Opswise Automation Center 5.1.0

Installation and Upgrade

System Configuration

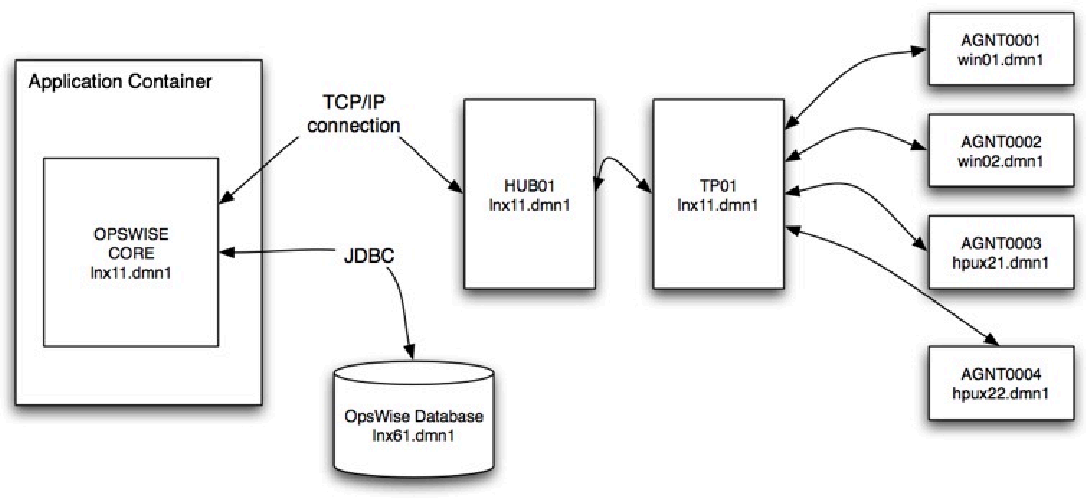
System Overview

Overview

Opswise consists of the following basic components:

- Core Processor (also referred to as Cluster Node)
- Application Container
- Database
- Message hub
- Transporter
- Multiple remote agents

The following diagram shows a sample configuration. The typical configuration will use Apache Tomcat as the application container; the Opswise Core Processor running inside Tomcat; a DBMS to contain the database, the outboard components, which include the hub (HUB01) and transporter (TP01), and one or more agents installed on various platforms.



Each component is described in more detail below.

Core Processor

The core processor provides the business logic of the system. The Opswise Core is a J2EE process that runs in an application container such as Apache Tomcat. This central processing server provides components that present the user interface, handle the scheduling logic, process all messages to and from the agents, and synchronize much of the High Availability operation of the product. In some places you may see the core processor referred to as the cluster node.

For information about how the core processor operates, see High Availability.

Application Container

The application container is third-party software that serves as a container for the Opswise core processor. Opswise currently runs under Apache Tomcat; future versions will support other application containers.

Database

The database management component supports SQL queries to a set of tables in the Opswise database. The following databases are supported:

- Oracle 10g, 11g
- MS SQL Server 2005, 2008
- MySQL 5.1, 5.5

Message Hub (MsgHub)

The message hub relays messages from the outbound message queue from the core to the agents, and writes messages to the inbound message queue from the agents to the core.

The message hub communicates with the agents using a "publish | subscribe" communications server known as the transport. Those messages are transformed to and from serialized message structures and moved to and from the agents via the messaging layer.

Transporter (Transport)

The transporter serves as the enterprise bus for the Unified Server Processes as well as the TCP/IP transport layer for the outboard components (the message hub and transporter). The message hub and agents are considered clients of the transporter. The transporter serves to abstract a group communications protocol referred to as "publish | subscribe." Clients subscribe to message queues to receive message traffic, and send data to subscribers by publishing messages to the transporter. Within Opswise, the outboard component names are normally assigned by the core component when the clients initially register their availability at process start-up; however, the user can also specify the names. See Agents and Connectors.

Universal Automation Center Agent

Universal Automation Center Agent (UAG) - the agent - is a process installed on each remote host system that launches and monitors applications or scripts, and returns status information and output when requested. The agent also performs event-oriented tasks such as file monitoring. When the agent starts, it establishes a connection with the message transporter and registers with the Opswise core. When an agent-based task is launched by the Opswise core, the core sends the agent a set of messages that instruct the agent to perform the requested services. See Agent-Based Resources.

Opswise Directory Structure

Shown below is the directory structure of Opswise

Core Component

```
/home//tomcat/
/home//tomcat/bin
/home//tomcat/conf
/home//tomcat/lib
/home//tomcat/logs
/home//tomcat/opswise_logs
/home//tomcat/opswise_export
/home//tomcat/temp
/home//tomcat/webapps
/home//tomcat/webapps/opswise
/home//tomcat/webapps/opswise/help
/home//tomcat/webapps/opswise/htmlarea
/home//tomcat/webapps/opswise/images
/home//tomcat/webapps/opswise/META-INF
/home//tomcat/webapps/opswise/portlet
/home//tomcat/webapps/opswise/scripts
/home//tomcat/webapps/opswise/styles
/home//tomcat/webapps/opswise/WEB-INF
/home//tomcat/webapps/opswise/WEB-INF/apps
/home//tomcat/webapps/opswise/WEB-INF/db.index
/home//tomcat/webapps/opswise/WEB-INF/ui.jtemplates
/home//tomcat/webapps/opswise/WEB-INF/dict
/home//tomcat/webapps/opswise/WEB-INF/plugins
/home//tomcat/webapps/opswise/WEB-INF/sys.scripts
/home//tomcat/webapps/opswise/WEB-INF/update
/home//tomcat/webapps/opswise/WEB-INF/import.templates
/home//tomcat/webapps/opswise/WEB-INF/lib
/home//tomcat/webapps/opswise/WEB-INF/properties
/home//tomcat/webapps/opswise/WEB-INF/ui.jforms
/home//tomcat/webapps/opswise/graph
/home//tomcat/webapps/opswise/hta
/home//tomcat/webapps/opswise/icons
/home//tomcat/webapps/opswise/mobile
/home//tomcat/webapps/opswise/portal
/home//tomcat/webapps/opswise/temp
/home//tomcat/work
```

Outboard Components (Message Hub, Transporter)

```
/opswise
/opswise/bin
/opswise/cache
/opswise/etc
/opswise/logs
/opswise/var
```

Universal Automation Center Agent

For Linux/Unix, click here.

For Windows, click here.

For z/OS, click here.

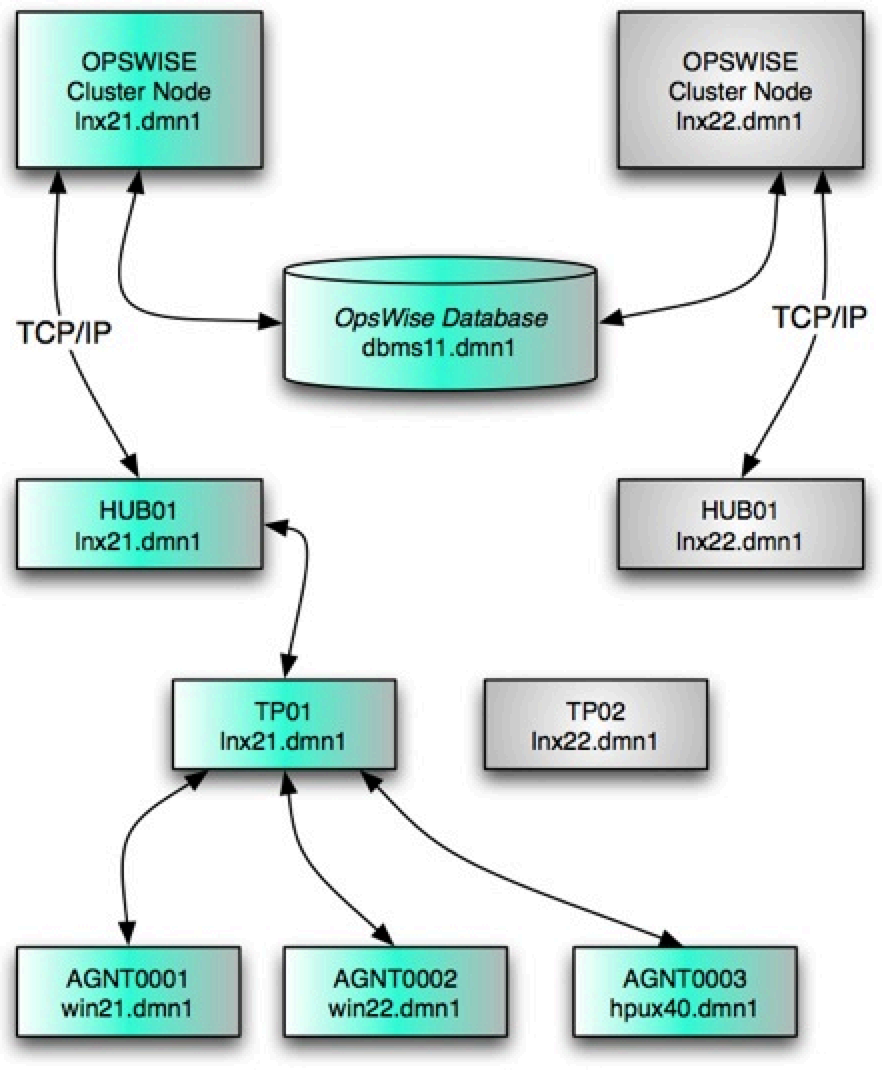
High Availability

Introduction

High availability means that the system continues to perform its duties through hardware or software failures. This document describes how Opswise system components recover in the event of such a failure, and what actions the users must take, if any.

Configuration for High Availability

Most configuration options for High Availability are performed internally and automatically by the components of Opswise. In other words, most of the configuration occurs once a component is connected to a database. The only exception to this is that of communications clients who must be told via configuration of one or more transporter processes. This section will describe in detail what must be done, but first, let us look at a typical, although simple, Opswise installation using High Availability.



In this case, there will be two core processes operating in cluster mode, two transporters, two msghubs, and three agents. There are surprisingly few manual steps to achieve High Availability. In this example, we need to set up several properties files. Not many surprises. The key thing to remember is that the core and msghub work together automatically, simply because they are connected the the same database. Another thing to be clear on is that the msghub and agent processes are really just clients of the transporter for communications purposes and are treated as peers in that regard. For High Availability, we need only to give them a selection list of transporter addresses from which they can choose.

Description of Operation

For the purposes of a high availability discussion, Opswise components are divided into two categories: the core processing cluster node(s) and the "outboard processes," which comprise agents and core-agent communications.

Core Processing "Cluster" Node

Each Opswise installation consists of one or more nodes, also referred to as "cluster nodes." Only one node is required to run Opswise; however, in order to run a high availability configuration, you must run at least two nodes. At any given time, one node operates in an Active mode and the remaining nodes operate in Passive mode. The active node performs all system processing functions and serves as the core processing node. Nodes running in Passive mode do not perform processing functions but allow the user to monitor and display data, and generate reports.

How High Availability Works

In a high availability (HA) configuration, running nodes play the role of "guardian" to one another. All running nodes issue heartbeats and check the status of other running nodes, both when they start up and continuously during operations. This process is described in more detail below.

Checking for an Active Node at Startup

The steps below describe the startup procedure for all Opswise nodes:

1. The node initially starts up in Passive/Unavailable mode.
2. If the node connects to a Message Hub, it transitions to Passive/Available mode.
3. The node checks for any Active nodes.
    - If no Active node is found:
        - If the node is in Passive/Available mode (connected to a Message Hub), it becomes the Active node.
        - If the node is in Passive/Unavailable mode:
            - If no other node is in Passive/Available mode, this node becomes the Active node.
            - If another node is in Passive/Available mode, this node remains in Passive/Unavailable.
    - If an Active node is found:
        - If that Active node is healthy, based on its heartbeat timestamp, the new node stays in Passive mode.
        - If the Active node is deemed stale, based on its heartbeat timestamp:
            - If the node is in Passive/Available mode (connected to a Message Hub), then it becomes the Active node and marks the previous Active node as offline.
            - If the node is in Passive/Unavailable mode:
                - If no other node is in Passive/Available mode, this node becomes the Active node and marks the previous Active node as offline.
                - If another node is in Passive/Available mode, this node remains in Passive/Unavailable.

Checking the Active Node During Operations

Once started, all nodes continuously monitor the heartbeats of other running nodes. If a Passive node determines that the Active node is no longer running, the Passive node automatically takes over as the Active node based upon the same criteria described above in the startup procedure. If the Passive node is in Available mode, then it will take over as Active. If the Passive node is in Unavailable mode, then it will only become Active if no other Passive node is in Available mode.

This determination is made as follows:

1. The Active node sends a "heartbeat" by updating a timestamp in the database. The heartbeat interval is specified using the **Heartbeat Processing Interval In Seconds** parameter in **Configuration > Properties**. The default is 10, meaning every 10 seconds.
2. All nodes in Passive mode check the Active node's timestamp to determine if it is current. The interval for the timestamp check is also specified using the **Heartbeat Processing Interval In Seconds** parameter in **Configuration > Properties**.
3. If a node in Passive mode determines that the Active node's timestamp has not updated after six checks, the Passive node changes the status of the Active node to "Offline" and takes over as Active node. If more than one node is operating in Passive mode, the first node eligible to become Active that determines that the Active node is not running becomes the Active node.

Active Node with no Message Hub

An Active node that does not have a connected Message Hub checks regularly to see of any Passive node is eligible to become Active. If a cluster node is in Passive/Available mode (has a connected Message Hub), then the Active node (when it has no Message Hub connected) will automatically stop and restart. This allows the eligible Passive/Available node to take over as Active. This stop and restart is done within the Tomcat process. That is, the Tomcat process remains running and the Opswise deployment just shuts down its internal processes (threads) and then starts them back up. Tomcat itself is not stopped and started.

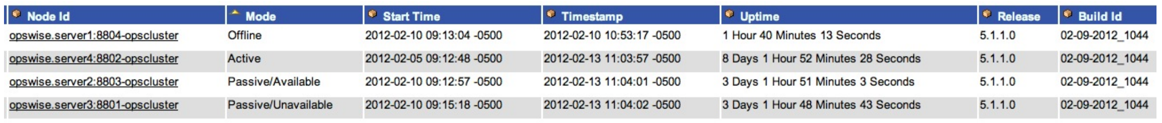
What To Do If a Failover Occurs

When a Passive node takes over as Active node, referred to here as "failover," the event is invisible unless you are using the Active node in a browser. If you are using a node in a browser and the node fails, you will see a browser error. In this case, take the following steps to continue working:

1. Access the new Active node in your browser. To determine which node is now Active, from the navigation pane select **Automation Center Resources > Cluster Nodes** and check the **Mode** column.
2. If you were adding, deleting, or updating records at the time of the failure, check the record you were working on. Any data you had not yet saved will be lost.

Viewing Node Status

To view a list of all nodes, select **Automation Center Resources > Cluster Nodes**. The system displays a list of all registered nodes, as shown in the example below. A node becomes registered the first time it starts. Once the node is registered, it always appears in the Cluster Nodes list, regardless of its current mode or status.



Node Id	Mode	Start Time	Timestamp	Uptime	Release	Build Id
opswise.server1:8804-opscluster	Offline	2012-02-10 09:13:04 -0500	2012-02-10 10:53:17 -0500	1 Hour 40 Minutes 13 Seconds	5.1.1.0	02-09-2012_1044
opswise.server4:8802-opscluster	Active	2012-02-05 09:12:48 -0500	2012-02-13 11:03:57 -0500	8 Days 1 Hour 52 Minutes 28 Seconds	5.1.1.0	02-09-2012_1044
opswise.server2:8803-opscluster	Passive/Available	2012-02-10 09:12:57 -0500	2012-02-13 11:04:01 -0500	3 Days 1 Hour 51 Minutes 3 Seconds	5.1.1.0	02-09-2012_1044
opswise.server3:8801-opscluster	Passive/Unavailable	2012-02-10 09:15:18 -0500	2012-02-13 11:04:02 -0500	3 Days 1 Hour 48 Minutes 43 Seconds	5.1.1.0	02-09-2012_1044

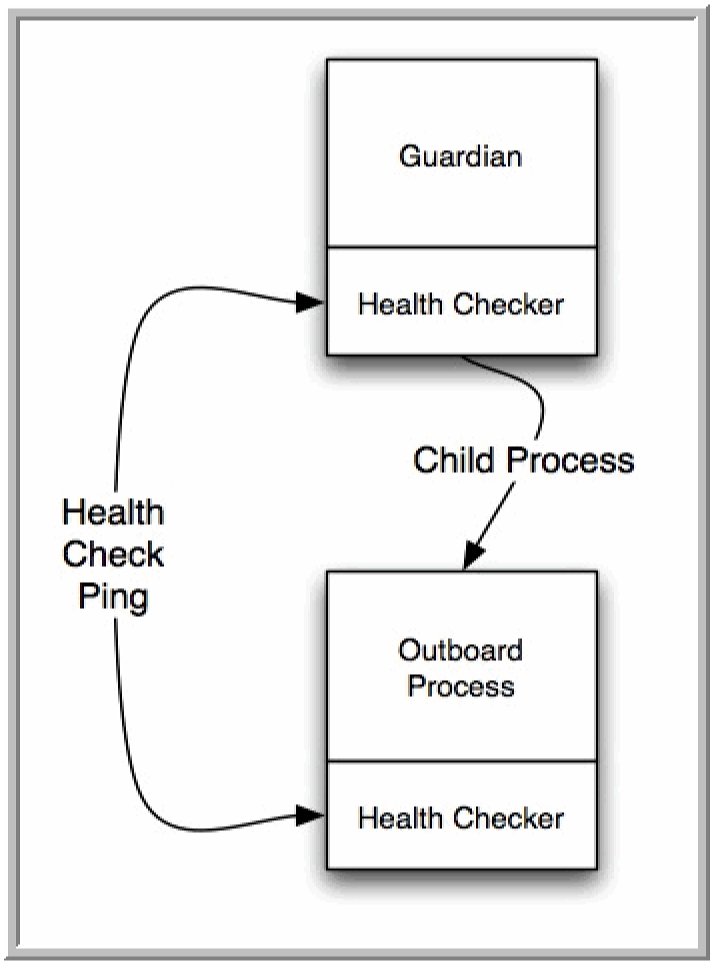
The following table provides column descriptions for the cluster node display.

Field Name	Description
Node ID	User-defined. The URL of the node.
Mode	The processing mode of the node. One of the following:\\ • **Active** -- An Active node processes events and messages and interfaces with the database. It is considered the active core processing node for automated operations. • **Passive/Available** -- This node is is running and connected to its Message Hub. An "Available" node performs the following tasks: ◦ Accepts HTTP requests for data. It can access the database, generate reports, monitor and display data. ◦ Does not process any events or messages. ◦ Takes over as Active node if it determines that the Active node is not running. • **Passive/Unavailable** -- This node is is running and is not connected to its Message Hub. An "Unavailable" node performs the following tasks: ◦ Accepts HTTP requests for data. It can access the database, generate reports, monitor and display data. ◦ Does not process any events or messages. ◦ Takes over as Active node if it determines that the Active node is not running and no other node is in Passive/Available mode. • **Offline** -- This node is not running.
Start Time	System-supplied. Date and time this node was last started.
Timestamp	System-supplied. Date and time of this node's last heartbeat.
Uptime	System-supplied. Amount of time this node has been running.
Release	System-supplied. Release number for this node. Support purposes only.
Build ID	System-supplied. Build ID for this node. Support purposes only.

Outboard Processes

Opswise outboard processes include the message hub and transporter. Each component has a guardian process that monitors the health of the component and restarts it if problems arise.

As shown in the illustration below, once the guardian process starts a component, a health checker is initialized in the component. At regular intervals the guardian process requests health status of its component. If a good status is received in a nominal amount of time, the component is considered operational. If the component does not respond, the guardian restarts the component.



The component also issues heartbeats to the core processing node to indicate that it is operational. A WatchDog component in the core processing node ensures the heartbeats are regular and available. Otherwise, the core node marks the component as inoperable.

Components interact using a guaranteed messaging system. A client (agent or message hub) sends a message and the receiver sends an acknowledgment that the message has been processed. This procedure guarantees that even a process failure would not prevent a message from eventually arriving at the intended destination.

Details specific to each component are provided below.

Agent

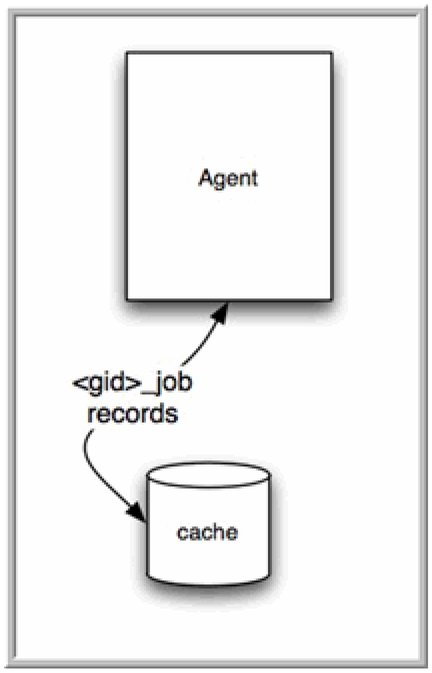
The agent runs as a Windows service or Linux/Unix daemon. The core sends a request to the agent to perform a function. The agent processes the request, gathers data about the operation of the client machine, and sends back status and results. It performs these functions by exchanging messages with the core.

Once an agent has registered with the core processing node, you can view it by selecting the type you want to view from the Resources section of the navigation pane. A list displays showing all the registered agents of the type you selected. See Agent-Based Resources for more information.

If an agent fails, its guardian process restarts it. The agent then attempts to determine what tasks or functions were in process at the time of failure. In order to support such a determination, agent task processing includes the following steps:

- Each time the agent receives a task, it writes to cache a record called [guid]_job, where [guid] is a unique tracking number assigned to the task instance.
- As the task runs, the agent updates the [guid]_job record with status information.
- When the task run completes, the agent deletes the [guid]_job record.
- If an agent is restarted, it looks in the cache for [guid]_job records. If any are found, the agent looks at the status. If the record indicates that the job is supposed to be running, the agent searches the system to locate it. If the agent is able to locate the task and resume tracking, it continues and marks the task resumed. If it is not able to resume tracking a task, it returns a message to the core, setting the status of the task instance to IN-DOUBT. This then requires manual follow-up to determine the state of the process.

As illustrated below, the agent reads/writes a record to its agent/cache directory for each task instance it manages.



Message Hub

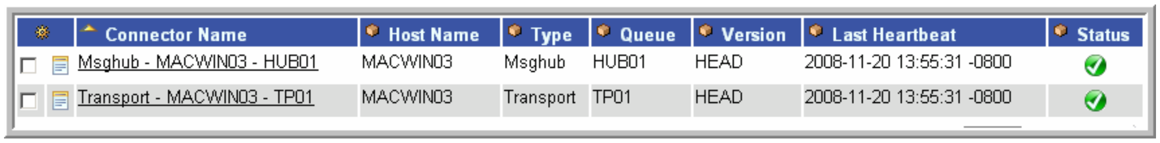
The Message Hub runs as a Windows service or Linux/Unix daemon. It listens for input messages from the agents (arriving via a transporter) and places them on input message queues for the node. It also polls the output message queue for messages to construct and send to agents. You can run one or more message hubs. Hubs can be on different nodes, but require access to the same database.

The message hub startup process is described below.

1. The message hub starts up initially in **Available** mode.
2. Its heartbeats reach the core and the core decides which hub is to be **Active**.
3. If instructed to do so, the message hub sets itself to Active and connects to the transport.

During an outage of the message hub, the core assigns the survivor as the Active message hub. In the case of HUB01 failing, HUB01 on the backup system will convert to Active mode and HUB01 on the initial system will be in Available mode when it eventually restarts.

Once a message hub has registered with the core processing node, the node creates a record of it in the database. You can view all registered message hubs by selecting **Resources > Connectors** from the navigation pane. The following example shows a message hub called MACWIN03 - HUB01.



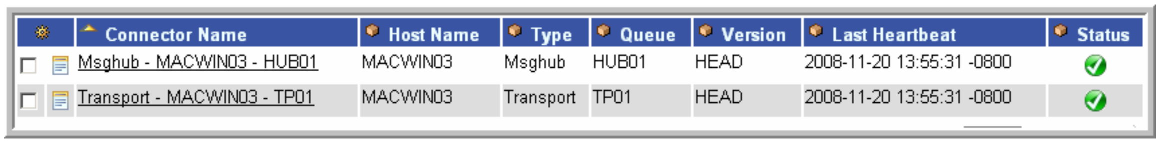
	Connector Name	Host Name	Type	Queue	Version	Last Heartbeat	Status
	Msghub - MACWIN03 - HUB01	MACWIN03	Msghub	HUB01	HEAD	2008-11-20 13:55:31 -0800	✓
	Transport - MACWIN03 - TP01	MACWIN03	Transport	TP01	HEAD	2008-11-20 13:55:31 -0800	✓

Transporter

The transporter runs as a Windows service or Linux/Unix daemon. The transporter provides a group communications capability using the publish-subscribe model to exchange messages among subscribed clients. All messaging is performed using TCP protocol. You can run one or more transporters and they are generally established on different machines. The transporter currently uses the following ports: 4803 for normal client communications, 4804 & 4805 for inter-transport communications.

Each client is provided a list of possible transporters to use. If one fails and reconnect attempts fail, the client switches to the next on the list. Transporters can restart at any time and be considered candidates for messaging traffic.

Once a transporter has registered with the core processing node, the node creates a record of it in the database. You can view all registered transporters by selecting **Resources > Connectors** from the navigation pane. The following example shows a transporter called MACWIN03 - TP01.



	Connector Name	Host Name	Type	Queue	Version	Last Heartbeat	Status
	Msghub - MACWIN03 - HUB01	MACWIN03	Msghub	HUB01	HEAD	2008-11-20 13:55:31 -0800	✓
	Transport - MACWIN03 - TP01	MACWIN03	Transport	TP01	HEAD	2008-11-20 13:55:31 -0800	✓

Configuring Notifications Based on Component Status

You can configure Opswise to generate email or SNMP notifications based on the status of your agents, cluster nodes (servers), and connectors (message hub and transporter). See Sending Notifications on Opswise Component Status for instructions.

Setting Up High Availability

Configuring the Core Processors (Cluster Nodes)

For both core processors running, you can use a single **glide.properties** file with the following lines to point to the single database image:

```
glide.db.rdbms = mysql
glide.db.url = jdbc:mysql://10.10.1.1/
```

Configuring the Outboard Components

Click here for a description of transporter properties.

The described configuration is supported by outboard version 1.5.0 build 107 and above.

Click here for a description of message hub properties.

Opswise Properties

Properties Set at Installation

Opswise uses several properties files that allow you to configure options that control its operation. In general, there is one property file for each of the Opswise components: core, msghub, transporter, and agent. These properties files are filled in by the installation process, based on information provided by the user during installation. If necessary, you can change these settings after the installation by editing the files. If you do so, the changes will take effect the next time you start the related process.

Opswise glide.properties

This properties file is read by the core processor, which is started by Tomcat. The table below describes the properties that must be set before Tomcat is started. Initially, the core installation process sets these properties using the values you supply for that command.

The **glide.properties** file resides here:

```
[tomcat directory]\webapps\opswise\WEB-INF\properties
```

Property Name	Description	Default
(For MySQL)		
`glide.db.rdbms=mysql`	Specify this if you are using a MySQL database.	
`glide.db.url=jdbc:mysql://localhost/`	Specify this if you are using a MySQL database.	
(For SQLServer)		
`glide.db.rdbms=sqlserver`	Specify this if you are using a SQLServer database.	
`glide.db.url=jdbc:sqlserver://localhost:1433;DatabaseName=opswise`	Specify this if you are using a SQLServer database.	
(For LDAP)		
`glide.ldap.groups.filter_indirect`	When set to true, any Groups synchronized indirectly (that is, through a User's memberOf attribute) will honor the Group search filter and Group OU filters under the LDAP Advanced Settings section.	false

`glide.ldap.groups.single_parent_per_child`	**IMPORTANT** This property should only be set to true if your Groups being synchronized from AD have at most one parent Group. When synchronizing Groups, the default behavior in Automation Center is to copy the members of a Sub Group into the Parent Group. When this property is set to true, Automation Center assumes that each Group has, at most, a single Parent Group and will use the Parent field on the Group definition to maintain the hierarchy instead of copying members.	false
(For all databases)		
`glide.db.user=`	Login ID that Opswise will use to log in to your database.	root
`glide.db.password=`	Password that Opswise will use to log in to your database.	(None)
`glide.db.name=`	Name for the Opswise database.	opswise
`glide.sys.boot_script=`	Script used to initialize the database. **Internal use only; do not change.**	zboot_opswise.js
`glide.db.pooler.connections=`	Minimum number of connections that can remain idle in the pool without extra connections being created, or zero to create none.	25
`glide.db.pooler.connections.max=`	Maximum number of active connections that can be allocated from this pool at the same time, or negative for no limit.	100
`glide.servlet.port=`	Port number used by Tomcat.	8080
`opswise.hub.host=`	Hostname or IP address of the machine where the MsgHub resides.	localhost
`opswise.hub.host=`	Port number used for MsgHub communications.	6776

`opswise.trustmanager.algorithm=`	Java trust manager algorithm. For IBM AIX, value must be IbmX509.The default works for all other platforms.	SunX509
`opswise.trustmanager.provider=`	Java trust manager provider. For IBM AIX, value must be IBMJSSE2. The default works for all other platforms.	SunJSSE

Sample **glide.properties** file:

```
# DB
glide.db.rdbms=mysql
glide.db.url=jdbc:mysql://localhost/
# MYSQL
# glide.db.rdbms=mysql
# glide.db.url=jdbc:mysql://localhost/
# MS SQLSERVER
# glide.db.rdbms=sqlserver
# glide.db.url=jdbc:sqlserver://localhost:1433;DatabaseName=opswise
# ORACLE
# glide.db.rdbms=oracle
# glide.db.url=jdbc:oracle:thin:@//localhost:1521/@oracle.db.name@
#
# COMMON
#
# hub host & port properties
opswise.hub.host=localhost
opswise.hub.port=6776
#
# trust manager algorithm & provider
#opswise.trustmanager.algorithm=SunX509
#opswise.trustmanager.provider=SunJSSE
#
glide.db.user=root
glide.db.password=pswd
glide.db.name=opswise
glide.sys.boot_script=zboot_opswise.js
glide.db.pooler.connections=2
glide.db.pooler.connections.max=40
glide.servlet.port=8080
glide.ui.session_timeout=60
```

Message Hub Properties msghub.props

The table below describes the properties that must be set prior to starting the Message Hub. Normally these properties are set when you run the outboard installer using the values you supply for that command.

The **msghub.props** file resides here:

```
[agent install directory]\etc
```

Property Name	**Description**	**Default**
`config.loglvl=[s |t]`	Level of logging for this process: • s = severe • e = error • w = warning • i = informational • d = debugging • t = trace	i (informational)

`config.txtdebug=[y|n]`	Specifies whether all messages are logged to the console, as opposed to being written to the log.	n
`network.transports=`	Port and network address of the Opswise Transporter(s) used for messaging. When using multiple Transporters, separate each port@network address by a comma or semicolon.	4803@127.0.0.1
`network.core=`	Queue name for the message hub.	HUB01
`network.tpqname=`	(Deprecated.)	TP01
`network.gmsg=`	Specifies whether Guaranteed messaging is used.	false
`core.bind_iface=`	Listening IPaddress of Hub for Core/Hub communications	0.0.0.0
`core.bind_port=`	Listening port of Hub for for Core/Hub communications	6776
SSL Properties		
`enable_ssl=`	Value = **yes** or **no**. Enables and disables SSL protocol for network communications between the Hub and the Opswise core processor. Supported on Linux/Unix, Windows, z/OS.	No.

Sample **msghub.props** file:

```
#
config.loglvl=I
#
network.transports=4803@127.0.0.1
network.core=HUB01
network.tpqname=TP01
#
#  Core connection properties
#
core.bind_port=6776
core.bind_iface=0.0.0.0
#
#  SSL is enabled with a value of YES.  The default is NO.
#
ssl.enable=no
```

Transporter Properties transport.props

The table below describes the properties that must be set prior to starting the Transporter. Normally these properties are set when you run the outboard installer using the values you supply for that command.

The **transport.props** file resides here:

```
[agent install directory]\etc
```

Property Name	Description	Default
`transport1.ipaddr=`	Network address for one of the Transporters in the configuration. This parameter is generally specified along with the companion parameters, transport[n].port and transport[n].name. Configurations with a single transport will have only one set of entries. For High Availability configurations, additional sets of entries are prefixed with "transport[n]" where [n] is a sequentially advancing integer.	127.0.0.0
`transport1.port=`	Port number used for client to client communication by this transporter.	4803
`transport1.name=`	Queue name used by this Transporter.	TP01
`debug.flags=`	ALL - show all messages. ACM - Access control log messages. CONFIGURATION - configuration processing log messages. DEBUG - Debug messages. DATA_LINK - Low-level communication log messages. EVENTS - Event processing log messages. EXIT - Process termination log messages. FLOW_CONTROL - Message flow control log messages. GROUPS - Groups processing log messages. HOP, TCP_HOP, ROUTE, RING - Inter-transport communications log messages. MEMBERSHIP - Subscribers membership log messages. MEMORY - Memory management log messages. NETWORK - Networking protocol log messages. OBJ_HANDLER - Objects processing log messages. PRINT - Informational messages. PROTOCOL - SPREAD protocol messages. QOS - QoS log messages. SYSTEM - System messages. SESSION - Client session management messages. STATUS - Status messages. SKIPLIST - Skiplist processing. SECURITY - Security-related messages. TIMESTAMP - Show timestamps at the beginning of the log messages.	SESSION PRINT EXIT MEMBERSHIP
`config.loglvl=[s|t]`	Not currently used. See debug.flags above for logging options.	
SSL Properties		
`ssl.public.cert=`	Specifies the path to the PEM-formatted digital certificate file to be used by the Transport for SSL network communications between the Transporter, Hub and agents. By providing a digital certificate and private key, the SSL protocol is enabled. When no digital certificate or private key is specified, SSL is disabled. Supported on Linux/Unix, Windows, z/OS.	(None)
`ssl.private.key=`	Specifies the path to the PEM-formatted private key file that corresponds to the digital certificate specified in the ssl.public.cert property. By providing a digital certificate and private key, the SSL protocol is enabled. When no digital certificate or private key is specified, SSL is disabled. If the private key file is password protected, the password must be specified with the ssl.keypassword property. Supported on Linux/Unix, Windows, z/OS.	(None)

`ssl.keypassword=`	Provides the password for a password-protected private key file specified with the ssl.private.key property. Supported on Linux/Unix, Windows, z/OS.	(None)

Sample **transport.props** file:

```
#win01
transport1.ipaddr=192.168.30.64
transport1.port=4803
transport1.name=TP01
#win03
transport2.ipaddr=192.168.30.66
transport2.port=4803
transport2.name=TP02
#win02
transport3.ipaddr=192.168.30.65
transport3.port=4803
transport3.name=TP03


debug.flags= SESSION PRINT EXIT MEMBERSHIP  NETWORK GROUPS TIMESTAMP

ssl.private.key=c:\opswise\etc\transport.pkey
ssl.public.cert=c:\opswise\etc\transport.cert
```

Universal Automation Center Agent Properties

Windows	Universal Automation Center Agent (UAG) properties are located in file **uags.conf**, which resides here: `DriveID:\ProgramData\Universal\conf\uags.conf`
Linux/Unix	Universal Automation Center Agent (UAG) properties are located in file **uags.conf**, which resides here: `/etc/universal/uags.conf`
z/OS	Configuration information for all components is in the configuration library, **UNV.UNVCONF**.

Click here for details on each option.

Sample **uags.conf** file:

```
installation_directory "C:\Program Files\Universal\UAGSrv"
message_level INFO
security DEFAULT
automation_center_transports 4803@127.0.0.1
automation_center_core HUB01
agent_clusters "Opswise - Default Linux/Unix Cluster, Opswise - Default Windows Cluster"
loglvl I
netname OPSAUTOCONF
```

Properties Set at Any Time

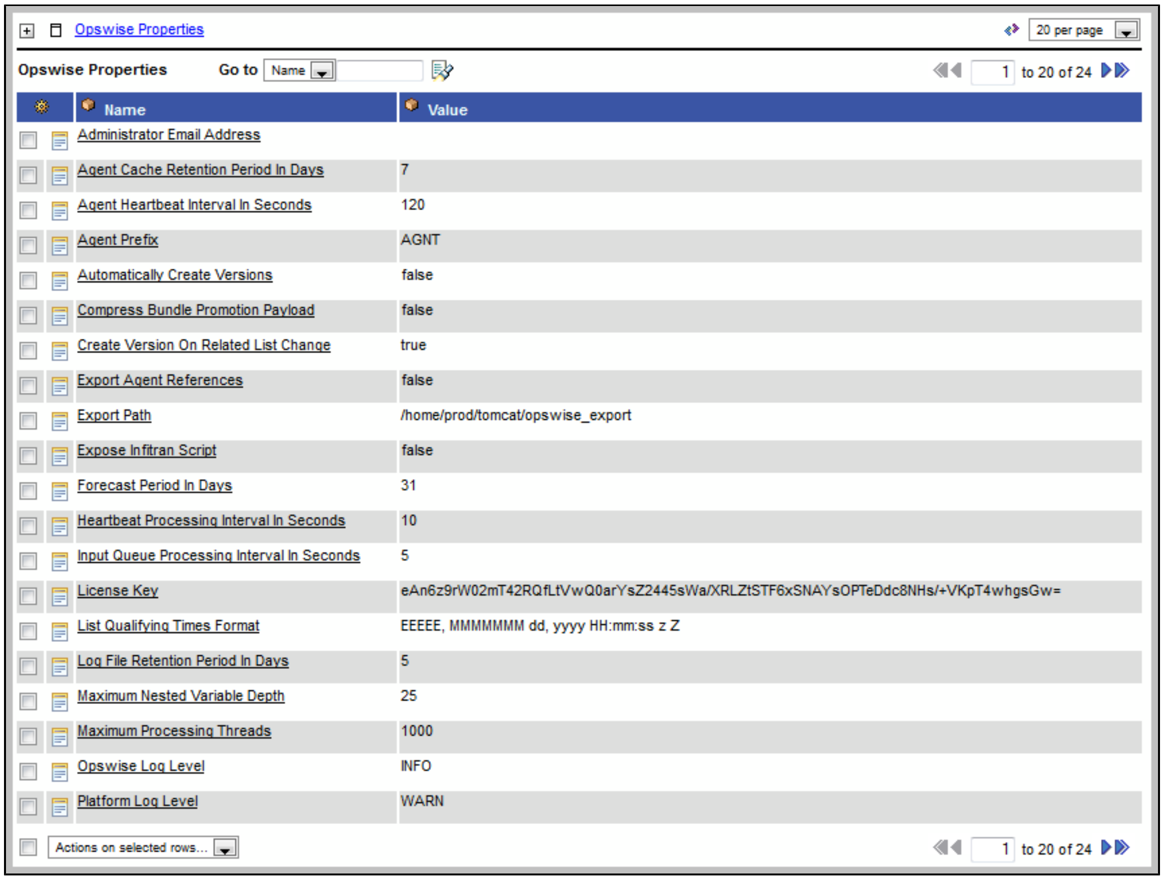
You can set the following properties any time after the system is in operation.

Opswise Properties

Opswise Properties are system properties that you can customize at any time. These properties do not reside in a properties file; you set them through the user interface.

To set Opswise system properties:

1. Select **Automation Center Administration > Configuration > Properties** from the navigation pane. The Opswise Properties screen displays.



Name	Value
Administrator Email Address	
Agent Cache Retention Period In Days	7
Agent Heartbeat Interval In Seconds	120
Agent Prefix	AGNT
Automatically Create Versions	false
Compress Bundle Promotion Payload	false
Create Version On Related List Change	true
Export Agent References	false
Export Path	/home/prod/tomcat/opswise_export
Expose Infitran Script	false
Forecast Period In Days	31
Heartbeat Processing Interval In Seconds	10
Input Queue Processing Interval In Seconds	5
License Key	eAn6z9rW02mT42RQfLtVwQ0arYsZ2445sWa/XRLZtSTF6xSNAYsOPTeDdc8NHs/+VKpT4whgsGw=
List Qualifying Times Format	EEEEE, MMMMMMM dd, yyyy HH:mm:ss z Z
Log File Retention Period In Days	5
Maximum Nested Variable Depth	25
Maximum Processing Threads	1000
Opswise Log Level	INFO
Platform Log Level	WARN

2. Click a property name to open the record for the property you want to change.

The following table describes the Opswise Properties:

Property Name	Description	Default
`Administrator Email Address`	Address specified in emails for administrator action.	(None)
`Agent Cache Retention Period in Days`	Number of days that cache files (stdout, stderr) are retained by the system.	7
`Agent Heartbeat Interval in Seconds`	Number of seconds between each heartbeat message sent by the agent to the core processor.	120

`Agent Prefix`	Prefix appended to the Queue name for newly registered agents. A 4 digit number is appended to this prefix.	AGNT
`Automatically Create Versions`	Affects system behavior when you make updates to records in your Opswise database, such as changing a task definition. If true, specifies that you want Opswise to retain copies of previous versions.	true
`Compress Bundle Promotion Payload`	Instructs Opswise to compress record bundles during a promotion.	false
`Create Version On Related List Change`	Specification for whether or not a record version will be created if the user changes a record associated with the current record. For example, if true, the system will create a version of the task when the user changes a task variable.	true
`Export Agent References`	Specification for whether or not Opswise will export referenced agents when exporting definition XMLs with the "Export References" option.	false
`Export Path`	Pathname where exported xml files are written. This applies only when using the List Export-->XML operations, not the bulk export.	/home/prod/tomcat/opswise_export or C:\apache-tomcat-6.0.32\opswise_export
`Expose Infitran Script`	Debugging use only. When Opswise launches a file transfer on an Infitran installation, it prepares a script. If troubleshooting is necessary, enabling this property allows the user to view the script in the Output tab on the task instance.	false
`Forecast Period in Days`	Number of days to be included in a trigger forecast. See Displaying Trigger Forecast Information.	31
`Heartbeat Processing Interval in Seconds`	Interval that the various watchdogs inspect the heartbeats of attached components.	10
`Input Queue Processing Interval in Seconds`	Number of seconds that are used to poll the incoming message queues.	5

`License Key`	License key for your installation; provided to you by your Opswise representative.	
`List Qualifying Times Format`	Format you want the system to use when listing qualifying times for Time and Cron Triggers. See List Qualifying Times.	EEEEE, MMMMMMM dd, yyyy HH:mm:ss z Z
`Log File Retention Period in Days`	Number of days that log files (opswise core, or outboard) are retained by the system.	5
`Maximum Nested Variable Depth`	Maximum number of nested variables allowed.	25
`Maximum Processing Threads`	Maximum number of processing threads used.	1000
`Opswise Log Level`	Level of logging for the Opswise core, as follows: • ALL • TRACE • DEBUG • INFO • WARN • ERROR • SEVERE • OFF	INFO
`Platform Log Level`	Level of logging for the platform where Opswise operates, as follows: • ALL • TRACE • DEBUG • INFO • WARN • ERROR • OFF	WARN
`Retrieve Output Default Maximum Lines`	Sets a limit for the number of lines retrieved when Automatic Output Retrieval is enabled on a task.	100
`Server Fail Over in Seconds If No Message Hub`	In a high availability configuration, if the communications between the Core and the Message Hub (MsgHub) are disrupted, the Core waits the number of seconds configured here before bringing itself down and allowing the backup Core server to take over.	120

`Start Server Paused`	When true, the start server process brings up the server in pause mode.	false
`Stop Unknown Application Monitors`	Stops any application monitors currently running on an Agent if the Controller is no longer managing those monitors (Windows and Linux/Unix only)	false

Installation and Upgrade

Installation and Upgrade Overview

If you are installing Opswise for the first time, see Installation Steps - Overview. If you are upgrading from a previous version of Opswise, follow the Upgrading procedure.

Installation Steps

Installation Steps - Overview

Introduction

This page provides links to Opswise installation instructions. If you are upgrading from a previous version of Opswise, click here for instructions.

Create Home Base Directory

We recommend you create a directory that will serve as a "home base" for these installation procedures. When you download software, copy it to this directory as a central repository. For our examples here, we are using the directory "opsinstall".

Install Prerequisite Software

Before downloading and installing Opswise, install the following prerequisites:

1. Java Runtime Environment
2. Apache Tomcat
3. Database

Download Opswise Components

Click here for steps to download the Opswise software components.

Install and Verify the Opswise Core

The Opswise Core is a Java application running within Apache Tomcat. For this reason, the software and installation procedure is basically the same for all platforms.

Install Message Hub, Transporter, and CLI (Outboard Components)

The Opswise Outboard comprises the components that provide communications with the Opswise agent - the transporter and message hub - along with the command line interface. The software and instructions differ depending on your platform. Click the link below for the platform you want to install:

- Windows
- Linux (Redhat, CentOS)
- IBM z/Linux
- HP-UX Itanium
- Unix AIX
- Solaris Intel
- Solaris Sparc

Install Universal Automation Center Agent

Click any of the following links for details and instructions on installing Universal Automation Center Agent (as a component of a Workload Automation 5 installation package):

- Windows
- z/OS
- Linux/Unix

Start/Stop Opswise Components

- Windows
- Linux (Suse, Redhat, CentOS)
- z/Linux
- HP-UX
- Unix AIX
- Solaris Intel
- Solaris Sparc
- z/OS

Verify the Outboard Installation

As a final step, you should verify that your outboard components are installed, running, and communicating with the Opswise core. Click here for instructions.

Copy of Installation Steps - Overview

Introduction

This page provides links to Opswise installation instructions. If you are upgrading from a previous version of Opswise, click here for instructions.

Network Requirements

Ports configured for Workload Automation 5 components and prerequisites cannot be blocked by a firewall.

The following table identifies the default ports, which can be changed during installation / configuration:

component / prerequisite	default port
MySQL	3306
Universal Broker	7887
Transporter	4803
Tomcat / Opswise Controller	8080
Opswise Controller to Message Hub	6776
etc.	
etc.	

Create Home Base Directory

We recommend you create a directory that will serve as a "home base" for these installation procedures. When you download software, copy it to this directory as a central repository. For our examples here, we are using the directory "opsinstall".

Install Prerequisite Software

Before downloading and installing Opswise, install the following prerequisites:

1. Java Runtime Environment
2. Apache Tomcat
3. Database

Download Opswise Components

Click here for steps to download the Opswise software components.

Install and Verify the Opswise Core

The Opswise Core is a Java application running within Apache Tomcat. For this reason, the software and installation procedure is basically the same for all platforms.

Install Message Hub, Transporter, and CLI (Outboard Components)

The Opswise Outboard comprises the components that provide communications with the Opswise agent - the transporter and message hub - along with the command line interface. The software and instructions differ depending on your platform. Click the link below for the platform you want to install:

- Windows
- Linux (Redhat, CentOS)
- IBM z/Linux
- HP-UX Itanium
- Unix AIX
- Solaris Intel
- Solaris Sparc

Install Universal Automation Center Agent

Click any of the following links for details and instructions on installing Universal Automation Center Agent (as a component of a Workload Automation 5 installation package):

- Windows
- z/OS
- Linux/Unix

Start/Stop Opswise Components

- Windows
- Linux (Suse, Redhat, CentOS)
- z/Linux
- HP-UX
- Unix AIX
- Solaris Intel
- Solaris Sparc
- z/OS

Verify the Outboard Installation

As a final step, you should verify that your outboard components are installed, running, and communicating with the Opswise core. Click here for instructions.

Apache Tomcat

Note
The download and installation procedure for Apache Tomcat may vary a bit for each platform.

Install Apache Tomcat

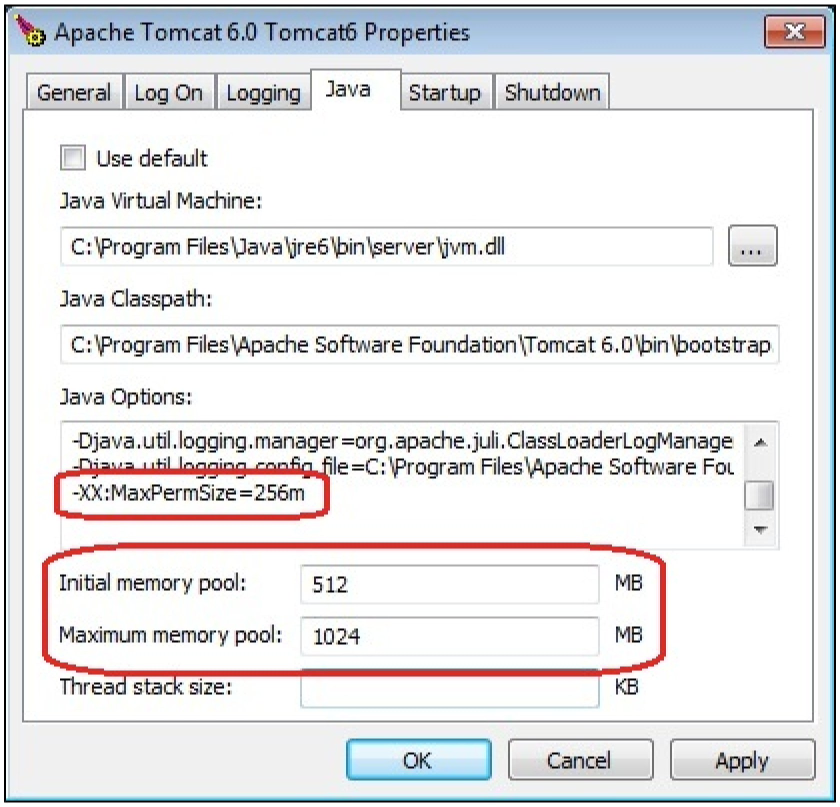
Perform the following steps to install Apache Tomcat:

1. Choose the appropriate method from the following options:
   - For Windows, we recommend using the GUI installer to create the Apache Tomcat Service:
     a. Download the "32-bit/64-bit Windows Service Installer" from Tomcat 6.0.xx.
     b. Follow the instructions to install the package.
   - For Windows or Linux/Unix, you can download a tar.gz or zip package that you unzip into a directory:
     a. Download an appropriate package from Tomcat 6.0.xx.
     b. Follow the instructions to unzip the appropriate package (tar.gz or zip) into a directory on your file system.
   - For Redhat and Centos distributions, you can also use the "yum" installer.
2. Choose one of the methods below to update the JVM variables to the following minimum values:

   ```
   CATALINA_OPTS="-Xms512m -Xmx1024m -XX:MaxPermSize=256m"
   ```

   You can use one of these methods:

   - In any operating system, you can add this value to $TOMCAT_HOME/bin/catalina.bat or $TOMCAT_HOME/bin/catalina.sh as the first line after the comment box.
   - In any operating system, you can add this parameter to the environment variables.
   - In Windows, if you installed Tomcat as a Windows service, you can set values using the $TOMCAT_HOME\bin\tomcatw.exe GUI tool, as shown below.
     Enter the parameters as follows:
     - Initial memory pool = minimum heap size (Xms)
     - Maximum memory pool = Maximum heap size (Xmx)
     - Enter the MaxPermSize parameter as a Java Option



Use default

Java Virtual Machine:
C:\Program Files\Java\jre6\bin\server\jvm.dll

Java Classpath:
C:\Program Files\Apache Software Foundation\Tomcat 6.0\bin\bootstrap

Java Options:
```
-Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManage
-Djava.util.logging.config.file=C:\Program Files\Apache Software Fou
-XX:MaxPermSize=256m
```

Initial memory pool:	512	MB
Maximum memory pool:	1024	MB
Thread stack size:		KB

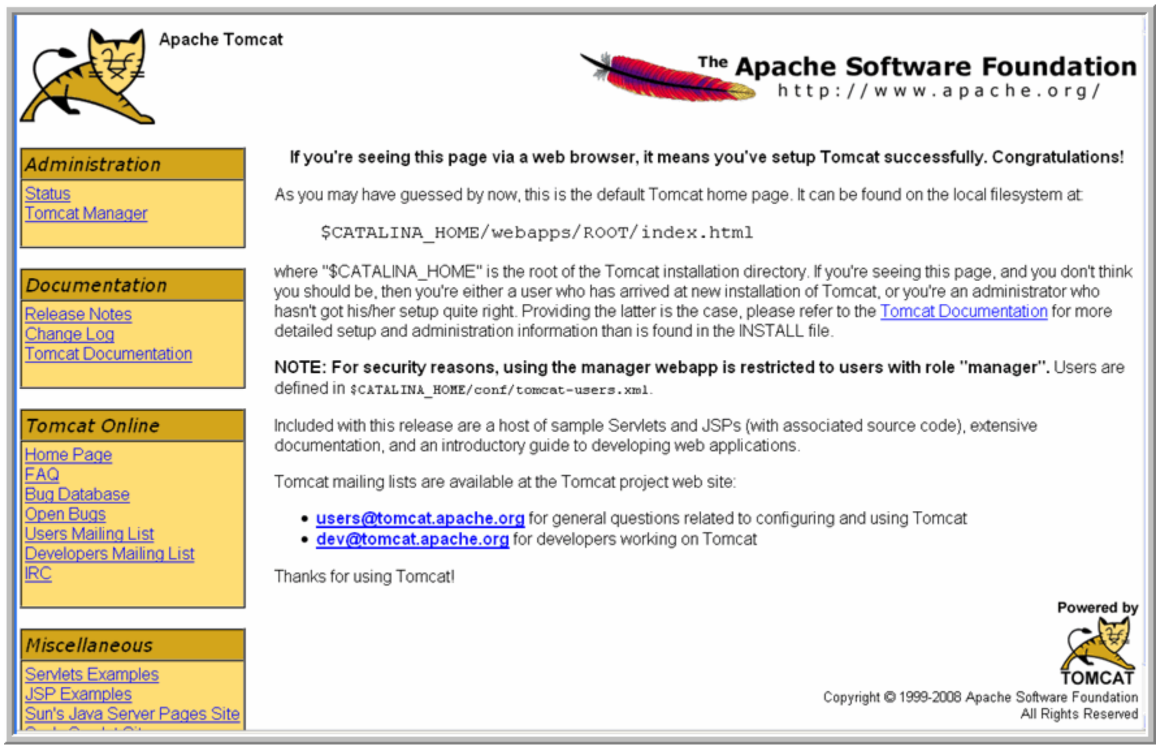
Note
Later, after you start Tomcat and log in to Opswise, you can validate these settings by running the memory_usage.js script, as follows:

1. From the navigation pane, select **Configuration > Maintenance Scripts**.
2. Run the memory_usage.js script. The min and max numbers on the top line (Heap) should be similar to the above settings.

Start and Validate Apache Tomcat

Perform the following steps to start and validate Apache Tomcat:

1. Tomcat is normally run as a system service or daemon. You can start Tomcat using the standard method for your operating system or by using a script, as follows:
   - In Windows, we recommend you use Windows Services to start Tomcat. Or, you can start Tomcat from the command line as follows:
     `net start`
   - In Linux, start the Tomcat daemon using the script placed in the /etc/init.d directory for Tomcat.
     `service start`
   - In either system, you can start the service using the $TOMCAT_HOME/bin/startup.bat or $TOMCAT_HOME/bin/startup.sh scripts.
2. Open a browser and go to the following URL:
   http://localhost:8080
3. The following screen displays, verifying that you have successfully installed and started Tomcat:



If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

As you may have guessed by now, this is the default Tomcat home page. It can be found on the local filesystem at:

```
$CATALINA_HOME/webapps/ROOT/index.html
```

where "$CATALINA_HOME" is the root of the Tomcat installation directory. If you're seeing this page, and you don't think you should be, then you're either a user who has arrived at new installation of Tomcat, or you're an administrator who hasn't got his/her setup quite right. Providing the latter is the case, please refer to the Tomcat Documentation for more detailed setup and administration information than is found in the INSTALL file.

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager". Users are defined in `$CATALINA_HOME/conf/tomcat-users.xml`.

Included with this release are a host of sample Servlets and JSPs (with associated source code), extensive documentation, and an introductory guide to developing web applications.

Tomcat mailing lists are available at the Tomcat project web site:

- **users@tomcat.apache.org** for general questions related to configuring and using Tomcat
- **dev@tomcat.apache.org** for developers working on Tomcat

Thanks for using Tomcat!

Copyright © 1999-2008 Apache Software Foundation
All Rights Reserved

Database

Overview

Opswise can use a database space of an existing database or you can install a database specifically for Opswise. We recommend an initial size of 100MB.

The following database management systems are supported:

- MySQL
- MS SQLServer
- Oracle

MySQL

1. Download MySQL installation instructions here.
2. Download MySQL from here (Windows only)
   - For Windows, select Windows (x86, 32-bit), MSI Installer
   - For Unix and Linux, you can use a tar.gz download or select a systems package installer appropriate for your environment, such as Yum.
3. Install MySQL as per the instructions.
4. Make a note of the userid and password to be used later when installing the Opswise core.
5. The database will be created automatically when you select MySQL during the Opswise installation process.

Microsoft SQL Server

Note
If you are using Microsoft SQL Server 2005, when you create the database, set it to be CASE INSENSITIVE.

1. Download and install MS SQLServer as per the Microsoft documentation.
2. Create the Opswise database. You can use any legal name, but we recommend the name "opswise."
3. Make a note of the userid and password to be used later when installing the Opswise core.

Oracle

1. Download and install Oracle as per the Oracle documentation.
2. Create the Opswise database. You can use any legal name, but we recommend the name "opswise."
3. Make a note of the userid and password to be used later when installing the Opswise core.

Java Runtime Environment

To download the Java Runtime Environment (JRE):

1. Access the Oracle site for Java JREs

```
http://www.oracle.com/technetwork/java/javase/downloads/index.html
```

2. Under Java, select JRE.
3. Select your operating system and download.

Note
A minimum JRE level of 1.6.0 is required.

Downloading Opswise Software

To download software, you need to log in to the Stonebranch web site. If you do not have a login, you can register for access on the login page.

You may also need to download maintenance software along with the Opswise software.

To download the Opswise software:

1. Go to the Current Product Downloads page on the Stonebranch website.
2. The Current Product Downloads page provides a series of tables that allow you to select the software and platforms you need. To install the entire Opswise package, you need the following:
    - Workload Automation N.N.N.N Agents
    - Opswise Automation Center Server (Core) N.N.N.N
    - Opswise Automation Center Outboard N.N.N.N
3. Locate the download file appropriate for your platform, if relevant, and click the **DOWNLOAD** button.
4. In response to the prompt, click **Save File** and browse to your save location.
5. Repeat this process for each component.
6. Return to the Installation Steps - Overview.

To download maintenance software:

1. Go to the Maintenance page on the Stonebranch website.
2. Select and download Opswise maintenance software, if any.
3. Extract the package and follow the instructions provided in the readme file.

Installing the Opswise Core

Introduction

These instructions describe how to install the Opswise Core. For a list of Opswise components and a sample configuration, click here.

This procedure assumes you have already completed the following:

- Installed the prerequisite software
- Downloaded Opswise components

Uncompress the Downloaded Tar File

Uncompress the Opswise Server (Core) tar file as follows:

Linux/Unix	`tar xvf opswise-core-N.N.N.N.tar`
Windows	Use an appropriate archiving / unzipping product.

Running the Installation Script

When you run the Opswise core installer, you need to provide certain information so that Opswise can access the database. You will do so by providing command switches to one of the following install commands, depending on your platform.

To start the installer, type:

Windows	`> install-core.bat`
Linux	`> sh install-core.sh`

Command Switches

The following table describes command line switches.

Note
All command switches are case-sensitive.

Command Switch	Req'd Y/N	Description
`--tomcat-dir`	Yes	Path to the Tomcat installation directory (contains the directories:/bin, /conf, /logs, webapps).
`--core-file`	Yes	Full path of the opswise.war file you downloaded.
`--rdbms`	Has default	Database type.Options are: mysql, sqlserver, oracle.

`--dburl`	Has default	JDBC connect URL. Format:
		`jdbc:[database type]://localhost`
		Examples:
		MYSQL
		`glide.db.rdbms=mysql`
		`glide.db.url=jdbc:mysql://localhost/`
		MS SQLSERVER
		`glide.db.rdbms=sqlserver`
		`glide.db.url=jdbc:sqlserver://localhost:1433;DatabaseName=opswise`
		ORACLE
		`glide.db.rdbms=oracle`
		`glide.db.url=jdbc:oracle:thin:@//localhost:1521/opswise`
		Where opswise is the database name (MS SQLServer and Oracle).

`--dbname`	Has default	Opswise database name.
`--dbuser`	Yes	Database user name.
`--dbpass`	Yes	Database user's password.
`--hubhost`	Has default	Hostname or IPaddress of msghub
`--hubport`	Has default	Port to use to communicate with msghub

Examples

Shown below are sample commands for installing the core processor on Linux and Windows platforms, using defaults for the database:

Linux	`sh install-core.sh --tomcat-dir ~/tomcat --core-file ./opswise-N.N.N-build.NN.war --dbuser root --dbpass userpass`
Windows	`install-core.bat --tomcat-dir "c:\Program Files\Apache Software Foundation\Tomcat 6.0" --core-file opswise-N.N.N-build.NN.war --dbuser root --dbpass userpass`

Deploy the Opswise Release

In this step, you will start Tomcat, which starts Opswise and builds your database tables. This process takes several minutes. When it is complete, Opswise is started and ready to use. If Tomcat was already running when you ran the Opswise installer, you do not need to stop and restart it; this process will happen automatically after you launch the Opswise installer.

1. Start Tomcat as follows:
    - In Windows, we recommend you use Windows Services to start Tomcat. Or, you can start Tomcat from the command line as follows:

```
net start [name of Tomcat service]
```

In Linux, start the Tomcat daemon using the script placed in the /etc/init.d directory for Tomcat.

```
service [name of Tomcat service] start
```

In either system, you can start the service using the `$TOMCAT_HOME/bin/startup.bat` or `$TOMCAT_HOME/bin/startup.sh` scripts.

During this initial startup, Opswise builds the database tables, a process that takes several minutes. You can view details in the Tomcat window or monitor the Opswise log, as described below:

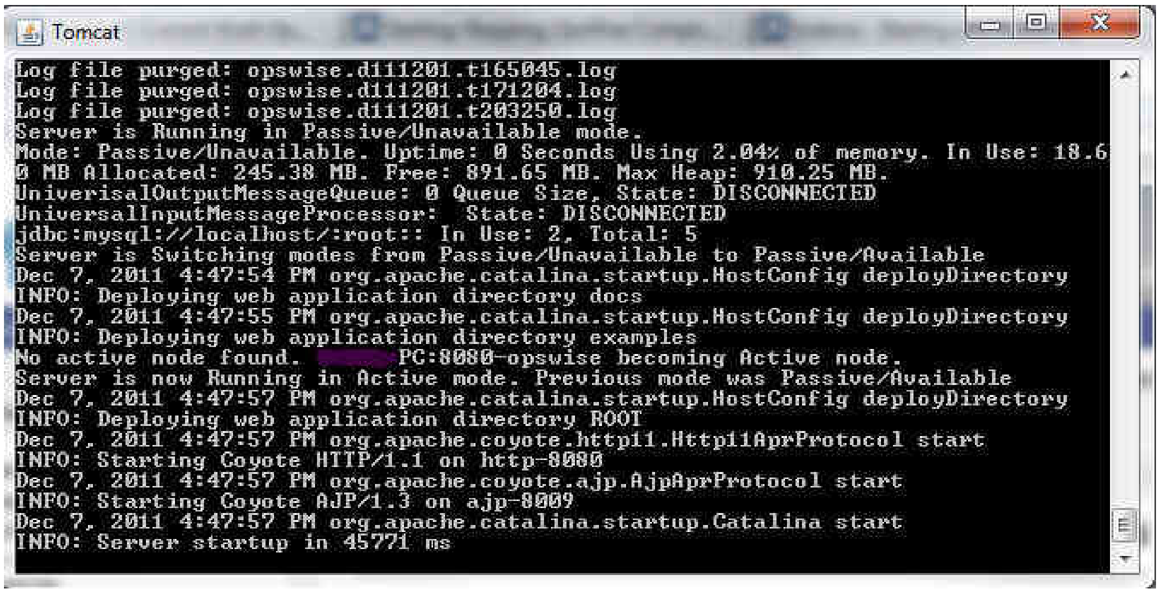
- Windows users can use a third-party tailing utility or open the log file using Notepad or other editor and scroll to the bottom to view the latest activity.

```
$TOMCAT_DIR/opswise_logs/opswise.log
```

- Linux/Unix users can tail the opswise.log to monitor the deployment process, as follows:

```
tail -f $TOMCAT_DIR/opswise_logs/opswise.log
```

- Do not continue until you see output in the log similar to the following:



```
Log file purged: opswise.d111201.t165045.log
Log file purged: opswise.d111201.t171204.log
Log file purged: opswise.d111201.t203250.log
Server is Running in Passive/Unavailable mode.
Mode: Passive/Unavailable. Uptime: 0 Seconds Using 2.04% of memory. In Use: 18.6
0 MB Allocated: 245.38 MB. Free: 891.65 MB. Max Heap: 910.25 MB.
UniverisalOutputMessageQueue: 0 Queue Size, State: DISCONNECTED
UniversalInputMessageProcessor:  State: DISCONNECTED
jdbc:mysql://localhost/:root:: In Use: 2, Total: 5
Server is Switching modes from Passive/Unavailable to Passive/Available
Dec 7, 2011 4:47:54 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory docs
Dec 7, 2011 4:47:55 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory examples
No active node found. [redacted]PC:8080-opswise becoming Active node.
Server is now Running in Active mode. Previous mode was Passive/Available
Dec 7, 2011 4:47:57 PM org.apache.catalina.startup.HostConfig deployDirectory
INFO: Deploying web application directory ROOT
Dec 7, 2011 4:47:57 PM org.apache.coyote.http11.Http11AprProtocol start
INFO: Starting Coyote HTTP/1.1 on http-8080
Dec 7, 2011 4:47:57 PM org.apache.coyote.ajp.AjpAprProtocol start
INFO: Starting Coyote AJP/1.3 on ajp-8009
Dec 7, 2011 4:47:57 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 45771 ms
```

When you see the phrase, **INFO: Server startup in nnnn ms**, the system is ready.

IBM AIX only. If you are running the core on IBM AIX, follow this procedure to change two default values in the glide.properties file, which is read by the core processor. The glide.properties file resides here:

```
/webapps/opswise/WEB-INF/properties
```

1. Change the following two parameters from the default to the IBM AIX values shown:

Property Name	Description	Default Value	IBM AIX Value
`> opswise.trustmanager.algorithm=`	Java trust manager algorithm.	SunX509	IbmX509
`> opswise.trustmanager.provider=`	Java trust manager provider.	SunJSSE	IBMJSSE2

2. Restart Tomcat.

You have now completed the install process and the server is running.

Verify the Installation

To make sure your system is installed and running properly:

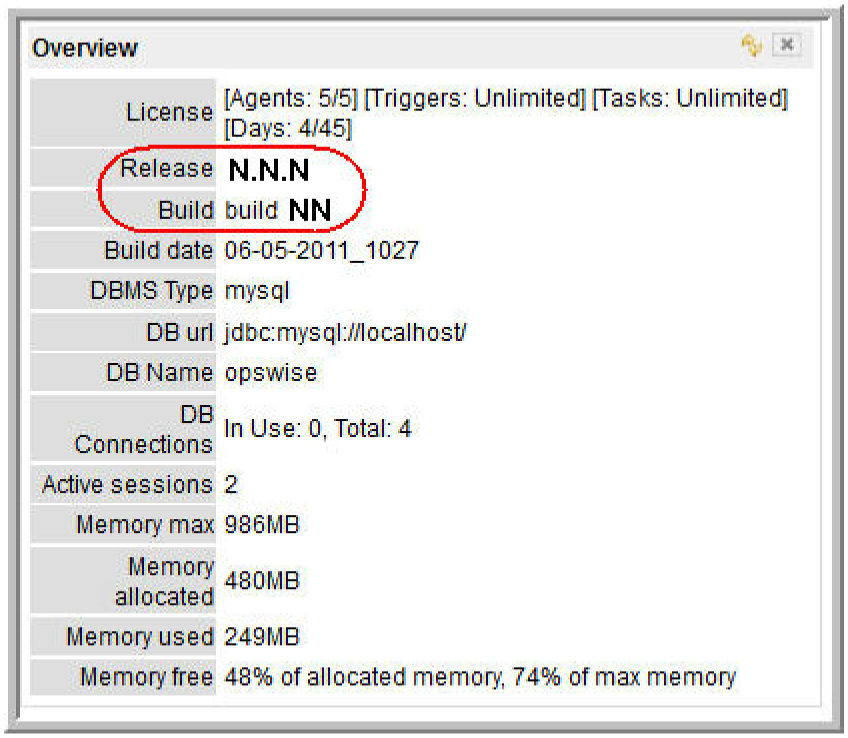
1. From your browser, open the Opswise page.

   ```
   http://localhost:8080/opswise
   ```

   Where localhost represents the machine name where you installed the server.

2. Log in with user ops.admin and no password.
3. Create a password when prompted. The system displays the home page.

4. The home page includes a gadget called Overview, which provides current system information. Check the Release information to verify that the latest version number is displayed, as shown in the following example.



Overview

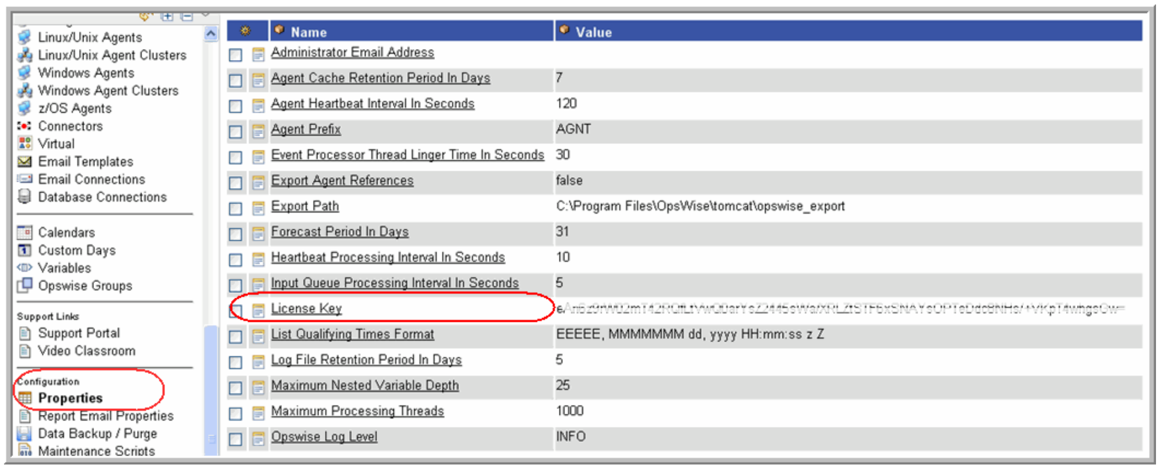
License	[Agents: 5/5] [Triggers: Unlimited] [Tasks: Unlimited] [Days: 4/45]
Release	N.N.N
Build	build NN
Build date	06-05-2011_1027
DBMS Type	mysql
DB url	jdbc:mysql://localhost/
DB Name	opswise
DB Connections	In Use: 0, Total: 4
Active sessions	2
Memory max	986MB
Memory allocated	480MB
Memory used	249MB
Memory free	48% of allocated memory, 74% of max memory

5. To get started using Opswise and become familiar with its features, we recommend you spend some time going through the exercises on the First Look page.

Apply the License Key

Although you do not normally need to enter a license key right away, at some point you will need to follow these steps to enter your key:

1. From the navigation pane, select **Configuration > Properties > License Key**.



Name	Value
Administrator Email Address	
Agent Cache Retention Period In Days	7
Agent Heartbeat Interval In Seconds	120
Agent Prefix	AGNT
Event Processor Thread Linger Time In Seconds	30
Export Agent References	false
Export Path	C:\Program Files\OpsWise\tomcat\opswise_export
Forecast Period In Days	31
Heartbeat Processing Interval In Seconds	10
Input Queue Processing Interval In Seconds	5
License Key	[illegible]
List Qualifying Times Format	EEEEE, MMMMMMM dd, yyyy HH:mm:ss z Z
Log File Retention Period In Days	5
Maximum Nested Variable Depth	25
Maximum Processing Threads	1000
Opswise Log Level	INFO

2. Enter your encrypted license key in the Value field and click **Update**.
3. Return to the Automation Center home page (click the Home icon) and review the License field on the Overview gadget to verify that the terms of your license are correct.

Before Installing Outboard

Before running the outboard installation procedures, you will need the following information:

- Opswise user name
- Root directory name for the Opswise software
- Transporter socket list.
- Port used by Transporter

Installing Opswise on AIX Unix

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does not, set it with the following command:

```
# chmod +x opswise-outboard-aix-powerpc-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-aix-powerpc-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-aix-powerpc-N.N.N-build.NN.sh --user opswise --user-home /home/opswise
--transports 4803@127.0.0.1
--tspport 4803 --bind-iface 0.0.0.0 --bind-port 6776 --install-transport --install-msghub
--install-cmdtools
```

Installing Opswise on HP-UX Itanium

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does not, set it with the following command:

```
# chmod +x opswise-outboard-hp-ux-ia64-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-hp-ux-ia64-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-hp-ux-ia64-N.N.N-build.NN.sh --user opswise --user-home /home/opswise
--transports 4803@127.0.0.1
--tspport 4803 --install-transport --install-msghub --install-cmdtools --bind-iface 0.0.0.0
--bind-port 6776
```

Installing Opswise on HP-UX PA-RISC

Introduction

These instructions assume you have already

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If not, set it with the following command:

```
# chmod +x opswise-outboard-hp-ux-hppa-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-hp-ux-hppa-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-hp-ux-hppa-N.N.Nrc3-build.NN.sh --user opswise --user-home /home/opswise
--transports 4803@127.0.0.1
--tspport 4803 --bind-iface 0.0.0.0 --bind-port 6776 --install-transport --install-msghub
--install-cmdtools
```

Installing Opswise on Linux

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does not, set it with the following command:

```
# chmod \+x opswise-outboard-linux-x86_64-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-linux-x86_64-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-linux-x86_64-N.N.N-build.NN.sh --user opswise --user-home /home/opswise
--transports 4803@127.0.0.1 --tspport 4803 --install-transport --install-msghub
--install-cmdtools
```

Installing Opswise on IBM zLinux

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does not, set it with the following command:

```
# chmod \+x opswise-outboard-linux-x86_64-N.N.N-build.NN.sh
```

Command Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
./opswise-outboard-linux-s390x-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-linux-s390x-N.N.N-build.NN.sh --user opswise --user-home /home/opswise
--transports 4803@127.0.0.1 --tspport 4803 --install-transport --install-msghub
--install-cmdtools
```

Installing Opswise on Solaris Intel

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does, not, set it with the following command:

```
# chmod +x opswise-outboard-SunOS-i386-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-SunOS-i386-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-SunOS-i386-N.N.N-build.NN.sh --user opswise --user-home /export/home/opswise
--transports 4803@127.0.0.1 --tspport 4803 --bind-iface 0.0.0.0 --bind-port 6776
--install-transport --install-msghub --install-cmdtools
```

Installing Opswise on Solaris Sparc

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

This section describes the shell script installer. For this install, you will use one of the packages that end in .sh. Running this installer will provide a complete installation of all Opswise Outboard features.

Before You Begin

1. Log in as the root user.
2. Make sure that the downloaded package has the executable bit set. If it does not, set it with the following command:

```
# chmod +x opswise-outboard-SunOS-sparc-N.N.N-build.NN.sh
```

Syntax

The command below installs the Opswise outboard components in a Linux environment. We recommend copying the sample install script into your command shell and editing the parameters with your site-specific information.

```
# ./opswise-outboard-SunOS-sparc-N.N.N-build.NN.sh [installer options]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.

`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.
`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Sample Install Script

The sample install script shown below can be cut and pasted into your command shell and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and build numbers.

```
./opswise-outboard-SunOS-sparc-N.N.N-build.NN.sh --user opswise --user-home /export/home/opswise
--transports 4803@127.0.0.1 --tspport 4803 --bind-iface 0.0.0.0 --bind-port 6776
--install-transport --install-msghub --install-cmdtools
```

Installing Opswise on Windows

Introduction

These instructions assume you have already:

- Installed the prerequisite software
- Downloaded Opswise components
- Installed the Opswise Core

Before Installing

You will need the information described here before installing the outboard components.

Install the Outboard Components

Two types of installs are available for Windows systems:

Windows GUI Install	This version runs the Windows install wizard and prompts you for the required information. For instructions, click here.
Command Line Install	This method allows you to build a command script or manually enter the installation command and switches in a DOS shell. For instructions, click here.

Install Windows Outboard Components Using Script

Command Syntax

The following command installs the Opswise outboard components. We recommend copying the sample install script into a batch file and editing the parameters with your site-specific information.

```
opswise-outboard-win32-N.N.N-build.NN.exe [command switches]
```

The following table describes the command switches for the outboard installation process. Note that all are case-sensitive. If a required switch is missing from the command line, you will be prompted to enter it during the installation process.

Windows	Linux/Unix	Default	Req'd Y/N	Description
`--upgrade`	`--upgrade`	none	No	Use only when performing a build-to-build software upgrade, such as an upgrade from 5.1 build 102 to 5.1 build 104. Specify this command switch **OR** one or more of `--install-msghub`, `--install-transport`.
`--install-transport`	`--install-transport`	none	No	Install the Transporter's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-msghub`	`--install-msghub`	none	No	Install the Message Hub's config file and its startup script. Use only when installing an agent or if you are including an agent install with the core install.
`--install-cmdtools`	`--install-cmdtools`	none	No	Installs the command line interface module.
`--tmpdir`	`--tmpdir`	none	No	Specifies an alternate directory used during installation for temporary files.
`--transports`	`--transports`	4803@localhost	Yes	Transporter socket list. Separate multiple entries with semicolons ';'.
`--tspport`	`--tspport`	4803	Yes	Port used by Transporter.

`--bind-iface`	`--bind-iface`	0.0.0.0	No	Listening IPaddress of Hub for Core/Hub communications.
`--bind-port`	`--bind-port`	6776	No	Listening port of Hub for for Core/Hub communications.
`--add-start-menu-items`	`N/A`	none	No	For Windows, indicates whether you want the installer to add Startup menu items.
`--user`	`--user`	none	Yes	Opswise user name. This is a normal Unix or Windows user id.
`--install-dir`	`--user-home`	Windows c:\opswise Linux/Unix /home/opswise	Yes	Specifies the root directory for the Opswise user directory structure.

Viewing the Installation Log

When the installation is complete, check the installsilent.log file in the Opswise log directory for any errors.

Sample Install Script

The following sample install script can be cut and pasted into a .bat file and edited with your site's information. Note that the release number N.N.N and build.NN in this command should be replaced with the actual release and "build.nn" numbers.

```
opswise-outboard-win32-N.N.N-build.NN.exe /S --install-dir d:\Opswise --transports 4803@127.0.0.1
--tspport 4803 --install-transport --install-msghub --install-cmdtools
```

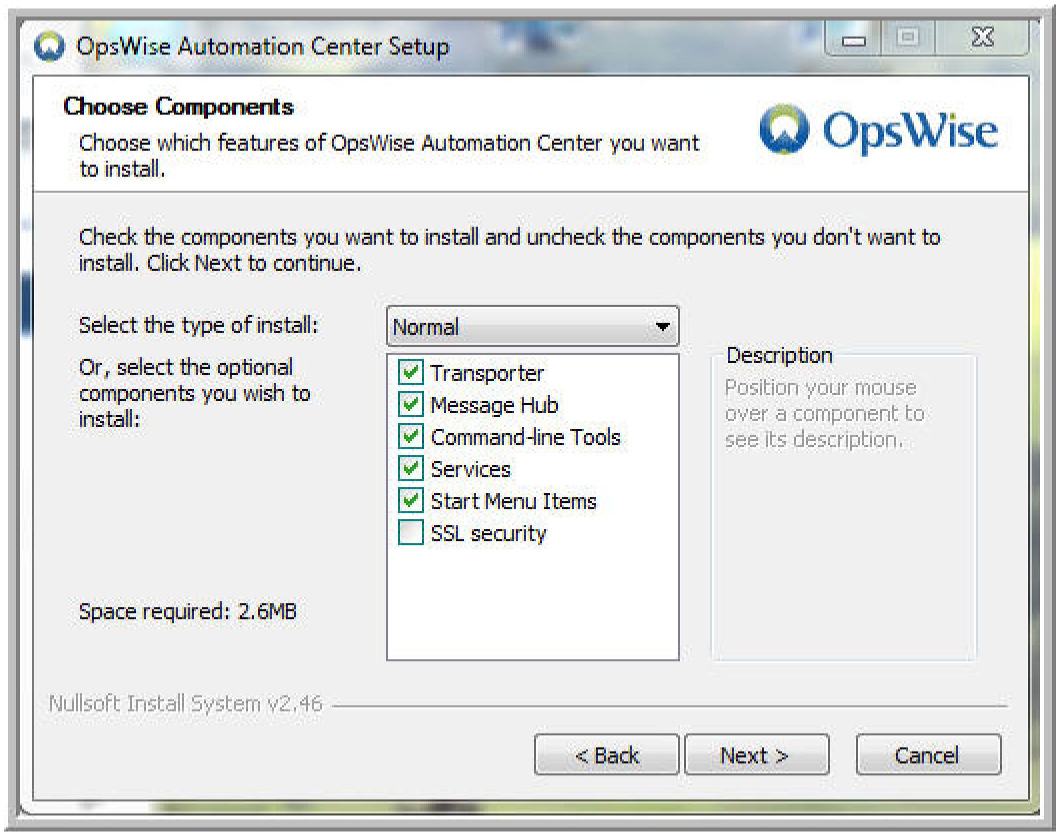
Install Windows Outboard Components Using Wizard

Perform the following instructions to install the Windows outboard components using the wizard. These steps assume you have already downloaded the outboard components.

1. In File Explorer, go to the OPSWISE install directory where you saved the downloaded components.
2. Click the .exe file. The file will be similar to the following:

```
opswise-outboard-win32-N.N.N-build.NN.exe
```

3. The wizard opens and prompts you to select the options you want to install. If you already have a message hub and transporter installed on another machine, you do not need to install a duplicate on the local machine.



OpsWise Automation Center Setup

Choose Components

Choose which features of OpsWise Automation Center you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select the type of install: Normal

Or, select the optional components you wish to install:

- ✓ Transporter
- ✓ Message Hub
- ✓ Command-line Tools
- ✓ Services
- ✓ Start Menu Items
- SSL security

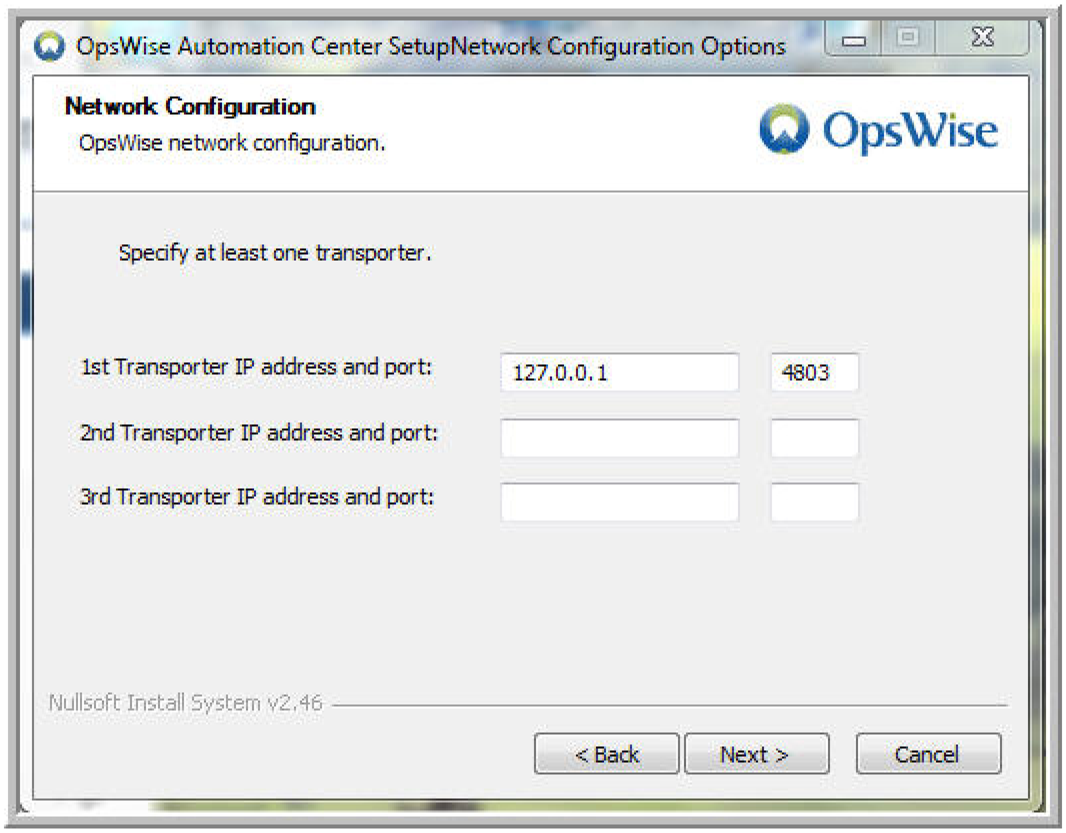
Description

Position your mouse over a component to see its description.

Space required: 2.6MB

Nullsoft Install System v2.46

4. Make your selections and click **Next**.
5. Specify the install directory and click Next.
6. The Network Configuration screen displays.



OpsWise Automation Center SetupNetwork Configuration Options

Network Configuration

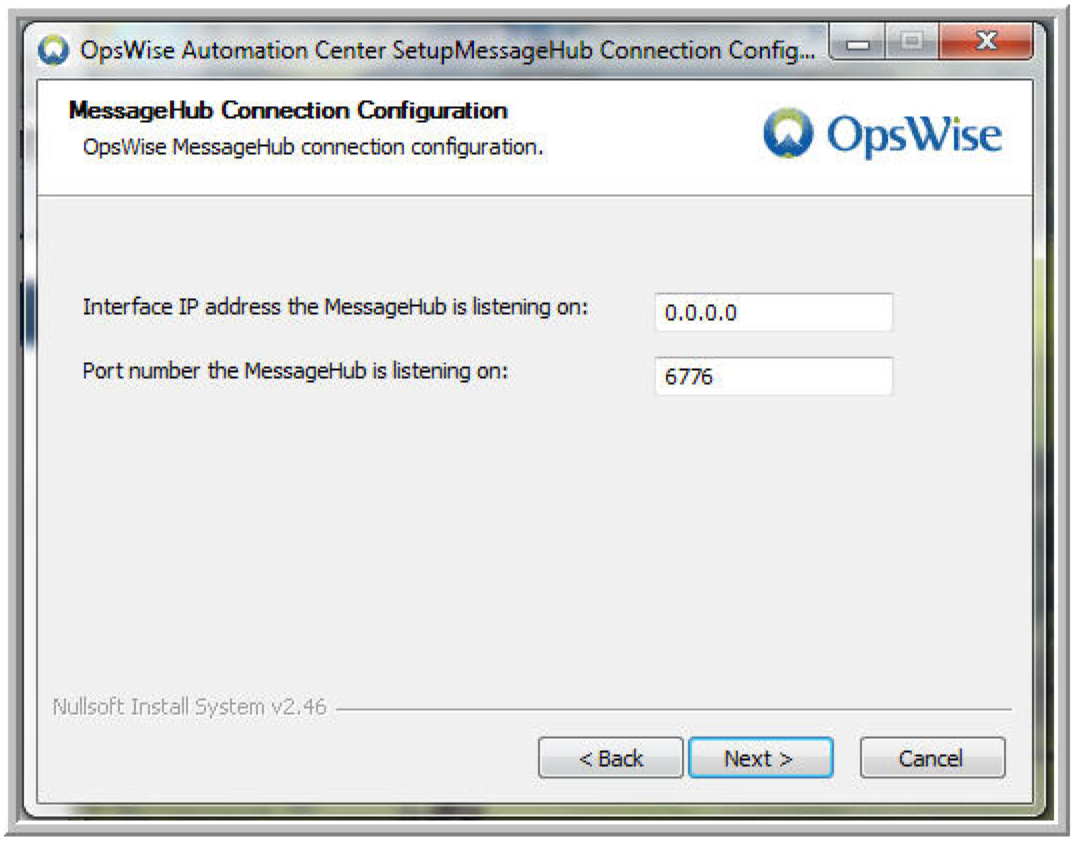
OpsWise network configuration.

Specify at least one transporter.

1st Transporter IP address and port:	127.0.0.1	4803
2nd Transporter IP address and port:		
3rd Transporter IP address and port:		

Nullsoft Install System v2.46

7. You must minimally specify the machine name where the transporter resides. Click **Next**.
8. The Message Hub Connection Configuration screen displays.



OpsWise Automation Center SetupMessageHub Connection Config...

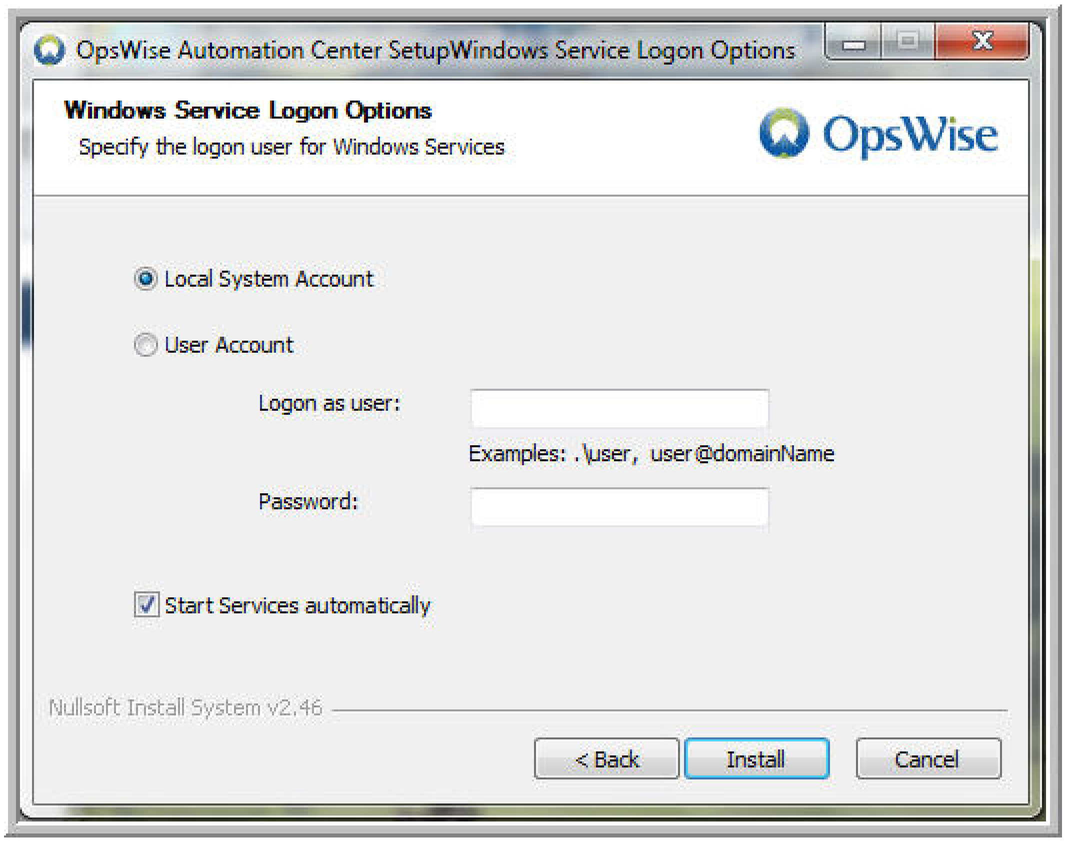
MessageHub Connection Configuration

OpsWise MessageHub connection configuration.

Interface IP address the MessageHub is listening on:	0.0.0.0
Port number the MessageHub is listening on:	6776

Nullsoft Install System v2.46

9. Type in the IP address and port number where the MsgHub will bind to and click **Next**.
10. The Windows Service Logon Options screen displays.


OpsWise Automation Center SetupWindows Service Logon Options

Windows Service Logon Options

Specify the logon user for Windows Services

[● Local System Account]

○ User Account

Logon as user:

Examples: .\user, user@domainName

Password:

[✓] Start Services automatically

Nullsoft Install System v2.46

11. Do not change the default selection, Local System Account. Also, indicate whether you want Windows services for the outboard component to start automatically when you start up Windows. Click **Install**, then **Next**.
12. The installation procedure installs the components you specified. If you did not specify that the Windows services should be started automatically, click here for instructions on how to start each service.

Starting-Stopping Opswise Components on Windows

Start/Stop Components - Opswise 5.1 and Later

From the DOS Command Prompt

To start or stop the Core, use the following commands:

```
c:\$CATALINA_HOME\bin\startup.bat
c:\$CATALINA_HOME\bin\shutdown.bat
```

To start or stop the Transporter, use the following commands:

```
sc start wmsgardt
sc stop wmsgardt
```

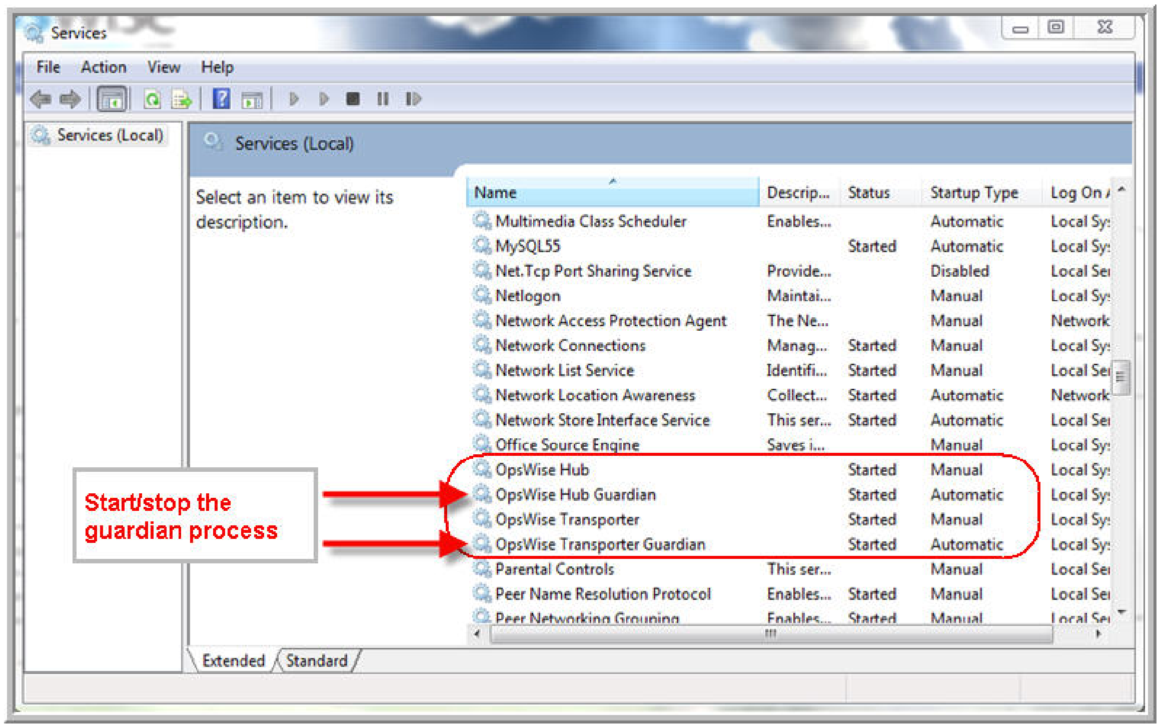
To start or stop the Message Hub, use the following commands:

```
sc start wmsgardh -s HUB
sc stop wmsgardh
```

To start or stop the agent (UAG), use Windows Services (see below).

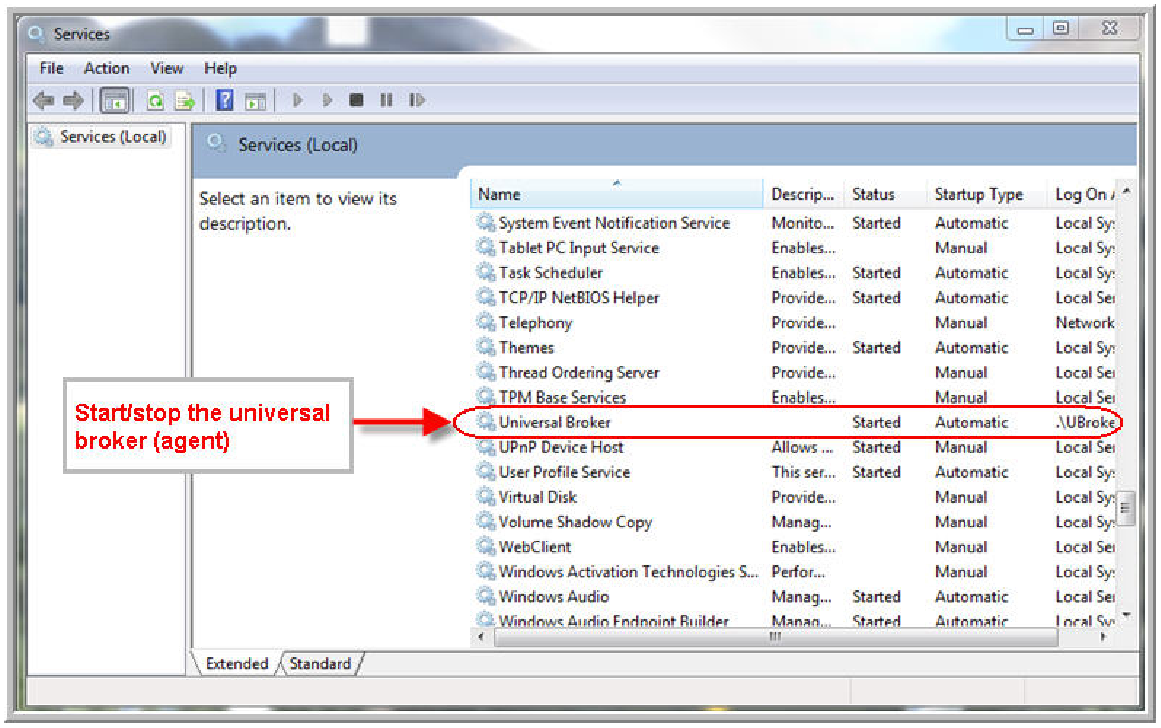
From Windows Services

The Opswise Windows Services are created automatically during the install process. A separate service entry is created for the message hub and transporter, plus a guardian service for each. You will start and stop these services by right-clicking the guardian service and selecting **Start** or **Stop**. The guardian service is used to start and stop the Opswise services and to monitor each service for status.



Name	Descrip...	Status	Startup Type	Log On A
Multimedia Class Scheduler	Enables...		Automatic	Local Sy:
MySQL55		Started	Automatic	Local Sy:
Net.Tcp Port Sharing Service	Provide...		Disabled	Local Se
Netlogon	Maintai...		Manual	Local Sy:
Network Access Protection Agent	The Ne...		Manual	Network
Network Connections	Manag...	Started	Manual	Local Sy:
Network List Service	Identifi...	Started	Manual	Local Se
Network Location Awareness	Collect...	Started	Automatic	Network
Network Store Interface Service	This ser...	Started	Automatic	Local Se
Office Source Engine	Saves i...		Manual	Local Sy:
OpsWise Hub		Started	Manual	Local Sy:
OpsWise Hub Guardian		Started	Automatic	Local Sy:
OpsWise Transporter		Started	Manual	Local Sy:
OpsWise Transporter Guardian		Started	Automatic	Local Sy:
Parental Controls	This ser...		Manual	Local Se
Peer Name Resolution Protocol	Enables...	Started	Manual	Local Se
Peer Networking Grouping	Enables...	Started	Manual	Local Se

The agent is referred to in the services window as the Universal Broker. It has no guardian service.


Services
File Action View Help
Services (Local)
Services (Local)
Select an item to view its description.
Start/stop the universal broker (agent)
Extended Standard

Name	Descrip...	Status	Startup Type	Log On A
System Event Notification Service	Monito...	Started	Automatic	Local Sy:
Tablet PC Input Service	Enables...		Manual	Local Sy:
Task Scheduler	Enables...	Started	Automatic	Local Sy:
TCP/IP NetBIOS Helper	Provide...	Started	Automatic	Local Se
Telephony	Provide...		Manual	Network
Themes	Provide...	Started	Automatic	Local Sy:
Thread Ordering Server	Provide...		Manual	Local Se
TPM Base Services	Enables...		Manual	Local Se
Universal Broker		Started	Automatic	.\UBroke
UPnP Device Host	Allows ...	Started	Manual	Local Se
User Profile Service	This ser...	Started	Automatic	Local Sy:
Virtual Disk	Provide...		Manual	Local Sy:
Volume Shadow Copy	Manag...		Manual	Local Sy:
WebClient	Enables...		Manual	Local Se
Windows Activation Technologies S...	Perfor...		Manual	Local Sy:
Windows Audio	Manag...	Started	Automatic	Local Se
Windows Audio Endpoint Builder	Manag...	Started	Automatic	Local Sy:

Note that the logon user for the services must have full access to the Opswise install directory.

Start/Stop Components - Opswise 1.7 and Earlier

From the DOS Command Prompt

To start or stop the Core, use the following commands:

```
c:\$CATALINA_HOME\bin\startup.bat
c:\$CATALINA_HOME\bin\shutdown.bat
```

To start or stop the Transporter, use the following commands:

```
sc start wmsgardt
sc stop wmsgardt
```

To start or stop the Message Hub, use the following commands:

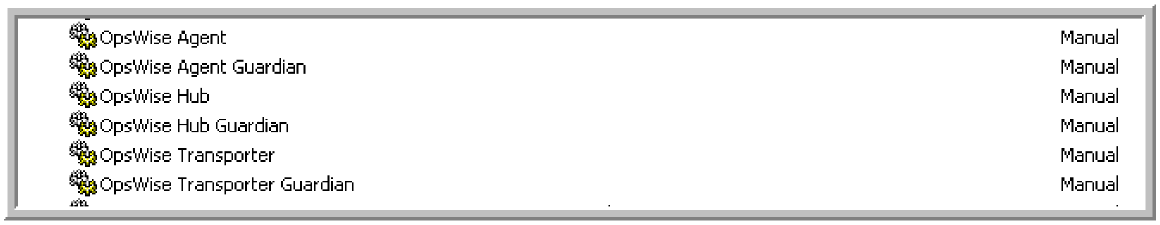
```
sc start wmsgardh -s HUB
sc stop wmsgardh
```

To start or stop the Opswise Agent, use the following commands:

```
sc start wmsgarda -s AGENTNAME
sc stop wmsgarda
```

From Windows Services

The Opswise Windows Services are created automatically during the install process. A separate service entry is created for the agent, message hub, and transporter, plus a guardian service for each component. The guardian service is used to start and stop the Opswise services and to monitor each service for status. Each service entry is identified as an Opswise service, as shown below:



OpsWise Agent	Manual
OpsWise Agent Guardian	Manual
OpsWise Hub	Manual
OpsWise Hub Guardian	Manual
OpsWise Transporter	Manual
OpsWise Transporter Guardian	Manual

Note that the logon user for the services must have full access to the Opswise install directory.

To start and stop the components:

1. Highlight its associated **guardian service**.
2. Right-click to display the services menu.
3. Click start or stop, as needed.

Starting-Stopping Opswise Components on HP-UX

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

Note
Enter the following commands as ROOT.

Itanium and PA-RISC: To start or stop the Opswise Agent Versions 1.7 and earlier, use the following commands:

```
/sbin/init.d/opsagent start
/sbin/init.d/opsagent stop
```

Itanium only: To start or stop the Transporter (all Opswise versions), use the following commands:

```
/sbin/init.d/opstransport start
/sbin/init.d/opstransport stop
```

Itanium only: To start or stop the Message Hub (all Opswise versions), use the following commands:

```
/sbin/init.d/opsmsghub start
/sbin/init.d/opsmsghub stop
```

Starting-Stopping Opswise Components on Solaris Intel

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

Note
Enter the following commands as ROOT.

To start or stop the Opswise Agent Versions 1.7 and earlier, use the following commands:

```
/etc/init.d/opsagent start
/etc/init.d/opsagent stop
```

To start or stop the Transporter (all Opswise versions), use the following commands:

```
/etc/init.d/opstransport start
/etc/init.d/opstransport stop
```

To start or stop the Message Hub (all Opswise versions), use the following commands:

```
/etc/init.d/opsmsghub start
/etc/init.d/opsmsghub stop
```

Starting-Stopping Opswise Components on Solaris Sparc

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

To start or stop the Opswise Agent versions 1.7 and earlier, enter the following commands as ROOT:

```
/etc/init.d/opsagent start
/etc/init.d/opsagent stop
```

Starting-Stopping Opswise Components on Unix AIX

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

Note
Enter the following commands as ROOT.

To start or stop the Opswise Agent Versions 1.7 and earlier, use the following commands:

```
/etc/rc.d/rc2.d/Sopsagent start
/etc/rc.d/rc2.d/Kopsagent stop
```

To start or stop the Transporter (all Opswise versions), use the following commands:

```
/etc/rc.d/rc2.d/Sopstransport start
/etc/rc.d/rc2.d/Kopstransport stop
```

To start or stop the Message Hub (all Opswise versions), use the following commands:

```
/etc/rc.d/rc2.d/Sopsmsghub start
/etc/rc.d/rc2.d/Kopsmsghub stop
```

Starting-Stopping Opswise Components on Linux

To start or stop the Core (all Opswise versions), use the following commands:

```
/$TOMCAT_HOME/bin/startup.sh
/$TOMCAT_HOME/bin/shutdown.sh
```

or

```
service tomcat start
service tomcat stop
```

If you have configured your system with **init.d**, you can use the following:

```
/etc/init.d/tomcat start
/etc/init.d/tomcat stop
```

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

> **Note**
> Enter the following commands as ROOT.

To start or stop the Opswise Agent Versions 1.7 and earlier, use the following commands:

```
service opsagent start
service opsagent stop
```

To start or stop the Transporter (all Opswise versions), use the following commands:

```
service opstransport start
service opstransport stop
```

To start or stop the Message Hub (all Opswise versions), use the following commands:

```
service opsmsghub start
service opsmsghub stop
```

Starting-Stopping Opswise Components on zLinux

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

To start or stop the Opswise Agent Version 1.7 and earlier, enter the following commands as ROOT:

```
service opsagent start
service opsagent stop
```

Starting-Stopping Opswise Components on zOS

To start or stop the Opswise 5.1 Agent (UAG), see Starting and Stopping Components.

To start or stop the Opswise Agent Versions 1.7 and earlier, use the following commands:

MVS Start command

```
 S [OPSPROC]          (where OPSPROC is the name of the procedure you added to the
system.PROCLIB)
```

MVS Stop command

```
 P [OPSPROC]
```

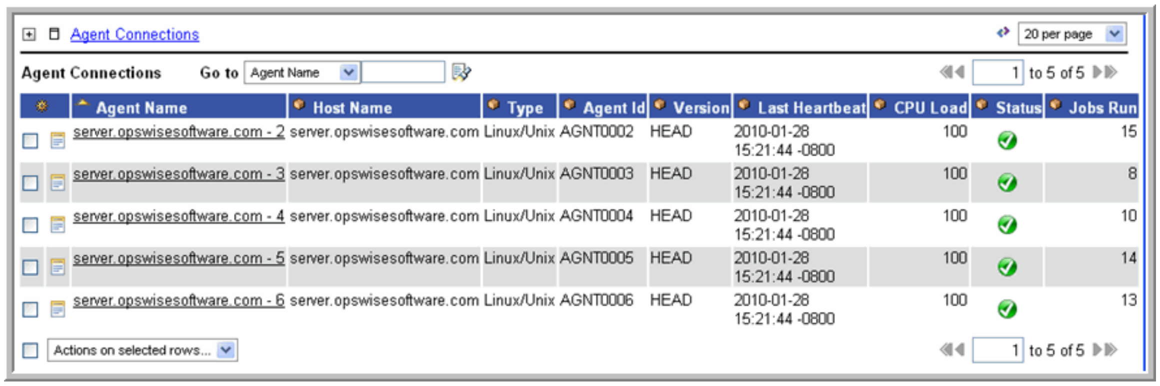
Verify Outboard Installation

Follow these steps to verify that your agent, message hub, and transporter are installed, running, and communicating with the core processor:

1. From your browser, open the Opswise main navigation page (where localhost is the name of the machine).

```
http://localhost:8080/opswise
```

2. Log in with a userid and password for this system.
3. To check for your newly installed agent, select **Automation Center Resources > All Agents**. You will see a list similar to the example shown below, including the agent you just installed. Make sure the agent Status is Active (green circle with a checkmark).



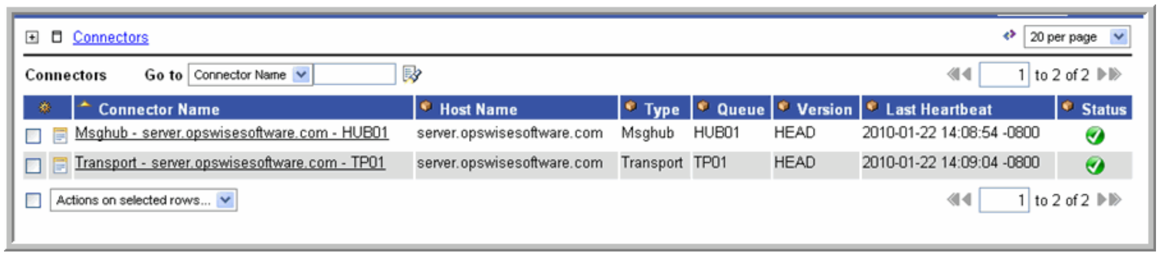
Agent Connections

Agent Connections Go to Agent Name — 20 per page — 1 to 5 of 5

Agent Name	Host Name	Type	Agent Id	Version	Last Heartbeat	CPU Load	Status	Jobs Run
server.opswisesoftware.com - 2	server.opswisesoftware.com	Linux/Unix	AGNT0002	HEAD	2010-01-28 15:21:44 -0800	100	✓	15
server.opswisesoftware.com - 3	server.opswisesoftware.com	Linux/Unix	AGNT0003	HEAD	2010-01-28 15:21:44 -0800	100	✓	8
server.opswisesoftware.com - 4	server.opswisesoftware.com	Linux/Unix	AGNT0004	HEAD	2010-01-28 15:21:44 -0800	100	✓	10
server.opswisesoftware.com - 5	server.opswisesoftware.com	Linux/Unix	AGNT0005	HEAD	2010-01-28 15:21:44 -0800	100	✓	14
server.opswisesoftware.com - 6	server.opswisesoftware.com	Linux/Unix	AGNT0006	HEAD	2010-01-28 15:21:44 -0800	100	✓	13

Actions on selected rows... — 1 to 5 of 5

4. To check for your newly installed message hub and transporter, select **Automation Center Resources > Connectors**. You will see a list similar to the example shown below, including the message hub and transporter you just installed. Make sure the message hub and transporter Statuses are Active (green circle with a checkmark).



Connectors

Connectors Go to Connector Name — 20 per page — 1 to 2 of 2

Connector Name	Host Name	Type	Queue	Version	Last Heartbeat	Status
Msghub - server.opswisesoftware.com - HUB01	server.opswisesoftware.com	Msghub	HUB01	HEAD	2010-01-22 14:08:54 -0800	✓
Transport - server.opswisesoftware.com - TP01	server.opswisesoftware.com	Transport	TP01	HEAD	2010-01-22 14:09:04 -0800	✓

Actions on selected rows... — 1 to 2 of 2

5. For more information about these components, go to Agent-Based Resources. To get started using the system and become familiar with its features, we recommend you spend some time going through the tutorials on the First Look page.

Upgrading Opswise Automation Center

Introduction

These instructions describe how to upgrade Opswise from one version to another. You can use these instructions for the supported upgrade paths shown in the table below. For any other upgrade path, consult your Stonebranch representative.

Upgrade to...	1.6	1.7	5.1
From 1.5	X	X	X
From 1.6	X	X	X
From 1.7	X	X	X

For each upgrade, you must install the new versions of the core, transporter and message hub components at the same time. Opswise agents do not need to be upgraded at the same time and older versions will still operate with updated core components. However, we recommend that you also upgrade the agents as soon as possible to avoid letting the software get too out of date.

For a system overview of Opswise 5.1, click here.

Back Up Your Database

Important
Before upgrading your system, back up your database. The database backup is a fail-safe measure; you will be using the Opswise export/import utility described below to migrate your data.

Run an Opswise Export

The Opswise export scripts copy and save the specified records to one or more XML files. The files can then be imported into the upgraded system. For best results, complete the tasks described below before running the export.

Make Sure No Records Are Being Processed

WARNING
If the core is processing task instances when you launch the export, the results are unpredictable.

Before running the export:

1. Log in with **ops.admin** or a user with administrator privileges.

2. Disable all active triggers to make sure no tasks are being processed.
3. Check the Activity screen to verify that there are no active task instances. If there are, wait until they complete before you start the export process. If necessary, you can force finish tasks.

Decide Which Records to Export

The list of export scripts changed from Opswise 1.7 to Opswise 5.1. Depending on what version you are upgrading from, select the appropriate upgrade script from the following descriptions.

Exporting Records from Opswise 1.7 or Earlier

The following export scripts are available:

- **opswise_bulk_exec_export.js** - Not recommended for migration. Exports all unfinished activity (task instances in the Activity display)
- **opswise_bulk_export.js** - Exports all record definitions and task instance history, which includes all task instances in an end state (cancelled, failed, skipped, finished, success)

Exporting Records from Opswise 5.1 or Later

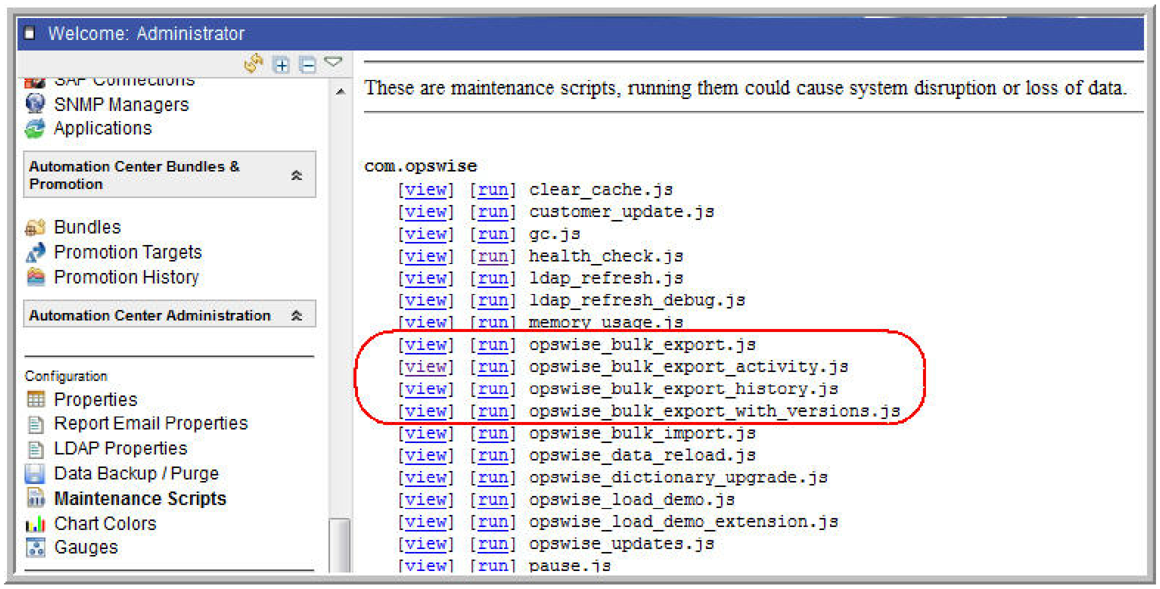
The following export scripts are available:

- **opswise_bulk_export_activity.js** - Not recommended for migration. Exports all unfinished activity (task instances in the Activity display)
- **opswise_bulk_export.js** - Exports all current record definitions, without versions
- **opswise_bulk_export_with_versions.js** - Exports all current records along with older (non-current) versions of record definitions
- **opswise_bulk_export_history.js** - Exports task instance history, which includes all task instances in an end state (cancelled, failed, skipped, finished, success)

Run the Export

Follow these steps to run a bulk export of your data:

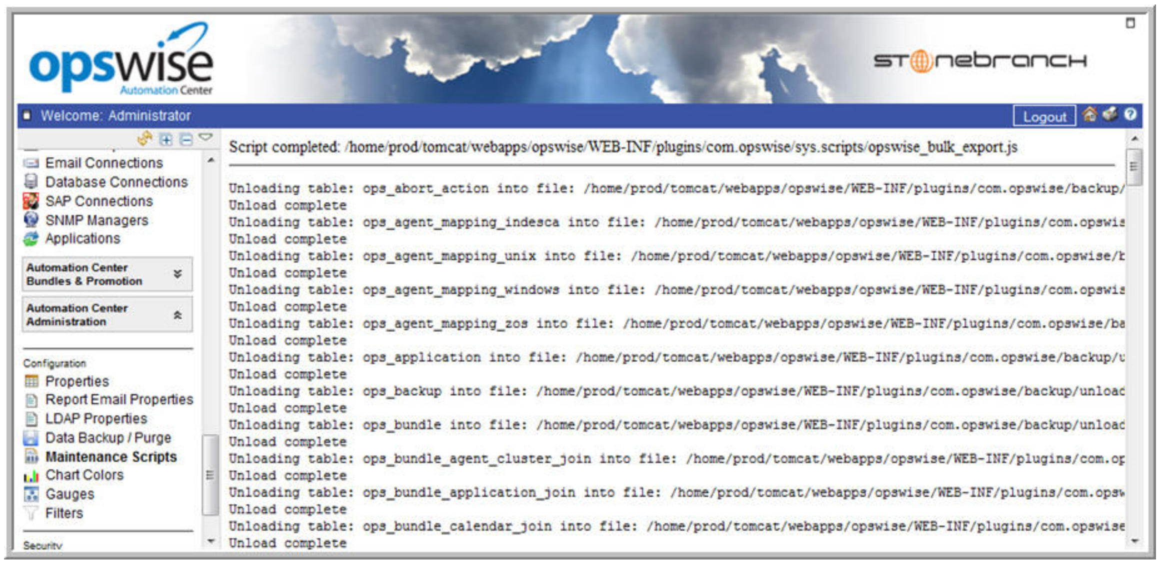
1. From the navigation pane, select **Automation Center Administration > Configuration > Maintenance Scripts**. The image below shows export script options in a 5.1 version of Opswise.



These are maintenance scripts, running them could cause system disruption or loss of data.

```
com.opswise
  [view] [run] clear_cache.js
  [view] [run] customer_update.js
  [view] [run] gc.js
  [view] [run] health_check.js
  [view] [run] ldap_refresh.js
  [view] [run] ldap_refresh_debug.js
  [view] [run] memory_usage.js
  [view] [run] opswise_bulk_export.js
  [view] [run] opswise_bulk_export_activity.js
  [view] [run] opswise_bulk_export_history.js
  [view] [run] opswise_bulk_export_with_versions.js
  [view] [run] opswise_bulk_import.js
  [view] [run] opswise_data_reload.js
  [view] [run] opswise_dictionary_upgrade.js
  [view] [run] opswise_load_demo.js
  [view] [run] opswise_load_demo_extension.js
  [view] [run] opswise_updates.js
  [view] [run] pause.js
```

2. Select the script you want to run and click **Run**.
3. The utility prompts for a confirmation. Click **Yes**.
4. As your data is exported, the output from the script is written to the screen, as shown.



Script completed: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/sys.scripts/opswise_bulk_export.js

```
Unloading table: ops_abort_action into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/backup/
Unload complete
Unloading table: ops_agent_mapping_indesca into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswis
Unload complete
Unloading table: ops_agent_mapping_unix into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/b
Unload complete
Unloading table: ops_agent_mapping_windows into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswis
Unload complete
Unloading table: ops_agent_mapping_zos into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/ba
Unload complete
Unloading table: ops_application into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/backup/u
Unload complete
Unloading table: ops_backup into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/backup/unload
Unload complete
Unloading table: ops_bundle into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise/backup/unload
Unload complete
Unloading table: ops_bundle_agent_cluster_join into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.op
Unload complete
Unloading table: ops_bundle_application_join into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opsw
Unload complete
Unloading table: ops_bundle_calendar_join into file: /home/prod/tomcat/webapps/opswise/WEB-INF/plugins/com.opswise
Unload complete
```

5. Look over the output for any error messages. If you see any, copy the output to a file and send it to support@stonebranch.com.
6. Zip or tar the contents of:

```
[tomcat directory]/webapps/opswise/WEB-INF/plugins/com.opswise/backup/unload/
```

7. Copy the zip/tar file to a safe place for use after the upgrade process.
8. Copy your glide.properties file to a safe place. You may need to consult this file later. The file is located here:

```
[tomcat directory]/webapps/opswise/WEB-INF/properties
```

9. Copy your license key from the Opswise Properties form and store in a safe place.
10. Copy the LDAP mapping file to a safe place for use after the upgrade process.

```
[tomcat directory]/webapps/opswise/WEB-INF/properties/users/ldapmap.xml
```

Stop Tomcat and Remove the Opswise Core

Important
Make sure you have copied to a safe location all the exported files from the previous step before continuing with this step. In this step you will stop and remove the core system (which does not include the agent components).

1. Stop the Tomcat process, as follows:
   - In Windows, you should use the services application to stop Tomcat. You may also issue the stop command on a command line:

     ```
     net stop [name of Tomcat service]
     ```

   - In Unix, stop the daemon using the script found in the /etc/init.d directory for Tomcat.

     ```
     service [name of Tomcat service] stop
     ```

   - In either system, you can stop the service using the $TOMCAT_HOME/bin/shutdown.bat or $TOMCAT_HOME/bin/shutdown.sh scripts:

Windows	Linux/Unix

`cd $CATALINA_HOME\bin`
`shutdown`
	`cd $CATALINA_HOME/bin`
	`./shutdown`

2. Delete the directory:

```
$CATALINA_HOME/webapps/opswise/
```

3. Delete the file:

```
$CATALINA_HOME/webapps/opswise.war
```

Stop Hub, Transporter, and Agent Components

For the agent and outboard components (the message hub and transporter), you do not need to remove the old versions. This process upgrades the existing software.

Note
You can run downlevel versions of Stonebranch agents with new versions of the core, hub, and transporter processes.

Stop the Opswise processes in the order shown. For instructions about how to stop Opswise processes for your specific platform, see Start/Stop Opswise Components .

1. Stop the Hub process.
2. Stop the Transporter process.
3. Stop all Agent processes (required only for agents you are updating to the new release).

Prepare Your Database

Delete or drop your database using the appropriate database admin tool.

Important
Before dropping your existing database, make sure you have created a backup as described at the beginning of the procedure.

Download the New Opswise Release

Click here for instructions about downloading the latest Opswise software.

Install and Verify the Opswise Core

The Opswise core processor is a Java application running within Apache Tomcat. For this reason, the core software and installation procedure is basically the same for all platforms.

Run an Opswise Import

In this step, you are importing your data that you exported earlier using the Opswise bulk export.

1. Unzip/untar the backup file that you created earlier using the Opswise export.
2. Copy the XML files to the following directory:

```
$CATALINA_HOME/webapps/opswise/WEB-INF/plugins/com.opswise/backup/unload
```

3. From the navigation pane, select **Automation Center Administration > Configuration > Maintenance Scripts**.
4. Locate and run the script

```
opswise_bulk_import.js
```

5. The utility prompts for a confirmation. Click **Yes**.
6. As your data is imported, the output from the script is written to the screen. Look over the output for any error messages. If you see any, copy the output to a file and send it to support@stonebranch.com.
7. Run the following script:

```
clear_cache.js
```

Check Your Data

At this point, your previous definitions, users and passwords have all been restored. Log out and in again, and review your records to make sure all your previous definitions, users, and passwords have been restored successfully.

Install Opswise Hub and Transporter (Outboard Components)

The Opswise Outboard components comprise the transporter, message hub and command line tools. The software and instructions differ depending on your platform. For instructions, click here and select your platform.

Install Universal Automation Center Agent (UAG)

The software and instructions for Universal Automation Center Agent (UAG) differ depending on your platform. For instructions, click here and select your platform.

If you are upgrading from Opswise 1.7, 1.6 or 1.5 to Opswise 5.1 or later, you will be converting from the old Opswise agent software to the new Opswise Universal Automation Center Agent (UAG) software. For Linux/Unix and Windows agents, the agent installation command parameters include the option CONVERT_OPSAGENT=yes. When you specify this option, the installation process invokes a script named opsmerge.sh (Linux/Unix) or opsmerge.vbs (Windows). This script stops the agent and converts the agent properties (agent.props) into the new agent properties file, uags.conf. The script also performs several other tasks needed for the conversion. For details, see Migrating an Opswise Agent to UAG for Workload Automation 5 for UNIX and Migrating an Opswise Agent to UAG for Workload Automation 5 for Windows.

Note
You cannot upgrade a z/OS agent from Opswise 1.7, 1.6, or 1.5 to Opswise 5.1 or later. To use the new Opswise Universal Automation Center Agent (UAG) software, you must install a Workload Automation 5 for z/OS package, which contains UAG.

Start/Stop Opswise Components

The start/stop procedure for the Opswise components may differ depending on your platform. For instructions, click here and select your platform.

Verify the Agent and Outboard Component Installation

As a final step, you should verify that your agent and outboard components are installed, running, and communicating with the Opswise core server. Click here for instructions.