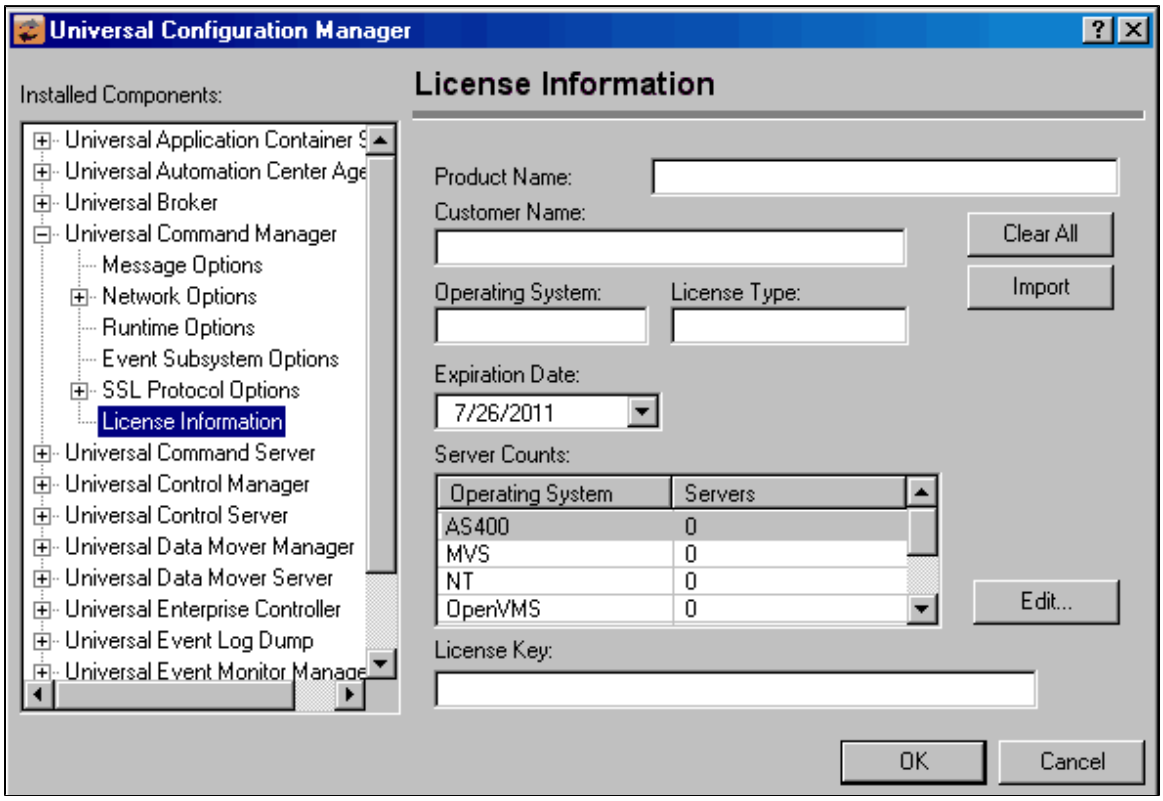
Universal Agent for Windows components can be configured with product licenses either by:

1. Entering the information from their product license file into their configuration file. It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration options.)
   - Universal Command Manager: **ucmd.conf**
   - Universal Data Mover Manager: **udm.conf**
   - Universal Connector: **usap.conf**
   - Universal Application Container Server: **uacs.conf**
2. Specifying the information on the Universal Configuration Manager License Information page for that component, either by.
   a. Entering the information specified in the license file.
   b. Importing the license file.

> **Note**
> Universal Configuration Manager is installed during the Universal Agent 6.2.x for Windows installation.

Licensing via Universal Configuration Manager

To enter license information via the Universal Configuration Manager, perform the following steps:

Step 1	On the Windows Control Panel, double-click the Universal Configuration Manager icon to display the Universal Configuration Manager screen.
Step 2	In the Installed Components tree, click **+** next to the component that you want to configure with license information.
Step 3	Click License Information to display the License Information page. 
Step 4	If you want to enter the license information on the page, make sure that you enter the information exactly as it is in the license information file. All information is case-sensitive; punctuation and spacing must be maintained.
Step 5	If you want to import the license file: 1. Click the Import button. 2. On the Select License File dialog that displays, select the license file to import. 3. Click the Open button. The information is imported to the License information page.
Step 6	Click the OK button to save the license information.

Restart Universal Broker

For Universal Broker to read the license information, you must start / restart it.

Via Windows Services

Step 1	Go to Windows Services.
Step 2	Locate and select Universal Broker.
Step 3	If Universal Broker is not running, click **Start**. If Universal Broker is running, click **Restart**.

Via Universal Configuration Manager

If you enter license information via the Universal Configuration Manager (see above), Universal Broker is restarted automatically, and the license information is read, when you click OK to save the license information.

UNIX Installation

- Introduction
- Installation Packages
- Universal Agent Installation Methods
  - System Installation
  - User Mode Installation
- Universal Agent Installation Summary
- Detailed Information

Introduction

These pages provide information on the installation of Stonebranch, Inc.'s Universal Agent on UNIX operating systems. Unless otherwise specified, all references to Universal Agent for UNIX refer to version 6.2.x.

Note
This information does not include installation on z/OS UNIX System Services (USS). See zOS USS Installation for z/OS USS installation instructions.

Installation Packages

Universal Agent is installed with one required installation package and one optional installation package:

Package	Distribution File	Required
Universal Agent for UNIX Installation	(See Universal Agent for UNIX - Distribution File.)	Yes
Universal Agent for SOA for UNIX Installation	(See Universal Agent for SOA for UNIX - Distribution File.)	No

Note
Starting with the 3.2.0 release of Universal Products, a Universal Broker must run on all systems on which a Universal Agent component is running, including manager components. The Broker maintains product configuration data for all components that have a configuration file.

Universal Agent Installation Methods

There are two installation methods supported on UNIX platforms for installing the Universal Agent package: system installation and user mode installation.

For detailed information on each method, see the platform-specific installation instructions:

- AIX installation
- HP-UX installation
- Solaris installation
- Linux installation

System Installation

The system installation method uses UNIX operating system vendor installation tools to install Universal Agent in the vendor-recommended installation directories.

This is the Stonebranch legacy method for installing UNIX Agents.

User Mode Installation

The Stonebranch install script, `unvinst`, offers a `-usermode_install` parameter that permits multiple Universal Agent installs on a single system. Additionally, a user mode install can create an environment in which an Agent executes as an unprivileged user (with significantly reduced functionality).

The user executing the install determines the environment in which the Agent executes. The root account may create the following environments:

- A fully-privileged system install to default locations.
- An "elevated" user mode install to custom locations. This is an install executed with the `-usermode_install` parameter, and permits user-specified install directories with some program execution under root authority.

When a non-root user executes the install script, the `-usermode_install` parameter is required. The result of an unprivileged install is one in which all application files are owned and execute as the (unprivileged) Broker user.

An unprivileged user mode environment will impose significant restrictions on Agent functionality, but may be desirable in test situations or for executing light workload.

The following requirements must be met in order to install and execute an Agent in an unprivileged user mode environment:

1. The Broker user must exist prior to the install. The account must have an existing home directory.
2. The account performing the install does not need to be root or the Broker user, but it must have authority to write to the Broker user's home directory.
3. A user mode install must be done by passing the `-usermode_install` parameter to unvinst.
4. The "security" configuration option must be set to "none" in `uags.conf`, `ucmds.conf`, `uctls.conf`, `udms.conf`, and `uems.conf`. This means that no user context switching is supported in an unprivileged user mode environment. All tasks execute as the Broker user.
5. Any task submitted via the Controller to execute on the Agent must not have credentials specified. This is necessary to ensure the process is spawned without requiring any kind of root privileges. One exception to this condition applies to any task that supports the "Run as sudo" option. In that case, credentials may be supplied, but must match an existing sudoers entry.
6. Any user that wants to execute Agent Managers (for example, ucmd, udm, or uem) or Utilities (for example, uctl or uquery) must source the `SB_ConfigSetup.env` file within their user profile. This file is installed in the `./etc` directory located under the install directory, and sets environment variables that point to configuration files and to the Broker's `.bif` and `.plf` files.

Universal Agent Installation Summary

Step 1	Download the distribution file from the Stonebranch Customer Portal.
Step 2	Copy the distribution file to the UNIX system.
Step 3	Logon to the UNIX system as root or execute **su** (substitute user) command to root.
Step 4	Uncompress the distribution file.
Step 5	Extract the files from the uncompressed **tar** file.
Step 6	Run the Stonebranch, Inc. installation script, **`unvinst`**.
Step 7	Run the Universal Broker daemon start-up script (located in **`./universal/ubroker`**) to start the Universal Broker: **`ubrokerd start`**.

Detailed Information

The following pages provide detailed information for UNIX Installation:

- Universal Agent for UNIX Installation
- Universal Agent for SOA for UNIX Installation
- Licensing

Universal Agent for UNIX Installation

Overview

The following information is provided for the installation of Universal Agent for UNIX:

- Installation Package
- Installation Requirements
- Distribution File
- Installation Procedures
- Customization
- File Inventory Lists

(For licensing information, see UNIX Installation - Licensing.)

Universal Agent for UNIX - Installation Package

- Components
- Component Compatibility
- Licensing

Components

The Universal Agent 6.2.x for UNIX package includes the following product components:

- Universal Broker 6.2.x
- Universal Automation Center Agent 6.2.x
- Universal Certificate 6.2.x
- Universal Command Manager and Server 6.2.x
- Universal Connector 6.2.x (for AIX, HP-UX, and Solaris)
- Universal Control Manager and Server 6.2.x
- Universal Data Mover Manager and Server 6.2.x
- Universal Encrypt 6.2.x
- Universal Event Monitor Manager and Server 6.2.x
- Universal Message to Exit Code Translator 6.2.x
- Universal Query 6.2.x
- Universal Message Service (OMS) 6.2.x
- Universal Controller Command Line Interface (CLI) 6.2.x

Component Compatibility

The following table identifies the compatibility of Universal Agent 6.2.x for UNIX components with previous component / product versions.

Component	Compatibility
Universal Broker 6.2.x	Universal Agent / Universal Products releases 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Command 6.2.x	Universal Command 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Control 6.2.x	Universal Control 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Data Mover 6.2.x	Universal Data Mover 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1 and 3.1.0.
Universal Encrypt 6.2.x	Universal Encrypt 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Query 6.2.x	Universal Broker 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, 2.2.0, and 2.1.0.
Universal Event Monitor 6.2.x	Universal Event Monitor 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, and 3.1.0.

The component references pertain to all supported UNIX platforms for that version.

Licensing

For licensing information, see UNIX Installation - Licensing.

Universal Agent for UNIX - Installation Requirements

- Installation Requirements
  - UNIX Versions
  - Additional Requirements
  - Platform Requirements
- Directories and Files
  - Command Reference Directory
  - Log Directory
  - Trace Directory
  - Spool Directory
  - UAG Cache

Installation Requirements

UNIX Versions

The following UNIX operating systems are supported:

- AIX 5.3 TL9 and above.
- HP-UX 11.23 and above (HP-UX IA64 package)
- Solaris 8 and above (SPARC-based).
- Solaris 10 and above (Intel-based).
- Linux Environments with the following qualifications:
  - 2.6 kernel
  - RedHat Package Manager (RPM).

Note
Universal Agent's native 64-bit package for x86_x64-based systems requires glibc 2.5-42 or later.

Note
In order to install Universal Agent components, you must be able to write to the directory from which the installation is launched.

Additional Requirements

- Superuser (root) access.
- TCP/IP Socket implementation.
- Approximately 400 megabytes of disk space for the installation. More disk space is required for variable files, such as log files, spool files, and trace files.
- Bourne shell or compatible.

Platform Requirements

Since platform requirements may change with new releases of a product, see Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x to make sure that your platform is supported before performing an installation.

Directories and Files

All product files that are written to during product execution are stored in the **/var/opt/universal** directory by default. This section documents the estimated amount of space required on the file system for all directories (and their sub-directories), required security access, and mount requirements.

Command Reference Directory

Universal Command Server can execute commands of type **cmdref**. A command reference is a predefined command or script to which the Universal Command Manager refers by its file name.

The command reference directory is **/var/opt/universal/cmdref**.

Space

The amount of space required is solely dependent on the number of command reference files you define. No command reference files are included in the installation.

Security

Universal Command Server requires read access to the **cmdref** directory. The product administrator requires read/write access in order to maintain the command reference files. No general user access is required.

Log Directory

Universal Broker can be configured to write its messages to a log file.

The current log file and previous log file generations are stored in the **/var/opt/universal/log** directory.

Space

A log file size grows to about 150,000 bytes and is then rolled over to a generation file. Five generations of log files are saved. The oldest generation log file is deleted. The amount of space required for the five generations and the current log file is about 900,000 bytes.

Security

Universal Broker requires read/write access to the log directory and read/write access to all files in the log directory. No other Universal Agent components use the log directory at this time. No general user access is required.

Trace Directory

Universal Broker and its server components (for example, Universal Command Server) create product trace files when configured to do so. A trace file is used by Stonebranch, Inc. Customer Support to resolve product problems.

Space

Trace files can grow to significant size depending on how long the trace is active and how much work the program is doing during the tracing period. A trace file size of about 10MB is not unusual.

Trace file sizes can be limited by setting the maximum number of lines to write to a file with the MAX_TRACE_LINES configuration option. Once the maximum is reached, the trace file will wrap to the beginning.

Under normal operation, no space is required for trace files. Trace files are requested by Stonebranch, Inc. Customer Support only for problem resolution. When trace files are required, at least 20MB of disk space should be available.

Security

Universal Broker and the Broker components (Universal Command Server and Universal Control Server) require read/write access to the trace directory. No other Universal Agent components access the trace directory. No general user access is required.

Spool Directory

The spool directory is used to store the following types of information:

- Execution information for Universal Agent components started by Universal Broker.
- Event definitions and event handlers managed by Universal Broker and used by Universal Event Monitor.
- Results of events tracked by Universal Event Monitor.
- Redirected standard I/O files (stdin, stdout, and stderr) captured by Universal Command when run with Manager Fault Tolerance enabled.

The default location for the spooled standard I/O files and other database files is the **/var/opt/universal/spool** directory.

Universal Command Server

Space

Spool files are located, by default, in directory **/var/opt/universal/spool**. The location of various product files can be can be configured via the product configuration files.

The amount of disk space required for the spool directory depends on:

1. Number of spooling user processes that will be executing simultaneously. A user process is created for each command requested by a Universal Command Manager. The default maximum number is 50.

When a user process ends and a Manager has received all the spool files, the spool files themselves are deleted.
2. Average size of the user processes standard input, output, and error files.

A Stonebranch Tip
Keep in mind that spooling is not a feature for file transfer purposes.

File transfer-related processes should execute without spooling enabled.

When these numbers have been determined, the average amount of disk space is calculated with the following formula:

`MAX-PROCESSES x AVERAGE-STDIO-SIZE x 2 = required disk space.`

For example:

If the maximum number of simultaneous user processes is estimated at 20 and the average size of processes standard I/O files is 100,000 bytes, the average amount of required disk space is 4MB (20 x 100000 x 2).

The Universal Command Server is configured with spooling disabled to prevent unintentional disk utilization. The feature must be turned on through the ALLOW_SPOOLING configuration settings.

For more information on the Manager Fault Tolerant feature and the spooling of redirected standard I/O files, see the Universal Agent 6.2.x User Guide.

Security

Universal Broker, Universal Command Server and Universal Event Monitor require read/write access to the spool directory. No other Universal Agent components access the spool directory. No general user access is required.

Mount

The spool directory must be mounted on a local device. It cannot be mounted on a network device, such as an NFS or SAMBA mount.

UAG Cache

UAG cache is used by Universal Automation Center Agent (UAG) for storing standard I/O files.

Space

Cache files are located, by default, in directory **`/var/opt/universal/uag/cache`**.

Cache files are created for each job that is run by the agent. They remain in the cache until they are purged by an automated purge process scheduled nightly by Universal Controller. You can configure the number of days that files remain in the cache via the Universal Controller user interface (see the Agent Cache Retention Period in Days Universal Controller system property).

The amount of disk space required for the cache directory depends on:

1. Number of jobs you estimate will run during the cache retention period you specified.
2. These files remain until they are purged by the automated cache purge process scheduled by the Automation Center Core daily at midnight.
3. Average size of the user processes standard output and error files.

When these numbers have been determined, the average amount of disk space is calculated with the following formula:

`(RETENTION PERIOD x MAX-JOBS) x (AVERAGE-STDOUT-SIZE + AVERAGE-STDERR-SIZE) = required disk space.`

For example:

If the files are purged every 7 days, and you run 1200 jobs on that agent server daily, and the average size of the STDOUT + STDERR files is 3,000 bytes, the average amount of required disk space is 25MB (7 x 1,200 x 3000).

The Universal Automation Center Agent Server automatically redirects the standard output and standard error files to the cache directory with no required user input.

Security

Universal Automation Center Agent require read/write access to the UAG cache directory. No other Universal Agent components access the cache directory. No general user access is required.

Universal Agent for UNIX - Distribution File

- UNIX Distribution File
- Obtaining the Distribution File
- Distribution File Format

UNIX Distribution File

Stonebranch, Inc. provides different Universal Agent for UNIX packages for different types of UNIX operating systems.

Obtaining the Distribution File

To obtain the Universal Agent for UNIX package for your type of UNIX operating system, you must download the corresponding product distribution file from the Stonebranch Customer Portal.

> **Note**
> A customer user ID and password — provided by Stonebranch, Inc. — are required to access this area.

After a distribution file has been downloaded, the installation files contained in that file must be extracted before the product can be installed (see Universal Agent for UNIX - Installation Procedures).

Distribution File Format

The name of each Universal Agent for UNIX distribution file has the following format:

`sb-Version.Release.Modification Level.Maintenance Level-operating system-version(.release)(-platform).tar.Z`

For example: **`sb-6.2.0.0-linux-2.6-ia64.tar.Z`**

In this format:

- **Version** is the current version of Universal Agent.
- **Release** is the current release of Universal Agent.
- **Modification Level** is the current Universal Agent feature set.
- **Maintenance Level** is the Universal Agent build level.
- **operating system** is the name of the operating system (for example, AIX or Linux).
- **version(.release)** is the supported version and, optionally, the release of the operating system.
- **platform** is the targeted hardware platform (for example, RS6000 or i386). It is included in the file name only if there is more than one platform available for the specified operating system.

Universal Agent for UNIX - Installation Procedures

Universal Agent for UNIX Installation Procedures

The following procedures are provided for the installation and modification of Universal Agent for UNIX:

- Universal Agent for AIX Installation
- Universal Agent for HP-UX Installation
- Universal Agent for Solaris Installation
- Universal Agent for Linux Installation

Universal Agent for AIX Installation

- Installation Process
- Extracting the Universal Agent for AIX Installation Files
  - Distribution File
- AIX Packages
- Installing Universal Agent for AIX
  - Component Selection
  - Starting the Installation Script
  - AIX Installation Script Parameters
  - Installation Script Example
  - User Mode Installation
  - AIX PAM Customization
- Listing Universal Agent for AIX Information
- Removing Universal Agent for AIX
  - System Install Removal
  - User Mode Install Removal

Installation Process

Installation of Universal Agent for AIX is a three-step process:

Step 1	Download the product distribution file.
Step 2	Extract the installation files from the distribution file.
Step 3	Install the extracted files.

Extracting the Universal Agent for AIX Installation Files

The Universal Agent for AIX product distribution file is in a compressed **tar** format.

To uncompress and extract the installation files from the distribution file, issue the following command:

```
zcat sb-6.2.x.x-aix-5.3.tar.Z | tar xvf -
```

This command assumes the following:

- Name of the distribution file is **`sb-6.2.x.x-aix-5.3.tar.Z`**.
- File is located in the current working directory.

Distribution File

The distribution file contains multiple files, including packages in the AIX backup file format, using extension **.bff**. The actual base name of the **.bff** file depends on the AIX version for which the distribution file is intended. (See Distribution File Format for distribution file naming conventions.)

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary of the installation procedure.
oms-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base OMS product.
opscli-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base Universal Controller CLI product.
uag-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UAG product.
ubr-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UBROKER product.
ucmd-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UCMD product.
ucom-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base COMMON package.
ucr-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UCERT package.

udm-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UDM product.
uem-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UEM product.
usap-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base USAP product.
utl-6.2.x.x-aix-5.3.bff	AIX backup file format package for the base UTILITIES package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

Note
If your Universal Agent for AIX distribution file does not contain all of these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

AIX Packages

The following table identifies the filesets contained in the AIX packages.

Package	Fileset Name	Description
UNVcom	**UNVcom.UNVcom.msg.en_US**	Universal Agent common files, such as message catalogs and translation tables.
UNVuag	**UNVuag.UNVuag.rte**	Universal Automation Center Agent
UNVubr	**UNVubr.UNVctl.rte** **UNVubr.UNVcts.rte** **UNVubr.UNVqry.rte** **UNVubr.UNVspl.rte** **UNVubr.UNVubr.rte**	Universal Control Manager Universal Control Server Universal Query Manager Universal Spool Maintenance Utilities Universal Broker
UNVucm	**UNVucm.UNVucm.rte** **UNVucm.UNVucs.rte**	Universal Command Manager Universal Command Server
UNVucr	**UNVucr.UNVucr.rte**	Universal Certificate
UNVudm	**UNVudm.UNVdmgr.rte** **UNVudm.UNVdsrv.rte**	Universal Data Mover Manager Universal Data Mover Server
UNVuem	**UNVuem.UNVuem.rte** **UNVuem.UNVues.rte**	Universal Event Monitor Manager Universal Event Monitor Server
UNVusp	**UNVusp.UNVusp.rte**	Universal Connector
UNVutl	**UNVutl.UNVutil.rte**	Universal Utilities
UNVoms	**UNVoms.UNVoms.rte**	Universal Message Service (OMS)
UNVopscli	**UNVopscli.UNVopscli.rte**	Universal Controller Command Line Interface (CLI)

Installing Universal Agent for AIX

Universal Agent for AIX is installed with the **`unvinst`** script, which executes the **installp** command. The command to start the script must be executed with the superuser ID.

Note
Stonebranch, Inc. strongly recommends the use of the **`unvinst`** script for the AIX installation above any other method.

Component Selection

The Universal Agent package contains many components, which are grouped into four categories. Components in some categories are installed

and activated optionally, as specified in the **unvinst** script by using **unvinst** command line parameters.

The following table describes each category, provides the default installation configuration for the components in that category, and identifies the command line parameters to use for components that are optionally installed and activated.

> **Note**
> The default installation configuration refers to new installs only. For upgrades, installed component configurations are not changed by the upgrade process.

Category	Default Configuration	Description
Base components	Always installed.	Base components are always installed and activated. They include the Universal Broker, Universal Command (UCM), Universal Data Mover (UDM), and others. Base components provide the core agent infrastructure and workload services.
Universal Automation Center Agent (UAG)	Always installed, but inactive.	Universal Controller utilizes UAG agents to provided distributed, workload automation services. UAG is activated using either the --oms_servers (recommended) or --ac_transports parameter.
Universal Message Service (OMS)	Always installed, but inactive.	OMS is message-oriented middleware that should be deployed on a small set of centrally located servers. It should not be activated on every Agent install. OMS is the network provider for UAG and the Universal Controller. OMS is activated using the --oms_autostart parameter.
Universal Controller command line programs	Not installed.	Universal Controller command line programs provide a command line interface to the Universal Controller. The installation of Universal Controller command line programs is optional. The command line programs are only required on Agents that need to interface with the Universal Controller via the command line. Universal Controller command line programs are installed using the --opscli parameter.

Starting the Installation Script

To start the **unvinst** installation script and install all of the AIX packages, issue the following command:

```
sh ./unvinst [--user username [--userdir directory] [--create_user {yes|no}] ] [--group group name
[--create_group {yes|no}] ]
            [--convert_opsagent [--opsdir directory] ] [--network_provider {oms|transport}]
            [--oms_servers network address] [--ac_transports ports/addresses] [--ac_core name]
[--oms_port port]
            [--oms_autostart {yes|no}] [--ac_netname ID] [--ac_enable_ssl {yes|no}] [--opscli {yes|no
}]
            [--usermode_install {yes|no} [--unvdir directory] [--unvcfgdir directory] [--unvdatadir
directory] [--unvport port] ]
```

See AIX Installation Script Parameters and Installation Script Example, below, for a description of the optional parameters that you can issue with **unvinst** and an example of **unvinst** with these parameters.

> **Note**
> If you want to install multiple Agents on the same machine, or add one or more additional Agents to a machine with a previously installed Agent, some of these installation parameters are required (see Installing Multiple Agents on a Single Machine).

This is a silent install. The output from **unvinst** is written to file **install.log** in the current directory.

An entry is added to the system initialization table, **/etc/inittab**, to start the **ubrokerd** daemon when the system boots to runlevel 2. The **inittab** entry is similar to the following:

```
ubroker:2:once:/opt/universal/ubroker-6.2.0/ubrokerd start
```

The Universal Broker daemon will be installed and run as the **username** and **groupname** specified with the installation script parameters, below.

> **Note**
> With the Solaris, HP, and AIX installs, the previous installation directories are removed when the native installer has detected that additional directories or files have not been added to the original installation directories. If they have been modified, the directories will remain and can be reviewed and removed, as desired, by your Administrator.

AIX Installation Script Parameters

The following table describes the optional parameters that are available in the installation script, **`unvinst`**, when installing Universal Agent.

The parameters are grouped into the following categories:

Category	Description
Base	Used for base install.
OMS	Used for an active OMS configuration.
UAG	Used for an active UAG configuration.
CLI	Used for Universal Controller CLI programs install.
User Mode	Used for user mode installation.

Parameter	Description	Default
Base Parameters		
-u **user** **--user**	Normal UNIX **username** that is used to execute the Universal Broker daemon. The install grants this user account ownership of all installed files, with the exception of the Universal Agent server components (for example: **ucmsrv**, **udmsrv**, and **uemsrv**) which, due to security requirements, are owned by root and will have their "set user ID on execution" bit set. • If the user account that you want to use already exists, specify that user account. • If the user account does not exist, the install creates it. • If you want to change the user account for an installed Universal Agent for AIX system, you must perform a re-installation and use this parameter to change the user account. • If **--user** is omitted from **`unvinst`**, the default is used. • If --usermode_install is **yes**, there is no default.	**ubroker**
create_user **create_user**	Specification (**yes** or **no**) for whether or not to create the user name that will own the installed files as a local user.	**yes**
userdir **userdir**	Home directory for the created user account specified by --user. • If this directory does not exist, it is created when the specified user is created. • If the user specified by --user already exists, but the home directory of that user is not the default directory (**`/home/<username>`**), **`--userdir`** must specify the path to that home directory. • If **--userdir** is omitted from **`unvinst`**, the default is used.	**`/home/<username>`**
-g **group** **group**	Normal UNIX **groupname**; the Universal Broker daemon will run as this specified group. All installed files will be assigned to this group. • If the group that you want to use already exists, specify that group. • If this group does not exist, the install creates it. • If **--group** is omitted from **`unvinst`**, the default is used.	**ubroker**
create_group **create_group**	Specification (**yes** or **no**) for whether or not to create the group that will own the installed files as a local group.	**yes**
UAG Parameters		

-c **convert** **convert_opsagent**	Converts an existing Opswise Automation Center Agent (1.5, 1.6, or 1.7) to a 5.2.0 Agent. The installation process will invoke a script, **`opsmerge.sh`**, which stops the Agent and converts the configuration options stored in its **`agent.props`** file to corresponding configuration options in the Universal Automation Center Agent (UAG) **`uags.conf`** file. The script also performs several other tasks needed for the conversion. **Note** You must include this parameter when upgrading an Opswise Automation Center Agent (1.5, 1.6, or 1.7) to Universal Agent 6.2.x.x.	
-d **opsdir** **opsdir**	If --convert_opsagent is specified: Identifies the primary install directory for Universal Agent. • If **--opsdir** is omitted from **`unvinst`**, the default is used.	**`/home/opswise`**
-network_provider **--network_provider**	Specifies the network communications provider that will be used between the Agent and the Controller. Valid values are: • **oms** The Agent connects to the Controller using the Universal Message Service (OMS). Value(s) for --oms_servers must also be specified. • **transport** The Agent connects to the Controller using the Opswise Transporter and Message Hub. Values for --ac_transports and --ac_core must also be specified. **Note** It is recommended that you always include **--network_provider** in **`unvinst`**. If you omit **--network_provider**, the Agent may not know which network communications provider to use and will not start up. **UAG configuration:** The value specified by this parameter sets the value of the UAG NETWORK_PROVIDER configuration option.	(none)
oms_servers **oms_servers**	If --network_provider is **oms**: Specifies a value, in the format **`port@host[,port2@host,...,portn@hostn]`**, for the port and network address of the OMS server(s) to be used as network communications providers. **Note** You should always include **--oms_servers** in **`unvinst`**; OMS should be used as the network communications provider between Controller 6.2.x.x and Agent 6.2.x.x. **UAG configuration:** The value specified for **--oms_servers** is set automatically for the UAG OMS_SERVERS configuration option.	(none)
-t **transports** **ac_transports**	If --network_provider is **transport**: Specifies a value, in the format **`port@ipaddr[,port2@ipaddr2,...,portn@ipaddrn]`**, for the port and network address of the Universal Automation Center Transporter(s) used for network communications. If --convert_opsagent is specified, the value specified in **--ac_transports** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_transports** is set automatically for the UAG AUTOMATION_CENTER_TRANSPORTS configuration option.	(none)

-r **core** **ac_core**	If --network_provider is **transport**: Specifies a value for the queue name of the Universal Automation Center Message Hub. If --convert_opsagent is specified*, the value specified in **--ac_core** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG AUTOMATION_CENTER_CORE configuration option.	(none)
ac_netname **ac_netname**	Specifies the network ID that Universal Agent will use. **UAG configuration:** The value specified by **--ac_netname** is set automatically for the UAG NETNAME configuration option.	(none)
ac_enable_ssl **ac_enable_ssl**	Specifies (**yes** or **no**) whether or not the SSL protocol is used for network communications between the Agent and the specified network communications provider. **UAG configuration:** The value specified by **--ac_enable_ssl** is set automatically for the UAG ENABLE_SSL configuration option.	**no**
OMS Parameters		
oms_port **oms_port**	Specifies the port to use to listen for OMS connection requests. **OMS configuration:** The value specified by **--oms_port** is set automatically for the OMS SERVICE_PORT configuration option.	(none)
oms_autostart **oms_autostart**	Specifies (**yes** or **no**) whether or not OMS is started automatically by Universal Broker when Universal Broker starts. **OMS component definition:** The value specified by **--oms_autostart** is set automatically for the OMS AUTOMATICALLY_START and RESTART component definition options.	**no**
CLI Parameters		
opscli **opscli**	Specifies (**yes** or **no**) whether or not the Universal Controller Command Line Interface (CLI) tools will be installed.	**no**
User Mode Parameters		
-**U** **-usermode_install** **--usermode_install**	Specifies (**yes** or **no**) for a user mode installation, which defines both of the following: • Whether or not this Agent can be installed on a machine where an Agent already is installed (see Installing Multiple Agents on a Single Machine). • Whether or not the user can specify installation parameters (--unvdir, --unvcfgdir, --unvdatadir, and --unvport) that control where the Agent will be installed and configured (see Changing the Default Installation Directories).	**no**
-unvdir **--unvdir**	If --usermode_install is set to **yes**: Specifies the Agent binaries (installation) directory.	(none)
unvcfgdir **unvcfgdir**	If --usermode_install is set to **yes**: Specifies the Agent configuration files directory.	**`<--unvdir>/etc`**
unvdatadir **unvdatadir**	If --usermode_install is set to **yes**: Specifies the Agent data files directory.	**`<--unvdir>/var`**
unvport **unvport**	If --usermode_install is set to **yes**: Specifies the Universal Broker port.	(none)
Additional Parameter		
-? **-h** **help** **info**	Displays command line help.	n/a

Installation Script Example

The following example illustrates Universal Agent for AIX installed with the installation script, **`unvinst`**, and optional parameters.

```
sh ./unvinst --user user1 --userdir /homedir/user --group usergroup
             --convert_opsagent --opsdir /homedir/ops --network_provider oms
             --oms_servers 7878@oms1,7979@oms2 --oms_port 7878 --oms_autostart yes
             --ac_netname OPSAUTOCONF --ac_enable_ssl yes --opscli yes
             --usermode_install yes --unvdir /opt/universal --unvcfgdir /etc/universal --unvdatadir
/var/opt/universal --unvport 7887
```

User Mode Installation

A user mode installation, implemented through use of the usermode_install installation parameter, lets you install multiple Agents on a single machine and change the default installation directories for any Agent being installed.

You must perform a user mode installation for installing an Agent on a machine where one or more Agents already have been installed.

However, you also can perform a user mode installation for the initial installation of an Agent on a machine.

Note
You can execute the install as a non-root user if you want to execute the Agent in an unprivileged user mode environment. See User Mode Installation for specific requirements and restrictions associated with an unprivileged user mode environment.

Installing Multiple Agents on a Single Machine

If you want to install multiple Agents on the same machine, you must set the following installation parameter values for each Agent being installed in addition to the initially installed Agent.

Note
You also can set these parameters values for the initial installation of an Agent on a machine.

Installation Parameter	Value
--usermode_install	**yes**
--user	A username that is different than the username specified for any other Agent installation on this machine.
--unvdir	An Agent binaries (installation) directory that is different than the installation directory specified for any other Agent installation on this machine.
--unvport	A Universal Broker port that is different than the port specified for any other Agent installation on this machine.

Changing the Default Installation Directories

By default, an Agent is installed and configured in default directories.

If you want to change these default directories, or if you want to install multiple Agents on the same machine, you must set the --usermode_install parameter to **yes** and specify new values in the following parameters. These directories must be different for each Agent on the same machine.

Note
You also can change these directories for the initial installation of an Agent on a machine.

Installation Parameter	Default Directory	Files Installed
--unvdir	**/opt/universal**	Agent binaries
--unvcfgdir	**/etc/universal**	Agent configuration files

--unvdatadir	**/var/opt/universal**	Agent data files

AIX PAM Customization

If security is set to PAM, the **pam.conf** file under **/etc** must be modified.

Below is the minimum required PAM service definition to make Universal Agent function:

ucmd	auth	required	/usr/lib/security/pam_aix
ucmd	account	required	/usr/lib/security/pam_aix

Listing Universal Agent for AIX Information

On AIX, information on an installed product is listed with the **lslpp** command. The command must be executed with the superuser ID.

Issue the following command to list information for Universal Agent for AIX:

```
lslpp -La 'UNV*'
```

Removing Universal Agent for AIX

System Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **`/etc/universal`** and **`/var/opt/universal`** directories.

Step 3	Using the superuser ID, remove all filesets of Universal Agent for AIX by issuing the following commands:
	`installp -u UNVopscli UNVoms UNVusp UNVuag UNVucr UNVutl UNVuem UNVudm UNVucm UNVubr UNVcom`
	All entries in the **`/etc/inittab`** file that reference **`ubroker`** are removed.
	`rm -rf /etc/universal`
	`rm -rf /var/opt/universal`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `rmuser ubroker`
Step 5	Delete the Agent group (`ubroker`): `rmgroup ubroker`

User Mode Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **<--unvcfgdir>** and **<--unvdatadir>** directories.

Step 3	Using the superuser ID, remove all filesets of Universal Agent for AIX by issuing the following commands:
	`installp -u UNVopscli UNVoms UNVusp UNVuag UNVucr UNVutl UNVuem UNVudm UNVucm UNVubr UNVcom`
	All entries in the **/etc/inittab** file that reference **ubroker** are removed.
	`rm -rf <--unvcfgdir>`
	`rm -rf <--unvdatadir>`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `rmuser ubroker`
Step 5	Delete the Agent group (`ubroker`): `rmgroup ubroker`

Agent
To remove an Agent executing in an unprivileged user mode environment (see User Mode Installation), simply stop the `ubrokerd` daemon and remove the `./universal` installation directory. To make sure that you do not mistakenly remove a system install directory, attempt the removal with a non-privileged user account or the Broker account.

Universal Agent for HP-UX Installation

- Installation Process
- Extracting Universal Agent for HP-UX Installation Files
- Distribution File
- HP-UX Package
- Installing Universal Agent for HP-UX
  - Component Selection
  - Starting the Installation Script
  - HP-UX Installation Script Parameters
  - Installation Script Example
  - User Mode Installation
  - HP-UX PAM Customization
- Listing Universal Agent for HP-UX Information
- Removing Universal Agent for HP-UX
  - System Install Removal
  - User Mode Install Removal

Installation Process

Installation of Universal Agent for HP-UX is a three-step process:

Step 1	Download product distribution file (see Universal Agent for UNIX - Distribution File).
Step 2	Extract the installation files from the distribution file.
Step 3	Install the extracted files.

Extracting Universal Agent for HP-UX Installation Files

Stonebranch, Inc. provides a product distribution file for the following HP-UX systems:

- IA64 Systems

The Universal Agent for HP-UX product distribution file is in a compressed **tar** format.

To uncompress and extract the installation files from the distribution file, issue the following command:

```
zcat sb-6.2.x.x-hpux-11.23-ia64.tar.Z | tar xvf -
```

This command assumes the following:

- Name of the distribution file is **`sb-6.2.x.x-hpux-11.23-ia64.tar.Z`**.
- File is located in the current working directory.

Distribution File

The distribution file contains multiple files, including a package in the HP-UX file format (extension **.depot**). The actual base name of the **.depot** file depends on the HP-UX version for which the distribution file is intended. (See Distribution File Format for distribution file naming conventions.)

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary of the installation procedure.
unv-6.2.x.x-hpux-11.23-ia64.depot	HP-UX .depot file-format package.
unv-opscli-6.2.x.x-hpux-11.23-ia64.depot	Universal Controller Command Line Interface (CLI) .depot file-format package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.

unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

If your Universal Agent for HP-UX distribution file does not contain these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

HP-UX Package

The Universal Agent for HP-UX is packaged as a depot file (extension **.depot**). The Universal Agent depot contains product UCM.

The following table identifies the sub-products contained in the HP-UX package.

Subproduct Name	Description
UNV.UNVcom	Universal Agent common files, such as message catalogs and translation tables.
UNV.UNVctl	Universal Control Manager
UNV.UNVcts	Universal Control Server
UNV.UNVqry	Universal Query
UNV.UNVspl	Universal Spool
UNV.UNVuag	Universal Automation Center Agent
UNV.UNVubr	Universal Broker
UNV.UNVucm	Universal Command Manager
UNV.UNVucr	Universal Certificate
UNV.UNVucs	Universal Command Server
UNV.UNVudm	Universal Data Mover Manager
UNV.UNVuds	Universal Data Mover Server
UNV.UNVuem	Universal Event Monitor Manager
UNV.UNVues	Universal Event Monitor Server
UNV.UNVusp	Universal Connector
UNV.UNVutl	Universal Utilities
UNV.UNVoms	Universal Message Service (OMS)
UNV.UNVopscli	Universal Controller Command Line Interface (CLI)

Installing Universal Agent for HP-UX

Universal Agent for HP-UX is installed with the **unvinst** script, which executes the **swinstall** command. The command to start the script must be executed with the superuser ID.

> **Note**
> Stonebranch, Inc. strongly recommends the use of the **unvinst** script for the HP-UX installation above any other method.

Component Selection

The Universal Agent package contains many components, which are grouped into four categories. Components in some categories are installed and activated optionally, as specified in the **`unvinst`** script by using **`unvinst`** command line parameters.

The following table describes each category, provides the default installation configuration for the components in that category, and identifies the command line parameters to use for components that are optionally installed and activated.

Note
The default installation configuration refers to new installs only. For upgrades, installed component configurations are not changed by the upgrade process.

Category	Default Configuration	Description
Base components	Always installed.	Base components are always installed and activated. They include the Universal Broker, Universal Command (UCM), Universal Data Mover (UDM), and others. Base components provide the core agent infrastructure and workload services.
Universal Automation Center Agent (UAG)	Always installed, but inactive.	Universal Controller utilizes UAG agents to provided distributed, workload automation services. UAG is activated using either the --oms_servers (recommended) or --ac_transports parameter.
Universal Message Service (OMS)	Always installed, but inactive.	OMS is message-oriented middleware that should be deployed on a small set of centrally located servers. It should not be activated on every Agent install. OMS is the network provider for UAG and the Universal Controller. OMS is activated using the --oms_autostart parameter.
Universal Controller command line programs	Not installed.	Universal Controller command line programs provide a command line interface to the Universal Controller. The installation of Universal Controller command line programs is optional. The command line programs are only required on Agents that need to interface with the Universal Controller via the command line. Universal Controller command line programs are installed using the --opscli parameter.

Starting the Installation Script

To start the installation script, **unvinst**, and install all of the HP-UX packages, issue the following command:

```
sh ./unvinst [--user username [--userdir directory] [--create_user {yes|no}] ] [--group group name
[--create_group {yes|no}] ]
            [--convert_opsagent [--opsdir directory] ] [--network_provider {oms|transport}]
            [--oms_servers network address] [--ac_transports ports/addresses] [--ac_core name]
[--oms_port port]
            [--oms_autostart {yes|no}] [--ac_netname ID] [--ac_enable_ssl {yes|no}] [--opscli {yes|no
}]
            [--usermode_install {yes|no} [--unvdir directory] [--unvcfgdir directory] [--unvdatadir
directory] [--unvport port] ]
```

See HP-UX Install Script Parameters and Installation Script Example, below, for a description of the optional parameters that you can issue with **unvinst** and an example of **unvinst** with these parameters.

Note
If you want to install multiple Agents on the same machine, or add one or more additional Agents to a machine with a previously installed Agent, some of these installation parameters are required (see Installing Multiple Agents on a Single Machine).

This is a silent install. The output from **unvinst** is written to file **install.log** in the current directory.

System initialization files **/sbin/init.d/ubrokerd** and **/sbin/rc3.d/S850ubrokerd** are created to start the **ubrokerd** daemon when the system boots to runlevel 3.

The Universal Broker daemon will be installed and run as the **username** and **groupname** specified with the installation script parameters, below.

Note
With the Solaris, HP-UX, and AIX installs, the previous installation directories are removed when the native installer has detected that additional directories or files have not been added to the original installation directories. If they have been modified, the directories will remain and can be reviewed and removed, as desired, by your Administrator.

HP-UX Installation Script Parameters

The following table describes the optional parameters that are available in the installation script, **unvinst**, when installing Universal Agent.

The parameters are grouped into the following categories:

Category	Description
Base	Used for base install.
OMS	Used for an active OMS configuration.
UAG	Used for an active UAG configuration.
CLI	Used for Universal Controller CLI programs install.
User Mode	Used for user mode installation.

Parameter	Description	Default
Base Parameters		
-u **user** **--user**	Normal UNIX **username** that is used to execute the Universal Broker daemon. The install grants this user account ownership of all installed files, with the exception of the Universal Agent server components (for example: **ucmsrv**, **udmsrv**, and **uemsrv**) which, due to security requirements, are owned by root and will have their "set user ID on execution" bit set. • If the user account that you want to use already exists, specify that user account. • If the user account does not exist, the install creates it. • If you want to change the user account for an installed Universal Agent for HP-UX system, you must perform a re-installation and use this parameter to change the user account. • If **--user** is omitted from **`unvinst`**, the default is used. • If --usermode_install is **yes**, there is no default.	**ubroker**
create_user **create_user**	Specification (**yes** or **no**) for whether or not to create the user name that will own the installed files as a local user.	**yes**
userdir **userdir**	Home directory for the created user account specified by --user. • If this directory does not exist, it is created when the specified user is created. • If the user specified by --user already exists, but the home directory of that user is not the default directory (**`/home/<username>`**), **`--userdir`** must specify the path to that home directory. • If **--userdir** is omitted from **`unvinst`**, the default is used.	**`/home/<username>`**
-g **group** **group**	Normal UNIX **groupname**; the Universal Broker daemon will run as this specified group. All installed files will be assigned to this group. • If the group that you want to use already exists, specify that group. • If this group does not exist, the install creates it. • If **--group** is omitted from **`unvinst`**, the default is used.	**ubroker**
create_group **create_group**	Specification (**yes** or **no**) for whether or not to create the group that will own the installed files as a local group.	**yes**
UAG Parameters		

-c **convert** **convert_opsagent**	Causes **unvinst** to execute **opsmerge.sh** (residing in **/opt/universal/uagsrv/bin**), which performs the following tasks: 1. Searches for an existing Opswise Automation Center Agent 1.6 or 1.7 install and converts properties stored in the **agent.props** file to corresponding configuration options in the Universal Automation Center Agent (UAG) configuration file, **uags.conf**. 2. Searches for an active Opswise Automation Center Agent 1.6 or 1.7 daemon process and attempts to stop it. 3. Assigns the ID used by the Opswise Automation Center Agent 1.6 or 1.7 to UAG by moving the **qname** file from the Opswise Automaton Center install directory to the **/var/opt/universal/uag/var** directory.	
-d **opsdir** **opsdir**	If --convert_opsagent is specified: Identifies the primary install directory for Universal Agent. • If **--opsdir** is omitted from **unvinst**, the default is used.	**/home/opswise**
-network_provider **--network_provider**	Specifies the network communications provider that will be used between the Agent and the Controller. Valid values are: • **oms** The Agent connects to the Controller using the Universal Message Service (OMS). Value(s) for --oms_servers must also be specified. • **transport** The Agent connects to the Controller using the Opswise Transporter and Message Hub. Values for --ac_transports and --ac_core must also be specified. **Note** It is recommended that you always include **--network_provider** in **unvinst**. If you omit **--network_provider**, the Agent may not know which network communications provider to use and will not start up. **UAG configuration:** The value specified by this parameter sets the value of the UAG NETWORK_PROVIDER configuration option.	(none)
oms_servers **oms_servers**	If --network_provider is **oms**: Specifies a value, in the format **port@host[,port2@host,...,portn@hostn]**, for the port and network address of the OMS server(s) to be used as network communications providers. **Note** You should always include **--oms_servers** in **unvinst**; OMS should be used as the network communications provider between Controller 6.2.x.x and Agent 6.2.x.x. **UAG configuration:** The value specified for **--oms_servers** is set automatically for the UAG OMS_SERVERS configuration option.	(none)
-t **transports** **ac_transports**	If --network_provider is **transport**: Specifies a value, in the format **port@ipaddr[,port2@ipaddr2,...,portn@ipaddrn]**, for the port and network address of the Universal Automation Center Transporter(s) used for network communications. If --convert_opsagent is specified, the value specified in **--ac_transports** will override any currently configured options, including those imported from **agent.props**. **UAG configuration:** The value specified by **--ac_transports** is set automatically for the UAG AUTOMATION_CENTER_TRANSPORTS configuration option.	(none)

-r **core** **ac_core**	If --network_provider is **transport**: Specifies a value for the queue name of the Universal Automation Center Message Hub. If --convert_opsagent is specified, the value specified in **--ac_core** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG AUTOMATION_CENTER_CORE configuration option.	(none)
ac_netname **ac_netname**	Specifies the network ID that Universal Agent will use. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG NETNAME configuration option.	(none)
ac_enable_ssl **ac_enable_ssl**	Specifies (**yes** or **no**) whether or not the SSL protocol is used for network communications between the Agent and the specified network communications provider. **UAG configuration:** The value specified by **--ac_enable_ssl** is set automatically for the UAG ENABLE_SSL configuration option.	**no**
OMS Parameters		
oms_port **oms_port**	Specifies the port to use to listen for OMS connection requests. **OMS configuration:** The value specified by **--oms_port** is set automatically for the OMS SERVICE_PORT configuration option.	(none)
oms_autostart **oms_autostart**	Specifies (**yes** or **no**) whether or not OMS is started automatically by Universal Broker when Universal Broker starts. **OMS component definition:** The value specified by **--oms_autostart** is set automatically for the OMS AUTOMATICALLY_START and RESTART component definition options.	**no**
CLI Parameters		
opscli **opscli**	Specifies (**yes** or **no**) whether or not the Universal Controller Command Line Interface (CLI) tools will be installed.	**no**
User Mode Parameters		
-U **-usermode_install** **--usermode_install**	Specifies (**yes** or **no**) for a user mode installation, which defines both of the following: • Whether or not this Agent can be installed on a machine where an Agent already is installed (see Installing Multiple Agents on a Single Machine). • Whether or not the user can specify installation parameters (--unvdir, --unvcfgdir, --unvdatadir, and --unvport) that control where the Agent will be installed and configured (see Changing the Default Installation Directories).	**no**
-unvdir **--unvdir**	If --usermode_install is set to **yes**: Specifies the Agent binaries (installation) directory.	(none)
unvcfgdir **unvcfgdir**	If --usermode_install is set to **yes**: Specifies the Agent configuration files directory.	**`<--unvdir>/etc`**
unvdatadir **unvdatadir**	If --usermode_install is set to **yes**: Specifies the Agent data files directory.	**`<--unvdir>/var`**
unvport **unvport**	If --usermode_install is set to **yes**: Specifies the Universal Broker port.	(none)
Additional Parameter		
-? **-h** **help** **info**	Displays command line help.	n/a

Installation Script Example

The following example illustrates Universal Agent for HP_UX installed with the installation script, **unvinst**, and optional parameters.

```
sh ./unvinst --user user1 --userdir /homedir/user --group usergroup
             --convert_opsagent --opsdir /homedir/ops --network_provider oms
             --oms_servers 7878@oms1,7979@oms2 --oms_port 7878 --oms_autostart yes
             --ac_netname OPSAUTOCONF --ac_enable_ssl yes --opscli yes
             --usermode_install yes --unvdir /opt/universal --unvcfgdir /etc/universal --unvdatadir
/var/opt/universal --unvport 7887
```

User Mode Installation

A user mode installation, implemented through use of the usermode_install installation parameter, lets you install multiple Agents on a single machine and change the default installation directories for any Agent being installed.

You must perform a user mode installation for installing an Agent on a machine where one or more Agents already have been installed.

However, you also can perform a user mode installation for the initial installation of an Agent on a machine.

Note
You can execute the install as a non-root user if you want to execute the Agent in an unprivileged user mode environment. See User Mode Installation for specific requirements and restrictions associated with an unprivileged user mode environment.

Installing Multiple Agents on a Single Machine

If you want to install multiple Agents on the same machine, you must set the following installation parameter values for each Agent being installed in addition to the initially installed Agent.

Note
You also can set these parameters values for the initial installation of an Agent on a machine.

Installation Parameter	Value
--usermode_install	**yes**
--user	A username that is different than the username specified for any other Agent installation on this machine.
--unvdir	An Agent binaries (installation) directory that is different than the installation directory specified for any other Agent installation on this machine.
--unvport	A Universal Broker port that is different than the port specified for any other Agent installation on this machine.

Changing the Default Installation Directories

By default, an Agent is installed and configured in default directories.

If you want to change these default directories, or if you want to install multiple Agents on the same machine, you must set the --usermode_install parameter to **yes** and specify new values in the following parameters. These directories must be different for each Agent on the same machine.

Note
You also can change these directories for the initial installation of an Agent on a machine.

Installation Parameter	Default Directory	Files Installed
--unvdir	**/opt/universal**	Agent binaries
--unvcfgdir	**/etc/universal**	Agent configuration files

--unvdatadir	**/var/opt/universal**	Agent data files

HP-UX PAM Customization

If security is set to PAM, the **pam.conf** file under **/etc** does not require modification. The **ucmd** service will use the modules defined in the "other" section of **pam.conf**.

Listing Universal Agent for HP-UX Information

On HP-UX, information on an installed product and sub-products is listed with the **swlist** command. The command must be executed with the superuser ID.

To list information for Universal Agent for HP-UX, issue the following command:

```
swlist -1 subproduct OPSCLI
swlist -l subproduct UNV
```

Removing Universal Agent for HP-UX

System Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **`/etc/universal`** and **`/var/opt/universal`** directories.

Step 3	Using the superuser ID, remove all Universal Agent for HP-UX sub-products by issuing the following commands:
	`swremove OPSCLI`
	`swremove UNV`
	System initialization files **`/sbin/init.d/ubrokerd`** and **`/sbin/rc3.d/S850ubrokerd`** are removed.
	`rm -rf /etc/universal`
	`rm -rf /var/opt/universal`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

User Mode Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **<--unvcfgdir>** and **<--unvdatadir>** directories.

Step 3	Using the superuser ID, remove all Universal Agent for HP-UX sub-products by issuing the following commands:
	`swremove OPSCLI`
	`swremove UNV`
	System initialization files **/sbin/init.d/ubrokerd** and **/sbin/rc3.d/S850ubrokerd** are removed.
	`rm -rf <--unvcfgdir>`
	`rm -rf <--unvdatadir>`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

Agent
To remove an Agent executing in an unprivileged user mode environment (see User Mode Installation), simply stop the `ubrokerd` daemon and remove the `./universal` installation directory. To make sure that you do not mistakenly remove a system install directory, attempt the removal with a non-privileged user account or the Broker account.

Universal Agent for Solaris Installation

- Installation Process
- Extracting the Universal Agent for Solaris Installation Files
  - Intel-Based Systems
  - SPARC-Based Systems
- Solaris Package
- Installing Universal Agent for Solaris
  - Component Selection
  - Starting the Installation Script
  - Solaris Installation Script Parameters
  - Installation Script Example
  - User Mode Installation
  - Solaris PAM Customization
- Listing Universal Agent for Solaris Information
- Removing Universal Agent for Solaris
  - System Install Removal
  - User Mode Install Removal

Installation Process

Installation of Universal Agent for Solaris is a three-step process:

Step 1	Download product distribution file (see Universal Agent for UNIX - Distribution File).
Step 2	Extract the installation files from the distribution file.
Step 3	Install the extracted files.

Extracting the Universal Agent for Solaris Installation Files

Stonebranch, Inc. provides a separate product distribution file for the following Solaris systems:

- Intel-based systems
- SPARC-based systems

Universal Agent for Solaris product distribution files are in a compressed **tar** format.

Intel-Based Systems

To uncompress and extract the installation files from the distribution file, issue the following command:

```
zcat sb-6.2.x.x-solaris-10-x64.tar.Z | tar xvf -
```

This command assumes that:

- Name of the distribution file is **`sb-6.2.x.x-solaris-10-x64.tar.Z`**.
- File is located in the current working directory.

Distribution File

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary of the installation procedure.
unv-6.2.x.x-solaris-10-x64.pkg	Solaris Intel-based file format package.
unv-opscli-6.2.x.x-solaris-10-x64.pkg	Universal Controller Command Line Interface (CLI) Intel-based file format package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.

usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

Note
If your Universal Agent for Solaris (Intel-based systems) distribution file does not contain these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

SPARC-Based Systems

To uncompress and extract the installation files from the distribution file, issue the following command:

```
zcat sb-6.2.x.x-solaris-9-sparc.tar.Z | tar xvf -
```

This command assumes that:

- Name of the distribution file is **`sb-6.2.x.x-solaris-9-sparc.tar.Z`**.
- File is located in the current working directory.

The actual base name of the **.pkg** file depends on the Solaris version for which the distribution file is intended. (See Distribution File Format for distribution file naming conventions.)

Distribution File

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary of the installation procedure.
unv-6.2.x.x-solaris-9-sparc.pkg	Solaris SPARC-based file format package.
unv-opscli-6.2.x.x-solaris-9-sparc.pkg	Universal Controller Command Line Interface (CLI) SPARC-based file format package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

Note
If your Universal Agent for Solaris (SPARC-based systems) distribution file does not contain all of these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

Solaris Package

Universal Agent for Solaris is packaged as a set of Solaris packages (extension **.pkg**).

The following table identifies the Universal Agent for Solaris package names.

Package Name	Description
UNVcom	Universal Agent common files, such as, message catalogs and translation tables.
UNVctl	Universal Control Manager
UNVcts	Universal Control Server

UNVqry	Universal Query
UNVspl	Universal Spool
UNVuag	Universal Automation Center Agent
UNVubr	Universal Broker
UNVucm	Universal Command Manager
UNVucr	Universal Certificate
UNVucs	Universal Command Server
UNVudm	Universal Data Mover Manager
UNVuds	Universal Data Mover Server
UNVuem	Universal Event Monitor Manager
UNVues	Universal Event Monitor Server
UNVusp *	Universal Connector
UNVutl	Universal Utilities
UNVoms	Universal Message Service (OMS)
UNVopscli	Universal Controller Command Line Interface (CLI)

Note
UNVusp is not provided with the Universal Agent for Solaris ***Intel-based*** file format package.

Installing Universal Agent for Solaris

Universal Agent for Solaris is installed with the **unvinst** script, which executes the **pkgadd** command. The command to start the script must be executed with the superuser ID.

Note
Stonebranch, Inc. strongly recommends the use of the **unvinst** script for the Solaris installation above any other method.

Component Selection

The Universal Agent package contains many components, which are grouped into four categories. Components in some categories are installed and activated optionally, as specified in the **`unvinst`** script by using **`unvinst`** command line parameters.

The following table describes each category, provides the default installation configuration for the components in that category, and identifies the command line parameters to use for components that are optionally installed and activated.

Note
The default installation configuration refers to new installs only. For upgrades, installed component configurations are not changed by the upgrade process.

Category	Default Configuration	Description
Base components	Always installed.	Base components are always installed and activated. They include the Universal Broker, Universal Command (UCM), Universal Data Mover (UDM), and others. Base components provide the core agent infrastructure and workload services.
Universal Automation Center Agent (UAG)	Always installed, but inactive.	Universal Controller utilizes UAG agents to provided distributed, workload automation services. UAG is activated using either the --oms_servers (recommended) or --ac_transports parameter.

Universal Message Service (OMS)	Always installed, but inactive.	OMS is message-oriented middleware that should be deployed on a small set of centrally located servers. It should not be activated on every Agent install. OMS is the network provider for UAG and the Universal Controller. OMS is activated using the --oms_autostart parameter.
Universal Controller command line programs	Not installed.	Universal Controller command line programs provide a command line interface to the Universal Controller. The installation of Universal Controller command line programs is optional. The command line programs are only required on Agents that need to interface with the Universal Controller via the command line. Universal Controller command line programs are installed using the --opscli parameter.

Starting the Installation Script

To start the installation script, **unvinst**, and install all of the Solaris packages, issue the following command:

```
sh ./unvinst [--user username [--userdir directory] [--create_user {yes|no}] ] [--group group name
[--create_group {yes|no}] ]
            [--convert_opsagent [--opsdir directory] ] [--network_provider {oms|transport}]
            [--oms_servers network address] [--ac_transports ports/addresses] [--ac_core name]
[--oms_port port]
            [--oms_autostart {yes|no}] [--ac_netname ID] [--ac_enable_ssl {yes|no}] [--opscli {yes|no
}]
            [--usermode_install {yes|no} [--unvdir directory] [--unvcfgdir directory] [--unvdatadir
directory] [--unvport port] ]
```

See Solaris Installation Script Parameters and Installation Script Example, below, for a description of the optional parameters that you can issue with **unvinst** and an example of **unvinst** with these parameters.

> **Note**
> If you want to install multiple Agents on the same machine, or add one or more additional Agents to a machine with a previously installed Agent, some of these installation parameters are required (see Installing Multiple Agents on a Single Machine).

This is a silent install. The output from **unvinst** is written to file **install.log** in the current directory.

System initialization files **/etc/init.d/ubrokerd** and **/etc/rc3.d/S85ubrokerd** are created to start the **ubrokerd** daemon when the system boots to runlevel 3.

The Universal Broker daemon will be installed and run as the **username** and **groupname** specified with the installation script parameters, below.

> **Note**
> With the Solaris, HP, and AIX installs, the previous installation directories are removed when the native installer has detected that additional directories or files have not been added to the original installation directories. If they have been modified, the directories will remain and can be reviewed and removed, as desired, by your Administrator.

Solaris Installation Script Parameters

The following table describes the optional parameters that are available in the installation script, **unvinst**, when installing Universal Agent.

The parameters are grouped into the following categories:

Category	Description
Base	Used for base install.
OMS	Used for an active OMS configuration.
UAG	Used for an active UAG configuration.
CLI	Used for Universal Controller CLI programs install.
User Mode	Used for user mode installation.

Parameter	Description	Default
Base Parameters		
-u **user** **--user**	Normal UNIX **username** that is used to execute the Universal Broker daemon. The install grants this user account ownership of all installed files, with the exception of the Universal Agent server components (for example: **ucmsrv**, **udmsrv**, and **uemsrv**) which, due to security requirements, are owned by root and will have their "set user ID on execution" bit set. • If the user account that you want to use already exists, specify that user account. • If the user account does not exist, the install creates it. • If you want to change the user account for an installed Universal Agent for Solaris system, you must perform a re-installation and use this parameter to change the user account. • If **--user** is omitted from **unvinst**, the default is used. • If --usermode_install is **yes**, there is no default.	**ubroker**
create_user **create_user**	Specification (**yes** or **no**) for whether or not to create the user name that will own the installed files as a local user.	**yes**
userdir **userdir**	Home directory for the created user account specified by --user. • If this directory does not exist, it is created when the specified user is created. • If the user specified by --user already exists, but the home directory of that user is not the default directory (**/export/home/<username>**), **--userdir** must specify the path to that home directory. • If **--userdir** is omitted from **unvinst**, the default is used.	**/export/home/<username>**
-g **group** **group**	Normal UNIX **groupname**; the Universal Broker daemon will run as this specified group. All installed files will be assigned to this group. • If the group that you want to use already exists, specify that group. • If this group does not exist, the install creates it. • If **--group** is omitted from **unvinst**, the default is used.	**ubroker**
create_group **create_group**	Specification (**yes** or **no**) for whether or not to create the group that will own the installed files as a local group.	**yes**
UAG Parameters		
-c **convert** **convert_opsagent**	Causes **unvinst** to execute **opsmerge.sh** (residing in **/opt/universal/uagsrv/bin**), which performs the following tasks: 1. Searches for an existing Opswise Automation Center Agent 1.6 or 1.7 install and converts properties stored in the **agent.props** file to corresponding configuration options in the Universal Automation Center Agent (UAG) configuration file, **uags.conf**. 2. Searches for an active Opswise Automation Center Agent 1.6 or 1.7 daemon process and attempts to stop it. 3. Assigns the ID used by the Opswise Automation Center Agent 1.6 or 1.7 to UAG by moving the **qname** file from the Opswise Automaton Center install directory to the **/var/opt/universal/uag/var** directory.	
-d **opsdir** **opsdir**	If --convert_opsagent is specified: Identifies the primary install directory for Universal Agent. • If **--opsdir** is omitted from **unvinst**, the default is used.	**/export/home/opswise**

-network_provider **--network_provider**	Specifies the network communications provider that will be used between the Agent and the Controller. Valid values are: • **oms** The Agent connects to the Controller using the Universal Message Service (OMS). Value(s) for --oms_servers must also be specified. • **transport** The Agent connects to the Controller using the Opswise Transporter and Message Hub. Values for --ac_transports and --ac_core must also be specified. ⚠ **Note** It is recommended that you always include **--network_provider** in **`unvinst`**. If you omit **--network_provider**, the Agent may not know which network communications provider to use and will not start up. **UAG configuration:** The value specified by this parameter sets the value of the UAG NETWORK_PROVIDER configuration option.	(none)
oms_servers **oms_servers**	If --network_provider is **oms**: Specifies a value, in the format **`port@host[,port2@host,...,portn@hostn]`**, for the port and network address of the OMS server(s) to be used as network communications providers. ⚠ **Note** You should always include **--oms_servers** in **`unvinst`**; OMS should be used as the network communications provider between Controller 6.2.x.x and Agent 6.2.x.x. **UAG configuration:** The value specified for **--oms_servers** is set automatically for the UAG OMS_SERVERS configuration option.	(none)
-t **transports** **ac_transports**	If --network_provider is **transport**: Specifies a value, in the format **`port@ipaddr[,port2@ipaddr2,...,portn@ipaddrn]`**, for the port and network address of the Universal Automation Center Transporter(s) used for network communications. If --convert_opsagent is specified, the value specified in **--ac_transports** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_transports** is set automatically for the UAG AUTOMATION_CENTER_TRANSPORTS configuration option.	(none)
-r **core** **ac_core**	If --network_provider is **transport**: Specifies a value for the queue name of the Universal Automation Center Message Hub. If --convert_opsagent is specified, the value specified in **--ac_core** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG AUTOMATION_CENTER_CORE configuration option.	(none)
ac_netname **ac_netname**	Specifies the network ID that Universal Agent will use. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG NETNAME configuration option.	(none)

ac_enable_ssl **ac_enable_ssl**	Specifies (**yes** or **no**) whether or not the SSL protocol is used for network communications between the Agent and the specified network communications provider. **UAG configuration:** The value specified by **--ac_enable_ssl** is set automatically for the UAG ENABLE_SSL configuration option.	**no**
OMS Parameters		
oms_port **oms_port**	Specifies the port to use to listen for OMS connection requests. **OMS configuration:** The value specified by **--oms_port** is set automatically for the OMS SERVICE_PORT configuration option.	(none)
oms_autostart **oms_autostart**	Specifies (**yes** or **no**) whether or not OMS is started automatically by Universal Broker when Universal Broker starts. **OMS component definition:** The value specified by **--oms_autostart** is set automatically for the OMS AUTOMATICALLY_START and RESTART component definition options.	**no**
CLI Parameters		
opscli **opscli**	Specifies (**yes** or **no**) whether or not the Universal Controller Command Line Interface (CLI) tools will be installed.	**no**
User Mode Parameters		
-U **-usermode_install** **--usermode_install**	Specifies (**yes** or **no**) for a user mode installation, which defines both of the following: • Whether or not this Agent can be installed on a machine where an Agent already is installed (see Installing Multiple Agents on a Single Machine). • Whether or not the user can specify installation parameters (--unvdir, --unvcfgdir, --unvdatadir, and --unvport) that control where the Agent will be installed and configured (see Changing the Default Installation Directories).	**no**
-unvdir **--unvdir**	If --usermode_install is set to **yes**: Specifies the Agent binaries (installation) directory.	(none)
unvcfgdir **unvcfgdir**	If --usermode_install is set to **yes**: Specifies the Agent configuration files directory.	**<--unvdir>/etc**
unvdatadir **unvdatadir**	If --usermode_install is set to **yes**: Specifies the Agent data files directory.	**<--unvdir>/var**
unvport **unvport**	If --usermode_install is set to **yes**: Specifies the Universal Broker port.	(none)
Additional Parameter		
-? **-h** **help** **info**	Displays command line help.	n/a

Installation Script Example

The following example illustrates Universal Agent for Solaris installed with the installation script, **unvinst**, and optional parameters.
&nbap;

```
sh ./unvinst --user user1 --userdir /homedir/user --group usergroup
             --convert_opsagent --opsdir /homedir/ops --network_provider oms
             --oms_servers 7878@oms1,7979@oms2 --oms_port 7878 --oms_autostart yes
             --ac_netname OPSAUTOCONF --ac_enable_ssl yes --opscli yes
             --usermode_install yes --unvdir /opt/universal --unvcfgdir /etc/universal --unvdatadir
/var/opt/universal --unvport 7887
```

User Mode Installation

A user mode installation, implemented through use of the usermode_install installation parameter, lets you install multiple Agents on a single machine and change the default installation directories for any Agent being installed.

You must perform a user mode installation for installing an Agent on a machine where one or more Agents already have been installed.

However, you also can perform a user mode installation for the initial installation of an Agent on a machine.

> **Note**
> You can execute the install as a non-root user if you want to execute the Agent in an unprivileged user mode environment. See User Mode Installation for specific requirements and restrictions associated with an unprivileged user mode environment.

Installing Multiple Agents on a Single Machine

If you want to install multiple Agents on the same machine, you must set the following installation parameter values for each Agent being installed in addition to the initially installed Agent.

> **Note**
> You also can set these parameters values for the initial installation of an Agent on a machine.

Installation Parameter	Value
--usermode_install	**yes**
--user	A username that is different than the username specified for any other Agent installation on this machine.
--unvdir	An Agent binaries (installation) directory that is different than the installation directory specified for any other Agent installation on this machine.
--unvport	A Universal Broker port that is different than the port specified for any other Agent installation on this machine.

Changing the Default Installation Directories

By default, an Agent is installed and configured in default directories.

If you want to change these default directories, or if you want to install multiple Agents on the same machine, you must set the --usermode_install parameter to **yes** and specify new values in the following parameters. These directories must be different for each Agent on the same machine.

> **Note**
> You also can change these directories for the initial installation of an Agent on a machine.

Installation Parameter	Default Directory	Files Installed
--unvdir	**/opt/universal**	Agent binaries
--unvcfgdir	**/etc/universal**	Agent configuration files
--unvdatadir	**/var/opt/universal**	Agent data files

Solaris PAM Customization

If security is set to PAM, the **pam.conf** file under **/etc** does not require modification. The **ucmd** service will use the modules defined in the "other" section of **pam.conf**.

Listing Universal Agent for Solaris Information

Information on an installed packages is listed with the **pkginfo** command. The command must be executed with the superuser ID.

To list information for Universal Agent for Solaris, issue the following command:

```
pkginfo UNVopscli UNVoms UNVucr UNVusp UNVutl UNVues UNVuem UNVuds UNVudm UNVucs UNVucm UNVqry UNVctl
UNVcts UNVuag UNVubr UNVspl UNVcom
```

Note
UNVusp is not provided with the Universal Agent for Solaris Intel-based file format package.

Removing Universal Agent for Solaris

System Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **/var/opt/universal** and **/etc/universal** directories.

Step 3	Using the superuser ID, remove all Universal Agent for Solaris packages by issuing the following commands:
	`pkgrm UNVopscli UNVoms UNVucr UNVusp UNVutl UNVues UNVuem UNVuds UNVudm UNVucs UNVucm UNVqry UNVctl UNVcts UNVuag UNVubr UNVspl UNVcom`
	System initialization files **/etc/init.d/ubrokerd** and **/etc/rc3.d/S85ubrokerd** are removed.
	Note
	UNVusp is not provided with the Universal Agent for Solaris Intel-based file format package.
	`rm -rf /etc/universal`
	`rm -rf /var/opt/universal`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

User Mode Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **<--unvcfgdir>** and **<--unvdatadir>** directories.

Step 3	Using the superuser ID, remove all Universal Agent for Solaris packages by issuing the following commands:
	`pkgrm UNVopscli UNVoms UNVucr UNVusp UNVutl UNVues UNVuem UNVuds UNVudm UNVucs UNVucm UNVqry UNVctl UNVcts UNVuag UNVubr UNVspl UNVcom`
	System initialization files **/etc/init.d/ubrokerd** and **/etc/rc3.d/S85ubrokerd** are removed.
	Note
	UNVusp is not provided with the Universal Agent for Solaris Intel-based file format package.
	`rm -rf <--unvcfgdir>`
	`rm -rf <--unvdatadir>`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

Agent
To remove an Agent executing in an unprivileged user mode environment (see User Mode Installation), simply stop the `ubrokerd` daemon and remove the `./universal` installation directory. To make sure that you do not mistakenly remove a system install directory, attempt the removal with a non-privileged user account or the Broker account.

Universal Agent for Linux Installation

- Installation Process
- Extracting the Universal Agent for Linux Installation Files
  - x86-Based Systems
  - x86_64-Based Systems
  - IBM S/390 and zSeries Systems
- Installing Universal Agent for Linux
  - Component Selection
  - Starting the Installation Script
  - Linux Installation Script Parameters
  - Installation Script Example
  - User Mode Installation
- Listing Universal Agent for Linux Information
- Removing Universal Agent for Linux
  - System Install Removal
  - User Mode Install Removal
- Linux PAM Customization
  - PAM Configuration File
  - Examples of Customized PAM Configuration Files
- Configuring the Agent to Run a Task without a Password

Installation Process

Installation of Universal Agent for Linux is a three-step process:

Step 1	Download product distribution file (see Universal Agent for UNIX - Distribution File).
Step 2	Extract the installation files from the distribution file.
Step 3	Install the extracted files.

Extracting the Universal Agent for Linux Installation Files

Stonebranch, Inc. provides separate product distribution files for the following Linux systems:

- x86-Based Systems
- x86_64-Based Systems
- IBM S/390 and zSeries Systems

Universal Agent for Linux product distribution files are in a compressed **tar** format.

x86-Based Systems

(The RPM package for x86-based systems contains the **i386** qualifier.)

To uncompress and extract the installation files from the product distribution file, issue the following command:

```
zcat sb-6.2.x.x-linux-2.6-i386.tar.Z | tar xvf -
```

This command assumes that the name of the distribution file is **`sb-6.2.x.x-linux-2.6-i386.tar.Z`**.

Distribution File

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary of the installation procedure.
unv-6.2.x.x-linux-2.6-i386.rpm	Linux RPM file format package.
unv-opscli-6.2.x.x-linux-2.6-i386.rpm	Universal Controller Command Line Interface (CLI) RPM file format package.
unvinst	Installation script.

upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

> **Note**
> If your Universal Agent for Linux (x86-based systems) distribution file does not contain all of these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

x86_64-Based Systems

(The RPM package for x86_64-based systems contains the **x86_64** qualifier.)

To uncompress and extract the installation files from the product distribution file, issue the following command:

```
zcat sb-6.2.x.x-linux-2.6-x86_64.tar.Z | tar xvf -
```

This command assumes that the name of the distribution file is **sb-6.2.x.x-linux-2.6-x86_64.tar.Z**.

Distribution File

The following table identifies the files contained in the distribution file.

File	Description*
Readme.unv	Summary of the installation procedure.
unv-6.2.x.x-linux-2.6-x86_64.rpm	Linux RPM file format package.
unv-opscli-6.2.x.x-linux-2.6-x86_64.rpm	Universal Controller Command Line Interface (CLI) RPM file format package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

> **Note**
> If your Universal Agent for Linux (x86_64-based systems) distribution file does not contain these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

IBM S/390 and zSeries Systems

(The RPM package for IBM S/390 and zArchitecture systems contains the **s390x** qualifier.)

To uncompress and extract the installation files from the product distribution file, issue the following command:

```
zcat sb-6.2.x.x-linux-2.6-s390x.tar.Z | tar xvf -
```

This command assumes that the name of the distribution file is **sb-6.2.x.x-linux-2.6-s390x.tar.Z**.

Distribution File

The following table identifies the files contained in the distribution file.

File	Description
Readme.unv	Summary the installation procedure.
unv-6.2.x.x-linux-2.6-s390x.rpm	Linux RPM file format package.
unv-opscli-6.2.x.x-linux-2.6-s390x.rpm	Universal Controller Command Line Interface (CLI) RPM file format package.
unvinst	Installation script.
upimerge.sh	Script that uses the Universal Installation Merge (UIM) module.
usrmode.inc	User-mode installer main script.
unvfiles.tar	User-mode installer modules archive; a set of scripts loaded and executed by **usrmode.inc**.

> **Note**
> If your Universal Agent for Linux (IBM S/390 and zSeries systems) distribution file does not contain these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

Installing Universal Agent for Linux

Universal Agent for Linux is installed with the **unvinst** script, which executes the **rpm** command. The command to start the script must be executed with the superuser ID.

> **Note**
> Stonebranch, Inc. strongly recommends the use of the **unvinst** script for the Linux installation above any other method.

Component Selection

The Universal Agent package contains many components, which are grouped into four categories. Components in some categories are installed and activated optionally, as specified in the **`unvinst`** script by using **`unvinst`** command line parameters.

The following table describes each category, provides the default installation configuration for the components in that category, and identifies the command line parameters to use for components that are optionally installed and activated.

> **Note**
> The default installation configuration refers to new installs only. For upgrades, installed component configurations are not changed by the upgrade process.

Category	Default Configuration	Description
Base components	Always installed.	Base components are always installed and activated. They include the Universal Broker, Universal Command (UCM), Universal Data Mover (UDM), and others. Base components provide the core agent infrastructure and workload services.
Universal Automation Center Agent (UAG)	Always installed, but inactive.	Universal Controller utilizes UAG agents to provided distributed, workload automation services. UAG is activated using either the --oms_servers (recommended) or --ac_transports parameter.
Universal Message Service (OMS)	Always installed, but inactive.	OMS is message-oriented middleware that should be deployed on a small set of centrally located servers. It should not be activated on every Agent install. OMS is the network provider for UAG and the Universal Controller. OMS is activated using the --oms_autostart parameter.

Universal Controller command line programs	Not installed.	Universal Controller command line programs provide a command line interface to the Universal Controller. The installation of Universal Controller command line programs is optional. The command line programs are only required on Agents that need to interface with the Universal Controller via the command line. Universal Controller command line programs are installed using the --opscli parameter.

Starting the Installation Script

To start the installation script, **unvinst**, issue the following command:

```
sh ./unvinst [--user username [--userdir directory] [--create_user {yes|no}] ] [--group group name
[--create_group {yes|no}] ]
            [--convert_opsagent [--opsdir directory] ] [--network_provider {oms|transport}]
            [--oms_servers network address] [--ac_transports ports/addresses] [--ac_core name]
[--oms_port port]
            [--oms_autostart {yes|no}] [--ac_netname ID] [--ac_enable_ssl {yes|no}] [--opscli {yes|no
}]
            [--usermode_install {yes|no} [--unvdir directory] [--unvcfgdir directory] [--unvdatadir
directory] [--unvport port] ]
```

See Linux Install Script Parameters and Installation Script Example, below, for a description of the optional parameters that you can issue with **unvinst** and an example of **unvinst** with these parameters.

> **Note**
> If you want to install multiple Agents on the same machine, or add one or more additional Agents to a machine with a previously installed Agent, some of these installation parameters are required (see Installing Multiple Agents on a Single Machine).

This is a silent install. The output from **unvinst** is written to file **install.log** in the current directory. The Agent is installed into directory **/opt**.

The Universal Broker daemon will be installed and run as the **username** and **groupname** specified with the installation script parameters, below.

> **Note**
> For this release of Linux RPM, the previous non-RPM version will not be uninstalled. The Administrator can remove the remaining files/directories as needed.

Linux Installation Script Parameters

The following table describes the optional parameters that are available in the UNIX install script (**unvinst**) when installing Universal Agent.

The parameters are grouped into the following categories:

Category	Description
Base	Used for base install.
OMS	Used for an active OMS configuration.
UAG	Used for an active UAG configuration.
CLI	Used for Universal Controller CLI programs install.
User Mode	Used for user mode installation.

Parameter	Description	Default
Base Parameters		

-u **user** **--user**	Normal UNIX **username** that is used to execute the Universal Broker daemon. The install grants this user account ownership of all installed files, with the exception of the Universal Agent server components (for example: **ucmsrv**, **udmsrv**, and **uemsrv**) which, due to security requirements, are owned by root and will have their "set user ID on execution" bit set. • If the user account that you want to use already exists, specify that user account. • If the user account does not exist, the install creates it. • If you want to change the user account for an installed Universal Agent for Linux system, you must perform a re-installation and use this parameter to change the user account. • If **--user** is omitted from **unvinst**, the default is used. • If --usermode_install is **yes**, there is no default.	**ubroker**
create_user **create_user**	Specification (**yes** or **no**) for whether or not to create the user name that will own the installed files as a local user.	**yes**
userdir **userdir**	Home directory for the created user account specified by --user. • If this directory does not exist, it is created when the specified user is created. • If the user specified by --user already exists, but the home directory of that user is not the default directory (**/home/<username>**), **--userdir** must specify the path to that home directory. • If **--userdir** is omitted from **unvinst**, the default is used.	**/home/<username>**
-g **group** **group**	Normal UNIX **groupname**; the Universal Broker daemon will run as this specified group. All installed files will be assigned to this group. • If the group that you want to use already exists, specify that group. • If this group does not exist, the install creates it. • If **--group** is omitted from **unvinst**, the default is used.	**ubroker**
create_group **create_group**	Specification (**yes** or **no**) for whether or not to create the group that will own the installed files as a local group.	**yes**
UAG Parameters		
-c **convert** **convert_opsagent**	Causes **unvinst** to execute **opsmerge.sh** (residing in **/opt/universal/uagsrv/bin**), which performs the following tasks: 1. Searches for an existing Opswise Automation Center Agent 1.6 or 1.7 install and converts properties stored in the **agent.props** file to corresponding configuration options in the Universal Automation Center Agent (UAG) configuration file, **uags.conf**. 2. Searches for an active Opswise Automation Center Agent 1.6 or 1.7 daemon process and attempts to stop it. 3. Assigns the ID used by the Opswise Automation Center Agent 1.6 or 1.7 to UAG by moving the **qname** file from the Opswise Automaton Center install directory to the **/var/opt/universal/uag/var** directory.	
-d **opsdir** **opsdir**	If --convert_opsagent is specified: Identifies the primary install directory for Universal Agent. • If **--opsdir** is omitted from **unvinst**, the default is used.	**/home/opswise**

-network_provider **--network_provider**	Specifies the network communications provider that will be used between the Agent and the Controller. Valid values are: • **oms** The Agent connects to the Controller using the Universal Message Service (OMS). Value(s) for --oms_servers must also be specified. • **transport** The Agent connects to the Controller using the Opswise Transporter and Message Hub. Values for --ac_transports and --ac_core must also be specified. **Note** It is recommended that you always include **--network_provider** in **`unvinst`**. If you omit **--network_provider**, the Agent may not know which network communications provider to use and will not start up. **UAG configuration:** The value specified by this parameter sets the value of the UAG NETWORK_PROVIDER configuration option.	(none)
oms_servers **oms_servers**	If --network_provider is **oms**: Specifies a value, in the format **`port@host[,port2@host,...,portn@hostn]`**, for the port and network address of the OMS server(s) to be used as network communications providers. **Note** You should always include **--oms_servers** in **`unvinst`**; OMS should be used as the network communications provider between a Controller 6.2.x.x and Agent 6.2.x.x. **UAG configuration:** The value specified for **--oms_servers** is set automatically for the UAG OMS_SERVERS configuration option.	(none)
-t **transports** **ac_transports**	If --network_provider is **transport**: Specifies a value, in the format **`port@ipaddr[,port2@ipaddr2,...,portn@ipaddrn]`**, for the port and network address of the Universal Automation Center Transporter(s) used for network communications. If --convert_opsagent is specified, the value specified in **--ac_transports** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_transports** is set automatically for the UAG AUTOMATION_CENTER_TRANSPORTS configuration option.	(none)
-r **core** **ac_core**	If --network_provider is **transport**: Specifies a value for the queue name of the Universal Automation Center Message Hub. If --convert_opsagent is specified, the value specified in **--ac_core** will override any currently configured options, including those imported from **`agent.props`**. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG AUTOMATION_CENTER_CORE configuration option.	(none)
ac_netname **ac_netname**	Specifies the network ID that Universal Agent will use. **UAG configuration:** The value specified by **--ac_core** is set automatically for the UAG NETNAME configuration option.	(none)
ac_enable_ssl **ac_enable_ssl**	Specifies (**yes** or **no**) whether or not the SSL protocol is used for network communications between the Agent and the specified network communications provider. **UAG configuration:** The value specified by **--ac_enable_ssl** is set automatically for the UAG ENABLE_SSL configuration option.	**no**
OMS Parameters		

oms_port **oms_port**	Specifies the port to use to listen for OMS connection requests. **OMS configuration:** The value specified by **--oms_port** is set automatically for the OMS SERVICE_PORT configuration option.	(none)
oms_autostart **oms_autostart**	Specifies (**yes** or **no**) whether or not OMS is started automatically by Universal Broker when Universal Broker starts. **OMS component definition:** The value specified by **--oms_autostart** is set automatically for the OMS AUTOMATICALLY_START and RESTART component definition options.	**no**
CLI Parameters		
opscli **opscli**	Specifies (**yes** or **no**) whether or not the Universal Controller Command Line Interface (CLI) tools will be installed.	**no**
User Mode Parameters		
-U **-usermode_install** **--usermode_install**	Specifies (**yes** or **no**) for a user mode installation, which defines both of the following: • Whether or not this Agent can be installed on a machine where an Agent already is installed (see Installing Multiple Agents on a Single Machine). • Whether or not the user can specify installation parameters (--unvdir, --unvcfgdir, --unvdatadir, and --unvport) that control where the Agent will be installed and configured (see Changing the Default Installation Directories).	**no**
-unvdir **--unvdir**	If --usermode_install is set to **yes**: Specifies the Agent binaries (installation) directory.	(none)
unvcfgdir **unvcfgdir**	If --usermode_install is set to **yes**: Specifies the Agent configuration files directory.	**<--unvdir>/etc**
unvdatadir **unvdatadir**	If --usermode_install is set to **yes**: Specifies the Agent data files directory.	**<--unvdir>/var**
unvport **unvport**	If --usermode_install is set to **yes**: Specifies the Universal Broker port.	(none)
Additional Parameter		
-? **-h** **help** **info**	Displays command line help.	n/a

Installation Script Example

The following example illustrates Universal Agent for Linux installed with the installation script, **unvinst**, and optional parameters.

```
sh ./unvinst --user user1 --userdir /homedir/user --group usergroup
             --convert_opsagent --opsdir /homedir/ops --network_provider oms
             --oms_servers 7878@oms1,7979@oms2 --oms_port 7878 --oms_autostart yes
             --ac_netname OPSAUTOCONF --ac_enable_ssl yes --opscli yes
             --usermode_install yes --unvdir /opt/universal --unvcfgdir /etc/universal --unvdatadir
/var/opt/universal --unvport 7887
```

User Mode Installation

A user mode installation, implemented through use of the usermode_install installation parameter, lets you install multiple Agents on a single machine and change the default installation directories for any Agent being installed.

You must perform a user mode installation for installing an Agent on a machine where one or more Agents already have been installed.

However, you also can perform a user mode installation for the initial installation of an Agent on a machine.

> **Note**
> You can execute the install as a non-root user if you want to execute the Agent in an unprivileged user mode environment. See User Mode Installation for specific requirements and restrictions associated with an unprivileged user mode environment.

Installing Multiple Agents on a Single Machine

If you want to install multiple Agents on the same machine, you must set the following installation parameter values for each Agent being installed in addition to the initially installed Agent.

> **Note**
> You also can set these parameters values for the initial installation of an Agent on a machine.

Installation Parameter	Value
--usermode_install	**yes**
--user	A username that is different than the username specified for any other Agent installation on this machine.
--unvdir	An Agent binaries (installation) directory that is different than the installation directory specified for any other Agent installation on this machine.
--unvport	A Universal Broker port that is different than the port specified for any other Agent installation on this machine.

Changing the Default Installation Directories

By default, an Agent is installed and configured in default directories.

If you want to change these default directories, or if you want to install multiple Agents on the same machine, you must set the --usermode_install parameter to **yes** and specify new values in the following parameters. These directories must be different for each Agent on the same machine.

> **Note**
> You also can change these directories for the initial installation of an Agent on a machine.

Installation Parameter	Default Directory	Files Installed
--unvdir	**/opt/universal**	Agent binaries
--unvcfgdir	**/etc/universal**	Agent configuration files
--unvdatadir	**/var/opt/universal**	Agent data files

Listing Universal Agent for Linux Information

Information on installed packages is listed with the **rpm** command. The command must be executed with the superuser ID.

To list information for the Universal Agent for Linux, issue the following command:

```
rpm -qi unv
rpm -qi unv-opscli
```

Removing Universal Agent for Linux

System Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **`/var/opt/universal`** and **`/etc/universal`** directories.

Step 3	Using the superuser ID, remove all Universal Agent for Linux packages by issuing the following commands:
	`rpm -e unv-opscli`
	`rpm -e unv`
	`rm -rf /etc/universal`
	`rm -rf /var/opt/universal`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

User Mode Install Removal

Step 1	Stop the **ubrokerd** daemon.
Step 2	Make a backup copy of the **`<--unvcfgdir>`** and **`<--unvdatadir>`** directories.

Step 3	Using the superuser ID, remove all Universal Agent for Linux packages by issuing the following commands:
	`rpm -e unv-opscli`
	`rpm -e unv`
	`rm -rf <--unvcfgdir>`
	`rm -rf <--unvdatadir>`

Step 4	Delete the Agent user account (`ubroker`) and its home directory: `userdel -r ubroker`
Step 5	Delete the Agent group (`ubroker`): `groupdel ubroker`

Agent
To remove an Agent executing in an unprivileged user mode environment (see User Mode Installation), simply stop the `ubrokerd` daemon and remove the `./universal` installation directory. To make sure that you do not mistakenly remove a system install directory, attempt the removal with a non-privileged user account or the Broker account.

Linux PAM Customization

Linux installations utilize Pluggable Authentication Modules (PAM) for user authentication. Many of the Universal Agent servers, such as Universal Command (UCMD), Universal Data Mover (UDM), and Universal Control (UCTL), must authenticate user accounts and passwords. Proper PAM configuration is essential for product operation.

There are many organizations and companies that package and distribute the Linux operating system. Most have consistent PAM implementations, but there are exceptions.

All supported Linux installations - except for SUSE 9 and below - require the sample PAM configuration file delivered with Universal Agent to be copied to directory **/etc/pam.d**:

```
cp /opt/universal/ucmdsrv/samp/ucmd.pam /etc/pam.d/ucmd
```

PAM Configuration File

All Universal Agent components utilize the same PAM configuration file.

32-Bit Linux Systems

For 32-bit Linux systems (X_86-Based systems), its contents are:

auth	required	/lib/security/pam_pwdb.so shadow nullok
auth	required	/lib/security/pam_nologin.so
account	required	/lib/security/pam_pwdb.so

64-Bit Linux Systems

For 64-bit Linux systems (x86_64-Based systems; S/390 and z/Series systems) its contents are:

auth	required	/lib64/security/pam_pwdb.so shadow nullok
auth	required	/lib64/security/pam_nologin.so
account	required	/lib64/security/pam_pwdb.so

Your Administrator must modify this sample PAM file to meet your local configuration.

Examples of Customized PAM Configuration Files

Universal Agent for Redhat systems 5.0 and greater

auth	include	system-auth
auth	required	pam_nologin.so
account	include	system-auth

Universal Agent for SUSE-based systems 10.0 and greater

auth	required	pam_unix2.so nullok
auth	required	pam_nologin.so
account	include	common-account

Alternative Universal Agent for SUSE 10.1

auth	required	pam_unix2.so nullok
auth	required	pam_nologin.so
account	include	common-account

Configuring the Agent to Run a Task without a Password

Prior to release 5.1, Universal Automation Center used the **ops_suexec.nopass file**, which listed all trusted users.

As of release 5.1, this file no longer is used. To apply your desired security configuration, access the UAG SECURITY configuration option in the **`uags.conf`** configuration file:

- If you do not want security, set the value to **none**.
- If you want security, set the value to **pam** and update the following files:
  - Check the **/etc/pam.d/ucmd** configuration file to ensure that it contains the minimum PAM security settings (see Examples of Customized PAM Configuration Files, above).
  - Add the following to **/etc/universal/uacl.conf** for each user: **uag_work_request [username],allow,noauth**
    Also, verify that the user has a **/home** directory defined.

Universal Agent for UNIX - Customization

- Overview
- Universal Broker Customization
  - Universal Broker Configuration
  - Universal Broker System Initialization
- Universal Automation Center Agent Customization
  - Universal Automation Center Agent Configuration
- Universal Command Manager Customization
  - Universal Command Manager Configuration
- Universal Command Server Customization
  - Universal Command Server Configuration
- Universal Connector Customization
  - Universal Connector Configuration
  - Universal Connector SAP RFC Configuration
- Universal Control Manager Customization
  - Universal Control Manager Configuration
- Universal Control Server Customization
  - Universal Control Server Configuration
- Universal Data Mover Manager Customization
  - Universal Data Mover Manager Configuration
- Universal Data Mover Server Customization
  - Universal Data Mover Server Configuration
- Universal Event Monitor Manager Customization
  - Universal Event Monitor Manager Configuration
- Universal Event Monitor Server Customization
  - Universal Event Monitor Server Configuration
- Universal Query Customization
  - Universal Query Configuration

Overview

This page provides the following information for the customization of Universal Agent components:

- Configuration
- System initialization (Universal Broker)

(For information on applying product licenses to installed Universal Agent for UNIX components, see UNIX Installation - Licensing.)

Universal Broker Customization

Universal Broker Configuration

Configuration options for Universal Broker are stored in configuration file, **ubroker.conf**, in directory **/etc/universal** by default.

See the Universal Broker 6.2.x Reference Guide for details on configuring Universal Broker.

Universal Broker System Initialization

A Broker daemon start-up script is provided in file **`/opt/universal/ubroker/ubrokerd`**. A single command line argument — either **start**, **stop**, **status**, or **restart** — instructs the script on the action to take.

See the Universal Broker 6.2.x Reference Guide for details on the Broker daemon script.

To start the Broker daemon automatically each time that the system is started, add this script to the system initialization process. This is performed by the AIX, HP-UX, Solaris, and Linux (Red Hat- and SUSE-style systems) installation process, but not the generic UNIX installation script.

Unless the Broker daemon is started by the system initialization process, it will inherit the environment of the user that starts it. In some cases, there may be environmental variables that should not be inherited. For those variables, the start-up script **`/opt/universal/ubroker-6.2.x/ubrokerd`** can be modified to unset the undesired environmental variables.

The format of the unset is as follows:

- **unset Variable1**
- **unset MAIL**
- **unset Variable2**

This above modifications would cause **Variable1**, **MAIL**, and **Variable2** to become UNSET within the environment of the Broker Daemon.

Universal Automation Center Agent Customization

Universal Automation Center Agent Configuration

Configuration options for Universal Automation Center Agent (UAG) are stored in configuration file, **uags.conf**, in directory **`/etc/universal`** by default. See the Universal Automation Center Agent 6.2.x Reference Guide for details on configuring UAG.

UAG runs as a component managed by Universal Broker. UAG provides a component definition file, **uag**, that Universal Broker uses to start it and establish its runtime environment. **uag** is located in directory **`/etc/universal/comp`** by default.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

UAG uses the Universal Access Control List (UACL) configuration file as a level of product security. How UAG utilizes the UACL file is described in the Universal Agent 6.2.x User Guide.

Universal Command Manager Customization

Universal Command Manager Configuration

Configuration options for Universal Command Manager are stored in configuration file, **ucmd.conf**, in directory **`/etc/universal`** by default.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Manager.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Command Server Customization

Universal Command Server Configuration

Configuration options for Universal Command Server are stored in configuration file, **ucmds.conf**, in directory **`/etc/universal`** by default.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Server.

Universal Command Server runs as a component managed by Universal Broker. Universal Command Server provides a component definition file, **ucmd**, that Universal Broker uses to start the Server and establish its runtime environment. **ucmd** is located in directory **`/etc/universal/comp`** by default.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Command Server uses the Universal Access Control List (UACL) configuration file as a level of product security. How Universal Command Server utilizes the UACL file is described in the Universal Agent 6.2.x User Guide.

Universal Connector Customization

Universal Connector Configuration

Configuration options for Universal Connector are stored in configuration file **usap.conf**, in directory **`/etc/universal`**, by default.

See the Universal Connector 6.2.x Reference Guide for details on configuring Universal Connector.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Connector SAP RFC Configuration

Universal Connector utilizes SAP's NW (Netweaver) RFC interface. The NW RFC interface uses configuration file **sapnwrfc.ini** to store information required for connecting to SAP systems. The **sapnwrfc.ini** file is an SAP resource that can be shared by multiple external tools using the SAP NW RFC interface. The **sapnwrfc.ini** file groups connection information into **destinations**. Each destination contains the connection information required to establish a connection to a particular SAP system.

> **Note**
> Universal Connector for AIX and Universal Connector for Solaris SPARC-based systems do not not require the SAP NW RFC libraries to run. The RFC interface for those version of Universal Connector use configuration file **sapnwrfc.ini** to store information required for connecting to SAP systems.

The connections defined in the **sapnwrfc.ini** file must be configured to meet your local SAP environment. A sample **sapnwrfc.ini** file is installed with Universal Connector to directory **`/opt/universal/usap/samp`**. The file is provided by SAP and contains complete documentation on its configuration.

In order for Universal Connector to find the configured **sapnwrfc.ini** file, it must be placed in the Universal Connector executable directory, or environment variable **RFC_INI** must be set to its full path name.

Universal Control Manager Customization

Universal Control Manager Configuration

Configuration options for Universal Control Manager are stored in configuration file, **uctl.conf**, in directory **/etc/universal** by default.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Manager.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Control Server Customization

Universal Control Server Configuration

Configuration options for Universal Control Server are stored in configuration file, **uctls.conf**, in directory **`/etc/universal`** by default.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Server.

Universal Control Server runs as a component managed by Universal Broker. Universal Control Server provides a component definition file, **uctl**, that Universal Broker uses to start the Server and establish its runtime environment. **uctl** is located in directory **/etc/universal/comp** by default

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Control Server uses the Universal Access Control List (UACL) configuration file as a level of product security. How Universal Control Server utilizes the UACL file is described in the Universal Agent 6.2.x User Guide.

Universal Data Mover Manager Customization

Universal Data Mover Manager Configuration

Configuration options for UDM Manager are stored in configuration file, **udm.conf**, in directory **`/etc/universal`** by default.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring UDM Manager.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Data Mover Server Customization

Universal Data Mover Server Configuration

Configuration options for UDM Server are stored in configuration file, **udms.conf**, in directory **`/etc/universal`** by default.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Command Server.

UDM Server runs as a component managed by Universal Broker. UDM Server provides a component definition file, **udm**, that Universal Broker uses to start the Server and establish its runtime environment. **udm** is located in directory **`/etc/universal/comp`** by default

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

UDM Server uses the Universal Access Control List (UACL) configuration file as a level of product security. How Universal Data Mover Server utilizes the UACL file is described in the Universal Agent 6.2.x User Guide.

Universal Event Monitor Manager Customization

Universal Event Monitor Manager Configuration

Configuration options for UEM Manager are stored in configuration file **uem.conf**, in directory **`/etc/universal`** by default.

See the Universal Event Monitor 6.2.x Reference Guide for details on configuring UEM Manager.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Event Monitor Server Customization

Universal Event Monitor Server Configuration

Configuration options for UEM Server are stored in configuration file, **uems.conf**, in directory **`/etc/universal`** by default.

See the [Universal Event Monitor 6.2.x Reference Guide] for details on configuring UEM Server.

UEM Server runs as a component managed by Universal Broker. UEM Server provides two component definition files, located in the default directory **`/etc/universal/comp`**, that Universal Broker uses to start the Server and establish its runtime environment:

- **uems** is used to start an event-driven UEM Server.
- **uemd** is used to start a demand-driven UEM Server.

The product executable files intended for command line use are located in the default directory **`/opt/universal/bin`**:

- **uem**
- **uemload**

This directory must be added to the PATH environment variable for intended users of the executable files.

UEM Server uses the Universal Access Control List (UACL) configuration file as a level of product security. How Universal Event Monitor Server utilizes the UACL file is described in the Universal Agent 6.2.x User Guide.

Universal Query Customization

Universal Query Configuration

Configuration options for Universal Query are stored in configuration file, **uquery.conf**, in directory **`/etc/universal`** by default.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Query.

The product executable files intended for command line use are located in the default directory **/opt/universal/bin**. This directory must be added to the PATH environment variable for intended users of the executable files.

Universal Agent for UNIX - File Inventory Lists

- Universal Agent for UNIX - File Inventory Lists
- Universal Automation Center Agent
- Universal Broker
- Universal Command Manager
- Universal Command Server
- Universal Connector
- Universal Control Manager
- Universal Control Server
- Universal Data Mover Manager
- Universal Data Mover Server
- Universal Event Monitor Manager
- Universal Event Monitor Server
- Universal Certificate
- Universal Query
- Universal Spool Utilities
- Universal Message Service (OMS)
- Universal Controller Command Line Interface (CLI)

Universal Agent for UNIX - File Inventory Lists

The Universal Agent installation for UNIX includes the files required for:

- Universal Automation Center Agent
- Universal Broker
- Universal Command Manager and Server
- Universal Connector
- Universal Control Manager and Server
- Universal Data Mover Manager and Server
- Universal Event Monitor Manager and Server
- Universal Certificate
- Universal Query
- Universal Spool Utilities
- Universal Message Service (OMS)
- Universal Controller Command Line Interface (CLI)

This page identifies the files installed with each of these Universal Agent components / utilities.

The file paths listed presume that the installation directory (**/opt**) and the variable file directory (**/var/opt**) are the defaults. These directories can be changed on most UNIX installations.

Universal Automation Center Agent

File	Description
/opt/universal/uagsrv/bin/uagsrv	UAG component program.
/opt/universal/uagsrv/samp	UAG sample directory.
/opt/universal/uagsrv/samp/OPSWISE-MIB.txt	MIB file.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/uags.conf	UAG configuration file.
/etc/universal/comp/uag	UAG component definition file.
/var/opt/universal/trace	UAG trace file directory.
/opt/universal/ubroker/tmpl/uagcfg	Template file for the UAG configuration.
/opt/universal/ubroker/tmpl/uagcmp	Template file for the UAG component definition.

Universal Broker

File	Description

/opt/universal/ubroker/ubrinst.src	Broker installation source file (included in Generic UNIX packaging only).
/opt/universal/ubroker/ubrokerd	Broker daemon start script.
/opt/universal/ubroker/bin/ubroker	Console broker program.
/opt/universal/ubroker/bin/ubrokerd	Daemon broker program.
/etc/universal/comp	Component definition directory.
/opt/universal/ubroker/samp	Broker sample directory.
/opt/universal/ubroker/tmpl	XML template files used by I-Management Console for remotely configuring Universal Agent.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/var/opt/universal/log	Broker message log directory.
/var/opt/universal/trace	Broker trace file directory.
/var/opt/universal/spool	Broker component database files.
/etc/universal/uacl.conf	Universal Access Control List configuration file.
/etc/universal/ubroker.conf	Broker configuration file.

Universal Command Manager

File	Description
/opt/universal/ucmdmgr/ucmcinst.src	Manager installation source file (included in Generic UNIX packaging only).
/opt/universal/ucmdmgr/bin/ucmd	Universal Command Manager program.
/opt/universal/ucmdmgr/bin/uencrypt	Universal Encrypt program file.
/opt/universal/ucmdmgr/samp	Manager sample file directory.
/opt/universal/bin/ucmd	Symbolic link to the Manager program.
/opt/universal/bin/uencrypt	Symbolic link to the Universal Encrypt program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/ucmd.conf	Manager configuration file.

Universal Command Server

File	Description
/opt/universal/ucmdsrv/ucmsinst.src	Server installation source file (included in Generic UNIX packaging only).
/opt/universal/ucmdsrv/bin/ucmsrv	Server component program.
/opt/universal/ucmdsrv/bin/ucopy	Utility used for binary file copies. Similar to the UNIX cat command.
/opt/universal/ucmdsrv/bin/umet	Universal Message Translator program.
/opt/universal/ucmdsrv/samp	Server sample directory.
/etc/universal/comp	Universal Command Server component definition file.
/opt/universal/bin/ucopy	Symbolic link to the Universal Copy program.
/opt/universal/bin/umet	Symbolic link to the Universal Message Translator program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/var/opt/universal/trace	Server trace file directory.

/var/opt/universal/spool	Server component database files and spool file directory.
/etc/universal/ucmds.conf	Server configuration file.

Universal Connector

File	Description
/opt/universal/usap/usapinst.src	Server installation source file (include in Generic UNIX packaging only).
/opt/universal/usap/bin/usap	USAP program.
/opt/universal/usap/samp	Server sample file directory. **Note** The **samp** directory contains the SAP NW (NetWeaver) RFC interface `sapnwrfc.ini` configuration file. However, for Universal Connector for AIX and Universal Connector for Solaris SPARC-based systems, **samp** contains SAP RFC interface `saprfc.ini` configuration file, as thise versions of Universal Connector do not require the SAP NW RFC libraries.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/usap.conf	Server configuration file.

Universal Control Manager

File	Description
/opt/universal/uctlmgr/uctcinst.src	Manager installation source file (include in Generic UNIX packaging only).
/opt/universal/uctlmgr/bin/uctl	Manager program.
/opt/universal/uctlmgr/samp	Manager sample file directory.
/opt/universal/bin/uctl	Symbolic link to the Manager program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/uctl.conf	Manager configuration file.

Universal Control Server

File	Description
/opt/universal/uctlsrv/uctsinst.src	Server installation source file (include in Generic UNIX packaging only).
/opt/universal/uctlsrv/bin/uctsrv	Server component program.
/opt/universal/uctlsrv/samp	Server sample file directory.
/etc/universal/comp	Universal Control Server component definition file.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/var/opt/universal/trace	Server trace file directory.
/etc/universal/uctls.conf	Server configuration file.

Universal Data Mover Manager

File	Description

/opt/universal/udmmgr/udmminst.src	Manager installation source file (included in Generic UNIX packaging only).
/opt/universal/udmmgr/bin/udm	Manager program.
/opt/universal/udmmgr/samp	Manager sample file directory.
/opt/universal/bin/udm	Symbolic link to the Manager program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/udm.conf	Manager configuration file.

Universal Data Mover Server

File	Description
/opt/universal/udmsrv/udmsinst.src	Server installation source file (include in Generic UNIX packaging only).
/opt/universal/udmsrv/bin/udmsrv	Server program.
/opt/universal/udmsrv/samp	Server sample file directory.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/udms.conf	Server configuration file.
/etc/universal/comp	Universal Data Mover Server component definition file.

Universal Event Monitor Manager

File	Description
/opt/universal/uemmgr/uemminst.src	Manager installation source file (included in Generic UNIX packaging only).
/opt/universal/uemmgr/bin/uem	Manager program.
/opt/universal/uemmgr/samp	Manager sample file directory.
/opt/universal/bin/uem	Symbolic link to the Manager program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/uem.conf	Manager configuration file.

Universal Event Monitor Server

File	Description
/opt/universal/uemsrv/uemsinst.src	Server installation source file (included in Generic UNIX packaging only).
/opt/universal/uemsrv/bin/uemsrv	Server program.
/opt/universal/uemsrv/bin/uemload	UEM database load utility.
/opt/universal/uemsrv/samp	Server sample file directory.
/opt/universal/bin/uemload	Symbolic link to the UEM database load utility.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/uems.conf	Server configuration file.
/etc/universal/comp/uems	Event-driven Universal Event Monitor Server component definition file.
/etc/universal/comp/uemd	Demand-driven Universal Event Monitor Server component definition file.

Universal Certificate

File	Description
/opt/universal/ucert/bin/ucert	UCERT program

Universal Query

File	Description
/opt/universal/uquery/uqryinst.src	Query installation source file (included in Generic UNIX packaging only).
/opt/universal/uquery/bin/uquery	Universal Query program.
/opt/universal/uquery/samp	Query sample file directory.
/opt/universal/bin/uquery	Symbolic link to the Query program.
/opt/universal/nls	Code page files used for text translation between different operating systems and product message catalogs.
/etc/universal/uquery.conf	Query configuration file.

Universal Spool Utilities

File	Description
/opt/universal/uspool/bin/uslist	Used to list the contents of Universal Spool files.
/opt/universal/uspool/bin/uslrm	Used to remove records from Universal Spool files.
/opt/universal/uspool/bin/udb_archive **/opt/universal/uspool/bin/udb_checkpoint** **/opt/universal/uspool/bin/udb_deadlock** **/opt/universal/uspool/bin/udb_dump** **/opt/universal/uspool/bin/udb_load** **/opt/universal/uspool/bin/udb_printlog** **/opt/universal/uspool/bin/udb_recover** **/opt/universal/uspool/bin/udb_stat** **/opt/universal/uspool/bin/udb_upgrade** **/opt/universal/uspool/bin/udb_verify**	Miscellaneous spool file utilities. Should be used only for debugging purposes, and only at the request of Stonebranch, Inc. Customer Support.

Universal Message Service (OMS)

File	Description
/opt/universal/omssrv/bin/omsadm	OMS Administration utility program.
/opt/universal/omssrv/bin/omssrv	OMS Server program.
/opt/universal/omssrv/samp/oms.def	OMS Server sample component definition file.
/opt/universal/omssrv/samp/omss.conf	OMS Server sample configuration file.
/opt/universal/nls/omsmceng.umc	OMS message catalog.
/opt/universal/ubroker/tmpl/omscfg	Template file for the OMS Server configuration.
/opt/universal/ubroker/tmpl/omscmp	Template file for the OMS Server component definition.
/etc/universal/omss.conf	OMS Server configuration file.
/etc/universal/comp/oms	OMS Server component definition file.

Universal Controller Command Line Interface (CLI)

File	Description
/opt/universal/opscli/bin/ops-agent-status	Lists the status of one or more Agents.

/opt/universal/opscli/bin/ops-change-credential-password	Changes the runtime password for the specified Universal Controller credentials resource.
/opt/universal/opscli/bin/ops-change-user-password	Changes the password for the specified Universal Controller User account.
/opt/universal/opscli/bin/ops-connector-status	Lists the status of one or more Connectors.
/opt/universal/opscli/bin/ops-export-bulk	Performs a bulk export of all Controller database records.
/opt/universal/opscli/bin/ops-export-trigger	Performs an export of the specified triggers and any associated records.
/opt/universal/opscli/bin/ops-import-bulk	Imports Controller database records that were exported with ops-export-bulk.
/opt/universal/opscli/bin/ops-import-trigger	Imports triggers and associated records that were exported using ops-export-trigger.
/opt/universal/opscli/bin/ops-manual-setcompleted	Sets Manual task status to Success.
/opt/universal/opscli/bin/ops-manual-setstarted	Changes Manual task status from Action Required to Started.
/opt/universal/opscli/bin/ops-resume-agent	Allows a suspended Agent to submit tasks once again.
/opt/universal/opscli/bin/ops-resume-agent-cluster	Allows a suspended Agent Cluster to execute tasks once again.
/opt/universal/opscli/bin/ops-resume-agent-cluster-membership	Allows a specified Agent to rejoin specified Agent Cluster.
/opt/universal/opscli/bin/ops-set-agent-cluster-task-execution-limit	Sets the task execution limit for an Agent Cluster.
/opt/universal/opscli/bin/ops-set-agent-task-execution-limit	Sets the task execution limit for an Agent.
/opt/universal/opscli/bin/ops-suspend-agent	Temporarily prevents a specified Agent from submitting tasks.
/opt/universal/opscli/bin/ops-suspend-agent-cluster	Temporarily prevents the Agents in the specified Cluster from submitting tasks.
/opt/universal/opscli/bin/ops-suspend-agent-cluster-membership	Temporarily removes an Agent from cluster membership.
/opt/universal/opscli/bin/ops-task-cancel	Cancels a task.
/opt/universal/opscli/bin/ops-task-forcefinish	Force Finishes a task.
/opt/universal/opscli/bin/ops-task-hold	Places a task on hold.
/opt/universal/opscli/bin/ops-task-launch	Executes a task.
/opt/universal/opscli/bin/ops-task-list	Lists the specified tasks.
/opt/universal/opscli/bin/ops-task-release	Releases a held task.
/opt/universal/opscli/bin/ops-task-rerun	Re-executes the specified task.
/opt/universal/opscli/bin/ops-task-setpriority	Changes the execution priority of a Started task.
/opt/universal/opscli/bin/ops-task-skip	Skips the specified task instance.
/opt/universal/opscli/bin/ops-task-status	Displays the status of the task instance(s) associated with a task.
/opt/universal/opscli/bin/ops-trigger-disable	Disables the specified trigger(s).
/opt/universal/opscli/bin/ops-trigger-enable	Enables the specified trigger(s).
/opt/universal/opscli/bin/ops-trigger-now	Satisfies all conditions in the specified trigger and launches its associated tasks.
/opt/universal/opscli/bin/ops-trigger-status	Displays the status of the specified trigger(s).
/opt/universal/opscli/bin/ops-update-resource-limit	Sets the resource limit for a resource.
/opt/universal/opscli/bin/ops-variable-list	Displays the specified variable's value.
/opt/universal/opscli/bin/ops-variable-set	Sets the specified variable's value.
/opt/universal/opscli/bin/uagcmd	CLI executable command.

/etc/universal/cmdtools.props	Optional CLI configuration file.

Universal Agent for SOA for UNIX Installation

Overview

The following information is provided for the installation of Universal Agent for SOA for UNIX:

- Installation Package
- Installation Requirements
- Pre-Installation-Upgrade Backups
- Distribution File
- Deployment Options
- Installation Procedures
- File Inventory Lists

Licensing

Universal Agent for SOA allows operation by connector and endpoint count, based on the license information it receives from the UAC Server component. (See your Stonebranch, Inc. account representative for details.)

(For licensing information, see UNIX Installation - Licensing.)

Universal Agent for SOA for UNIX - Installation Package

Package Components

The Universal Agent for SOA 6.2.x for UNIX package includes the following product components:

- Universal Application Container Server
- Universal Application Container
- Universal Application Interface

Component Compatibility

The following table identifies the compatibility of Universal Agent for SOA for UNIX 6.2.x with previous component / product versions.

Component	Compatibility
Universal Agent for SOA 6.2.x	Universal Command Manager 5.2.0, 5.1.0, 4.3.0, 4.2.0, 4.1.0, and 3.2.0.

The component references pertain to all supported platforms for that version.

Universal Agent for SOA for UNIX - Installation Requirements

UNIX Versions

To install Universal Agent for SOA for UNIX, you must have one of the following versions of UNIX:

- AIX 5.3 TL9 and above
- Linux 2.6 kernel and greater
    - RedHat Package Manager (RPM)
    - Intel (x86) Compatible Systems

Additional Requirements

In addition, you must have:

- 512MB RAM minimum, 1 GB or more preferred.
- 150 MB free disk space.
- TCP/IP socket implementation.
- Superuser (root) access.
- Bourne shell or compatible.
- Universal Agent 5.2.0.0 or later (32-bit packages only).

Platform Requirements

Since platform requirements may change with new releases of a product, see Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x to make sure that your platform is supported before performing an installation.

Universal Agent for SOA for UNIX - Pre-Installation-Upgrade Backups

Universal Agent for SOA for UNIX - Pre-Installation / Upgrade Backups

The installation process overwrites the current files (exception: see the Note for Log4jConfiguration.xml in the following table), effectively removing your modifications. Backing up these files will optimize the time it takes you to get up and running after installing or upgrading.

The following list identifies the files - and their locations - that should be backed up or copied before you install a new release or upgrade a current release.

File	Location
UAC.xml File	**`/etc/universal`**
Log4jConfiguration.xml File	**`/etc/universal`** (UAC) **`/etc/universal`** (UAI) **Note** The installation process does not overwrite Log4jConfiguration.xml files.
JMS Provider Client Jar Files	**`/opt/universal/uac/container/webapps/axis2/WEB-INF/lib`** **Note** The names of the jar files differ depending on which JMS Provider you are using.
JMS Provider Client Properties Files	**`/opt/universal/uai/xml`** **Note** These are suggested locations; you can place these files anywhere on the files system. If you have located these files under the **uai** directory, you should back them up.
Payload Files	Normally, payload files should be located on the host system where Universal Command Manager is installed. If you have chosen to store them elsewhere, the suggested location is: **`/opt/universal/uai/xml`** **Note** You can place these files anywhere on the files system. If you have located these files under the **uai** directory, you should back them up.

Universal Agent for SOA for UNIX - Distribution File

- UNIX Distribution File
- Obtaining the Distribution File
- Distribution File Format

UNIX Distribution File

Stonebranch, Inc. provides different Universal Agent for SOA for UNIX packages for different types of UNIX operating systems.

Obtaining the Distribution File

To obtain the Universal Agent for SOA for UNIX package for your type of UNIX operating system, you must download the corresponding product distribution file from the Stonebranch Customer Portal.

> **Note**
> A customer user ID and password — provided by Stonebranch, Inc. — are required to access this area.

After a distribution file has been downloaded, the installation files contained in that file must be extracted before the product can be installed (see Universal Agent for SOA for UNIX - Installation Procedures).

Distribution File Format

The name of each Universal Agent for SOA for UNIX distribution file has the following format:

```
sb-soa-Version.Release.Modification Level.Maintenance Level-operating
system-version(.release)(-platform).tar.Z
```

For example: **`sb-soa-6.2.x.x-linux-2.4-i386.tar.Z`**

In this format:

- **Version** is the current version of Universal Agent for SOA.
- **Release** is the current release of Universal Agent for SOA.
- **Modification Level** is the current Universal Agent feature set.
- **Maintenance Level** is the Universal Agent build level.
- **operating system** is the name of the operating system (for example, AIX or Linux).
- **version(.release)** is the supported version and, optionally, the release of the operating system.
- **platform** is the targeted hardware platform (for example, i386). It is included in the file name only if there is more than one platform available for the specified operating system.

Universal Agent for SOA for UNIX - Deployment Options

- Deployment Options
- Single-Server Deployment (SSD)
  - SSD Deployment Flow of Events
- Distributed Server Deployment (DSD)
  - DSD Deployment Flow of Events

Deployment Options

Deployment of Universal Agent for SOA has been designed to be flexible in order to fit the needs of your Enterprise IT.

There are two main deployment options:

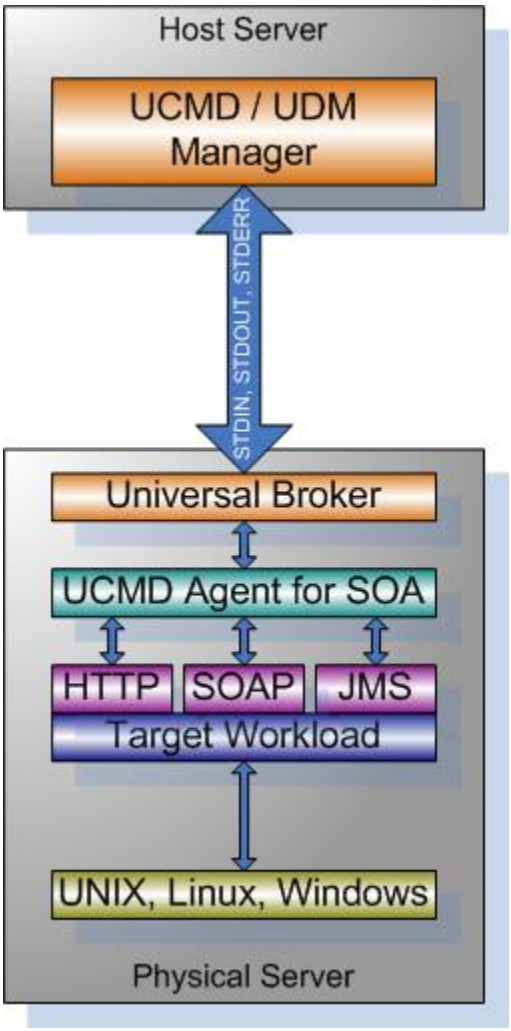
1. Single-Server Deployment (SSD)
2. Distributed Server Deployment (DSD)

SSD is the default deployment.

Single-Server Deployment (SSD)

Single-Server Deployment (SSD) is where all components, with the exception of the Universal Command Agent managers, are deployed to the same physical server. This includes Universal Command Agent for SOA, Universal Broker and associated components, and your target workload(s).

The following figure illustrates the Single Server Deployment (SSD).



SSD Deployment Flow of Events

The following list identifies the flow of events that occur with SSD deployment:

1. The calling application invokes the Universal Command (UCMD) Manager to execute a remote workload. For this example the remote workload, or target workload, is an internet or message based workload.

> **Note**
> Although this figure implies the use of a Java-based workload, you can execute any remote workload, regardless of what language the workload is implemented in, as long as it has an interface that supports HTTP, SOAP, or JMS.

2. The request is forwarded to the Universal Broker specified in the invocation of the Universal Command Manager, which then spawns the UCMD Server and passes the workload execution request to Universal Command Agent for SOA.
3. Universal Command Agent for SOA will execute the workload deployed on the same physical server and return any messages or data back to the Universal Command Manager.

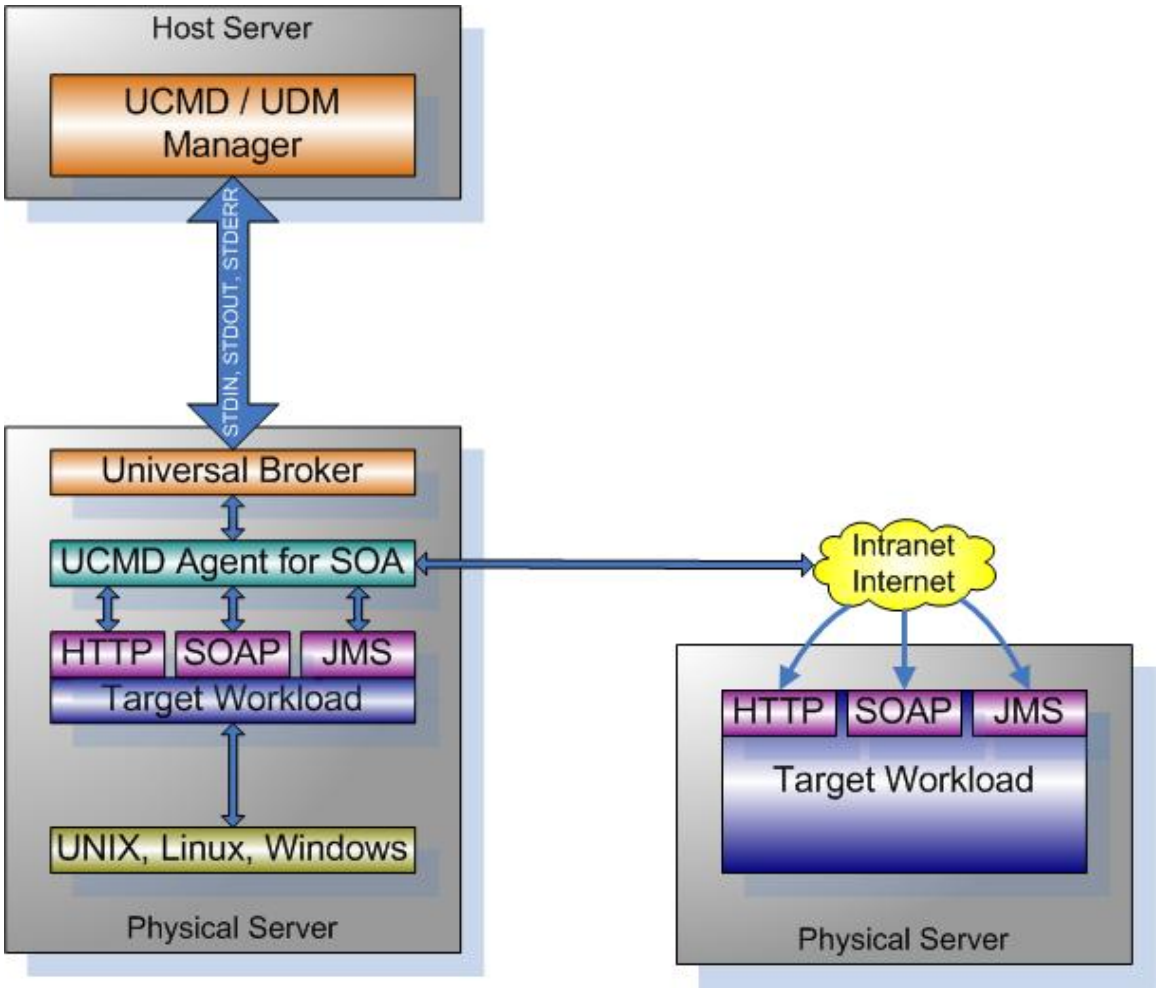
Distributed Server Deployment (DSD)

Distributed Server Deployment (DSD) is where:

- Universal Broker and Universal Command Agent for SOA are located together on one physical server.
- Target workload is located on a different physical server.

> **Note**
> The target workload location is user-defined. The SERVICE_URL option specifies how Universal Command Agent for SOA knows where to look for the target workload.

The following figure illustrates the Distributed Server Deployment (DSD).



DSD Deployment Flow of Events

The flow is nearly the same as with the Single Server Deployment except that the location of the target workload is not **localhost**.

Universal Agent for SOA for UNIX - Installation Procedures

Universal Agent for SOA for UNIX Installation Procedures

The following procedures are provided for the installation and modification of Universal Agent for SOA for UNIX:

- Universal Agent for SOA for AIX Installation
- Universal Agent for SOA for Linux Installation

Installation Process

Installation is a straightforward process using the platform-specific package. The installed environment is self-contained and follows current Universal Agent installation standards.

Installation Prerequisite

Before installing Universal Command Agent for SOA 6.2.x, you must install Universal Products 3.2.0 or later.

Universal Agent for SOA for AIX Installation

- Product Distribution File
- Unpacking and Installation Procedures
- Removing Universal Agent for SOA for AIX
- Listing Universal Agent for SOA for AIX Information

Product Distribution File

The Universal Agent for SOA for AIX product distribution file is in a compressed **tar** format.

The name of the Universal Automation Center 5 for SOA for AIX distribution file has the following format:

`sb-soa-6.2.x.x-aix-5.3.tar.Z`

Unpacking and Installation Procedures

To unpack and install Universal Agent for SOA, perform the following steps:

Step 1	Create a directory (or select an existing directory) in which to save the package file.
Step 2	Save the package file into that directory.

Step 3	Uncompress and extract the installation files in the current working directory. The command to extract the files is:
	`zcat sb-soa-6.2.x.x-aix-5.3.tar.Z | tar xvf -`
	If your operating system does not support the **zcat** command, use the following command:
	`gunzip sb-soa-6.2.x.x-aix-5.3.tar.Z`
	The output of the **gunzip** command provides the following **tar** file:
	`tar -xvf sb-soa-6.2.x.x-aix-5.3.tar`

Step 4	After the extraction is complete, run the installation script, **upsinst**, which executes the **installp** command: `./upsinst` An installation log is written to file **install.log** in the current directory. **upsinst** automatically restarts the Universal Broker daemon, **ubrokerd**, at the end of the install.
Step 5	From the license file that was sent to you by Stonebranch, Inc., add the license information to the following file: **`/etc/universal/uacs.conf`**

Step 6	Recycle Universal Broker using the following commands (cd to **/opt/universal/ubroker**)
	
	First:
	
	`./ubrokerd stop`
	
	Then:
	
	`./ubrokerd start`

Step 7	Use Universal Query (cd to **`/opt/universal/bin`**) to validate that the Universal Application Container Server component of Universal Agent for SOA 6.2.x is running:
	
	`uquery -i localhost` (or the name of your server)
	
	The output for Universal Application Container Server should have the following format:
	
	`Component ID.............: 1360109684`
	`Component Name...........: uac (Server)`
	`Component Description....: Universal Application Container Server`
	`Component Version........: 6.2.0 Level 0 Release Build 101`
	`Component Type...........: uac`
	`Component Process ID.....: 23331000`
	`Component Start Time.....: 18:14:42`
	`Component Start Date.....: 06/05/15`
	`Component Command ID.....: uac`
	`Component State..........: REGISTERED`
	`Component MGR UID........:`
	`Component MGR Work ID.....:`
	`Component MGR Host Name...:`
	`Component MGR IP Address..:`
	`Component MGR Port.......:`
	`Component Comm State.....: ESTABLISHED`
	`Component Comm State Time.: 18:14:44`
	`Component Comm State Date.: 06/05/15`
	`Component MGR Restartable.: NO`
	`Component Comment........:`

Removing Universal Agent for SOA for AIX

> **Note**
> Before removing Universal Agent for SOA for AIX, stop the **ubrokerd** daemon. Also, it is strongly recommended that you back up existing data before removing Universal Agent for SOA for AIX.

Universal Agent for SOA for AIX is removed with the **installp** command. The command must be executed with the superuser ID.

To remove AIX, issue the following command:

```
installp -u UPSuac
```

All entries in the **/etc/inittab** file that reference **ubroker** are removed.

Listing Universal Agent for SOA for AIX Information

On AIX, information on an installed product is listed with the **lslpp** command. The command must be executed with the superuser ID.

Issue the following command to list information for Universal Agent for SOA for AIX:

```
lslpp -La 'UPS*'
```

Universal Agent for SOA for Linux Installation

- Product Distribution File
- Unpacking and Installation Procedures
- Removing Universal Agent for SOA for Linux
- Listing Universal Agent for SOA for Linux Information

Product Distribution File

Universal Agent for SOA 6.3.x is packaged as an RPM file (extension **.rpm**). It is installed using the Linux **rpm** command.

The name of the Universal Agent for SOA for Linux distribution file has the following format:

`sb-soa-6.2.x.x-linux-2.4-i386.tar.Z`

Unpacking and Installation Procedures

To unpack and install Universal Agent for SOA, perform the following steps:

Step 1	Create a directory (or select an existing directory) in which to save the package file.
Step 2	Save the package file into that directory.
Step 3	Uncompress and extract the installation files in the current working directory. The command to extract the files is: `zcat sb-soa-6.2.x.x-linux-2.4-i386.tar.Z | tar xvf -` If your operating system does not support the **zcat** command, use the following command: **`gunzip sb-soa-6.2.x.x-linux-2.4-i386.tar.Z`** The output of the gunzip command provides the following **tar** file: **`tar -xvf sb-soa-6.2.x.x-linux-2.4-i386.tar`**
Step 4	After the extraction is complete, run the installation script, ups*inst*, which executes the **rpm** command: `./upsinst` An installation log is written to file **install.log** in the current directory. **upsinst** automatically restarts the Universal Broker daemon, **ubrokerd**, at the end of the install.
Step 5	From the license file that was sent to you by Stonebranch, Inc., add the license information to the **`/etc/universal/uacs.conf`** file.

Step 6	Recycle Universal Broker using the following commands (cd to **`/opt/universal/ubroker`**):
	
	First:
	
	`./ubrokerd stop`
	
	Then:
	
	`./ubrokerd start`

Step 7	Use Universal Query (cd to **`/opt/universal/bin`**) to validate that the Universal Application Container Server component of Universal Agent for SOA 6.2.x is running:
	
	`uquery -i localhost` (or the name of your server)
	
	The output for Universal Application Container Server should have the following format:
	
	`Component ID.............: 1360109684`
	`Component Name...........: uac (Server)`
	`Component Description....: Universal Application Container Server`
	`Component Version........: 6.2.0 Level 0 Release Build 101`
	`Component Type...........: uac`
	`Component Process ID.....: 23331000`
	`Component Start Time.....: 18:14:42`
	`Component Start Date.....: 06/05/15`
	`Component Command ID.....: uac`
	`Component State..........: REGISTERED`
	`Component MGR UID........:`
	`Component MGR Work ID.....:`
	`Component MGR Host Name...:`
	`Component MGR IP Address..:`
	`Component MGR Port.......:`
	`Component Comm State.....: ESTABLISHED`
	`Component Comm State Time.: 18:14:44`
	`Component Comm State Date.: 06/05/15`
	`Component MGR Restartable.: NO`
	`Component Comment........:`

Removing Universal Agent for SOA for Linux

> **Note**
> Before removing Universal Agent for SOA for Linux, stop the **ubrokerd** daemon. Also, it is strongly recommended that you back up existing data before removing Universal Agent for SOA for Linux.

Universal Agent for SOA for Linux is removed with the **rpm** command. The command must be executed with the superuser ID.

To remove all packages, issue the following command:

```
rpm -e ups
```

Listing Universal Agent for SOA for Linux Information

Information on installed packages is listed with the **rpm** command. The command must be executed with the superuser ID.

To list information for the Universal Agent for SOA for Linux, issue the following command:

```
rpm -qi ups
```

Universal Agent for SOA for UNIX - File Inventory Lists

- Overview
- Parent Directories
- Product Directories and Files
  - /opt/universal Parent Directory
  - /var/opt/universal Parent Directory
  - /etc/universal Parent Directory

Overview

This page identifies the Universal Agent for SOA file system hierarchy and its contents.

The parent directories under which Universal Agent for SOA operates are based on the existingUniversal Agent deployment. There should be no product components in these directories, just the product directories.

The product directories contain the product components either directly or in sub directories and are divided into two categories: runtime and logging.

- Runtime directories are for runtime operation of the product; they are not written to.
- Logging directories are where database components, audit and logging files reside, all components that are written to.

Parent Directories

For UNIX, the parent directories are:

- **/opt/universal**
- **/var/opt/universal**

The following sections identify the directories and files located under each of these parent directories.

Product Directories and Files

/opt/universal Parent Directory

The following table identifies the Universal Agent for SOA for UNIX product directories and files located under the **/opt/universal** parent directory.

Directory / File	Sub-Directory / File	Description
uac		Directory containing artifacts for the UAC component. It includes the following sub-directories and files.
	bin	Sub-directory containing the **uacsrv** executable. It requires no user interaction.
	lib	Sub-directory containing the **uacsrv.jar** file needed for communication between the **uacsrv** and **uac** components. It requires no user interaction.
	container	Sub-directory containing the libraries and other deployable objects needed for UAC operation. The only reason to explore this directory is if you are running the JMS Connector and need to deploy your JMS provider client jar files.
	shutdownUAC.sh	Shutdown script for UAC. You usually will not need to use this script, as it is the responsibility of Universal Broker to shut down UAC.
	startUAC.sh	Start-up script for UAC. You usually will not need to use this script, as it is the responsibility of Universal Broker to start UAC.
	uacValidateInbound.sh	Script that validates the contents of the UAC.xml file using the schema definition contained in **UAC.xsd**.
	UAC.xsd	File used by UAC to validate content of the **UAC.xml** inbound configuration file. It requires no user interaction.
uai		Directory containing the artifacts for the UAI component. It includes the following sub-directories and files.

	lib	Sub-directory containing the libraries needed for UAI operation. It usually requires no user interaction.
	UAI.sh	Shell script used to invoke the UAI component. It usually requires no user interaction.
jre		Directory containing the Java Runtime Environment for the UAC, UAI, and UAC Server components. It usually requires no user interaction.
nls/uacmceng.umc		Universal Application Container Server English message catalog.
Readme.ups		Universal Agent for SOA for UNIX installation README file.
ubroker/tmpl/uaccfg		Universal Application Container Server configuration template file.
ubroker/tmpl/uaccmp		Universal Application Container Server component definition template file.

/var/opt/universal Parent Directory

The following table identifies the Universal Agent for SOA for UNIX product directories and files located under the **/var/opt/universal** parent directory.

Directory	Sub-Directory / File	Description
log/uac		Directory containing logging and work artifacts for the UAC component. s; It includes the following sub-directories and files.
	temp	Sub-directory used by UAC for temporary file persistence. It requires no user interaction.
	work	Sub-directory used by the web services framework for temporary operations. It requires no user interaction.
	catalina.out	Container engine log file.
	container.log	Web services framework log file.
	derby.log	Database log file.
	uac.log	UAC log file.
log/uai		Directory containing the logging and work artifacts for the UAI component. It is empty by default. If **trace** or **info** is enabled, the output would be written to this directory.
uac_db		Directory containing the database components, log, property, and lock files. It requires no user interaction.

/etc/universal Parent Directory

The following table identifies the Universal Agent for SOA for UNIX files located under the **/etc/universal** parent directory.

File	Description
uac_log4jConfiguration.xml	Logger configuration file that you may need to modify if you want to change the logging level of the UAC component.
uai_log4jConfiguration.xml	Logger configuration file that you may need to modify if you want to change the logging level of the UAI component.
uacs.conf	Universal Application Container Server configuration file.
UAC.xml	Inbound operation configuration file that you may need to modify to support your inbound workload operations.
comp/uac	Universal Application Container Server component definition file.

UNIX Installation - Licensing

- Licensing Universal Agent for UNIX Components
- Product License File
  - Format
  - Sample
- Entering License Information
- Restart Universal Broker

Licensing Universal Agent for UNIX Components

After Universal Agent for UNIX has been installed, you must configure the following components with product licenses before they can be used:

- Universal Command Manager
- Universal Data Mover Manager
- Universal Connector
- Universal Application Container Server

Note
Universal Application Container Server (as a component of Universal Command Agent for SOA 6.2.x for UNIX) is packaged, and licensed, separately.

Product License File

For each component, product license information (license parameter keywords and their values) is contained in a separate text file provided by your Stonebranch, Inc. account representative.

Format

The format of the product license file name is: <*component name*>_<*customer name*>_<*operating system*>_<*schedule or solution*>.txt. For example: **Indesca_Stonebranch_UNIX_A1.txt**

- For Universal Command Manager, **Indesca** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.
- For Universal Data Mover Manager, **Infitran** is used as the <*component name*> in the product license file name and as the name of the product in the file itself.

Sample

The following is a sample Universal Command Manager for UNIX product license file:

```
License_Product "INDESCA"
License_Customer "STONEBRANCH"
License_OS_Type "UNIX"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

Enter each component's product license file information into its configuration file:

- Universal Command Manager: **ucmd.conf**
- Universal Data Mover Manager: **udm.conf**
- Universal Connector: **usap.conf**

- Universal Application Container Server: **uacs.conf**

It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration options.)

Restart Universal Broker

For Universal Broker to read the license information, you must stop and restart it:

Stop Universal Broker	`ubrokerd stop`
Start Universal Broker	`ubrokerd start`

IBM i Installation

- Introduction
- Installation Summary
- Naming Conventions
- Detailed Information

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Introduction

These pages provide information on the installation of Stonebranch Inc's Workload Automation 5 on the IBM i operating system. Unless otherwise specified, all references to Workload Automation 5 for IBM i in these pages refer to version 5.1.0.

All Workload Automation 5 for IBM i components are provided in an easily installed, one-time installation package.

Note
Starting with the 3.2.0 release of Universal Products, a Universal Broker must run on all systems on which a Workload Automation 5 component is running, including manager components. The local Broker maintains product configuration data for all components that have a configuration file.

Installation Summary

Warning
This installation procedure is intended only for default installations into simple environments. Installing into high availability environments may require a customized installation. For customized installation, parallel installation or multiple system roll out see the Installation Guide for IBM i.

Step 1	Download the distribution file from the Stonebranch Customer Portal to a computer running a supported version of Windows or UNIX.
Step 2	Extract the Licensed Program Save File from the distribution file.
Step 3	If transferring the installation file to the native, QSYS file system, create a save file, UNV510, on the target IBM I system in library QGPL. This step is not necessary if transferring the file to the IFS.
Step 4	FTP the Licensed Program Save File to either the save file created in step 1 or to /QSYS.LIB/QGPL.LIB/UNV510.SAVF.
Step 5	Execute the command RSTLICPGM LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/UNV510) to install Workload Automation 5 to the default libraries.
Step 6	Verify the correctness of the install by executing the following command: CHKPRDOPT OUNV510.
Step 7	Optionally, rename the UNVUBR510 subsystem to UBROKER for consistent operation across future installations.

Naming Conventions

In Workload Automation 5 for IBM i, some library names and object names include a **vrm** (version, release, and modification) suffix, **510**. This **vrm** suffix will change, as appropriate, for subsequent releases.

Additionally, some library names can be changed. These pages use the default names for these libraries:

- **UNVPRD510** (product library)
- **UNVTMP510** (temporary library)
- **UNVSPL510** (spool file library)
- **UNVCMDREF** (command reference library)

You can change the **UNVPRD510**, **UNVTMP510**, and **UNVSPL510** library names at installation time.

You can change **UNVCMDREF** only via the Universal Command Server configuration file.

Detailed Information

The following pages provide detailed information for IBM i Installation:

- Installation Package
- Installation Requirements
- Distribution File
- Transferring to IBM i
- Installation Procedures
- Customization
- Object Inventory Lists
- Licensing

IBM i Installation - Installation Package

- Package Components
- Component Compatibility

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Package Components

The Workload Automation for IBM i package includes the following Workload Automation components:

- Universal Broker 5.1.0
- Universal Command (Manager and Server) 5.1.0
- Universal Control (Manager and Server) 5.1.0
- Universal Copy 5.1.0
- Universal Data Mover (Manager and Server) 5.1.0
- Universal Display Log File for AS/400 log files 5.1.0
- Universal Encrypt 5.1.0
- Universal Message Translator 5.1.0
- Universal Query 5.1.0
- Universal Submit Job with z/OS WTO support 5.1.0

Workload Automation for IBM i is packaged with product identifier **0UNV510**. The product can be managed using the IBM i commands for working with licensed programs (such as **RSTLICPGM**, **SAVLICPGM**, and **DLTLICPGM**).

Note
For the 5.1.0 release, Universal Command and Universal Encrypt are packaged as one IBM i licensed program.

Component Compatibility

The following table identifies the compatibility of Workload Automation for IBM i with previous component / product versions.

Component	Compatibility
Universal Broker 5.1.0	Stonebranch Solutions / Universal Products 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Command 5.1.0	Universal Command 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Control 5.1.0	Universal Control 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Copy 5.1.0	Universal Copy 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Data Mover 5.1.0	Universal Data Mover 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Display Log File for IBM i log files 5.1.0	Universal Display Log File for OS/400 log files 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.

Universal Encrypt 5.1.0	Universal Encrypt 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Message Translator 5.1.0	Universal Message Translator 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Query 5.1.0	Universal Broker 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.
Universal Submit Job with z/OS WTO support 5.1.0	Universal Submit Job with z/OS WTO support 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, 3.1.0, and 1.2.1.

The component references pertain to all support platforms for that version.

IBM i Installation - Installation Requirements

- System Requirements
- Platform Requirements
- Libraries and Logs
  - Universal Broker Job Log
  - Command Reference Library
  - Trace File Location Library
  - Spool Library

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

System Requirements

The requirements for installation of Workload Automation 5 for IBM i are:

- IBM i V5R4M0 or above.
- TCP/IP.
- User profile with *ALLOBJ, *SPLCTL, *JOBCTL, and *SECADM authorities.
- About 360 megabytes of disk space.
- Network-attached workstation.
- PTF SI27629 installed on V5R4 systems.
  For additional information, see the documentation for APAR SE28859 and PTF SI27629 at both:
  - https://www-912.ibm.com/n_dir/nas4apar.nsf/aaf5d88f9cc2ee1086257102005863 5c/a2b5a3b3ac874a9d862572d8003c7457?Op
  - https://www-912.ibm.com/a_dir/as4ptf.nsf/c2fd98f5d2eccb83862574ce00520341/7529dd654f63941b862572eb0058109

The Workload Automation 5 for IBM i installation process creates a product user profile, **UNVUBR510**, that is given ***ALLOBJ** special authority. **UNVUBR510**, with ***ALLOBJ** special authority, is required to successfully complete the installation.

Note
Some organizations and companies require the removal of ***ALLOBJ** authority from non-administrative user profiles. Workload Automation may be configured to run without ***ALLOBJ** authority; however, to do so requires additional administrative overhead. The steps required to remove product ***ALLOBJ** authority are described in Universal Broker Security.

The ***BASE** option contains the complete installation. This includes all of the components and utilities listed above.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

Libraries and Logs

Under the IBM i native file system, Workload Automation write to product files residing in various libraries and write information to job logs.

This section specifies the following information for various libraries and logs:

- Estimated amount of space required
- Required security access
- Location of the production files

Universal Broker Job Log

Under IBM i, Universal Broker writes its messages to the **UBROKER** job log.

Space

Job log file growth is dependent on use of the Workload Automation Servers. The disposition and size of the job log depends on the job definition

as well as system variables **QJOBMSGQMX** (maximum job log size) and **QJOBMSGQFL** (action when job log is full).

Security

Since Universal Broker uses a normal job log, no special security is required.

Command Reference Library

Universal Command Server can execute commands of type **cmdref**. A command reference is a predefined command or script to which the Universal Command Manager refers by its file name.

The default command reference library name is **UNVCMDREF**. For security reasons, the name of this library can be changed only via the Universal Command Server configuration, located in the **UNVCONF** file and **UCMDS** member. The configuration file entry has precedence over the default value. If the name is changed to something other than the default or configured value, Universal Command Manager will no longer be able to read the command reference files; it will generate appropriate error messages if an attempt is made to do so.

Space

The amount of space is dependent solely on the number of command reference files defined.

Security

Universal Command Server requires read access to the **UNVCMDREF** library files. Administrator accounts require appropriate access in order to maintain the command reference files. No general user access is required.

Trace File Location Library

Universal Broker and its server components (for example, Universal Command Server) create product trace files when configured to do so. A trace file is used by Stonebranch, Inc. Customer Support to resolve product problems.

On IBM i, Workload Automation trace files normally are written to ***CURLIB** (current library). Under Universal Broker and the associated servers, the current library is the temporary library designated at installation time (**UNVTMP510**).

Otherwise, current library is the default current library of the user.

Space

Trace files can grow to significant size, depending on how long the trace is active and how much work the program is doing during the tracing period. Due to the information associated with IBM i pointers and fact that physical database files are fixed record lengths, trace files quickly can grow very large on a system with high Universal Broker and server activity.

If the trace file size is increased beyond 500,000 records, the maximum file size must be changed or the associated job will hang. The hang results from a system generated inquiry message, which is issued when the maximum file size is exceeded. By default, trace files wrap before reaching the maximum file size, thus avoiding the system inquiry message.

The MAX_TRACE_LINES configuration option sets the number of records at which the trace file wrapping occurs. When the maximum size is reached, the trace file will wrap to the beginning.

No space is required under normal operation for trace files. Trace files are requested by Stonebranch, Inc. Customer Support only for problem resolution. When trace files are required, at least 40MB of disk space should be available.

Security

The **UNVUBR510** user profile requires at least ***CHANGE** authority to the **UNVTMP510** library to create and use the Universal Broker and server trace files. No general user access is required.

Spool Library

The spool library is used to store the following types of information:

- Execution information for Workload Automation components started by Universal Broker, **UBR_CMP_DB**.
- Universal Command Server status, **SRV_CMP_DB**.
- Redirected standard I/O files (stdin, stdout, and stderr) captured by Universal Command when run with manager fault tolerance enabled.

Spool files are stored in the **UNVSPL510** library.

Space

The spool files are located in the **UNVSPL510** library. The amount of disk space required for the spool directory depends on these factors:

1. Number of spooling user processes that will be executing simultaneously.
   A user process is created for each command requested by a Universal Command Manager. The default maximum number is 50. Once a user process ends and a Manager has received all the spool files, the spool files themselves are deleted.
2. Average number of records of the user processes for standard input, output, and error files.
   Each record is 32764 bytes in length. Keep in mind that spooling is not intended as a feature for file transfer purposes. File transfer related processes should execute without spooling enabled.

Once these numbers are determined, the average amount of disk space is calculated with the following formula:

```
MAX-PROCESSES x AVERAGE NUMBER-OF-REC0RDS x 32764 = required disk space.
```

As an example, if the maximum number simultaneous user processes is estimated at 20 and the average number of records in files is 1,000 bytes, the average amount of required disk space is 655MB (20 x 1000 x 32764).

The Universal Command Server is configured with spooling disabled to prevent unintentional disk utilization. The feature must be turned on through the ALLOW_SPOOLING configuration settings.

Further details on the Manager Fault Tolerant feature and the spooling of redirected standard I/O files can be found in the Universal Agent 6.2.x User Guide.

Security

The **UNVUBR510** user profile requires at least ***CHANGE** authority to the **UNVSPL510** library to create and use the spool files. No general user access is required.

No other Workload Automation 5 for IBM i components access the spool library.

IBM i Installation - Distribution File

- IBM i Distribution File
- Obtaining the Distribution File
- Distribution File Format
  - Full Release
  - Maintenance Release
- Distribution File Contents
  - Full Release
  - Maintenance Release

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

IBM i Distribution File

The Workload Automation for IBM i product distribution file contains all of the files required for the installation of the Workload Automation for IBM i package.

Obtaining the Distribution File

To obtain the Workload Automation for IBM i package, you must download the corresponding product distribution file from the To download a product distribution file:

Step 1	Log in to the Stonebranch Customer Portal.
Step 2	Click the **Software** link.
Step 3	Click the **Universal Agent** link.
Step 4	Click the Universal Agent package link appropriate for your platform.
Step 5	Click **Save File** and browse to your save location. You can then install the software. .

Note
A customer user name and password — provided by Stonebranch, Inc. — are required to access the Customer Portal.

Distribution File Format

The format of the Workload Automation for IBM i distribution file name is different for a full release than it is for a maintenance release.

Full Release

For a full release installation, the product distribution file name has the following format:

`sb-Version.Release.Modification Level.Maintenance Level-operating system-version.release.tar.Z`

For example: **`sb-5.1.0.0-as400-5.4.tar.Z`**

In this format:

- **`Version`** is the current version of Workload Automation.
- **`Release`** is the current release of Workload Automation.
- **`Modification Level`** is the current Workload Automation feature set.
- **`Maintenance Level`** is the Workload Automation build level.
- **`operating system`** is the name of the operating system.
- **`version.release`** is the minimum supported version and release of the operating system.

Maintenance Release

For a maintenance release installation, the product distribution file name has a modified format:

- **`-ptfs`** is included after the **`operating system`** name.
- **`version.release`** is not included after the **`operating system`** name.

For example: **`sb-5.1.0.1-as400-ptfs.tar.Z`**

(See Transferring for a Maintenance Release for information about PTFs in maintenance release distribution.)

Distribution File Contents

The distribution file contains the installation files required for the installation of Workload Automation for IBM i.

The Workload Automation Licensed Program for IBM i is distributed as an IBM i Save File.

Depending on whether the installation is for a full release or a maintenance release of Workload Automation for IBM i, the distribution file contains different installation files.

Full Release

The following table lists the installation files included in the distribution file for a full release of Workload Automation for IBM i.

File Name	Description
UNV510.SAVF	Workload Automation Licensed Program Save File.
README.TXT	Summary of the installation procedure.

Maintenance Release

The following table lists the installation files included in the distribution file for a maintenance release of Workload Automation for IBM i.

File Name	Description
CUMUNV510	Workload Automation Cumulative PTF Save File.
README.TXT	Summary of the installation procedure.

IBM i Installation - Transferring to IBM i

- Workload Automation for IBM i - Transferring Workload Automation to IBM i
- Transferring for a Full Release
- Transferring for a Maintenance Release

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Workload Automation for IBM i - Transferring Workload Automation to IBM i

In order to install Workload Automation for IBM i, you must first transfer the Workload Automation Licensed Program for IBM i to the IBM i operating system.

The transfer is comprised of three procedures:

1. Download the product distribution file from the Stonebranch Customer Portal (see IBM i Installation - Distribution File) to Windows, UNIX, or IBM i shell with tar and zcat utilities.
2. Extract all files from the distribution file.
3. Transfer the IBM i Save File (the Workload Automation Licensed Program) to a library on an iSeries.

This page provides separate transferring procedures for:

- Transferring for a Full Release
- Transferring for a Maintenance Release

Transferring for a Full Release

To transfer a full release package to IBM i — whether for a new installation of a full release or an upgrade installation to a new release — perform the following steps:

Step 1	Download the distribution file to a computer running a supported version of Windows or UNIX.
Step 2	Extract the installation files from the distribution file: • To extract the installation files on UNIX, QShell, or PASE shell, run: `zcat *.Z | tar xvf -` **Note** The space and the **-** character after **xvf** must be included. • To extract the installation files on Windows, use a utility capable of extracting files from a **.Z** file. The files will be extracted to the specified location.
Step 3	Create a save file on the target IBM i system in library **QGPL**. On the IBM i command line, execute the following command: `CRTSAVF FILE(QGPL/UNV510)`

Step 4	If needed, FTP the SAVF file extracted in Step 2 to the save file created in Step 3.
	
	Note
	The following example assumes that the SAVF file resides in: **`c:\temp\Workload Automation for as400`**
	
	`ftp your_as400`
	`Name: your_name`
	`Password:  your_password`
	`ftp> cd QGPL`
	`ftp> bin`
	`ftp> lcd "c:\temp\Workload Automation for as400"`
	`ftp> put UNV510.SAVF UNV510`
	`ftp> quit`

Transferring for a Maintenance Release

To upgrade Workload Automation from a currently installed 5.1.0 release to 5.1.0 maintenance release, cumulative PTFs are used. These cumulative PTFs are distributed as IBM i Save Files and can be downloaded from the Stonebranch Customer Portal. PTFs should be installed as user **QSECOFR** or a user with equivalent authority.

To transfer a maintenance release package to IBM i, perform the following steps:

Step 1	Download the distribution file to a computer running a supported version of Windows or UNIX.
Step 2	Extract the distribution file: • To extract the ***.tar.Z** file on UNIX, run: `zcat *.Z | tar xvf -` **Note** The space and the **-** character after **xvf** must be included. • To extract the distribution file for Windows, use a utility capable of extracting files from a **.Z** file. The files will be extracted to the specified location.
Step 3	Create a save file on the target IBM i system in library **QGPL**. On the IBM i command line, execute the following command: `CRTSAVF FILE(QGPL/CUMUNV510)`

Step 4	FTP the **.SAVF** file extracted in Step 2 to the save file created in Step 3.
	For example:
	
	`ftp your_as400`
	`Name: your_name`
	`Password:  your_password`
	`ftp> cd QGPL`
	`ftp> bin`
	`ftp> lcd c:\temp`
	`ftp> put CUMUNV510 CUMUNV510`
	`ftp> quit`

IBM i Installation - Installation Procedures

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Installation Processes

There are six different processes for installing Workload Automation on an IBM i system:

1. New Installation (Default)
2. New Installation (Custom)
3. Re-Installation of Same Release
4. Upgrade Installation to New Release
5. Propagating New Release to Additional Systems
6. Upgrade Installation for Maintenance Release

Your installation environment, currently installed Workload Automation 5 for IBM i release (if any), and Workload Automation 5 for IBM i release to be installed determine the process to use.

There are two additional procedures related to installation:

- UCHGRLS (Change Release Tag) Program for changing Workload Automation for IBM i command names in the IBM i **QSYS** library.
- Product Removal procedures that are required or optional, depending on the installation process.

IBM i Installation - New Installation (Default)

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

New Installation (Default)

The default process for a new installation of Workload Automation for IBM i installs to the following libraries:

- Product library (**UNVPRD510**)
- Temporary library (**UNVTMP510**)
- Spool file library (**UNVSPL510)**
- Command reference library (**UNVCMDREF**)

V5R4 systems
Before installing Workload Automation for IBM i, either download and apply PTF SI27629 or verify that it is applied to the system.

Step 1	Execute the following command to install Workload Automation to the default libraries: `RSTLICPGM LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/UNV510)`
Step 2	Verify that the installation was correct by executing the following command: `CHKPRDOPT 0UNV510` The following message should display: **No errors detected by CHKPRDOPT.**
Step 3	Optionally, rename the UNVUBR510 subsystem to UBROKER for consistent operation across future installations.
Step 4	Optionally, use the UCHGRLS (Change Release Tag) Program to change the names of the Workload Automation for IBM i commands in the IBM i **QSYS** library.

IBM I Installation - New Installation (Custom)

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

New Installation (Custom)

The custom method for a new installation of Workload Automation for IBM i allows installation to libraries other than **UNVPRD510**, **UNVTMP510**, and **UNVSPL510**, as required by your environment.

V5R4 systems
Before installing Workload Automation for IBM i, either download and apply PTF SI27629 or verify that it is applied to the system.

Step 1	Execute the following command to install Workload Automation to user-specified libraries: `RSTLICPGM LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/UNV510) LIB(prodLib tmpLib spoolLib)` **Note** Replace **prodLib** with the desired product library name, **tmpLib** with the desired temporary library name, and **spoolLib** with the desired spool library name.
Step 2	Verify that the installation was correct by executing the following command: `CHKPRDOPT 0UNV510` The following message should display: **No errors detected by CHKPRDOPT.**
Step 3	Optionally, rename the UNVUBR510 subsystem to UBROKER for consistent operation across future installations.
Step 4	Optionally, use the UCHGRLS (Change Release Tag) Program to change the names of the Workload Automation for IBM i commands in the IBM i **QSYS** library.

IBM I Installation - Re-Installation of Same Release

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Re-Installation Using Same Libraries

If you are reinstalling a Workload Automation for IBM i release (5.1.0 and later) into the same libraries, you do not have to remove the currently installed release.

Note
Re-installing to the same libraries will preserve configuration and other user-modified files, as well as objects created as part of the installation process. The re-installation process will replace programs, commands, etc.

Re-installation to the same libraries can be performed without deleting the current release.

Step 1	Back up all file objects in all of the Workload Automation libraries - product, temporary, spool, and command reference - prior to re-installation (in case errors occur during the re-installation process).
Step 2	End the Universal Broker subsystem, **UNVUBR510** (**UBROKER** if renamed).
Step 3	Run **RSTLICPGM** to install Workload Automation using the library names used during the initial 5.1.0 installation.
Step 4	After installation is complete, check the configuration and component files to ensure that the re-installation process preserved all changes previously made. Restore any files not correctly preserved from the backed up file objects.
Step 5	Restart **UNVUBR510**.
Step 6	Make sure that previous events needed by the customer are displayed via the I-Activity Monitor and that any disconnected jobs, as well as any appropriate MFT spool files, are correctly displayed using the Universal Query. If problems exist, restore the spool library from the backed up file objects.
Step 7	If **UCHGRLS** was previously run to remove 510 tags from the Workload Automation commands in **QSYS**, manually delete the newly created, 510-tagged Workload Automation commands in **QSYS**. A list of those commands can be found in QSYS Library. **Note** Do not remove the **UNVUBR510** user profile from **QSYS**.

Re-Installation Using Different Library Names

Before reinstalling a Workload Automation for IBM i release (5.1.0 and later) into different libraries, you must remove the currently installed release.

Step 1	Save copies of all configuration files and component definition files; otherwise, the configuration and component information will be lost.
Step 2	Remove the currently installed release (see Product Removal).
Step 3	Re-install the 5.1.0 release. See New Installation (Default) and New Installation (Custom) to determine which process is most appropriate for your environment.
Step 4	Merge the saved configuration into the newly installed configuration files manually. Conversely, the saved configuration files can be copied in place of the reinstalled configuration files.
Step 5	Optionally, use the UCHGRLS (Change Release Tag) Program to change the names of the Workload Automation for IBM i commands in the IBM i **QSYS** library.

IBM i Installation - Upgrade Installation to New Release

- Upgrade Procedures
  - Print Files Removal
  - Currently Installed Release
- Install (Default or Custom)
- Post-Install Tasks
- Pre-Production Tasks
  - Change Release Tags
  - Change Port Address

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Upgrade Procedures

To upgrade Workload Automation for IBM i from a currently installed pre-5.1.0 release (Stonebranch Solutions release 4.3.0 or earlier) to a 5.1.0 release, the following procedures are required:

1. Install (Default or Custom)
2. Post-Install Tasks
3. Pre-Production Tasks

V5R4 systems
Before installing Workload Automation for IBM i, either download and apply PTF SI27629 or verify that it is applied to the system.

Print Files Removal

In releases 4.1.0 - 4.3.0, Job Log Copy Keep used the **USBMJPRTF** and **UCMSVRPRTF** print files.

Starting with release 5.1.0, these print files are no longer used and will be removed during the installation process. The spooled job logs will not be lost because they are spooled to the **QEZJOBLOG** output queue.

To print job logs from a previous release, use the **CHGSPLFA** command to associate the job logs with a different printer file.

Currently Installed Release

You are not required to delete a currently installed pre-5.1.0 release (Stonebranch Solutions 4.3.0 or earlier) of Workload Automation for IBM i before installing a 5.1.0 release. You can do so either before or after installing the 5.1.0 release (see Product Removal).

However, if you want to install a 5.1.0 release to the previously used installation libraries of a pre-5.1.0 release, you must first uninstall that pre-5.1.0 release:

Step 1	Save configuration, component, NLS, and any other files from the product library (**UNVPRD*** / **UNIVERSAL**) that may contain information needed after the new installation.
Step 2	Save any files from the command reference library (**UNVCMDREF)** and temporary library (**UNVTMP*** / **UNVTMP**) that are needed, as these libraries will be deleted by the uninstall process.
Step 3	Follow the release-appropriate uninstall process (see Product Removal) and the New Release installation procedure (see Install (Default or Custom), below, to install the new release.
Step 4	After the new installation is complete, merge required configuration and any other saved information as well as any needed files and/or members from the previous release into the new release.

Install (Default or Custom)

The processes (default or custom) for the upgrade installation to a new Workload Automation release (5.1.0) are the same as the processes for a new installation.

See New Installation (Default) and New Installation (Custom) to determine which process is most appropriate for your environment.

Post-Install Tasks

After the installation upgrade of Stonebranch Solutions 4.3.0 (or previous release) to Workload Automation, perform the following steps:

Step 1	Copy any customized code page files from the **UNVNLS** file members from the previous release to the newly installed **UNVNLS** file. **Note** Do not simply copy the old **UNVNLS** file to replace the new **UNVNLS** file. The new files contain many updates.
Step 2	If needed, manually update the newly installed configuration file, **UNVCONF**, from the file of the previous release. The new file contains many new configuration options. At a minimum, the Workload Automation license entries must be entered into the configuration file members for the managers of each licensed product.
Step 3	If the original component file (**UNVCOMP**) was modified, update the newly installed **UNVCOMP** file for each of the modified members.
Step 4	If either or both of the exit programs (**UCMSJOBI** and **UCMSJOBT**) in the previous release had been customized, you likewise must customize the newly installed exit programs. The exit programs are located in the product library, **UNVPRD510**. **Note** You cannot replace **UCMSJOBI** for 5.1.0 with programs of the same name from previous releases without a possible conflict. If you must use older programs because their source is no longer available, rename them and call them from the new exit program. See **UCMSJOBI** in the Universal Command 5.1.0 Reference Guide for additional information.

Pre-Production Tasks

Change Release Tags

When you are ready to take the new release to production, you can use the **UCHGRLS** (Change Release Tag) program to:

- Rename the Workload Automation for IBM i commands in **QSYS** to the untagged command names in **UNVPRD510**.
- Tag the pre-5.1.0 release command names in **QSYS** with the version / release / modification number of that release (for example, **320**).

See UCHGRLS (Change Release Tag) Program for information on using **UCHGRLS**.

Change Port Address

No two Workload Automation installed under the same IBM-i can have the same port number associated with the Universal Broker subsystem (**UBROKER**). Change the default port address, 7887, to another port number for all but one of the **UBROKER** subsystems. Workload Automation components and utilities intended to use the altered port number must specify a port number in the commands or program calls. Whether or not a command or program call is used depends on the platform from which the request is sent.

Note
Do not replace a configuration file with one from a different release.

IBM I Installation - Propagating New Release to Additional Systems

- Propagating New Release to Additional Systems
- SAVLICPGM and RSTLICPGM Method

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Propagating New Release to Additional Systems

You may want to install Workload Automation on a pre-production system, implement needed changes such as customizing configuration files, and then propagate these changes to other systems.

One method to propagate the customized product is via the **SAVLICPGM** and **RSTLICPGM** commands (see SAVLICPGM and RSTLICPGM Method, below).

Warning
A number of Stonebranch objects are created during the installation process. These include, but are not limited to:

- Commands in QSYS
- UBROKER job queue and class
- Job definitions in the product library
- Universal Broker Subsystem
- Print files in the product library.

The default command reference library, **UNVCMDREF**, is created during installation unless a library of that name exists on the system. This process only recreates the objects previously identified. It will not propagate changes made to those objects.

V5R4 systems
Before installing Workload Automation for IBM i, either download and apply PTF SI27629 or verify that it is applied to the system.

SAVLICPGM and RSTLICPGM Method

The following steps are recommended:

Step 1	Install Workload Automation (see New Installation (Default) and New Installation (Custom)).
Step 2	Download the latest product maintenance from the Stonebranch Customer Portal and temporarily apply the PTFs. Allow the system to automatically handle any permanent PTF application via the **APYPTF** command. See Upgrade Installation for Maintenance Release for detailed instructions on applying a maintenance release.
Step 3	Customize the Workload Automation installation as described in Customization.
Step 4	Verify that the 5.1.0 Universal Broker Subsystem, **UNVUBR510**, is not active.
Step 5	To create a save file in which to save the customized product, invoke: `CRTSAVF FILE(QGPL/UNV510CUST) TEXT('Customized installation file for SB Prod 5.1.0')`

Step 6	Invoke: `SAVLICPGM LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/UNV510CUST)`
Step 7	If needed, restart the Universal Broker Subsystem: `STRSBS UNVPRD510/UNVUBR510`
Step 8	Via what ever mechanism you prefer, move the save file to each system on which you want to restore the product.
Step 9	Invoke: `RSTLICPGM LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/UNV510CUST)`

Note
Although this procedure describes use of a save file, any other available save and restore device can be used.

IBM i Installation - Upgrade Installation for Maintenance Release

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Upgrade Installation for Maintenance Release

To upgrade Workload Automation for IBM i from a currently installed release to a maintenance release, perform the following steps to install the cumulative PTFs:

Step 1	If it is running, end the Universal Broker subsystem (**UNVUBR510**) on the IBM i system. On the IBM i, enter the following command: `===> ENDSBS SBS(UNVUBR510)`
Step 2	Load the PTF. On the IBM i, enter the following command: `===> LODPTF LICPGM(0UNV510) DEV(*SAVF) SAVF(QGPL/CUMUNV510)`

Step 3	Apply the PTF.
	On the IBM i, enter the following command:
	`===> APYPTF LICPGM(0UNV510)`
	This command applies all PTFs contained in the cumulative PTF file.
	If you want to apply only selected PTFs, enter the following command:
	`===>APYPTF LICPGM(0UNV510) SELECT(ptf1 ptf2 ...)`
	This command applies the PTF temporarily.
	If you want to apply the PTFs permanently after successful testing, re-execute the command with the parameter **APY(*PERM)**.

Step 4	Restart **UNVUBR510**.

IBM i Installation - UCHGRLS (Change Release Tag) Program

- Overview
- Using UCHGRLS
- UCHGRLS Examples

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Overview

All installations of Workload Automation for IBM i, except an upgrade for maintenance, place the Workload Automation for IBM i commands in the product library (**UNVPRD510**) and the IBM i **QSYS** library.

Starting with the 3.2.0 release of Universal Products for IBM i, the command names in **QSYS** are tagged with the Workload Automation for IBM i version / release / modification number.

To maintain consistency across releases, you may prefer to use untagged names in your production environment. The **UCHGRLS** (Change Release Tag) program lets you change command names in the IBM i **QSYS** library from tagged to untagged.

Using UCHGRLS

UCHGRLS resides in **UNVPRD510**. It provides two parameters, **SETTAGVRM** and **RMVTAGVRM**.

- **SETTAGVRM** specifies the release number of the release to be tagged.
- **RMVTAGVRM** specifies the release number of the tag to be removed.

Removing the tags changes the command names in **QSYS** to the normal production command names. (See UCHGRLS Examples, below.)

For a new installation, specify the following **UCHGRLS** command options and their values:

- Set tags (**SETTAGVRM**) option; value = ***NEW**.
- Remove tags (**RMVTAGVRM**) option; value = **510**.

This changes the tagged command names in **QSYS** to the untagged command names in **UNVPRD510**.

For an upgrade installation (from a pre-5.1.0 release to 5.1.0), **UCHGRLS** tags the pre-5.1.0 release command names in **QSYS** with the version / release / modification number of that release (for example, **510**). See Object Inventory Lists for tagged command name details.

UCHGRLS Examples

This example runs **UCHGRLS** with the default values, following installation of **0UNV510** with **0UNV410** existing on the system.

```
UNVPRD510/UCHGRLS
```

This example runs **UCHGRLS** to revert back to **0UNV320** for the default names in **QSYS** with **0UNV510** installed. (This should be done prior to re-installing **0UNV510**, assuming that **0UNV320** remains installed on the system.)

```
UNVPRD510/UCHGRLS SETTAGVRM(510) RMVTAGVRM(320)
```

This example runs **UCHGRLS** to change the **0UNV510** commands in **QSYS** to the default names and to tag the **0UCM121** and **0UEN121** commands in **QSYS** with 121.

```
UNVPRD510/UCHGRLS SETTAGVRM(121) RMVTAGVRM(510)
```

This example runs **UCHGRLS** to set the **0UNV510** commands in **QSYS** to the default names, where **0UNV510** is the first Workload Automation installation on an IBM i system.

```
UNVPRD510/UCHGRLS SETTAGVRM(*NEW) RMVTAGVRM(510)
```

IBM i Installation - Product Removal

- Overview
- Universal Encrypt 1.2.1
- Universal Command 1.2.1
- Universal Products 3.1.1
- Universal Products 3.2.0
- Universal Products 4.1.0
- Stonebranch Solutions 4.2.0
- Stonebranch Solutions 4.3.0
- Workload Automation 5.1.0

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Overview

This page provides processes for the removal of current (5.1.0) and previous (Stonebranch Solutions 4.3.0 or earlier) releases of Workload Automation for IBM i.

If you remove the currently installed release before installing or re-installing a 5.1.0 release, the removal process deletes the default command reference library, **UNVCMDREF**, in which the command reference files are located.

If you remove the currently installed release after installing or re-installing a 5.1.0 release, **UNVCMDREF** is not deleted.

To keep your command reference files from being deleted, rename **UNVCMDREF** and change the library owner. Release 5.1.0 will let you specify this library - or any library - as your command reference library via the CMD_REFERENCE_DIRECTORY option.

Before removing a 5.1.0 release, check CMD_REFERENCE_DIRECTORY in the Universal Command server member (**UCMDS**) of the Workload Automation configuration file (**UNVCONF**) - located in the product library (**UNVPRD510**); an alternate command reference library previously may have been selected. If it has, manually clear and delete the default command reference library, **UNVCMDREF**, after removing the 5.1.0 release.

Warning
If the **UCHGRLS** command was used to rename the 510-tagged names to normal production names (see UCHGRLS (Change Release Tag) Program), you must perform the following steps prior to uninstalling either 5.1.0, 4.3.0, 4.2.0, 4.1.0, 3.2.0, 3.1.1, or 1.2.1:

Step 1	Run the **UCHGRLS** command to temporarily rename the 5.1.0 Workload Automation commands in **QSYS** to the 510-tagged names and the tagged names of the older product to the normal production names. Without this step, Workload Automation may no longer run as expected.
Step 2	After removal of the older product is complete, use the ***NEW** option with the **UCHGRLS** command to change the 510-tagged names back to the normal, production names.

Universal Encrypt 1.2.1

Note
Before performing this uninstall, read the Overview.

You must uninstall Universal Encrypt 1.2.1 before uninstalling Universal Command 1.2.1.

The user ID used for uninstalling Universal Encrypt must have authority to delete all Universal Encrypt product-related objects.

To uninstall Universal Encrypt, perform the following steps:

Step 1	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUEN121)`
Step 2	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUEN121 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 1.

Universal Command 1.2.1

> **Note**
> Before performing this uninstall, read the Overview.

Before uninstalling Universal Command 1.2.1, you must uninstall Universal Encrypt 1.2.1.

The user ID used for uninstalling Universal Command must have *ALLOBJ and *SECADM special authorities.

To uninstall Universal Command, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UBROKER**). On the IBM i command line, execute the following command: `===>ENDSBS UBROKER`
Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUCM121)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUCM121 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

Universal Products 3.1.1

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Universal Products 3.1.1 must have *ALLOBJ and *SECADM special authorities.

To uninstall Universal Products 3.1.1, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UBROKER**). On the IBM i command line, execute the following command: `===>ENDSBS UBROKER`

Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV311)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUNV311 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

Universal Products 3.2.0

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Universal Products 3.2.0 must have *ALLOBJ and *SECADM special authorities.

To uninstall Universal Products 3.2.0, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UNVUBR320**). On the IBM i command line, execute the following command: `===>ENDSBS UNVUBR320`
Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV320)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUNV320 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

Universal Products 4.1.0

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Universal Products 4.1.0 must have *ALLOBJ and *SECADM special authorities.

To uninstall Universal Products 4.1.0, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UNVUBR410**). On the IBM i command line, execute the following command: `===>ENDSBS UNVUBR410`

Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV410)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: {**Objects for product OUNV410 option *ALL release *ONLY not deleted."**}} Correct the problem and repeat Step 2.

Stonebranch Solutions 4.2.0

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Stonebranch Solutions 4.2.0 must have *ALLOBJ and *SECADM special authorities.

To uninstall Stonebranch Solutions 4.2.0, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UNVUBR420**). On the IBM i command line, execute the following command: `===>ENDSBS UNVUBR420`
Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV420)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUNV510 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

Stonebranch Solutions 4.3.0

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Stonebranch Solutions 4.3.0 must have *ALLOBJ and *SECADM special authorities.

To uninstall Stonebranch Solutions 4.3.0, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UNVUBR430**). On the IBM i command line, execute the following command: `===>ENDSBS UNVUBR430`

Step 2	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV430)`
Step 3	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUNV430 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

Workload Automation 5.1.0

> **Note**
> Before performing this uninstall, read the Overview.

The user ID used for uninstalling Workload Automation 5.1.0 must have *ALLOBJ and *SECADM special authorities.

To uninstall Workload Automation 5.1.0, perform the following steps:

Step 1	If it is active, end the Universal Broker subsystem (**UNVUBR510**). On the IBM i command line, execute the following command: `===>ENDSBS UNVUBR510`
Step 1	To remove the licensed program, execute the following command: `===>DLTLICPGM LICPGM(OUNV510)`
Step 1	If the following message displays, use the **DSPJOBLOG** command to identify the problem: **`Objects for product OUNV510 option *ALL release *ONLY not deleted."`** Correct the problem and repeat Step 2.

IBM i Installation - Customization

- Workload Automation for IBM i - Customization
- Modify the IPL Start-up Program
- Multiple-Installation Implementation
  - Universal Broker
  - Active Jobs
- Universal Broker Customization
  - Configuration
  - System Initialization
- Universal Command Manager Customization
  - Configuration
- Universal Command Server Customization
  - Configuration
- Universal Control Manager Customization
  - Configuration
- Universal Control Server Customization
  - Configuration
- Universal Data Mover Manager Customization
  - Configuration
- Universal Data Mover Server Customization
  - Configuration
- Universal Query Customization
  - Configuration
- National Language Customization

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Workload Automation for IBM i - Customization

(For information on applying product licenses to installed Workload Automation 5 for IBM i components, see IBM i Installation - Licensing.)

Modify the IPL Start-up Program

Modify the IPL Start-up Program so that the Universal Broker subsystem **(UNVUBR510)** is started at IPL time.

The subsystem start command is:

```
QSYS/STRSBS UNVPRD510/UNVUBR510
```

The IPL Start-up program name is displayed by:

```
DSPSYSVAL SYSVAL(QSTRUPPGM)
```

Refer to the AS/400 publication AS/400e series: Basic System Operation, Administration, and Problem Handling for complete details on modifying the IPL Start-up program.

Multiple-Installation Implementation

Workload Automation for IBM i provides the ability to install, configure, and test a new release prior to placing it into production.

You can keep an older release available during this initial production phase in case problems occur. This allows you to manually merge component files, configuration files, and exit programs without having to save copies of those files and programs prior to installation of the newer

release.

When initially installed, the names of the Workload Automation commands in the **QSYS** library are tagged with the *v*ersion / *r*elease / *m*odification number, **510**. This renaming allows currently installed pre-5.1.0 releases to remain fully functional without modifying production code until pre-production testing is completed.

When you are ready to bring the newly installed release into production, run the UCHGRLS (Change Release Tag) Program to untag the 5.1.0 command names and tag the pre-5.1.0 command names with that pre-5.1.0 release number.

After running **UCHGRLS**, check the job log for errors because missing objects do not result in program termination. There will generally be one or more missing commands flagged; only those commands included with the installed release present a problem if they are missing.

Universal Broker

For the Universal Broker, both a job queue (***JOBQ**) and a class (***CLS**) were added; they are both named **UBROKER** and are located in the product library, **UNVPRD510**.

The **UNVUBR510** subsystem and Workload Automation job descriptions use the **UNVPRD510** / **UBROKER** class and job queue instead of the **QGPL** / **QBATCH** class and job queue.

To revert back to the **QGPL** / **QBATCH** class:

Step 1	Invoke the CHGRTGE command using SBSD(**UNVPRD510** / **UNVUBR510**), SEQNBR(**10**), and CLS(**QGPL** / **QBATCH**).
Step 2	Invoke the CHGPJE command using SBSD(**UNVPRD510** / **UNVUBR510**), PGM(**QSYS** / **QP0ZSPWP**), JOB(**UCMSINIT**), and CLS(**QGPL** / **QBATCH**).
Step 3	Invoke the CHGPJE command using SBSD(**UNVPRD510** / **UNVUBR510**), PGM(**QSYS** / **QP0ZSPWT**), JOB(**UNVSRV**), and CLS(**QGPL** / **QBATCH**).
Step 4	Use the RMVJOBQE command with SBSD(**UNVPRD510** / **UNVUBR510**) and JOBQ(**UNVPRD510** / **UBROKER**) to remove the job queue entry from the **UNVUBR510** subsystem.

To revert back to the **QGPL/QBATCH** job queue, invoke the CHGJOBD command specifying JOBQ(**QGPL** / **QBATCH**) for the **UNVPRD510** / **UBROKER**, **UNVPRD510** / **UCMSINIT**, and **UNVPRD510** / **UNVSRV** job descriptions.

Active Jobs

Whereas the **QGPL** / **QBATCH** job queue entry for the **QBASE** subsystem limits the maximum active jobs to only six (by default), the **UNVUBR510** / **UBROKER** job queue has no limit. Thus, it allows many additional Workload Automation jobs to run in parallel. Using the new settings, the Universal Broker RUNNING_MAX configuration option controls the maximum number of components allowed to run simultaneously.

For consistency with earlier releases, the initial settings of the **UNVUBR510** class is the same as the **QGPL** / **QBATCH** class. Using the new class and job description, you may better tune your systems without affecting those jobs running under the default **QGPL** / **QBATCH** class and job description.

Universal Broker Customization

Configuration

Configuration options for Universal Broker are stored in configuration file **UNVCONF**(**UBROKER**), in product library **UNVPRD510**.

See the Universal Broker 6.2.x Reference Guide for details on configuring Universal Broker.

The file can be edited using the Source Entry Utility (SEU).

System Initialization

Universal Broker runs as a job, UBROKER, in the Universal Broker subsystem (**UNVUBR510**) located in product library **UNVPRD510**.

To start Universal Broker, enter the following on the command line: **STRSBS UNVPRD510/UNVUBR510**. There are two sets of pre-start jobs initiated: **UNVSRV** and **UCMSINIT**:

- **UNVSRV** is the pre-start job for any of the Workload Automation servers.
- **UCMSINIT** is the pre-start job for the **UCMSINIT** program.

The Universal Command Server initiates **UCMSINIT**, which initiates and monitors the command originated from either the Universal Command Manager or the **exec** command under the Universal Data Mover Manager.

Universal Command Manager Customization

Configuration

Configuration options for Universal Command Manager are stored in its configuration file, member **UNVCONF**(**UCMD**) in product library **UNVPRD510**.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Manager.

The file can be edited using the Source Entry Utility (SEU).

Universal Command Server Customization

Configuration

Configuration options for Universal Command Server are stored in configuration file **UNVCONF**(UCMDS), in product library **UNVPRD510**.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Server.

Universal Command Server runs as a component managed by Universal Broker. It provides a component definition file that Universal Broker uses to start the Server and establish its runtime environment. The component definition file, **UNVCOMP**(**UCMD**), is located in product library **UNVPRD510**.

Universal Command Server uses the Universal Access Control List (UACL) configuration file as a level of product security. The UACL file is located in **UNVCONF**(**UACL**), in product library **UNVPRD510**. See the Universal Agent Security for information on how Universal Command Server utilizes the UACL file.

The file can be edited using the Source Entry Utility (SEU).

Universal Control Manager Customization

Configuration

Configuration options for Universal Control Manager are stored in configuration file **UNVCONF**(**UCTL**), in product library **UNVPRD510**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Manager.

The file can be edited using the Source Entry Utility (SEU).

Universal Control Server Customization

Configuration

Configuration options for Universal Control Server are stored in configuration file **UNVCONF**(**UCTLS**), in product library **UNVPRD510**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Server.

Universal Control Server runs as a component managed by Universal Broker. It provides a component definition file that Universal Broker uses to start the Server and establish its runtime environment. Component definition file **UNVCOMP**(**UCTL**) is located in library **UNVPRD510** by default

Universal Control Server uses the Universal Access Control List (UACL) configuration file as a level of product security. The UACL file **UNVCONF** (**UACL**) is located in product library **UNVPRD510**. See the Universal Agent Security for information on how Universal Control Server utilizes the UACL file.

The file can be edited using the Source Entry Utility (SEU).

Universal Data Mover Manager Customization

Configuration

Configuration options for Universal Data Mover Manager are stored in its configuration file, member **UNVCONF**(**UDM**) in product library **UNVPRD510**.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Data Mover Manager.

The file can be edited using the Source Entry Utility (SEU).

Universal Data Mover Server Customization

Configuration

Configuration options for Universal Data Mover Server are stored in configuration file **UNVCONF**(**UDMS**), in product library **UNVPRD510**.

See the Universal Data Mover 6.2.x Reference Guide for details on configuring Universal Data Mover Server.

Universal Data Mover Server runs as a component managed by Universal Broker. It provides a component definition file that Universal Broker uses to start the Server and establish its runtime environment. Component definition file **UNVCOMP(UDM)** is located in product library **UNVPRD510**.

Universal Data Mover Server uses the Universal Access Control List (UACL) configuration file as a level of product security. The UACL file is located in **UNVCONF**(**UACL**) in product library **UNVPRD510**. See the Universal Agent Security for information on how Universal Data Mover Server utilizes the UACL file.

The file can be edited using the Source Entry Utility (SEU).

Universal Query Customization

Configuration

Configuration options for Universal Query are stored in configuration file **UNVCONF**(**UQRY**), in product library*UNVPRD510*.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Query.

The file can be edited using the Source Entry Utility (SEU).

National Language Customization

The default IBM i CCSID value (QCCSID) is pre-set to 65535 or *HEX, both of which mean: Do not interpret data unless specifically required. You can set QCCSID to a value consistent with your country CCSID.

However, Workload Automation configuration files contain non-invariant characters (such as #, which marks the start of a comment). Some CCSIDs, such as 278, result in the translation of these characters into characters that the Workload Automation software cannot recognize. As a result, Workload Automation components, including Universal Broker, fail to start.

To resolve this, either:

- Use a Universal Broker job CCSID of 65535. Both the Workload Automation configuration files and the internal Universal Broker code are CCSID 037-based. Start Universal Broker and check the job log output to verify the correct translation.
- Use 037 as the CCSID for the Universal Broker user profile (**UNVUBR510**). This sets the CCSID of the associated Universal Broker job to 037. The job's 37 CCSID forces compatibility with the configuration files and the internal program CCSID value of 037, which is consistent with standard IBM i development practice.

The **CRTPGM** command sets the program CCSID to 65535, which allows the program to inherit the job CCSID. Writing to the job log should result in the correct translation of characters between the job CCSID and the system CCSID.

IBM i Installation - Object Inventory Lists

- Workload Automation for IBM i Objects
- Installation Libraries
  - QSYS Library Command Names
- Source File Record Lengths
- Product Library (UNVPRD510)
- Temporary Library (UNVTMP510)
- Spool Library (UNVSPL510)
- Command Reference Library (UNVCMDREF)
- QSYS Library

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Workload Automation for IBM i Objects

The Workload Automation for IBM i installation includes the objects required for:

- Universal Broker
- Universal Command Manager and Server
- Universal Control Manager and Server
- Universal Data Mover Manager and Server
- Universal Encrypt

Installation Libraries

These objects are contained in four user-designated installation libraries:

- Product library **UNVPRD510** (formerly **UNIVERSAL**)
  Library containing all Workload Automation for IBM i products.
- Temporary library **UNVTMP510** (formerly **UNVTMP**)
  Library used as a temporary work area by Workload Automation. It may contain such items as trace files and temporary job log files.
- Spool library **UNVSPL510** (formerly **UNVSPOOL**)
  Library containing a spool database of all Workload Automation spool files. The database is allocated the first time that Universal Broker is started.
- Command reference library **UNVCMDREF**
  Library containing user-defined command references

In addition, an installation places the following in the IBM i **QSYS** library:

- Copy of the Workload Automation for IBM i commands contained in **UNVPRD510**.
- Universal Broker user profile, **UNVUBR510**.

QSYS Library Command Names

The names of the commands in **QSYS** are tagged with the Workload Automation for IBM i version / release / modification number, **510**. However, since command names can have a maximum nine characters, the following names are shortened:

- **UDSPLOGF** becomes **UDSPLF510**.
- **USBMJOB** becomes **USBMJ510**.
- **UMSGHNDLR** becomes **UMSGH510**.
- **UJOBINIT** becomes **UJOBI510**.

Note
For further information on library and object names, see Naming Conventions.

Source File Record Lengths

Workload Automation for IBM i source file record lengths must be a maximum 240 characters in order to use the Source Edit Utility, SEU.

The configuration file (**UNVCONF**), component file (**UNVCOMP**), template file (**UNVTMPL**), National Language Support file (**UNVNLS**), and the source files - all of which are located in the Workload Automation installation library (**UNVPRD510**, by default) - are editable by SEU as they are initially installed.

Product Library (UNVPRD510)

Name	Type	Description
PRCDSPLOGF	*PGM	Universal Display Log File program.
PUMSGHNDLR	*PGM	Universal Submit Job, WTO msg callback program.
STRUCP	*PGM	UCOPY command program.
STRUEN	*PGM	Universal Encrypt command program.
STRUME	*PGM	UMET command program.
UBROKER	*PGM	Universal Broker program.
UCMCPP	*PGM	UCMD Manager command processing program.
UCHGRLS	*PGM	Universal Change Release program.
UCMD	*PGM	UCMD Manager program.
UCMDEF01	*PGM	UCMD object definition program.
UCMDEXIT01	*PGM	UCMD install exit program.
UCMMSG01	*PGM	UCMD message definition program.
UCMSINIT	*PGM	UCMD Server Initiator program.
UCMSJOBI	*PGM	UCMD Server Initiator job initialization program.
UCMSJOBT	*PGM	UCMD Server Initiator job termination program.
UCMSRV	*PGM	Universal Command Server program.
UCOPY	*PGM	Universal Copy program.

UCTCPP	*PGM	UCTL Manager command processing program.
UCTL	*PGM	Universal Control Manager program.
UCTSRV	*PGM	Universal Command Server program.
UDM	*PGM	Universal Data Mover Manager program.
UDMCPP	*PGM	UDM Manager command processing program.
UDMSRV	*PGM	UDM Server program.
UDSPLOGF	*PGM	Universal Display Log File cmd processing program.
UENCRYPT	*PGM	Universal Encrypt program.
UJOBINIT	*PGM	USBMJOB initialization program.
ULSTSECPP	*PGM	Universal Spool List utility command processing program.
UMET	*PGM	Universal Message Translator program.
UMSGHNDLR	*PGM	Universal Submit Job, PUMSGHNDLR interface.
UQRCPP	*PGM	Universal Query command program.
UQUERY	*PGM	Universal Query program.
URMVSECPP	*PGM	Universal Spool Remove utility command processing program.
USBMJOB	*PGM	Universal Submit Job program.
USBMUSRJOB	*PGM	Universal Submit User Job program.
USLIST	*PGM	Universal Spool List Job program.
USLRM	*PGM	Universal Spool Remove Job program.

UCMINST	*MSGF	UCMD install message file.
UNVMSG	*MSGF	Workload Automation message file.
UNVSEQMSGF	*MSGF	Universal Products Sequential Message File.
USBMJOB	*MSGF	Universal Submit Job message file.
CP2CCSID_X	*FILE	Example code page to CCSID mapping file.
INSTL.INFO	*FILE	Installation Information file for library locations.
UCMNLSTMP	*FILE	Temporary National Language Support file.
UCMSVRPRTF	*FILE	Universal Command server job log print file.
UNVCLSRC	*FILE	CL source.
UNVCOMP	*FILE	Universal component definition members.
UNVCONF	*FILE	Universal configuration members.
UNVDDSSRC	*FILE	DDS Src for Stonebranch Products.
UNVNLS	*FILE	National Language Support.
UNVTMPL	*FILE	Universal template members.
USBMJPRTF	*FILE	USBMJOB job log print file.
UBROKER	*JOBD	Job Definition for Universal Broker.
UCMSINIT	*JOBD	Job Definition for UCMD Server Initiator.
UNVSRV	*JOBD	Job Definition for Universal Servers.
UBROKER	*JOBQ	Universal Broker job queue.
UBROKER	*CLS	Universal Broker class.

STRUCM	*CMD	Universal Command Manager command.
STRUCP	*CMD	Universal Copy command.
STRUCT	*CMD	Universal Control Manager command.
STRUDM	*CMD	Universal Data Mover Manager command.
STRUEN	*CMD	Universal Encrypt command.
STRUME	*CMD	Universal Message Translator command.
STRUQR	*CMD	Universal Query command.
UCHGRLS	*CMD	Universal Change Release command.
UDSPLOGF	*CMD	Universal Display Log File command.
UJOBINIT	*CMD	USBMJOB initialization program command.
ULSTSE	*CMD	Universal Spool List utility command.
UMSGHNDLR	*CMD	WTO message callback program command.
URMVSE	*CMD	Universal Spool Remove utility command.
USBMJOB	*CMD	Universal Submit Job command.
UNVUBR510	*SBSD	Subsystem Definition for Universal Broker
STRUCM	*PNLGRP	UCMD Manager panel definition.
STRUCP	*PNLGRP	UCOPY panel definition.
STRUCT	*PNLGRP	UCTL Manager panel definition.
STRUDM	*PNLGRP	UDM Manager panel definition.
STRUEN	*PNLGRP	Universal Encrypt panel definition.

STRUME	*PNLGRP	UMET panel definition.
STRUQR	*PNLGRP	Universal Query panel definition.
UCHGRLS	*PNLGRP	Universal Change Release panel definition.
UDSPLOGF	*PNLGRP	Universal Display Log File panel definition.
UJOBINIT	*PNLGRP	USBMJOB initialization program panel definition.
ULSTSE	*PNLGRP	Universal Spool List utility panel definition.
UMSGHNDLR	*PNLGRP	WTO message callback program panel definition.
URMVSE	*PNLGRP	Universal Spool Remove utility panel definition.
USBMJOB	*PNLGRP	Universal Submit Job panel definition.
UNV510	*PRDDFN	Workload Automation product definition.
UNV510	*PRDLOD	Workload Automation product load.

Temporary Library (UNVTMP510)

Name	Type	Description
UCMDEF02	*PGM	UNVTMP object definition program.
UCMDEXIT02	*PGM	UNVTMP install exit program.

Spool Library (UNVSPL510)

Name	Type	Description
UNVDEFSPL	*PGM	**UNVSPL510** object definition program.
UNVEXITSPL	*PGM	**UNVSPL510** install exit program.

Command Reference Library (UNVCMDREF)

Name	Type	Description
UNVCRFSRC	*FILE	Universal Command command reference file.

QSYS Library

Name	Type	Description
STRUCM510	*CMD	Universal Command Manager command.
STRUCP510	*CMD	Universal Copy command.
STRUCT510	*CMD	Universal Control Manager command.
STRUDM510	*CMD	Universal Data Mover Manager command.
STRUEN510	*CMD	Universal Encrypt command.
STRUME510	*CMD	Universal Message Translator command.
STRUQR510	*CMD	Universal Query command.
UDSPLF510	*CMD	Universal Display Log File command.
UJOBI510	*CMD	Universal Job Initializer command.
ULSTSE510	*CMD	Universal Spool List utility command.
UMSGH510	*CMD	Universal Message Handler command.
URMVSE510	*CMD	Universal Spool Remove utility command.
USBMJ510	*CMD	Universal Submit Job command.
UNVUBR510	*USRPRF	Universal Broker user profile.

IBM i Installation - Licensing

- Licensing Workload Automation 5 for IBM i Components
- Product License File
  - Format
  - Sample
- Entering License Information
- Restart Universal Broker

Currently, IBM i runs Workload Automation 5.1.0. These pages provide information for that version.

Licensing Workload Automation 5 for IBM i Components

After Workload Automation 5 for IBM i has been installed, you must configure the following components with product licenses before they can be used:

- Universal Command Manager
- Universal Data Mover Manager

Product License File

Product license information (license parameter keywords and their values) is contained in a text file provided by your Stonebranch, Inc. account representative.

Format

The format of the product license file name is: *<component name>_<customer name>_<operating system>_<schedule or solution>*.txt. For example: **Indesca_Stonebranch_OS400_A1.txt**.

- For Universal Command Manager, **Indesca** is used as the *<component name>* in the product license file name and as the name of the product in the file itself.
- For Universal Data Mover Manager, **Infitran** is used as the *<component name>* in the product license file name and as the name of the product in the file itself.

Sample

The following is a sample Universal Command Manager for IBM i product license file:

```
License_Product "INDESCA"
License_Customer "STONEBRANCH"
License_OS_Type "OS400"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

In the **UNVPRD510** product library:

- Enter the Universal Command Manager license parameters into the Universal Command Manager configuration file, member **UNVCONF**(**UCMD**).
- Enter the Universal Data Mover Manager license parameters into the Universal Data Mover Manager configuration file, member **UNVCONF**(**UDM**).

It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration options.)

Restart Universal Broker

For Universal Broker to read the license information, you must stop and restart it:

Stop Universal Broker	`ENDSBS UBROKER *CNTRLD` (`*CNTRLD` performs a controlled shutdown.)
Start Universal Broker	`STRSBS UBROKER`

HP NonStop Installation

- Introduction
- Installation Summary
- Detailed Information

Currently, HP NonStop runs Universal Command 2.1.1. These pages provide information for that version.

Introduction

These pages provide information on the installation of Stonebranch, Inc.'s Universal Command 2.1.1 on the HP NonStop operating system.

All Universal Products 2.1.1 for HP NonStop (see HP NonStop Installation Package) are included as part of Universal Command 2.1.1.

Installation Summary

Step 1	Download the distribution file from the Stonebranch Customer Portal.
Step 2	Extract the HP NonStop installation files from the windows self-extracting executable.
Step 3	Transfer the extracted files to the HP NonStop Guardian environment using ftp.
Step 4	Logon to the Guardian environment of the HP NonStop system as super.super.
Step 5	Alter the file code of the installation script and set its file code to 180.
Step 6	Convert the installation script to a Guardian EDIT style file.
Step 7	Run the installation obey script.

Detailed Information

The following pages provide detailed information for HP NonStop Installation:

- HP NonStop Components
- Installation Requirements
- Distribution File
- Installation Procedures
- Customization
- File Inventory Lists
- Licensing

HP NonStop Installation - Installation Package

Components

The Universal Command 2.1.1 for HP NonStop installation package includes the following Universal Products components:

- Universal Broker 2.1.1
- Universal Command Manager and Server 2.1.1
- Universal Control Manager and Server 2.1.1
- Universal Query 2.1.1
- Universal Encrypt 2.1.1
- Universal Message Translator 2.1.1

Product Compatibility

The following table identifies the compatibility of Universal Command 2.1.1 for HP NonStop with previous product versions.

Product	Compatibility
Universal Command 2.1.1	Universal Command 2.2.0, 2.1.0, and 1.2.0.

HP NonStop Installation - Installation Requirements

- Installation Requirements
- Platform Requirements
- $SYSTEM Volume
  - Log Directory
  - Trace Directory

Installation Requirements

Universal Command 2.1.1 installation requirements for HP NonStop are:

- HP NonStop system:
  - HP NSK S-series (running the G06.13 or greater version of the OS)
  - HP Integrity (running the H06 OS)
- Open System Services (OSS) configured and running. The TACL command TESTOSS can be used to check the OSS configuration on the HP NonStop.
- TACL shell.
- Open System Services Local Services (OSSLS).
- TCP/IP Socket implementation.
- One available TCP/IP port.
- Approximately 20 megabytes of disk space for the installation. More disk space is required for variable files, such as, log files and trace files.
- super.super access.

Note
In order to install Universal Command 2.1.1, you must be able to write to the directory from which the installation is launched.

Platform Requirements

Since platform requirements may change with new releases of a product, please consult the Platform Support for Universal Controller 6.2.x and Universal Agent 6.2.x page to make sure that your platform is supported before performing an installation.

$SYSTEM Volume

All product files that are written to during product execution are stored in the **$SYSTEM** volume. The following sections describe the space and security requirements for all of the subdirectories.

Log Directory

Universal Broker can be configured to write its messages to a log file. The current log file and previous log file generations are stored in the **$SYSTEM.UNVLOG** subvolume.

Space

A log file size grows to about 150,000 bytes and is then rolled over to a generation file. Five generations of log files are saved. The oldest generation log file is deleted. The amount of space required for the five generations and the current log file is about 900,000 bytes.

Security

Universal Broker requires read/write access to the log subvolume and read/write access to all files in the log subvolume. No other Universal Products use the log subvolume at this time. No general user access is required.

Trace Directory

Universal Broker and its server components (for example, Universal Command Server) create product trace files when configured to do so. A trace file is used by Stonebranch, Inc. Customer Support to resolve product problems. The trace files are stored in the **$SYSTEM.UNVTRACE** subvolume.

Space

Trace files can grow to significant size depending on how long the trace is active and how much work the program is doing during the tracing period. A trace file size of about 10MB is not unusual.

No space is required under normal operation for trace files. Trace files are requested by Stonebranch, Inc. Customer Support only for problem resolution. When trace files are required, at least 20MB of disk space should be available.

Security

Universal Broker and the Broker components (Universal Command Server and Universal Control Server) require read/write access to the trace subvolume. No other Universal Products access the trace subvolume. No general user access is required.

HP NonStop Installation - Distribution File

- HP NonStop Distribution File
- Obtaining the Distribution File
- NSK S-series System
  - Extracting the Installation Files
- Integrity System
  - Extracting the Installation Files
- Distribution File Contents

HP NonStop Distribution File

The product distribution file contains the installation files required for the installation of Universal Command 2.1.1 for HP NonStop.

Obtaining the Distribution File

To obtain the distribution file, you must download it from the Stonebranch Customer Portal.

> **Note**
> A customer user ID and password - provided by Stonebranch, Inc. - are required to access this area.

Stonebranch, Inc. provides a separate product distribution file for the following HP NonStop systems:

- NSK S-series system
- Integrity system

NSK S-series System

The Universal Command 2.1.1 for HP NonStop (NSK S-series systems) product distribution file is in a Windows, self-extracting archive file format.

The name of the distribution file has the following format:

`ucmd-Version.Release.Modification Level.Maintenance Level-operating system-platform.exe`

In this format:

- **`Version`** is the current version of Universal Products.
- **`Release`** is the current release of Universal Products.
- **`Modification Level`** is the current Universal Products feature set.
- **`Maintenance Level`** is the Universal Products build level.
- **`operating system`** is the name of the operating system.
- **`platform`** is the targeted hardware platform.

For example, the name of the distribution file for Universal Products 2.1.1 Level 2, HP NonStop version NSK, release G06 would be:

`ucmd-2.1.1.2-tandem-NSK-G06.exe`

Extracting the Installation Files

To extract the installation files from the distribution file, simply execute the file.

Integrity System

The Universal Command 2.1.1 for HP NonStop (Integrity systems) product distribution file is in a compressed **tar** format.

The name of the distribution file has the following format:

`ucmd-Version.Release.Modification Level.Maintenance Level-operating system-platform.tar.Z`

In this format:

- **`Version`** is the current version of Universal Products.

- **`Release`** is the current release of Universal Products.
- **`Modification Level`** is the current Universal Products feature set.
- **`Maintenance Level`** is the Universal Products build level.
- **`operating system`** is the name of the operating system.
- **`platform`** is the targeted hardware platform.

For example, the name of the distribution file for Universal Products 2.1.1 Level 3, HP NonStop version NSK, release H06 would be:

`ucmd-2.1.1.3-tandem-NSK-H06.tar.z`

Extracting the Installation Files

To uncompress and extract the installation files from the distribution file, issue the following command:

```
zcat ucmd-2.1.1.3-tandem-NSK-H06.tar.z | tar xvf -
```

This command assumes that:

- Name of the distribution file is **`ucmd-2.1.1.3-tandem-NSK-H06.tar.z`**.
- File is located in the current working directory.

Distribution File Contents

The following table lists the installation files included in the distribution file for Universal Command 2.1.1 for HP NonStop (NSK S-series and Integrity systems).

File Name	Description
Readme.unv	Summary of the installation procedure.
ucmdins	Installation script.
ucmdpkg	Installation package file.

If your Universal Command 2.1.1 for HP NonStop distribution file does not contain these files, contact Stonebranch, Inc. Customer Support to obtain a correct distribution file.

HP NonStop Installation - Installation Procedures

Universal Command for HP NonStop - Installation

Installation of Universal Command 2.1.1 for HP NonStop is performed with the Universal Command installation script, **ucmdinst**.

To install Universal Command, perform the following steps:

Step 1	Transfer the installation files (in a BINARY transfer mode, not ASCII) to the HP NonStop operating system.

Step 2	To start the install as user **super.super**, the installation script file (**ucmdins**) must be converted to a Guardian EDIT file by using the **ctoedit** function.
	
	Execute the following commands from the subvolume in which the files are located:
	
	`fup alter ucmdins, code 180`
	
	`ctoedit ucmdins, ucmdinst`
	
	This prepares the script to be used with the **obey** command.

Step 3	To start the installation, execute the following **obey** command: `obey ucmdinst` This copies the files to the proper locations, after which the system will be ready to configure.

HP NonStop Installation - Customization

- Introduction
- Universal Broker Customization
  - Configuration
  - System Initialization
- Universal Command Manager Customization
  - Configuration
- Universal Command Server Customization
  - Configuration
- Universal Control Manager Customization
  - Configuration
- Universal Control Server Customization
  - Configuration
- Universal Query Customization
  - Configuration

Introduction

This page provides information on customizing the Universal Products components that comprise Universal Command 2.1.1 for HP NonStop.

(For information on applying product licenses to installed Universal Command 2.1.1 for HP NonStop components, see HP NonStop Installation - Licensing.)

Universal Broker Customization

Configuration

Configuration options for Universal Broker are stored in configuration file, **UBRCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Broker 6.2.x Reference Guide for details on configuring Universal Broker.

Once installed, the **`$SYSTEM.UNVBIN.UBROKERD`** startup script should be modified and the appropriate time zone set for the TZ variable.

TZ Environment Variable Values

Value	Description
North America	
EST5EDT	Eastern Standard Time, Eastern Daylight Time
CST6CDT	Central Standard Time, Central Daylight Time
MST7MDT	Mountain Standard Time, Mountain Daylight Time
PST8PDT	Pacific Standard Time, Pacific Daylight Time
AKST9AKDT	Alaska Standard Time, Alaska Daylight Time
Europe	
GMT0BST	Greenwich Mean Time, British Summer Time
WET0WEST	Western Europe Time, Western Europe Summer Time
CET-1CEST	Central Europe Time, Central Europe Summer Time
EET-2EES	Eastern Europe Time, Eastern Europe Summer Time

Note
Other common time zone abbreviations can be used; all possible values are not shown.

System Initialization

A Broker daemon start-up script is provided in file **$SYSTEM.UNVBIN.UBROKERD**. A single command line argument — either **-start**, **-stop**, **-status**, or **-restart** — instructs the script on the action to take.

See the Universal Agent 6.2.x User Guide for details on the Universal Broker daemon script.

Universal Command Manager Customization

Configuration

Configuration options for Universal Command Manager are stored in configuration file, **UCMDCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Manager.

The product executable files intended for command line use are located in the subvolume **$SYSTEM.UNVBIN**.

Universal Command Server Customization

Configuration

Configuration options for Universal Command Server are stored in configuration file, **UCMDSCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Command 6.2.x Reference Guide for details on configuring Universal Command Server.

Universal Command Server runs as a component managed by Universal Broker. Universal Command Server provides a component definition file that Universal Broker uses to start the Server and establish its runtime environment. Component definition file UCMD is located in subvolume **$SYSTEM.UNVCOMP**.

The product executable files intended for command line use are located in the subvolume **$SYSTEM.UNVBIN**.

Universal Command Server uses the Universal Access Control List (UACL) configuration file as a level of product security. See the Universal Command 6.2.x Reference Guide for information on how Universal Command Server utilizes the UACL file.

Universal Control Manager Customization

Configuration

Configuration options for Universal Control Manager are stored in configuration file, **UCTLCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Manager.

The product executable files intended for command line use are located in the subvolume **$SYSTEM.UNVBIN**.

Universal Control Server Customization

Configuration

Configuration options for Universal Control Server are stored in configuration file, **UCTLSCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Control Server.

Universal Control Server runs as a component managed by Universal Broker. Universal Control Server provides a component definition file that Universal Broker uses to start the Server and establish its runtime environment. Component definition file UCTL is located in subvolume **$SYSTEM.UNVCOMP**.

The product executable files intended for command line use are located in subvolume **$SYSTEM.UNVBIN**.

Universal Control Server uses the Universal Access Control List (UACL) configuration file as a level of product security.

See the Universal Agent 6.2.x User Guide for information on how Universal Control Server utilizes the UACL file.

Universal Query Customization

Configuration

Configuration options for Universal Query are stored in configuration file, **UQRYCFG**, in subvolume **$SYSTEM.UNVCONF**.

See the Universal Agent Utilities 6.2.x Reference Guide for details on configuring Universal Query.

The product executable files intended for command line use are located in the subvolume **$SYSTEM.UNVBIN**.

HP NonStop Installation - File Inventory Lists

- File Inventory Lists
- Universal Broker
- Universal Command Manager
- Universal Command Server
- Universal Control Manager
- Universal Control Server
- Universal Query

File Inventory Lists

The Universal Command 2.1.1 for HP NonStop installation includes the files required for the following Universal Products components:

- Universal Broker
- Universal Command Manager and Server
- Universal Control Manager and Server
- Universal Query

This section identifies the files installed with each component.

Universal Broker

File	Description
$SYSTEM.UNVBIN.UBROKERD	Broker daemon start script.
$SYSTEM.UNVBIN.UBROKER	Console broker program.
$SYSTEM.UNVBIN.UBRD	Daemon broker program.
$SYSTEM.UNVCOMP	Component definition directory.
$SYSTEM.UNVCOMP.UCMD	Universal Command Server component definition file.
$SYSTEM.UNVCOMP.UCTL	Universal Control Server component definition file.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVLOG	Broker message log directory.
$SYSTEM.UNVTRACE	Broker trace file directory.
$SYSTEM.UNVCONF.UBRCFG	Broker configuration file.

Universal Command Manager

File	Description
$SYSTEM.UNVBIN.UCMD	Universal Command Manager program.
$SYSTEM.UNVBIN.UENCRYPT	Universal Encrypt program file.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVCONF.UCMDCFG	Manager configuration file.

Universal Command Server

File	Description
$SYSTEM.UNVBIN.UCMSRV	Server component program.

$SYSTEM.UNVBIN.UCOPY	Utility used for binary file copies. Similar to the UNIX cat command.
$SYSTEM.UNVBIN.UMET	Universal Message Translator program.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVTRACE	Server trace file directory.
$SYSTEM.UNVCONF.UCMDSCFG	Server configuration file.

Universal Control Manager

File	Description
$SYSTEM.UNVBIN.UCTL	Manager program.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVCONF.UCTLCFG	Manager configuration file.

Universal Control Server

File	Description
$SYSTEM.UNVBIN.UCTSRV	Server component program.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVTRACE	Server trace file directory.
$SYSTEM.UNVCONF.UCTLSCFG	Server configuration file.

Universal Query

File	Description
$SYSTEM.UNVBIN.UQUERY	Universal Query program.
$SYSTEM.UNVNLS	Code page files used for text translation between different operating systems and platforms and product message catalogs.
$SYSTEM.UNVCONF.UQRYCFG	Query configuration file.

HP NonStop Installation - Licensing

- Licensing Universal Command 2.1.1 for HP NonStop Components
- Product License File
  - Format
  - Sample
- Entering License Information

Licensing Universal Command 2.1.1 for HP NonStop Components

After Universal Command 2.1.1 for HP NonStop has been installed, you must configure the following component with a product license before it can be used:

- Universal Command Manager

Product License File

Product license information (license parameter keywords and their values) is contained in a text file provided by your Stonebranch, Inc. account representative.

Format

The format of the product license file name is: *<component name>_<customer name>_<operating system>_<schedule or solution>*.txt. For example: **Universal Command_Stonebranch_Tandem_A1.txt**

Sample

The following is a sample Universal Command Manager for HP NonStop product license file:

```
License_Product "UNIVERSAL COMMAND"
License_Customer "STONEBRANCH"
License_OS_Type "Tandem"
License_Type "PERPETUAL"
License_Expiration_Date 2029.12.31          YYYY.MM.DD
License_NT_Servers 100
License_UNIX_Servers 100
License_OS400_Servers 10000
License_OS390_Servers 10000
License_Tandem_Servers 10000
License_OS390_Unix_Servers 10000
License_Key ABCD-1234-EFGH-5678-IJKL-MNOP-9999
```

Entering License Information

In subvolume **$SYSTEM.UNVCONF**, enter the Universal Command Manager license parameters into the Universal Command Manager configuration file, member **UCMDCFG**

It is recommended that you enter license information at the end of the file. (The values are specified in the same syntax as all other configuration options.)

Note
You do not have to restart Universal Broker in order for it to read the license information.

Verifying Universal Agent Installation

Verifying Agent Installation

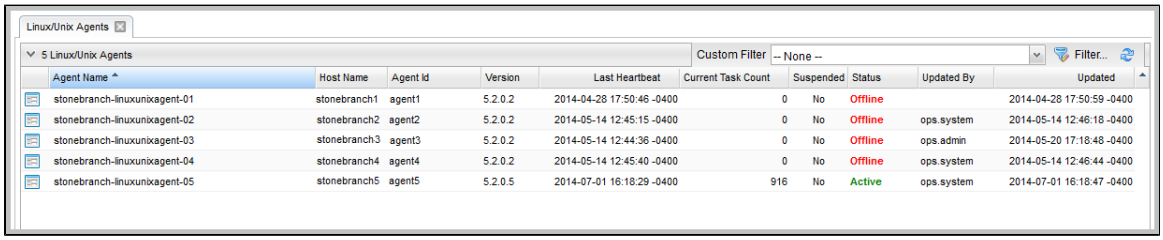
If an installed Agent is configured to communicate with Universal Controller, you can verify that it is installed, running, and communicating with the Controller by performing the following steps:

Step 1 From your browser, open the Universal Controller user interface (where `localhost` is the name of the machine):

```
http://localhost:8080/opswise
```

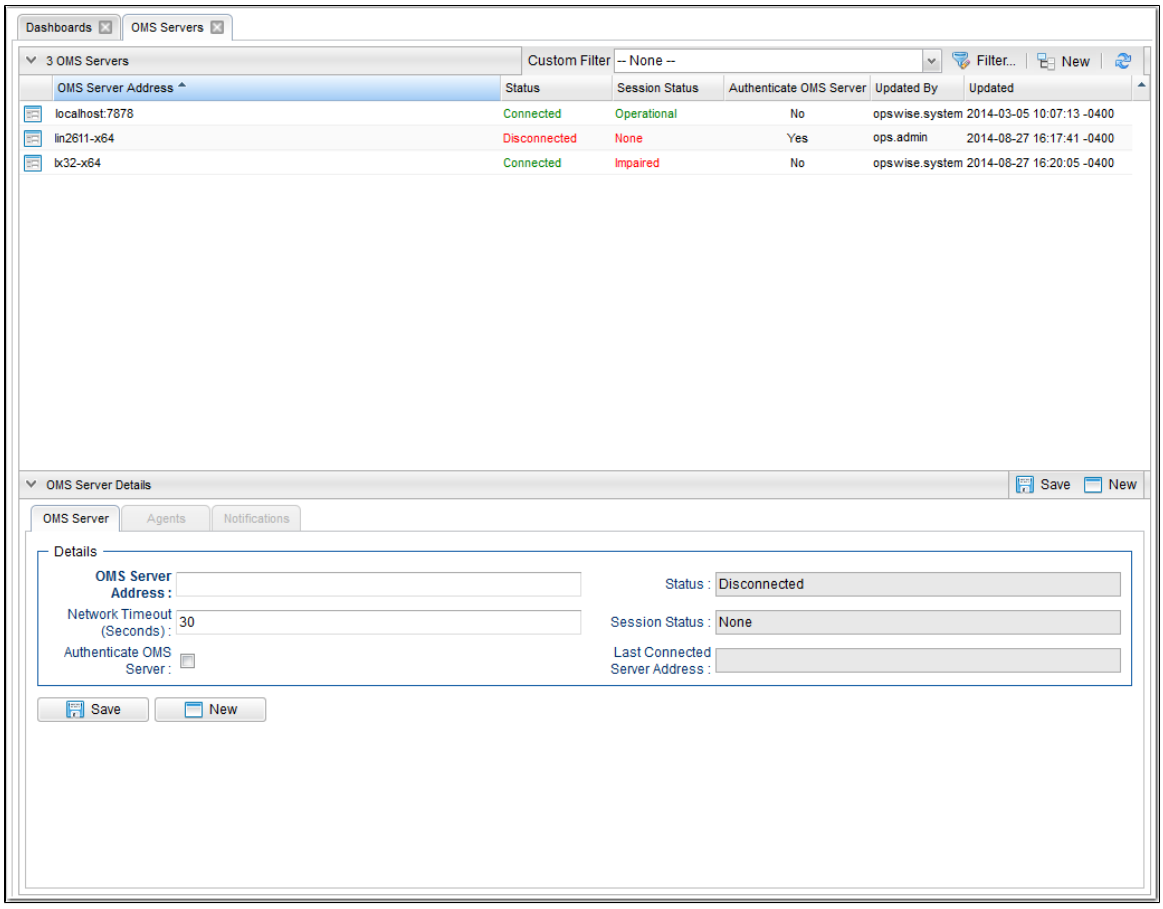
Step 2 Log in with a user name and password for this system.

Step 3 To check for your newly installed Agent, select **Agents > All Agents** from the Agents & Connections navigation pane. You will see a list similar to the following example, which includes the Agent that you just installed. Make sure the Agent Status is **Active**.



Step 4

To check for your newly installed OMS Server, select **System > OMS Servers** from the Agents & Connections navigation pane. You will see a list similar to the following example, which includes the OMS Server that you just installed as part of the Agent package. Make sure the OMS Server Status is **Active**.



Step 5

For more information about these components in the Universal Controller user interface, see:

- Agents
- OMS Servers

Upgrading Universal Agent

- Overview
- Licensing
- Upgrade Procedures
  - Upgrading Universal Agent for z/OS
  - Upgrading Universal Agent for Windows
  - Upgrading Universal Agent for SOA for Windows
  - Upgrading Universal Enterprise Controller (UEC) for Windows
  - Upgrading UEC Client Applications for Windows
  - Upgrading Universal Agent for UNIX
  - Upgrading Universal Agent for SOA for UNIX
  - Upgrading Universal Agent for IBM i

Overview

Upgrading Universal Agent refers to the increase of a currently installed pre-6.2.x Version, Release, or Modification level (see Versioning) of Universal Agent (5.2.0, 5.1.0, 4.x, 3.x, 2.x) to Universal Agent 6.2.x.

Universal Agent refers to all Universal components that have been packaged under the following names:

- Universal Agent 6.2.x
- Opswise Universal Agent 5.2.0
- Workload Automation 5.1.0 (Indesca, Infitran, UAG)
- Indesca / Infitran 4.x
- Universal Products 3.x
- Universal Command 2.x

You do not have to remove a pre-6.2.x Agent to upgrade to Agent 6.2.x.

(Applying maintenance to an Agent refers to the increase of its currently installed Maintenance level to a later Maintenance level; for example, applying maintenance to Agent 6.2.0.0 to increase its maintenance level to Agent 6.2.0.2.)

Note
You cannot upgrade an Opswise Automation Center Agent (releases 1.7, 1.6, and 1.5) to Universal Agent 6.2.x.

IMPORTANT
Before upgrading Universal Connector, you must first install the appropriate SAP NW RFC Libraries for the platform on which the connector agent is installed (see Installing Universal Agent - Universal Connector Prerequisite).

Licensing

All upgrades to Universal Agent 6.2.x from versions 4.x, 5.1.0, 5.2.0 do not require new licenses for its licensed components, except for the UEC, UEM, and SOA components of version 4.1.0, which do require new licenses.

All upgrades to Universal Agent 6.2.x from versions 3.2.0 and earlier do require new licenses for its licensed components.

Upgrade Procedures

The upgrade procedure actually is an un-install / re-install of the Agent. All configuration files (*.conf) are updated, not replaced. Local modifications to the configuration files are preserved.

Upgrading Universal Agent for z/OS

Step 1	From the Stonebranch Customer Portal, download the Agent product distribution file for z/OS (for instructions, see Product Distribution File Download).
Step 2	Stop all started tasks (STC) on the Agent that you are updating (for instructions on how to stop an Agent on a specific platform, see Starting and Stopping Agent Components).
Step 3	Install the Agent. For version-specific details, see z/OS Installation - SMP/E Installation.
Step 4	Start the Agent. The start/stop procedure may differ depending on your platform (see Starting and Stopping Agent Components and select your platform).
Step 5	Verify that the Agent is installed and running. If the Agent is configured to communicate with a Controller, verify that the Controller shows the Agent as being online (see Verifying Universal Agent Installation).

Upgrading Universal Agent for Windows

Step 1	From the Stonebranch Customer Portal, download the Agent product distribution file that is appropriate for your platform (for instructions, see Product Distribution File Download).
Step 2	Install the Agent. For details, see Universal Agent for Windows Installation.
Step 3	Verify that the Agent is installed and running. If the Agent is configured to communicate with a Controller, verify that the Controller shows the Agent as being online (see Verifying Universal Agent Installation).

Upgrading Universal Agent for SOA for Windows

Step 1	From the Stonebranch Customer Portal, download the Universal Agent for SOA for Windows product distribution file (for instructions, see Product Distribution File Download).
Step 2	Perform the recommended back-up of Universal Agent for SOA files (see Universal Agent for SOA for Windows - Pre-Installation / Upgrade Backups).
Step 3	Install Universal Agent for SOA. For details, see Universal Agent for SOA for Windows Installation.

Upgrading Universal Enterprise Controller (UEC) for Windows

Step 1	From the Stonebranch Customer Portal, download the Universal Enterprise Controller (UEC) for Windows product distribution file (for instructions, see Product Distribution File Download).
Step 2	Install UEC. For details, see Universal Enterprise Controller for Windows Installation.

Upgrading UEC Client Applications for Windows

Step 1	From the Stonebranch Customer Portal, download the UEC Client Applications for Windows product distribution file (for instructions, see Product Distribution File Download).
Step 2	Log out of active UEC Client Applications instances on the machine where you are upgrading UEC Client Applications.
Step 3	Install UEC Client Applications. For details, see UEC Client Applications Installation.

Note
If you are upgrading a per-user installation of UEC Client Applications from any release prior to 5.2.0.1, you must uninstall the older version before installing the new version. There is an error in the older releases that prevents the upgrade from retaining a per-user installation environment.

Upgrades from 5.2.0.1 to any newer release should behave as expected.

Upgrading Universal Agent for UNIX

Step 1	From the Stonebranch Customer Portal, download the Agent product distribution file that is appropriate for your platform (for instructions, see Product Distribution File Download).
Step 2	Install the Agent. For details, see Universal Agent for UNIX Installation.
Step 3	If you are installing the Agent on AIX or Linux, you must review and perform PAM customization: • AIX PAM Customization • Linux PAM Customization
Step 4	Verify that the Agent is installed and running. If the Agent is configured to communicate with a Controller, verify that the Controller shows the Agent as being online (see Verifying Universal Agent Installation).

Upgrading Universal Agent for SOA for UNIX

Step 1	From the Stonebranch Customer Portal, download the Universal Agent for SOA for UNIX product distribution file that is appropriate for your platform (for instructions, see Product Distribution File Download).
Step 2	Perform the recommended back-up of Universal Agent for SOA files (see Universal Agent for SOA for UNIX - Pre-Installation / Upgrade Backups).
Step 3	Install the Universal Agent for SOA. For details, see Universal Agent for SOA for UNIX Installation.

Upgrading Universal Agent for IBM i

Currently, IBM i runs Workload Automation 5.1.0. For information on upgrading to that version, see IBM i Installation - Upgrade Installation to New Release.

Applying Maintenance to Universal Agent

- Overview
- Applying Maintenance to Linux/Unix and Windows Agents
- Applying Maintenance to z/OS Agents

Overview

Applying maintenance to Universal Agent refers to the increase from its currently installed Maintenance level (see Versioning) to a later Maintenance level; for example, applying maintenance to a 6.2.0.0 Agent to increase its maintenance level to 6.2.0.1.

(Upgrading an Agent refers to the increase from its currently installed Version, Release, or Modification level to a later Version, Release, or Modification level; for example, upgrading a 5.2.0.11 Agent to a 6.2.0.0 Agent.)

Applying Maintenance to Linux/Unix and Windows Agents

The procedure for applying maintenance to Linux/Unix and Windows Agents is the same as the procedure for upgrading Linux/Unix and Windows Agents, which includes downloading and installing a platform-specific Agent from the Stonebranch Customer Portal (see Upgrading Universal Agent).

Applying Maintenance to z/OS Agents

To apply maintenance to Universal Agent for z/OS:

Step 1	From the Stonebranch Customer Portal, download the **z/OS PTFs** file from the Maintenance page (for instructions, see Product Distribution File Download).
Step 2	Stop all Agent processes on the Agents to which you are applying maintenance (for instructions on how to stop the Agent on a specific platform, see Starting and Stopping Agent Components).
Step 3	Apply the maintenance to the Agent using the instructions in the README file that is included with **z/OS PTFs**.
Step 4	Verify that the Agent is communicating with Universal Controller (see Verifying Universal Agent Installation).

Licenses for Third-Party Libraries

- Overview
- Berkeley DB License
- OpenSSL License
- zlib License
- ICU License
- Boost Software License

Overview

This page provides the following license files for the third-party libraries used within Universal Agent:

- Berkeley DB License
- OpenSSL License
- zlib License
- ICU License
- BOOST Software License

Berkeley DB License

The Berkeley DB library is used in binary form on z/OS, Windows, and UNIX ports.

The following is the Berkeley DB library license.

/*
* Copyright (c) 1990-2005
* Sleepycat Software. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. Redistributions in any form must be accompanied by information on
* how to obtain complete source code for the DB software and any
* accompanying software that uses the DB software. The source code
* must either be included in the distribution or be available for no
* more than the cost of distribution plus a nominal fee, and must be
* freely redistributable under reasonable conditions. For an
* executable file, complete source code means the source code for all
* modules it contains. It does not include source code for modules or
* files that typically accompany the major components of the operating
* system on which the executable file runs.
*
* THIS SOFTWARE IS PROVIDED BY SLEEPYCAT SOFTWARE ``AS IS'' AND ANY EXPRESS
* OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
* WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR
* NON-INFRINGEMENT, ARE DISCLAIMED. IN NO EVENT SHALL SLEEPYCAT SOFTWARE
* BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR
* CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF
* SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS
* INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN
* CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF
* THE POSSIBILITY OF SUCH DAMAGE.
*/
/*
* Copyright (c) 1990, 1993, 1994, 1995
* The Regents of the University of California. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions

* are met:
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. Neither the name of the University nor the names of its contributors
* may be used to endorse or promote products derived from this software
* without specific prior written permission.
*
* THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*/
/*
/* Copyright (c) 1995, 1996
/* The President and Fellows of Harvard University. All rights reserved.
/*
/* Redistribution and use in source and binary forms, with or without
/* modification, are permitted provided that the following conditions
/* are met:
/* 1. Redistributions of source code must retain the above copyright
/* notice, this list of conditions and the following disclaimer.
/* 2. Redistributions in binary form must reproduce the above copyright
/* notice, this list of conditions and the following disclaimer in the
/* documentation and/or other materials provided with the distribution.
/* 3. Neither the name of the University nor the names of its contributors
/* may be used to endorse or promote products derived from this software
/* without specific prior written permission.
/*
/* THIS SOFTWARE IS PROVIDED BY HARVARD AND ITS CONTRIBUTORS ``AS IS'' AND
/* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
/* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
/* ARE DISCLAIMED. IN NO EVENT SHALL HARVARD OR ITS CONTRIBUTORS BE LIABLE
/* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
/* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
/* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
/* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
/* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
/* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
/* SUCH DAMAGE.
*/

OpenSSL License

The OpenSSL toolkit is used in binary form on z/OS, Windows, UNIX, and IBM i ports.

The following is the OpenSSL license.

/* ==
* Copyright (c) 1998-2007 The OpenSSL Project. All rights reserved.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
*
* 1. Redistributions of source code must retain the above copyright
* notice, this list of conditions and the following disclaimer.
*
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in
* the documentation and/or other materials provided with the
* distribution.
*
* 3. All advertising materials mentioning features or use of this

* software must display the following acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit. (http://www.openssl.org/)"
*
* 4. The names "OpenSSL Toolkit" and "OpenSSL Project" must not be used to
* endorse or promote products derived from this software without
* prior written permission. For written permission, please contact
* openssl-core@openssl.org.
*
* 5. Products derived from this software may not be called "OpenSSL"
* nor may "OpenSSL" appear in their names without prior written
* permission of the OpenSSL Project.
*
* 6. Redistributions of any form whatsoever must retain the following
* acknowledgment:
* "This product includes software developed by the OpenSSL Project
* for use in the OpenSSL Toolkit (http://www.openssl.org/)"
*
* THIS SOFTWARE IS PROVIDED BY THE OpenSSL PROJECT ``AS IS'' AND ANY
* EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR
* PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE OpenSSL PROJECT OR
* ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,
* SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT
* NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
* LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
* STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
* ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
* OF THE POSSIBILITY OF SUCH DAMAGE.
* ==
*
* This product includes cryptographic software written by Eric Young
* (eay@cryptsoft.com). This product includes software written by Tim
* Hudson (tjh@cryptsoft.com).
*
*/

Original SSLeay License

/* Copyright (C) 1995-1998 Eric Young (eay@cryptsoft.com)
* All rights reserved.
*
* This package is an SSL implementation written
* by Eric Young (eay@cryptsoft.com).
* The implementation was written so as to conform with Netscapes SSL.
*
* This library is free for commercial and non-commercial use as long as
* the following conditions are adhered to. The following conditions
* apply to all code found in this distribution, be it the RC4, RSA,
* lhash, DES, etc., code; not just the SSL code. The SSL documentation
* included with this distribution is covered by the same copyright terms
* except that the holder is Tim Hudson (tjh@cryptsoft.com).
*
* Copyright remains Eric Young's, and as such any Copyright notices in
* the code are not to be removed.
* If this package is used in a product, Eric Young should be given attribution
* as the author of the parts of the library used.
* This can be in the form of a textual message at program startup or
* in documentation (online or textual) provided with the package.
*
* Redistribution and use in source and binary forms, with or without
* modification, are permitted provided that the following conditions
* are met:
* 1. Redistributions of source code must retain the copyright
* notice, this list of conditions and the following disclaimer.
* 2. Redistributions in binary form must reproduce the above copyright
* notice, this list of conditions and the following disclaimer in the
* documentation and/or other materials provided with the distribution.
* 3. All advertising materials mentioning features or use of this software
* must display the following acknowledgement:
* "This product includes cryptographic software written by
* Eric Young (eay@cryptsoft.com)"
* The word 'cryptographic' can be left out if the routines from the library

* being used are not cryptographic related :-).
* 4. If you include any Windows specific code (or a derivative thereof) from
* the apps directory (application code) you must include an acknowledgement:
* "This product includes software written by Tim Hudson (tjh@cryptsoft.com)"
*
* THIS SOFTWARE IS PROVIDED BY ERIC YOUNG ``AS IS'' AND
* ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
* IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
* ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE
* FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
* DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
* OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
* HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
* LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
* OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
* SUCH DAMAGE.
*
* The licence and distribution terms for any publicly available version or
* derivative of this code cannot be changed. i.e. this code cannot simply be
* copied and put under another distribution licence
* [including the GNU Public Licence.]
*/

zlib License

The zlib library is used in binary form on z/OS, Windows, UNIX, and IBM i ports.

The following is the zlib library license.

(C) 1995-2002 Jean-loup Gailly and Mark Adler

This software is provided 'as-is', without any express or implied
warranty. In no event will the authors be held liable for any damages
arising from the use of this software.

Permission is granted to anyone to use this software for any purpose,
including commercial applications, and to alter it and redistribute it
freely, subject to the following restrictions:

1. The origin of this software must not be misrepresented; you must not
claim that you wrote the original software. If you use this software
in a product, an acknowledgment in the product documentation would be
appreciated but is not required.
2. Altered source versions must be plainly marked as such, and must not be
misrepresented as being the original software.
3. This notice may not be removed or altered from any source distribution.

Jean-loup Gailly Mark Adler
jloup@gzip.org madler@alumni.caltech.edu

ICU License

ICU License - ICU 1.8.1 and later

COPYRIGHT AND PERMISSION NOTICE

Copyright (c) 1995-2015 International Business Machines Corporation and others

All rights reserved.

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, provided that the above copyright notice(s) and this permission notice appear in all copies of the Software and that both the above copyright notice(s) and this permission notice appear in supporting documentation.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF THIS SOFTWARE.

Except as contained in this notice, the name of a copyright holder shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Software without prior written authorization of the copyright holder.

All trademarks and registered trademarks mentioned herein are the property of their respective owners.
Third-Party Software Licenses
This section contains third-party software notices and/or additional terms for licensed third-party software components included within ICU libraries.
1. Unicode Data Files and Software

COPYRIGHT AND PERMISSION NOTICE

Copyright © 1991-2015 Unicode, Inc. All rights reserved.
Distributed under the Terms of Use in http://www.unicode.org/copyright.html.

Permission is hereby granted, free of charge, to any person obtaining
a copy of the Unicode data files and any associated documentation
(the "Data Files") or Unicode software and any associated documentation
(the "Software") to deal in the Data Files or Software
without restriction, including without limitation the rights to use,
copy, modify, merge, publish, distribute, and/or sell copies of
the Data Files or Software, and to permit persons to whom the Data Files
or Software are furnished to do so, provided that
(a) this copyright and permission notice appear with all copies
of the Data Files or Software,
(b) this copyright and permission notice appear in associated
documentation, and
(c) there is clear notice in each modified Data File or in the Software
as well as in the documentation associated with the Data File(s) or
Software that the data or software has been modified.

THE DATA FILES AND SOFTWARE ARE PROVIDED "AS IS", WITHOUT WARRANTY OF
ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE
WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
NONINFRINGEMENT OF THIRD PARTY RIGHTS.
IN NO EVENT SHALL THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS
NOTICE BE LIABLE FOR ANY CLAIM, OR ANY SPECIAL INDIRECT OR CONSEQUENTIAL
DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM LOSS OF USE,
DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER
TORTIOUS ACTION, ARISING OUT OF OR IN CONNECTION WITH THE USE OR
PERFORMANCE OF THE DATA FILES OR SOFTWARE.

Except as contained in this notice, the name of a copyright holder
shall not be used in advertising or otherwise to promote the sale,
use or other dealings in these Data Files or Software without prior
written authorization of the copyright holder.

2. Chinese/Japanese Word Break Dictionary Data (cjdict.txt)

1. The Google Chrome software developed by Google is licensed under the BSD license. Other software included in this distribution is provided under other licenses, as set forth below.
3. The BSD License
4. http://opensource.org/licenses/bsd-license.php
5. Copyright (C) 2006-2008, Google Inc.
7. All rights reserved.
9. Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:
11. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
12. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
13. Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.
16. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
19. The word list in cjdict.txt are generated by combining three word lists listed
20. below with further processing for compound word breaking. The frequency is generated
21. with an iterative training against Google web corpora.
23. * Libtabe (Chinese)
24. - https://sourceforge.net/project/?group_id=1519

25. - Its license terms and conditions are shown below.
27. * IPADIC (Japanese)
28. - http://chasen.aist-nara.ac.jp/chasen/distribution.html
29. - Its license terms and conditions are shown below.
31. --------~~COPYING.libtabe----BEGIN~~----------
33. /*
34. * Copyrighy (c) 1999 TaBE Project.
35. * Copyright (c) 1999 Pai-Hsiang Hsiao.
36. * All rights reserved.
37. *
38. * Redistribution and use in source and binary forms, with or without
39. * modification, are permitted provided that the following conditions
40. * are met:
41. *
42. * . Redistributions of source code must retain the above copyright
43. * notice, this list of conditions and the following disclaimer.
44. * . Redistributions in binary form must reproduce the above copyright
45. * notice, this list of conditions and the following disclaimer in
46. * the documentation and/or other materials provided with the
47. * distribution.
48. * . Neither the name of the TaBE Project nor the names of its
49. * contributors may be used to endorse or promote products derived
50. * from this software without specific prior written permission.
51. *
52. * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
53. * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
54. * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
55. * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
56. * REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
57. * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
58. * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
59. * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
60. * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
61. * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
62. * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
63. * OF THE POSSIBILITY OF SUCH DAMAGE.
64. */
66. /*
67. * Copyright (c) 1999 Computer Systems and Communication Lab,
68. * Institute of Information Science, Academia Sinica.
69. * All rights reserved.
70. *
71. * Redistribution and use in source and binary forms, with or without
72. * modification, are permitted provided that the following conditions
73. * are met:
74. *
75. * . Redistributions of source code must retain the above copyright
76. * notice, this list of conditions and the following disclaimer.
77. * . Redistributions in binary form must reproduce the above copyright
78. * notice, this list of conditions and the following disclaimer in
79. * the documentation and/or other materials provided with the
80. * distribution.
81. * . Neither the name of the Computer Systems and Communication Lab
82. * nor the names of its contributors may be used to endorse or
83. * promote products derived from this software without specific
84. * prior written permission.
85. *
86. * THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS
87. * "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT
88. * LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS
89. * FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE
90. * REGENTS OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT,
91. * INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
92. * (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR
93. * SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
94. * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT,
95. * STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE)
96. * ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED
97. * OF THE POSSIBILITY OF SUCH DAMAGE.
98. */
100. Copyright 1996 Chih-Hao Tsai @ Beckman Institute, University of Illinois
101. c-tsai4@uiuc.edu http://casper.beckman.uiuc.edu/~c-tsai4
103. ----------~~COPYING.libtabe----END~~----------
106. ----------~~COPYING.ipadic----BEGIN~~----------
108. Copyright 2000, 2001, 2002, 2003 Nara Institute of Science

109. and Technology. All Rights Reserved.
111. Use, reproduction, and distribution of this software is permitted.
112. Any copy of this software, whether in its original form or modified,
113. must include both the above copyright notice and the following
114. paragraphs.
116. Nara Institute of Science and Technology (NAIST),
117. the copyright holders, disclaims all warranties with regard to this
118. software, including all implied warranties of merchantability and
119. fitness, in no event shall NAIST be liable for
120. any special, indirect or consequential damages or any damages
121. whatsoever resulting from loss of use, data or profits, whether in an
122. action of contract, negligence or other tortuous action, arising out
123. of or in connection with the use or performance of this software.
125. A large portion of the dictionary entries
126. originate from ICOT Free Software. The following conditions for ICOT
127. Free Software applies to the current dictionary as well.
129. Each User may also freely distribute the Program, whether in its
130. original form or modified, to any third party or parties, PROVIDED
131. that the provisions of Section 3 ("NO WARRANTY") will ALWAYS appear
132. on, or be attached to, the Program, which is distributed substantially
133. in the same form as set out herein and that such intended
134. distribution, if actually made, will neither violate or otherwise
135. contravene any of the laws and regulations of the countries having
136. jurisdiction over the User or the intended distribution itself.
138. NO WARRANTY
140. The program was produced on an experimental basis in the course of the
141. research and development conducted during the project and is provided
142. to users as so produced on an experimental basis. Accordingly, the
143. program is provided without any warranty whatsoever, whether express,
144. implied, statutory or otherwise. The term "warranty" used herein
145. includes, but is not limited to, any warranty of the quality,
146. performance, merchantability and fitness for a particular purpose of
147. the program and the nonexistence of any infringement or violation of
148. any right of any third party.
150. Each user of the program will agree and understand, and be deemed to
151. have agreed and understood, that there is no warranty whatsoever for
152. the program and, accordingly, the entire risk arising from or
153. otherwise connected with the program is assumed by the user.
155. Therefore, neither ICOT, the copyright holder, or any other
156. organization that participated in or was otherwise related to the
157. development of the program and their respective officials, directors,
158. officers and other employees shall be held liable for any and all
159. damages, including, without limitation, general, special, incidental
160. and consequential damages, arising out of or otherwise in connection
161. with the use or inability to use the program or any product, material
162. or result produced or otherwise obtained by using the program,
163. regardless of whether they have been advised of, or otherwise had
164. knowledge of, the possibility of such damages at any time during the
165. project or thereafter. Each user will be deemed to have agreed to the
166. foregoing by his or her commencement of use of the program. The term
167. "use" as used herein includes, but is not limited to, the use,
168. modification, copying and distribution of the program and the
169. production of secondary products from the program.
171. In the case where the program, whether in its original form or
172. modified, was distributed or delivered to or received by a user from
173. any person, organization or entity other than ICOT, unless it makes or
174. grants independently of ICOT any specific warranty to the user in
175. writing, such person, organization or entity, will also be exempted
176. from and not be held liable to the user for any such damages as noted
177. above as far as the program is concerned.
179. ----------COPYING.ipadic---END----------

3. Lao Word Break Dictionary Data (laodict.txt)

    1. Copyright (c) 2013 International Business Machines Corporation
    2. and others. All Rights Reserved.
       #
    3. Project: http://code.google.com/p/lao-dictionary/
    4. Dictionary: http://lao-dictionary.googlecode.com/git/Lao-Dictionary.txt
    5. License: http://lao-dictionary.googlecode.com/git/Lao-Dictionary-LICENSE.txt
    6. (copied below)
       #
    7. This file is derived from the above dictionary, with slight modifications.
    8. ----------
    9. Copyright (C) 2013 Brian Eugene Wilson, Robert Martin Campbell.

10. All rights reserved.
    #
11. Redistribution and use in source and binary forms, with or without modification,
12. are permitted provided that the following conditions are met:
    #
13. Redistributions of source code must retain the above copyright notice, this
14. list of conditions and the following disclaimer. Redistributions in binary
15. form must reproduce the above copyright notice, this list of conditions and
16. the following disclaimer in the documentation and/or other materials
17. provided with the distribution.
    #
18. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
19. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
20. WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
21. DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
22. ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
23. (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
24. LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
25. ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
26. (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
27. SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
28. ----------

4. Burmese Word Break Dictionary Data (burmesedict.txt)

1. Copyright (c) 2014 International Business Machines Corporation
2. and others. All Rights Reserved.
   #
3. This list is part of a project hosted at:
4. github.com/kanyawtech/myanmar-karen-word-lists
   #
5. ----------
6. Copyright (c) 2013, LeRoy Benjamin Sharon
7. All rights reserved.
   #
8. Redistribution and use in source and binary forms, with or without modification,
9. are permitted provided that the following conditions are met:
   #
10. Redistributions of source code must retain the above copyright notice, this
11. list of conditions and the following disclaimer.
    #
12. Redistributions in binary form must reproduce the above copyright notice, this
13. list of conditions and the following disclaimer in the documentation and/or
14. other materials provided with the distribution.
    #
15. Neither the name Myanmar Karen Word Lists, nor the names of its
16. contributors may be used to endorse or promote products derived from
17. this software without specific prior written permission.
    #
18. THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
19. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
20. WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
21. DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
22. ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
23. (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
24. LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
25. ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
26. (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
27. SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
28. ----------

5. Time Zone Database

ICU uses the public domain data and code derived from Time Zone Database for its time zone support. The ownership of the TZ database is explained in BCP 175: Procedure for Maintaining the Time Zone Database section 7.

7. Database Ownership

The TZ database itself is not an IETF Contribution or an IETF
document. Rather it is a pre-existing and regularly updated work
that is in the public domain, and is intended to remain in the public
domain. Therefore, BCPs 78 RFC5378 and 79 RFC3979 do not apply
to the TZ Database or contributions that individuals make to it.
Should any claims be made and substantiated against the TZ Database,
the organization that is providing the IANA Considerations defined in

this RFC, under the memorandum of understanding with the IETF, currently ICANN, may act in accordance with all competent court orders. No ownership claims will be made by ICANN or the IETF Trust on the database or the code. Any person making a contribution to the database or code waives all rights to future claims in that contribution or in the TZ Database.

Boost Software License

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.