

RedundancyMaster Help

© 2010 Kepware Technologies

Table of Contents

1	Getting Started.....	3
	RedundancyMaster.....	3
	Introduction	3
	User Interface	6
	Requirements	7
	Installing as an NT Service.....	8
2	Setting Up Redundancy.....	9
	Adding Redundancy.....	9
	Aliasing Redundancy.....	10
	Unaliasing Redundancy.....	11
	General Settings	12
	Monitoring Settings.....	14
	Diagnostics Settings.....	16
	Notifications Settings.....	17
	Applying Your Settings.....	19
	Enable/Disable Redundancy.....	19
	Removing Redundancy.....	20
	Deployment	21
3	Runtime Diagnostics.....	21
	Runtime Diagnostics.....	21
	Attempt to add monitor item '<item id>' for '<server name>' on primary machine '<machine name>'.....	22
	failed. This monitor item will be considered in error	
	Attempt to add monitor item '<item id>' for '<server name>' on secondary machine '<machine name>'	22
	Connected to '<server name>' on primary machine '<machine name>'.....	22
	Connected to '<server name>' on secondary machine '<machine name>'.....	23
	Disconnected from '<server name>' on primary machine '<machine name>'.....	23
	Disconnected from '<server name>' on secondary machine '<machine name>'.....	23
	Failed to connect to '<server name>' on primary machine '<machine name>'.....	23
	Failed to connect to '<server name>' on secondary machine '<machine name>'.....	23
	Monitor item '<item id>' for '<server name>' on primary machine '<machine name>' contains array data.....	23
	which is not supported for the 'specific value' trigger condition	
	Monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' contains array.....	24
	Monitor item test in error for '<server name>' on primary machine '<machine name>'.....	24
	Monitor item test in error for '<server name>' on secondary machine '<machine name>'.....	24
	Monitor item test passed for '<server name>' on primary machine '<machine name>'.....	24
	Monitor item test passed for '<server name>' on secondary machine '<machine name>'.....	24
	Promoted active connection for server '<server name>' to primary machine '<machine name>'.....	25
	Promoted active connection for server '<server name>' to secondary machine '<machine name>'.....	25
	Received shutdown notification from server '<server name>' on primary machine '<machine name>'.....	25
	Received shutdown notification from server '<server name>' on secondary machine '<machine name>'.....	25
	Unable to retrieve status for server '<server name>' on primary machine '<machine name>'. Server.....	25
	communications has been lost	
	Unable to retrieve status for server '<server name>' on secondary machine '<machine name>'. Server.....	26
4	Purchase and Unlock a RedundancyMaster License.....	26
	Licensing Overview.....	26
	Purchase a License.....	26

Unlock a License.....	28
5 License Transfer.....	28
License Transfer Instructions.....	28
License Transfer - Step 1: Agreement.....	29
License Transfer - Step 2: Prepare Removable Media for License.....	29
License Transfer - Step 3: Transfer License from Source PC to Removable Media.....	31
License Transfer - Step 4: Transfer License from Removable Media to Target PC.....	33
License Transfer - Step 5: Completion.....	36
 Index	 37



Help version 1.014

CONTENTS

[Introduction](#)

[User Interface](#)

[Requirements](#)

[Setting Up Redundancy](#)

[Runtime Diagnostics](#)

[Purchase and Unlock a License](#)

[License Transfer](#)

Introduction

OPC Data Access (DA) Technology is one of the most successful developments made on the industrial automation software landscape. Virtually every application developed now uses some form of OPC DA technology. In this regard, OPC DA technology has proven to be reliable in virtually every possible situation that requires consistent data access to devices and systems. With the success of OPC technology, it is no wonder that so many applications and solutions have come to depend on the availability of data supplied by some form of OPC DA. Applications (such as data collection for compliance, alarm detection, product asset management, and human machine interface and SCADA) cannot afford to lose their connection to the underlying OPC server, target devices or systems.

There are many factors that can impact the quality and reliability of the data. The following may cause an OPC server to lose data:

1. The PC running the OPC server is shut down.
2. An error caused by the user makes the OPC server exit.
3. The network connection to OPC server is lost or unreliable.
4. The network setting changes cause a link failure.
5. The OPC server itself fails for any reason.
6. The log-in is changed on the OPC server's PC.

In most of these cases, the OPC DA server fails to provide data due to an actual failure underlying the OPC server or the connection to that server. These types of failures are called Object-based failures, and they occur when the actual link between the OPC client application and the target OPC server breaks down.

Loss of Data

Since communications have a number of factors that can dramatically affect reliability (specifically in the industrial environment), the application can lose data in several ways. Examples of these physical factors are as follows:

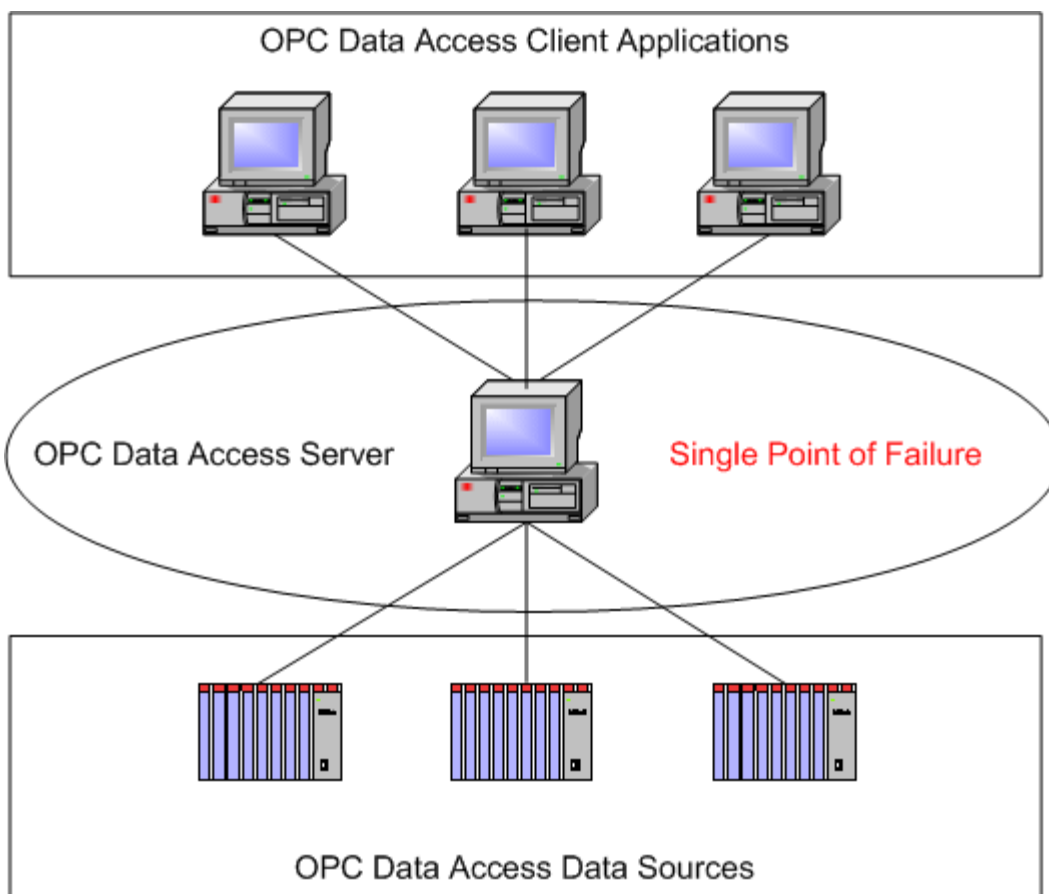
1. Physical connection failure. For example, the cable is pulled.
2. Hardware failure. For example, the router fails.
3. Electrical interference. For example, high current discharge.
4. Delays due to signal propagation. For example, radio links.
5. Environmental factors. For example, lightning.

6. Random accidents.

In these cases, the connection between the OPC server and the client may be perfectly intact but the physical link to the underlying device or system may be broken. These types of failures are called link-based failures, and they occur when the connection to the target device or system has been lost. The OPC server in use is usually still completely operational, but simply cannot supply the data.

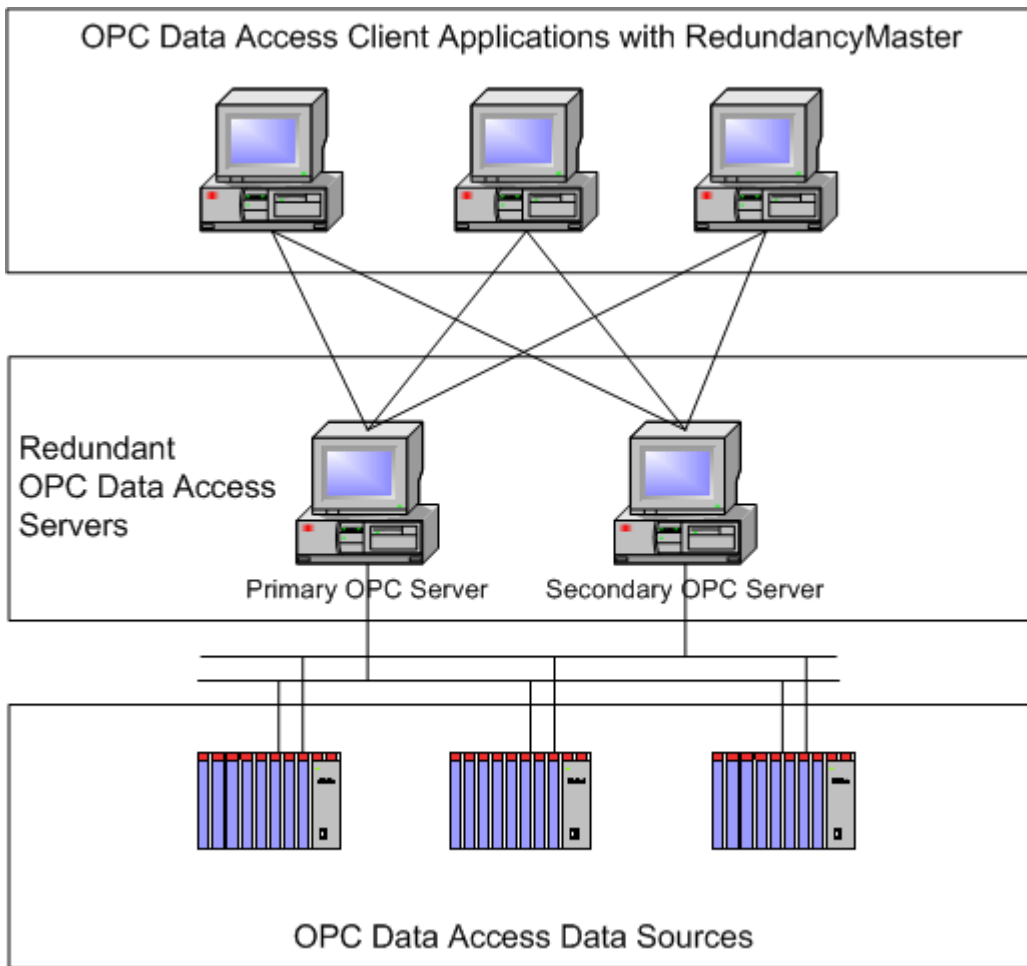
Increasing OPC Reliability

The following diagram demonstrates how a typical OPC application is configured and how it is susceptible to failure. As can be seen, the OPC DA client applications are all accessing a single OPC server; thus, the potential exists for both an Object-based failure and a Link-based failure. If for any reason the single OPC server fails to operate, then an Object-based failure occurs. Additionally, since this single PC is responsible for data collection from the underlying devices, a single point of failure exists for the device connection as well. To increase the reliability of the OPC system, users must remove these single points of failure.



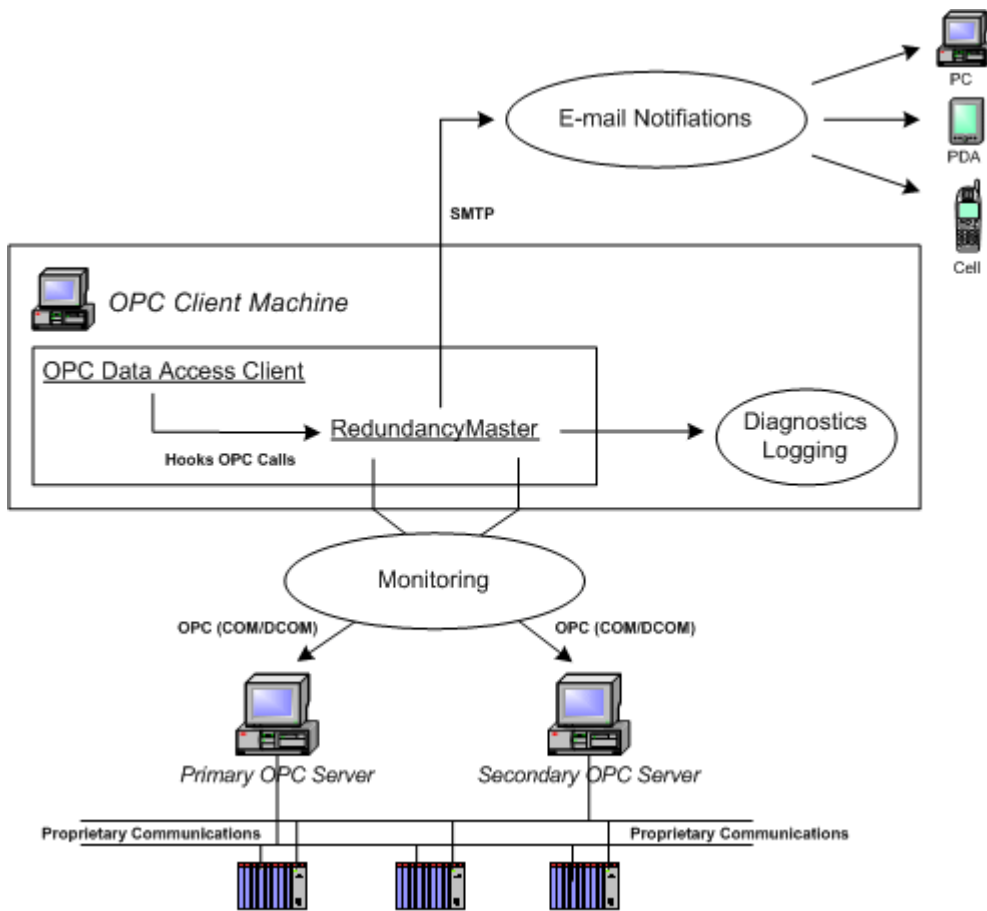
A Better Way

To eliminate the single point of failure, users can redesign the OPC applications to use more than one OPC server. The problem is that this requires the OPC client applications to have some form of redundancy built into the product. In either case, users may be required to redevelop the OPC client from the ground up to take advantage of the redundant technology. With an existing application of any considerable size, it may not be practical to go back and completely redesign and reconfigure the application. What is needed is a seamless way to drop redundancy into the OPC application without making any changes to the OPC client.



As shown above, the original OPC application has been redesigned using two OPC servers instead of a single OPC server. To facilitate the redundant operation of the OPC servers, each OPC client has been paired with RedundancyMaster. The design of RedundancyMaster allows this to occur with no configuration changes to the OPC client application. Using the configurable options within RedundancyMaster, the use of either the Primary or Secondary OPC server can be controlled directly. Based on the modes selected, RedundancyMaster will keep both servers active or if configured to do so, start the secondary server only when the primary server fails.

In regard to either Object-based failures or Link-based failures, RedundancyMaster can be configured to monitor these conditions. Subsequent sections of this manual will detail how users can configure RedundancyMaster to monitor Object failures and Link failures (as well as what actions to take when they occur).



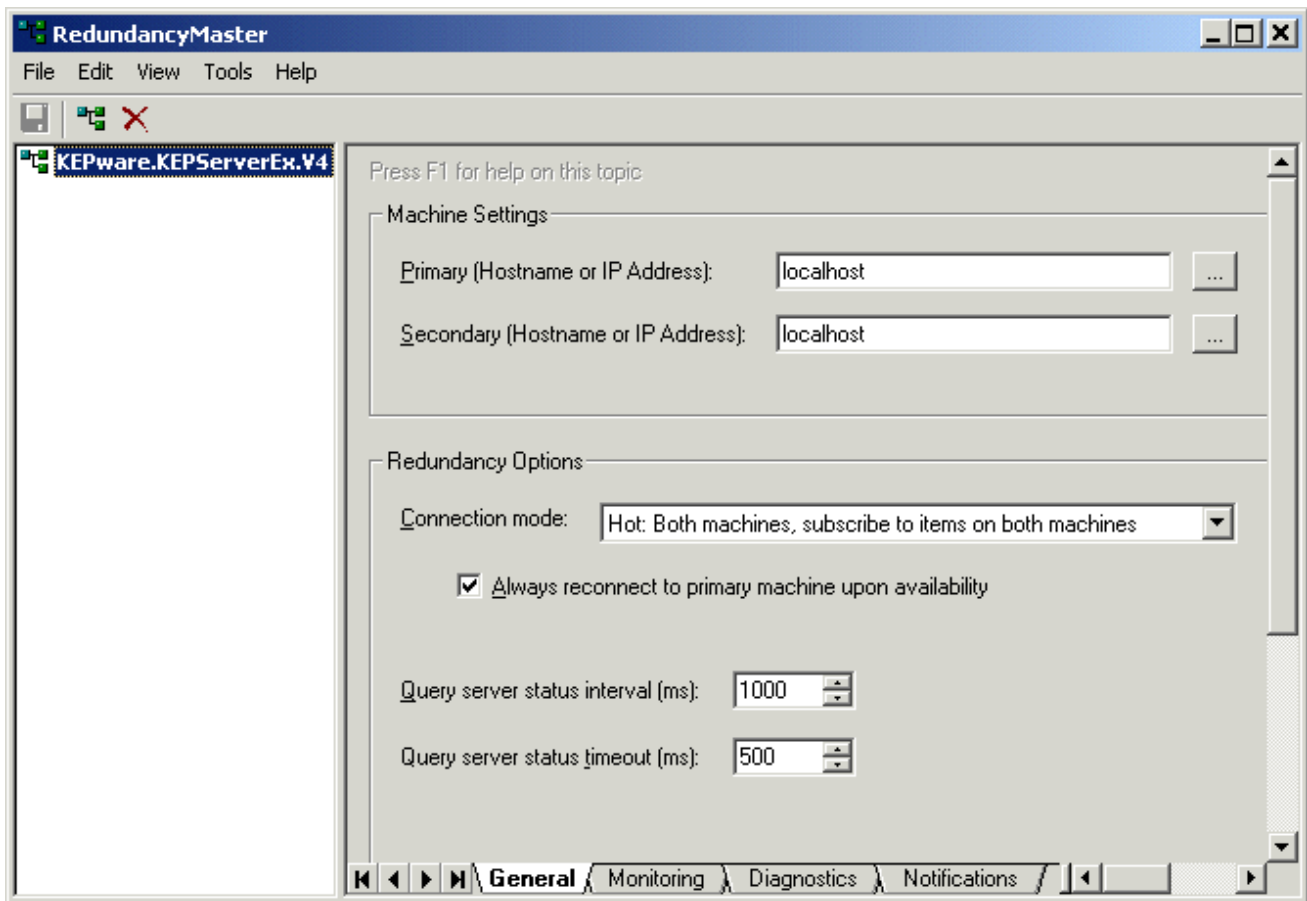
RedundancyMaster Architecture

RedundancyMaster's architecture seamlessly integrates redundancy. To prevent users from having to change the OPC client application, RedundancyMaster hooks all of the OPC calls that the application normally makes directly to an OPC server. Once the OPC calls have been hooked by RedundancyMaster, it can automatically route those calls to either the primary or secondary OPC server. The OPC client application never knows that this switching action occurs.

While it is important that the OPC client application has no idea that RedundancyMaster has switched between the primary or secondary OPC server, it is crucial to be aware that the switch has occurred since this may signal a possible loss of data if remedial action is not taken soon. To address this, RedundancyMaster logs all events that occur within its environment. RedundancyMaster can also generate email notifications for each logged event. To ensure that the email notifications go through, however, even the Simple Mail Transfer Protocol (SMTP) servers allow for both a primary and secondary server to be defined.

User Interface

To learn more about RedundancyMaster's user interface and other specific elements, click on any area in the image below.



Requirements

Operating System

Because this application can run as an NT-based service, it is required to be installed on one of the following operating systems.

- Windows XP Service Pack 1
- Windows Server 2003
- Windows 2000 Service Pack 4
- Windows NT 4.0 Service Pack 6a

Note: RedundancyMaster is capable of providing redundancy to any compliant OPC server running remotely on an OS platform. The OPC server must conform to the server side components required by the OPC Foundation.

See Also: [Installing as an NT Service](#)

Client System

This application must be installed on the same machine as the OPC clients that are being provided redundancy. This requirements is beneficial to the application by allowing it to provide redundancy to client projects without making any modifications to the client configuration.

OPC Servers

RedundancyMaster provides redundancy for OPC Data Access servers adhering to specifications 1.0 through 2.05a. Note the following:

- The OPC Data Access servers can reside on any machine that is accessible from the system running RedundancyMaster.
- OPC servers are not limited to the operating systems listed above. Those listed above are required for

RedundancyMaster.

- If an OPC server does not reside on the same machine that RedundancyMaster is installed on, a minimal OPC server configuration will be imported to the RedundancyMaster machine to enable redundancy to be achieved.
- The OPC server must run as an EXE-based, out-of-process server or RedundancyMaster will not function properly.

RedundancyMaster can provide redundancy for multiple OPC servers and will support up to two machines per OPC server. The OPC server on each machine must use the same Prog ID and conform to the same OPC item namespace in order for redundancy to be achieved. For example, if the client requires an item named "Channel.Device.Tag," then this item must be configured on both machines.

See Also: [Adding Redundancy](#) and [Deployment](#).

OPC Clients

RedundancyMaster provides redundancy for any client that can communicate with OPC Data Access servers adhering to specifications 1.0 through 2.05a. The client must reside on the same machine as this application; that is, the machine on which RedundancyMaster is installed. It attaches itself to a client application as a DLL-based, in-process server. Therefore, the OPC client application must not prevent connections to a DLL-based, in-process server or RedundancyMaster will not function properly.

Installing as an NT Service

RedundancyMaster has the ability to run as a service under Windows NT/2000/XP/Server2003. This ability is required for client applications that run as a service and continue to run when the machine experiences a user logout. If the client application does not run as a service, then users are not required to install this application as a service.

Installing as an NT Service

To install the application as a service, select **Tools | Install As NT Service**.



Note: Users must restart the application (as well as any clients currently utilizing redundancy) in order to complete the operation.

Canceling the Install as NT Service

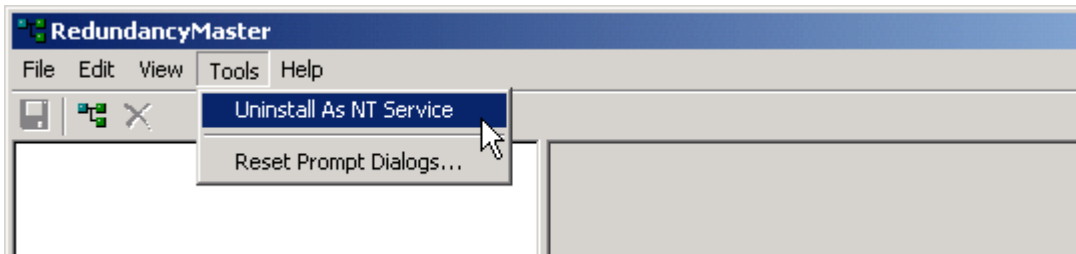
If users decide not to install the application as a service, they may cancel the request by selecting **Tools | Cancel Install As NT Service**.



Note: The application and clients do not need to be restarted after canceling.

Uninstalling as NT Service

To uninstall the application as an NT service, select **Tools | Uninstall As NT Service**.



Note: Users must restart the application (as well as any clients currently utilizing redundancy) in order to complete the operation.

Canceling the Uninstall as NT Service

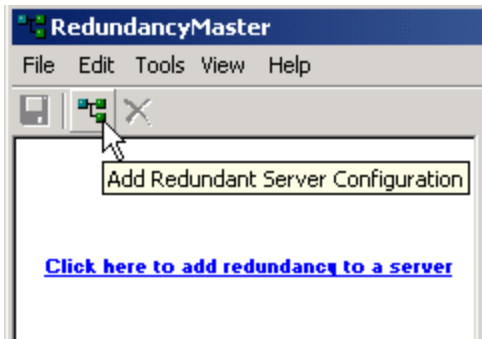
If users decide not to uninstall the application as a service, they may cancel the request by selecting **Tools | Cancel Uninstall As NT Service**. The application and clients do not need to be restarted after canceling.

Adding Redundancy

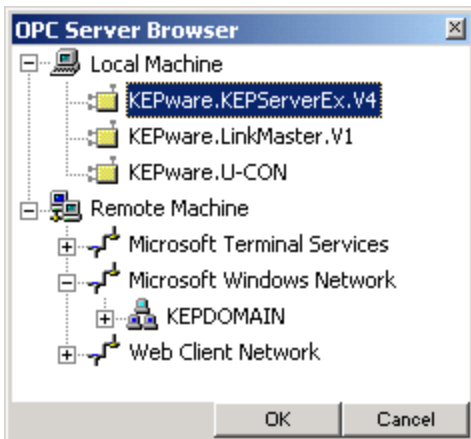
For information on adding redundancy to an OPC server, follow the instructions below.

Note: When planning to create multiple redundant pairs that will use the same OPC server, each redundant pair must be assigned a ProgID alias. For more information and instructions, refer to [Aliasing Redundancy](#).

1. Verify that all system requirements have been met.
2. Next, click **Add Redundant Server Configuration**. Alternatively, click **Click here to add redundancy to a server**.



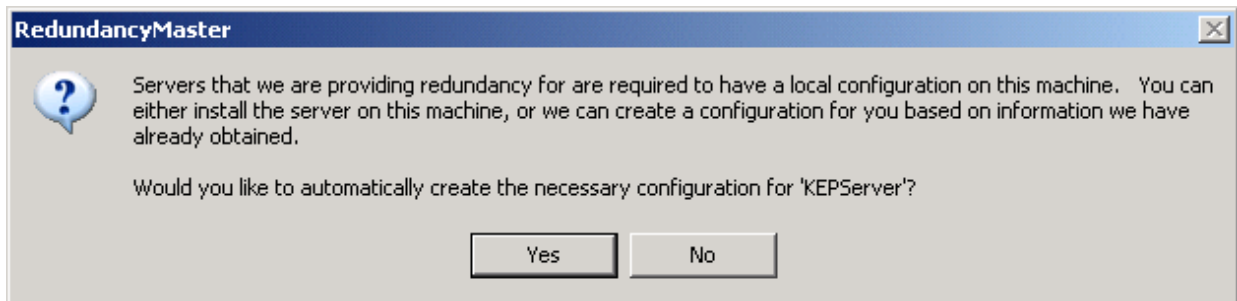
3. Browse to the ProgID of the OPC server to which redundancy will be added and then click to select. This OPC server may be on the local or remote machine.



4. Click **OK** to continue.

5. If the server chosen is on a remote machine and is not installed on the local machine, a message window will be invoked. Click **Yes** to create the necessary configurations on the local machine.

Note: If this redundant server configuration is ever removed or if the RedundancyMaster application is ever uninstalled from the local machine, the minimal configuration for the remote server will be automatically cleaned up from the local machine. For more information, refer to [Removing Redundancy](#).



6. To set up the minimal configuration requirements, select the **General Settings** tab. For more information, refer to [General Settings](#).

Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

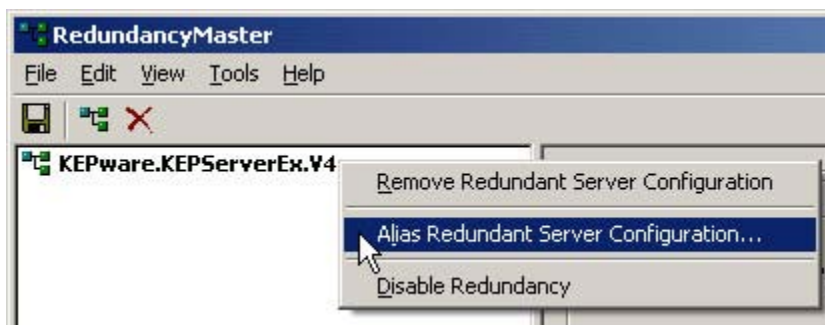
Aliasing Redundancy

If planning on creating multiple redundant OPC server pairs in RedundancyMaster that use the same ProgID, users will need to create an aliased ProgID for each of the redundant pairs. This will allow OPC clients to connect to the specific redundant pair by referring to the aliased ProgID of that redundant pair.

Note: If planning to use an existing OPC client project to connect to redundant pairs in RedundancyMaster that use aliased ProgIDs, users may be required to make a change to the existing client project. In the client project, references to the original ProgID may need to be changed to the alias.

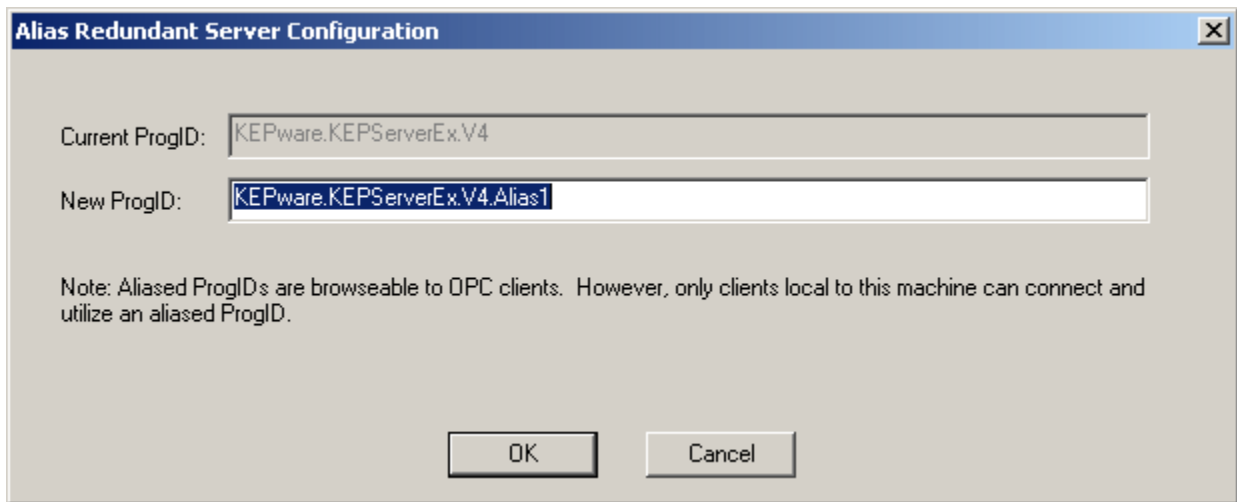
For an example on how to alias an existing redundant pair, refer to the instructions below.

1. Create an aliased ProgID from an existing redundant pair's assigned ProgID. To do so, right-click on the existing ProgID and then select **Alias Redundant Server Configuration...**

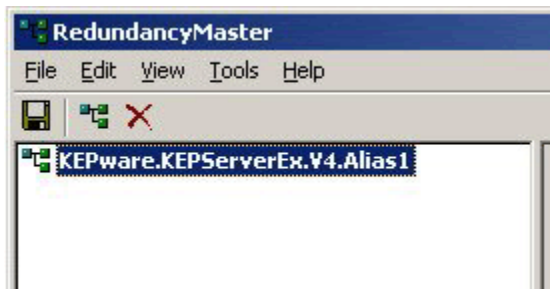


2. Next, click **OK** to accept the default ProgID alias that is displayed. Alternatively, type a different name and then click **OK**.

Note: ProgID alias names cannot include the characters <, > and &.



Note: The left pane of the screen should now display the ProgID with its alias.

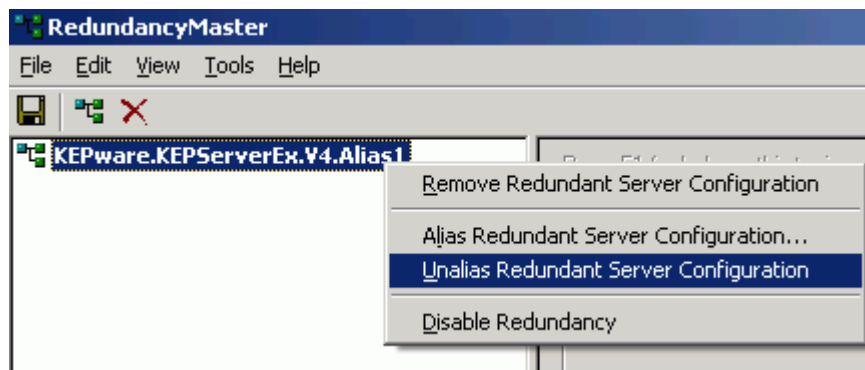


- Once one redundant pair has been aliased, another redundant pair can be created. An alias can then be assigned to that pair (such as, *ServerProgID.Alias2*).

See Also: [Unaliasing Redundancy](#)

Unaliasing Redundancy

To remove an alias from a redundant pair, right-click on the alias in the left pane of the main RedundancyMaster screen and then select **Unalias Redundant Server Configuration**. Alternatively, click **Edit | Unalias Redundant Server Configuration**.



Note: Unalias is not allowed if the original ProgID of the alias is currently being used as a redundant server.

General Settings

The General Settings tab provides the minimal configuration requirements that need to be defined in order to provide redundancy for the selected OPC server. These settings define where the underlying servers are located, how to connect to the servers and how often to test communications with the servers. Descriptions of these settings are below.

The screenshot shows the 'General Settings' tab of the RedundancyMaster application. It is divided into two main sections: 'Machine Settings' and 'Redundancy Options'.
In the 'Machine Settings' section, there are two text input fields. The first is labeled 'Primary (Hostname or IP Address):' and contains the text 'bob-xp'. The second is labeled 'Secondary (Hostname or IP Address):' and contains the IP address '192.168.111.67'. Both fields have a small '...' button to their right.
The 'Redundancy Options' section contains a dropdown menu for 'Connection mode:' set to 'Warm: Both machines, subscribe to items on active machine'. Below this is a checked checkbox labeled 'Always reconnect to primary machine upon availability'. At the bottom of this section are two spinners: 'Query server status interval (ms):' set to 2500 and 'Query server status timeout (ms):' set to 1000.
At the bottom of the dialog, there is a tabbed interface with 'General' selected, and other tabs for 'Monitoring', 'Diagnostics', and 'Notifications'.

Machine Settings

This parameter defines the machines that the redundancy application should use in order to connect to the underlying primary and secondary servers.

Primary Machine Name

The Primary Machine specifies the preferred connection that should be made to an OPC server. Every time a new client connection is made to the underlying server, the application will first attempt to make a connection to the server running on this machine. In the event that the connection to the primary fails or communications to the primary are lost, a connection to the secondary server will be attempted and established (if available). Depending on the connection mode, users can configure the application to automatically establish communications with the primary machine when it becomes available. For more information, refer to [Connection Mode](#).

To set the primary machine's name, manually type in the host name or IP address into the primary edit control. Alternatively, click the browse button (...) in order to invoke the Machine Browser.

Important: If the primary machine is the same machine running this application, set the machine name to "localhost."

Secondary Machine Name

The Secondary Machine specifies the fallback connection that should be made to an OPC server when communications to the primary machine are unavailable. Depending on the connection mode, users can minimize fail-over time by configuring the application to establish a connection to the secondary machine even if the primary is available. For more information, refer to [Connection Mode](#).

To set the secondary machine name, manually type in the host name or IP address into the secondary edit control. Alternatively, click the browse button (...) in order to invoke the Machine Browser.

Important: If the secondary machine is the same machine running this application, set the machine name to "localhost."

Connection Mode

The connection mode defines how and when the redundancy application should connect to the underlying primary and secondary servers. The mode of operation affects the amount of time it takes to fail over from one OPC server to the other. Some modes allow users to automatically promote communications to the primary (when available).

Cold: Active Machine Only

In this mode, the application will only connect to one underlying server at a time. On startup, a connection to the primary server will be made and all client related requests will be forwarded onto the primary. In the event that the connection to the primary fails (or communications to the primary are lost) a connection to the secondary will be made. If the redundancy application is unable to obtain a connection to the secondary, it will continue to ping-pong between the two servers until it makes a successful connection. Upon a successful connection, any subscriptions that the client has previously subscribed to will be automatically added to the newly 'active' server. Whenever communications to the active server are lost, the redundancy application will attempt to promote the inactive server in the ping-pong fashion just described.

Cold connection mode minimizes the amount of system resources that are allocated since only one connection is made to one server at any given time. This also reduces network traffic, since there is no need to poll the inactive machine in addition to the active machine.

The amount of time it takes to fail-over to the inactive server is detrimental, however. When the loss of communications with the active server is detected, the application must establish the connection to the inactive server, subscribe to all items on behalf of the client and then initiate the appropriate callback mechanisms. This time will vary depending on the underlying server (as well as network conditions). During this fail-over time, the client will not receive any data change or update notifications. Users should utilize one of the other connection modes if loss of data is crucial to the application.

Note: This mode does not support the "Always reconnect to Primary Machine" setting.

Warm: Both Machines, Subscribe to Items on Active Machine

In this mode, the application will attempt to maintain a connection to both the primary and secondary servers at all times. On startup, the application will initialize data callbacks for the primary server only. In the event that the connection to the primary fails (or communications to the primary are lost) a data callback will be initialized for the secondary server. In any event, the application will only receive data from and/or write data to the active server.

Periodically, both servers will be pinged to determine if the connection is still valid. If at anytime the redundancy application loses communications to either server, it will periodically attempt to reconnect to the failed server. If the redundancy application is only communicating with one server, this will be considered the active server. Upon successfully connecting to the primary server, the primary will automatically be made the active server if "Always connect to primary machine upon availability" is selected.

Warm connection mode increases the amount of system resources that are allocated since two server connections are made on behalf of the client. There is also a minimal increase in network traffic due to periodically pinging two servers instead of one. It is beneficial in that the fail-over time is minimized, since the redundancy application will only have to initialize data callbacks to the inactive server to begin receiving data. Users should utilize this connection mode in order to minimize network traffic and the loss of data in the application.

Hot: Both Machines, Subscribe to Items on Both Machines

In this mode, the application will attempt to maintain a connection to both the primary and secondary servers at all times. On startup, the application will initialize data callbacks for both primary and secondary servers so that both servers will send data change notifications. The data received from the primary will be forwarded onto the client. In the event that the connection to the primary fails (or communications to the primary are lost) data received for the secondary will immediately be forwarded onto the client. In either case, writes will only be forwarded to the active server.

Periodically, both servers will be pinged to determine if the connections are still valid. If at anytime the redundancy application loses communications to either server, it will periodically attempt to reconnect to the failed server. Upon successfully connecting to the primary server, the primary will automatically be made the active server if "Always connect to primary machine upon availability" is selected.

Hot connection mode increases the amount of system resources that are allocated since two server connections are made on behalf of the client. There is also an increase in network traffic due to receiving data change notifications from both underlying servers, as well as periodically pinging both servers to determine if they are still available. This setting

is beneficial in that fail-over time occurs immediately after detecting the loss of the active server. Users should utilize this connection mode if data loss is crucial to the application.

Always Connect to Primary Machine Upon Availability

This parameter enables the redundancy application to automatically promote communications back to the primary machine when the OPC server becomes available.

Server Status Settings

Query Server Status Interval

This interval determines how often the redundancy application will ping the underlying servers to determine if there has been a loss of communications. By querying at a faster rate, users can minimize fail-over time since failure detection occurs more frequently. It is specified in milliseconds.

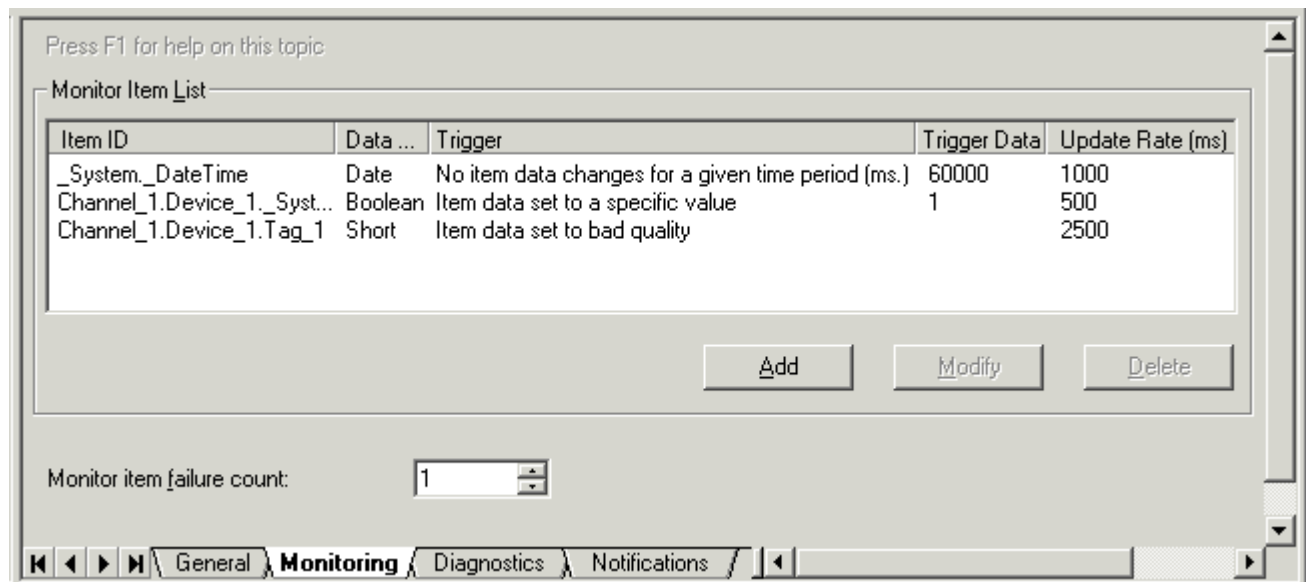
Query Server Status Timeout

This interval determines how long the redundancy application will wait for a ping response from the underlying servers before considering there to be a loss of communications. It is specified in milliseconds.

Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

Monitoring Settings

The Monitoring Settings tab is used to configure conditions that will initiate a fail-over to the inactive server. These conditions can be used to monitor server items for specific states in order to determine the health of the underlying servers/devices (beyond the automatic fail-over that occurs when communications are lost).

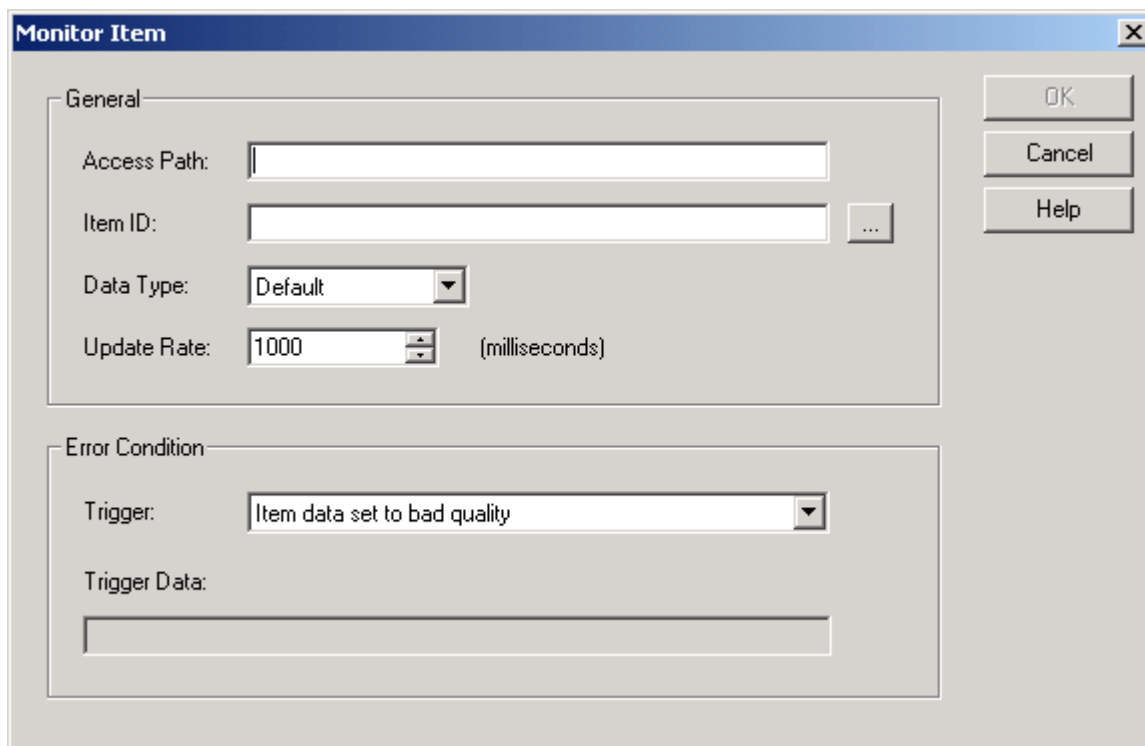


Monitor Item List

This list contains a collection of monitor items that can be assigned user-defined conditions. These items are monitored and tested to determine the health of the underlying servers and devices.

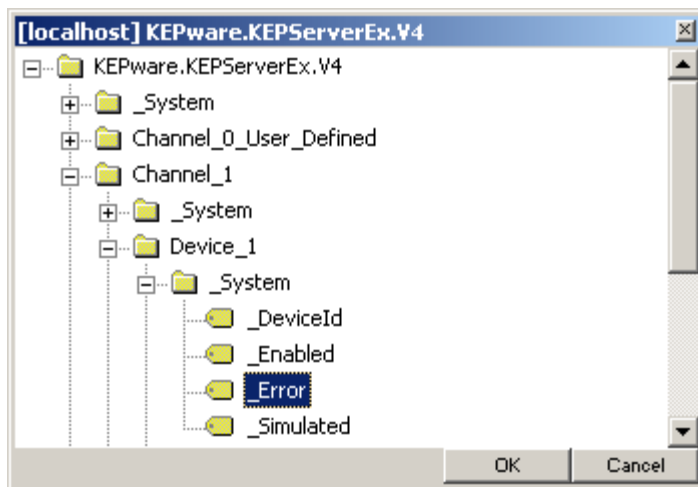
Monitor Item

This dialog is used to set the underlying OPC item definition along with the error condition for the monitor item. To add a monitor item, click **Add**.



Access Path

This parameter is used to manually type in the access path in the Access Path edit control (for those servers that require or allow one as part of an OPC item definition). If the underlying OPC server supports item browsing, users can browse the server's name space by clicking the browse button (...) instead. To select an item, locate the appropriate item and then press **OK**. The item definition will be filled in automatically.



Item ID

This parameter specifies the fully-qualified path which defines an item in the underlying OPC server. For servers that use or require access paths, this may be a partially qualified string. Users can manually type in the string which defines the item in the Item ID edit control. Alternatively, users can also browse the server's name space by clicking the browse button (...).

Data Type

This parameter specifies how the application will request the format of the data for the monitor item. Users can specify Default (which allows the server to return the data in the format of the item's native data type) or select a specific data type using the drop-down list. If the item definition is automatically filled in by using the item browse functionality

described above, the data type selection will indicate the format of the data that will be returned by the server by default.

Update Rate

This parameter specifies how often the underlying server should attempt to provide the application with the latest data, quality and time stamp for the monitor item. This rate is specified in milliseconds.

Trigger Condition

This parameter specifies how the application will test the monitor item to determine if it is in error. There are currently three supported conditions. Descriptions are as follows:

1. **Item data set to bad quality:** In this trigger condition, the item will be periodically tested to determine if the quality of the data received by the underlying server is "bad". When a quality of bad is detected, this particular monitor item will be considered in error. Depending on the "monitor item failure count" setting, this error condition may force the application to fail-over to the inactive server. When a quality of "good" is detected, the monitor item error will be cleared.
2. **Item data set to a specific value:** In this trigger condition, the item will be periodically tested to determine if the value of the data received by the underlying server is set to a specific value. Users must set the 'Trigger Data' to the specific value that they want monitored. When the specific value is detected, this particular monitor item will be considered in error. Depending on the "monitor item failure count" setting, this error condition may force the application to fail-over to the inactive server. When a value other than the specific value is detected, the monitor item error will be cleared.

Note 1: Some servers allow users to set an item to a specific 'dead value' when in error. Users might consider using this particular test and set the trigger data to the appropriate 'dead value' that will be reported by the server.

Note 2: Array data for this particular trigger condition is not currently supported by this application.

3. **No item data changes for a given time (ms)** - In this trigger condition, the item will be periodically tested to determine if the server has sent a data change notification within a specific time period (specified in milliseconds). The trigger data must be at least two times the update rate associated with the monitor item. If the trigger data is less than two times the update rate, an informational prompt will notify the user that the value was auto-adjusted will be displayed. Users will need to set the 'Trigger Data' to above the maximum acceptable allowed time that can occur without a data change notification. When the allotted time expires, this particular monitor item will be considered in error. Depending on the "monitor item failure count" setting, discussed below, this error condition may force the application to fail-over to the inactive server. Whenever an update is received, the monitor item error will be cleared and the expiration time will be reset.

Note: Most control systems implement error detection using watchdog values, which are values that change at a particular rate. If there are known values in the system that should be continuously changing, this trigger condition would be the most appropriate. Caution must be taken when selecting an appropriate timeout period. For example, even though a particular watchdog value may be changing every 100 milliseconds, the underlying server may not be able to poll this particular value at the same rate due to processing other tasks or polling other values. Users should determine the rate at which the underlying server can actually return a data change notification for this value to this application and multiply it by 2.

Monitor Item Failure Count

If the number of items in error meets or exceeds this count, a fail-over to the inactive server will occur. This count can range from 1 to the number of monitor items configured.

Note: Items that cannot be added to the underlying servers will be considered in error and an appropriate error message indicating the failure to add the monitor items will be logged.

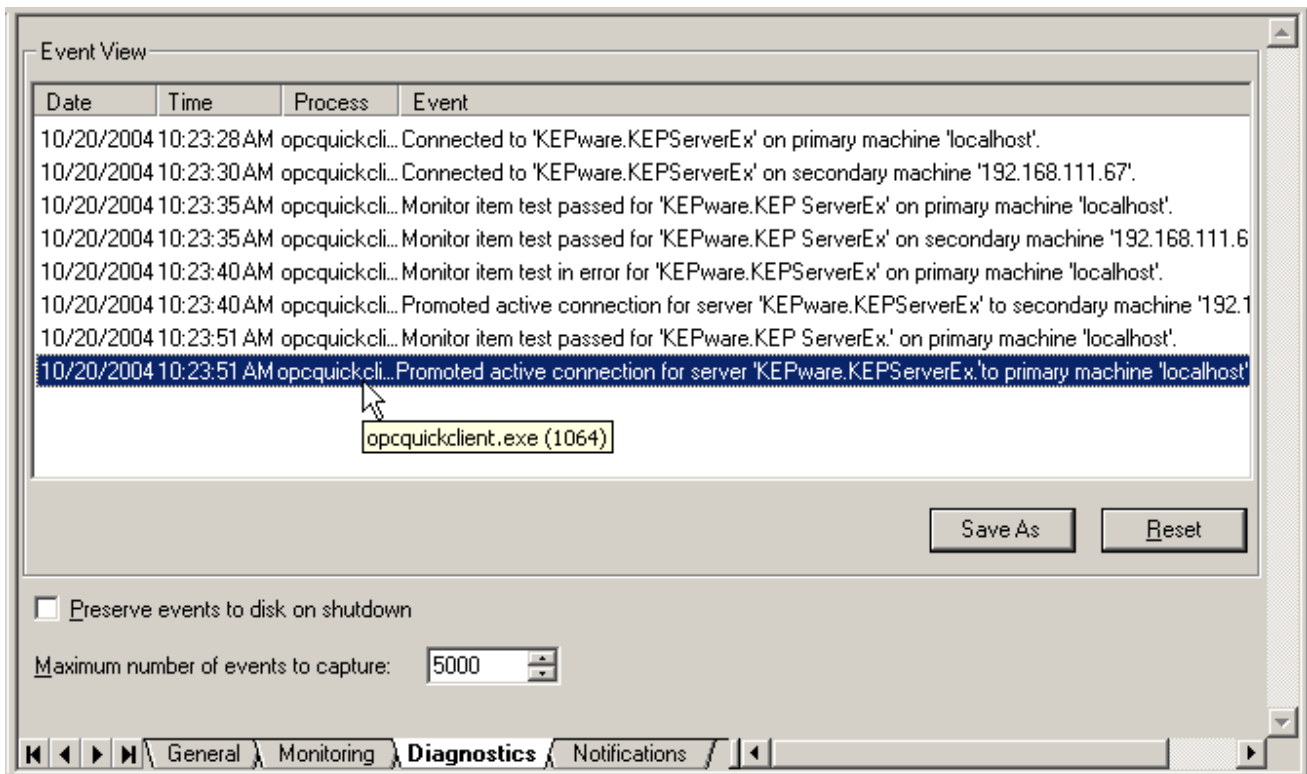
Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

Diagnosics Settings

The **Diagnosics Settings** tab provides an event view which displays information and/or errors that have occurred during Runtime operations. Each event contains the date and time it occurred, as well as the client process to which the message pertains. For a comprehensive list of Runtime events that can occur and their importance, refer to [Runtime Diagnosics](#).

Note: If the data in a particular column cannot be fully displayed, move the cursor over the entry. This will invoke a

tool tip that displays all the text.



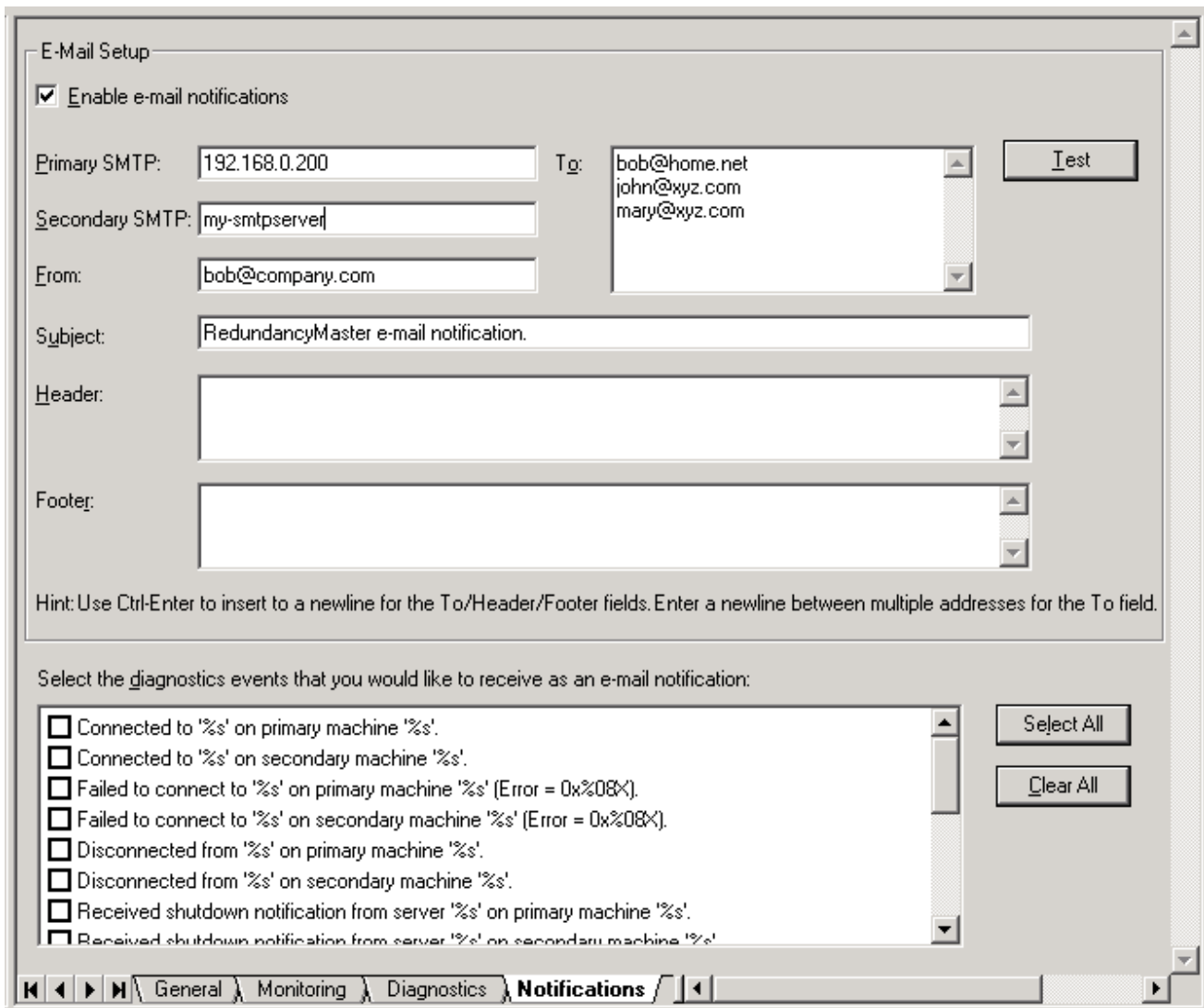
Descriptions of the parameters are as follows:

- **Save As:** This button saves the current diagnostics to a text file.
- **Reset:** This button clears the Event View and any diagnostics preserved to disk.
- **Preserve events to disk on shutdown:** When selected, events will be preserved to disk when the application is shutdown. The next time the application is started, the events will be displayed and any new events will be concatenated to the end of the view.
- **Maximum number of events to capture:** Since diagnostics utilize memory and storage resources, users may want to limit the number of diagnostics that are preserved at any given time. Once the maximum number of events has been reached, the oldest events will be discarded as necessary. The allowed range for this setting is 1000 to 30000. 5000 is the default.

Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

Notifications Settings

The Notifications Settings tab is used to configure one or more recipients to receive email notifications for one or more diagnostics events. The events visible under the local Diagnostics Settings View are available to send as email notifications.



E-Mail Setup

Enable e-mail notifications

Primary SMTP: 192.168.0.200 To: bob@home.net
john@xyz.com
mary@xyz.com Test

Secondary SMTP: my-smtpserver

From: bob@company.com

Subject: RedundancyMaster e-mail notification.

Header:

Footer:

Hint: Use Ctrl-Enter to insert to a newline for the To/Header/Footer fields. Enter a newline between multiple addresses for the To field.

Select the diagnostics events that you would like to receive as an e-mail notification:

- Connected to '%s' on primary machine '%s'.
- Connected to '%s' on secondary machine '%s'.
- Failed to connect to '%s' on primary machine '%s' (Error = 0x%08X).
- Failed to connect to '%s' on secondary machine '%s' (Error = 0x%08X).
- Disconnected from '%s' on primary machine '%s'.
- Disconnected from '%s' on secondary machine '%s'.
- Received shutdown notification from server '%s' on primary machine '%s'.
- Received shutdown notification from server '%s' on secondary machine '%s'.

Select All

Clear All

General Monitoring Diagnostics **Notifications**

Enable Email Notifications

This parameter enables the email notification system.

Primary SMTP

This parameter specifies the host name or IP address of the preferred machine capable of sending email utilizing the Simple Mail Transport Protocol. To set the primary SMTP, manually type in the host name or IP address into the primary SMTP edit control

When an initial email notification is to be sent, the application will first attempt to send the notification through this machine. In the event that the connection to the primary SMTP fails, an attempt to send the email notification through the secondary will be attempted (if configured).

Note: If the same email notification fails to be sent through the primary and secondary, the email will be immediately discarded and an error will be reported in the diagnostics event view. For more information, refer to [Diagnostics Settings](#).

Secondary SMTP (Optional)

This parameter specifies host name or IP address of the secondary machine capable of sending email utilizing the Simple Mail Transport Protocol. To set the secondary SMTP, manually type in the host name or IP address into the primary SMTP edit control.

When an attempt to send an email notification through the primary fails, an attempt to use the secondary will occur. If the application can successfully send mail through the secondary, it will continue to use this machine until a failure occurs. At that point, it will retry the primary.

Note: If the same email notification fails to be sent through the primary and secondary, the email will be immediately discarded and an error will be reported in the diagnostics event view. For more information, refer to [Diagnostics Settings](#).

From

This parameter specifies the email address of the sender that will appear as the originator of all email notifications sent from the application.

To

This parameter contains one or more recipients' email addresses that all notifications should be sent to. Each email address must appear on its own line. To start a new line with the cursor flashing in the To field, press **Ctrl+Enter**.

Subject (Optional)

This parameter contains the subject text that will appear in all email notifications sent from this application.

Header (Optional)

This parameter contains any text that will appear before the event text in the body of the email message.

Footer (Optional)

This parameter contains any text that will appear after the event text in the body of the email message.

List of Available Email Notifications

The events selected in this list will be sent to recipients as they occur. All events described in [Runtime Diagnostics](#) are available as email notifications.

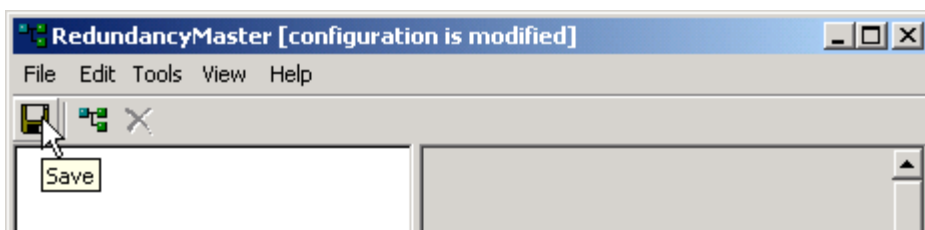
Testing the Setup

To test the email settings, simply press **Test**. An automated test message will be sent to all recipients.

Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

Applying Your Settings

RedundancyMaster can perform changes for many settings without having to stop and restart the client applications. The title bar will indicate that changes have been made to the settings by displaying "[configuration is modified]." In order to apply the changes, users are required to save the configuration. To do so, either click the **Save** toolbar button or click **File | Save**.



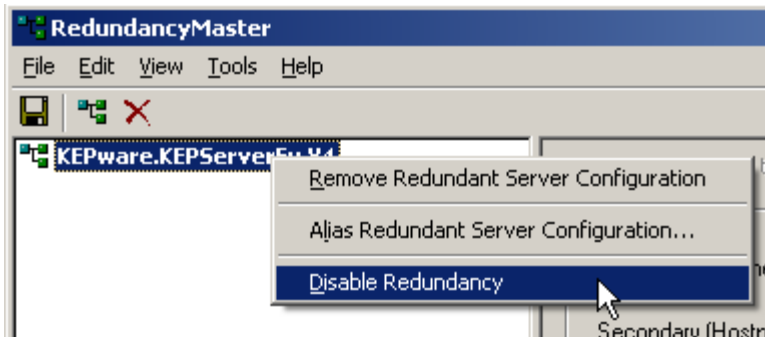
Note: If configuration has been newly added and clients are already running and connected to the OPC server to which redundancy will be added, the client applications will need to be shutdown and restarted for redundancy to be initiated. Likewise, if users are removing redundancy from an OPC server that clients are currently using, the client applications will need to be shutdown and restarted for redundancy to be fully removed.

Enable/Disable Redundancy

Disabling Redundancy

There may be times when redundancy should be unhooked from the client application. When doing so, it is not necessary to delete the redundant configuration from the application. To disable redundancy for a server, right-click on

the server and then select **Disable Redundancy**.

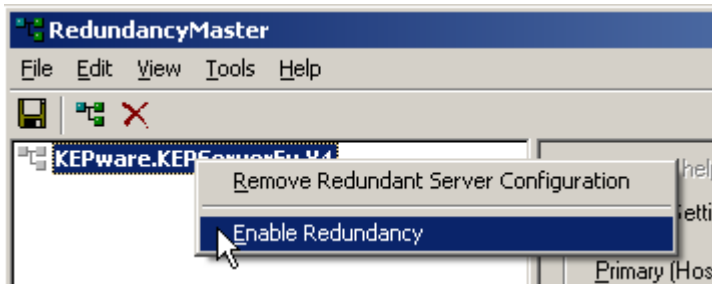


Note: The icon that was previously green will turn to gray to indicate its disabled state.

Important: Enabling/Disabling redundancy on a server is not considered a configuration change because the required redundancy hooks must be immediately registered/unregistered (respectively) with the system registry. Users will not be prompted to save the configuration for this particular operation. As such, this is an operation that cannot be accomplished spontaneously. All client applications will have to be completely shutdown and restarted to ensure that redundancy is either enabled or disabled properly.

Enabling Redundancy

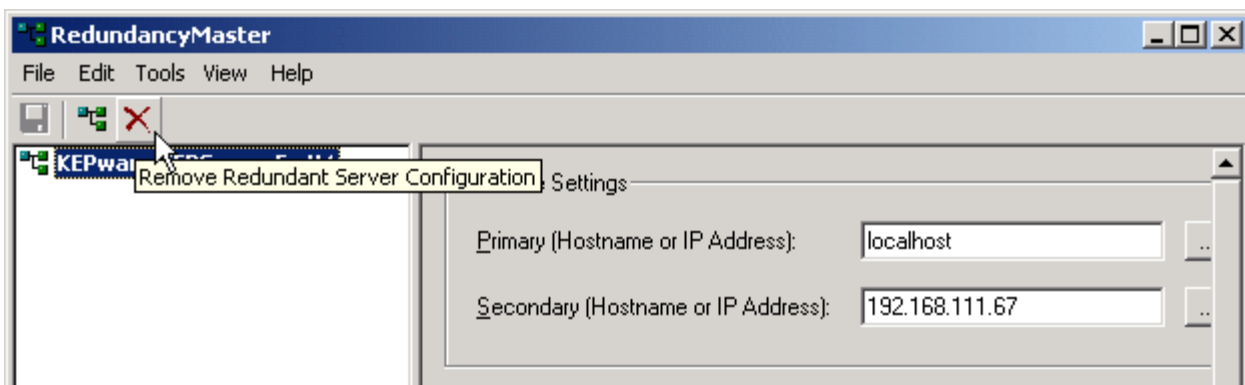
To re-enable redundancy, simply right-click the server as before and select **Enable Redundancy** from the context menu.



Note: The icon that was previously gray will turn to green to indicate its enabled state.

Removing Redundancy

To delete the redundant configuration for a server, select the server in the left-hand pane and then click the **Remove Redundant Server Configuration** toolbar button (denoted by a red X).



After a redundant server configuration has been removed, the redundant server will still be displayed in the OPC Server Browser window. To update the list of servers, click **Refresh**. To refresh the browser window, right-click on the

machine name and then click **Refresh**. Afterwards, the redundant server that was removed should no longer be displayed.



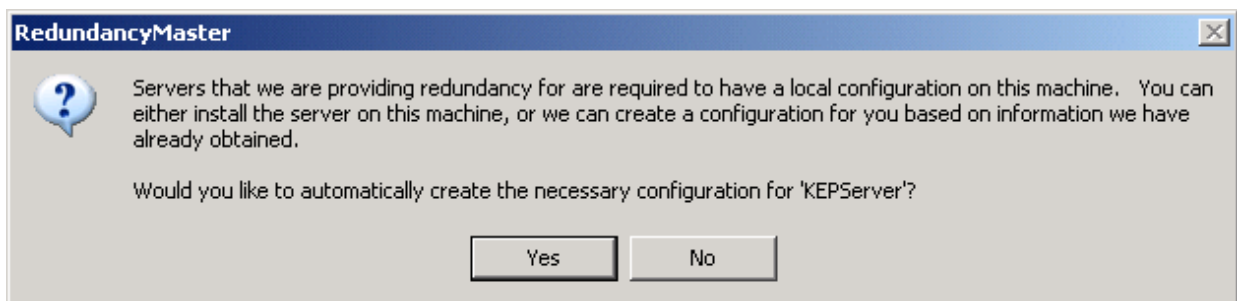
Important: Once changes have been made, users must remember to save the new settings. For more information, refer to [Applying Your Settings](#).

Deployment

There may be times where a redundancy project is configured and tested on one machine, but must be deployed on another machine (target). For information on deploying the project on another machine, refer to the instructions below.

1. Install the RedundancyMaster application on the Target Machine. The Target Machine must conform to the application requirements. If RedundancyMaster has been started on the Target Machine, shutdown the application before continuing.
2. Copy the **redundancymaster.xml** file (located in the **\RedundancyMaster** installation directory of the test machine) to the **\RedundancyMaster** installation directory on the Target Machine.
3. Start RedundancyMaster on the Target Machine in order to activate the project that was copied.
4. All redundant servers will be disabled on the Target Machine. Users must enable these servers in order to provide redundancy to the underlying servers because the enable/disable state is machine-dependent and not contained in the configuration file. To enable the disabled redundant servers, right-click on the server name and then select **Enable Redundancy**.
5. If the server to which redundancy is being added is not installed on the local machine, a message window will be invoked. A minimal configuration for the remote server needs to be registered on the local machine in order for Redundancy Master to provide redundancy. Answer the prompt "Would you like to automatically create the necessary configuration for [server name]?" by clicking **Yes**.

Note: If this redundant server configuration is ever removed or if the RedundancyMaster application is ever uninstalled from the local machine, the minimal configuration for the remote server will be automatically cleaned up from the local machine.



6. In the confirmation message, click **Yes**.

Note: At this point, the project has been successfully deployed to the Target Machine. Any client applications that require redundancy will need to be restarted. Users should ensure that the appropriate security/DCOM security rights are assigned to the Target Machine.

See Also: [Enable/Disable Redundancy](#) and [Removing Redundancy](#).

Runtime Diagnostics

RedundancyMaster logs events that provide description of information and errors that have occurred during Runtime. Events may be viewed in the diagnostics event view and may also be sent to one or more recipients as an email notification. Click on a link for a description of the event.

[Attempt to add monitor item '<item id>' for '<server name>' on primary machine '<machine name>' failed. This monitor item will be considered in error](#)

[Attempt to add monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' failed. This monitor item will be considered in error](#)

[Connected to '<server name>' on primary machine '<machine name>'](#)

[Connected to '<server name>' on secondary machine '<machine name>'](#)

[Disconnected from '<server name>' on primary machine '<machine name>'](#)

[Disconnected from '<server name>' on secondary machine '<machine name>'](#)

[Failed to connect to '<server name>' on primary machine '<machine name>'](#)

[Failed to connect to '<server name>' on secondary machine '<machine name>'](#)

[Monitor item '<item id>' for '<server name>' on primary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition](#)

[Monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition](#)

[Monitor item test in error for '<server name>' on primary machine '<machine name>'](#)

[Monitor item test in error for '<server name>' on secondary machine '<machine name>'](#)

[Monitor item test passed for '<server name>' on primary machine '<machine name>'](#)

[Monitor item test passed for '<server name>' on secondary machine '<machine name>'](#)

[Promoted active connection for server '<server name>' to primary machine '<machine name>'](#)

[Promoted active connection for server '<server name>' to secondary machine '<machine name>'](#)

[Received shutdown notification from server '<server name>' on primary machine '<machine name>'](#)

[Received shutdown notification from server '<server name>' on secondary machine '<machine name>'](#)

[Unable to retrieve status for server '<server name>' on primary machine '<machine name>'. Server communications has been lost](#)

[Unable to retrieve status for server '<server name>' on secondary machine '<machine name>'. Server communications has been lost](#)

See Also: [Diagnostics Settings](#) and [Notifications Settings](#).

Attempt to add monitor item '<item id>' for '<server name>' on primary machine '<machine name>' failed. This monitor item will be considered in error

Severity:

Error

Description:

A monitoring item that has been configured cannot be added to the underlying server on the primary machine. Items that cannot be added to the underlying servers will be considered in error. An appropriate error message will be logged that indicates the failure to add the monitor items.

Attempt to add monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' failed. This monitor item will be considered in error

Severity:

Error

Description:

A monitoring item that has been configured cannot be added to the underlying server on the secondary machine. Items that cannot be added to the underlying servers will be considered in error. An appropriate error message will be logged that indicates the failure to add the monitor items.

Connected to '<server name>' on primary machine '<machine name>'

Severity:

Informational

Description:

A successful connection has been made to the underlying server on the primary machine.

Connected to '<server name>' on secondary machine '<machine name>'**Severity:**

Informational

Description:

A successful connection has been made to the underlying server on the secondary machine.

Disconnected from '<server name>' on primary machine '<machine name>'**Severity:**

Informational

Description:

The redundancy application dropped the connection to the underlying server on the primary machine.

Disconnected from '<server name>' on secondary machine '<machine name>'**Severity:**

Informational

Description:

The redundancy application dropped the connection to the underlying server on the secondary machine.

Failed to connect to '<server name>' on primary machine '<machine name>'**Severity:**

Error

Description:

The redundancy application cannot connect to the underlying server on the primary machine. Even though the redundancy application will continue to attempt this connection (depending on the connection mode settings), this error is not continuously posted: it is only posted on the initial attempt or on the transition from a connected to a failed state.

Failed to connect to '<server name>' on secondary machine '<machine name>'**Severity:**

Error

Description:

The redundancy application cannot connect to the underlying server on the secondary machine. Even though the redundancy application will continue to attempt this connection (depending on the connection mode settings), this error is not continuously posted: it is only posted on the initial attempt or on the transition from a connected to a failed state.

Monitor item '<item id>' for '<server name>' on primary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition**Severity:**

Error

Description:

A monitoring item that has been configured with the 'specific value' trigger condition received array data from the underlying server on the primary machine. This application does not support array data for this particular trigger

condition.

Monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition

Severity:

Error

Description:

A monitoring item that has been configured with the 'specific value' trigger condition received array data from the underlying server on the secondary machine. This application does not support array data for this particular trigger condition.

Monitor item test in error for '<server name>' on primary machine '<machine name>'

Severity:

Error

Description:

The monitoring item test failed for the underlying server on the primary machine. This application will fail-over to the secondary machine. Even though the monitor item test may continue to fail for a given period of time, the redundancy application will not continuously post this error: it is only posted on the initial monitor test or on the transition to a failed state.

Monitor item test in error for '<server name>' on secondary machine '<machine name>'

Severity:

Error

Description:

The monitoring item test failed for the underlying server on the secondary machine. This application will fail-over to the primary machine. Even though the monitor item test may continue to fail for a given period of time, the redundancy application will not continuously post this error: it is only posted on the initial monitor test or on the transition to a failed state.

Monitor item test passed for '<server name>' on primary machine '<machine name>'

Severity:

Informational

Description:

The monitoring item test succeeded for the underlying server on the primary machine. This application will promote the primary machine, as required by the connection mode settings. Even though the monitor item test may continue to pass for a given period of time, the redundancy application will not continuously post this message: it is only posted on the initial monitor test or on the transition to a passed state.

Monitor item test passed for '<server name>' on secondary machine '<machine name>'

Severity:

Informational

Description:

The monitoring item test succeeded for the underlying server on the secondary machine. Even though the monitor item test may continue to pass for a given period of time, the redundancy application will not continuously post this message: it is only posted on the initial monitor test or on the transition to a passed state.

Promoted active connection for server '<server name>' to primary machine '<machine name>'

Severity:

Informational

Description:

The redundancy application successfully failed-over to the underlying server on the primary machine.

Promoted active connection for server '<server name>' to secondary machine '<machine name>'

Severity:

Informational

Description:

The redundancy application successfully failed-over to the underlying server on the secondary machine.

Received shutdown notification from server '<server name>' on primary machine '<machine name>'

Severity:

Informational

Description:

The redundancy application received an OPC shutdown notification from the underlying server on the primary machine. This notification is server specific and usually occurs if the server application is terminated by a user, the server application demo period times out or the machine the server is running on is shutdown. In either case, this application will then fail-over to the secondary machine. An attempt to reconnect to the primary will occur, as required by the connection mode settings.

Received shutdown notification from server '<server name>' on secondary machine '<machine name>'

Severity:

Informational

Description:

The redundancy application received an OPC shutdown notification from the underlying server on the secondary machine. This notification is server specific and usually occurs if the server application is terminated by a user, the server application demo period times out or the machine the server is running on is shutdown. In either case, this application will then fail-over to the primary machine. An attempt to reconnect to the secondary will occur, as required by the connection mode settings.

Unable to retrieve status for server '<server name>' on primary machine '<machine name>'. Server communications has been lost

Severity:

Error

Description:

The redundancy application failed to receive a ping response from the underlying server on the primary machine after establishing a connection successfully. This application will then fail-over to the secondary machine and attempt to reconnect to the primary, as required by the connection mode settings.

Unable to retrieve status for server '<server name>' on secondary machine '<machine name>'. Server communications has been lost

Severity:

Error

Description:

The redundancy application failed to receive a ping response from the underlying server on the secondary machine after establishing a connection successfully. This application will then fail-over to the primary machine and attempt to reconnect to the secondary, as required by the connection mode settings.

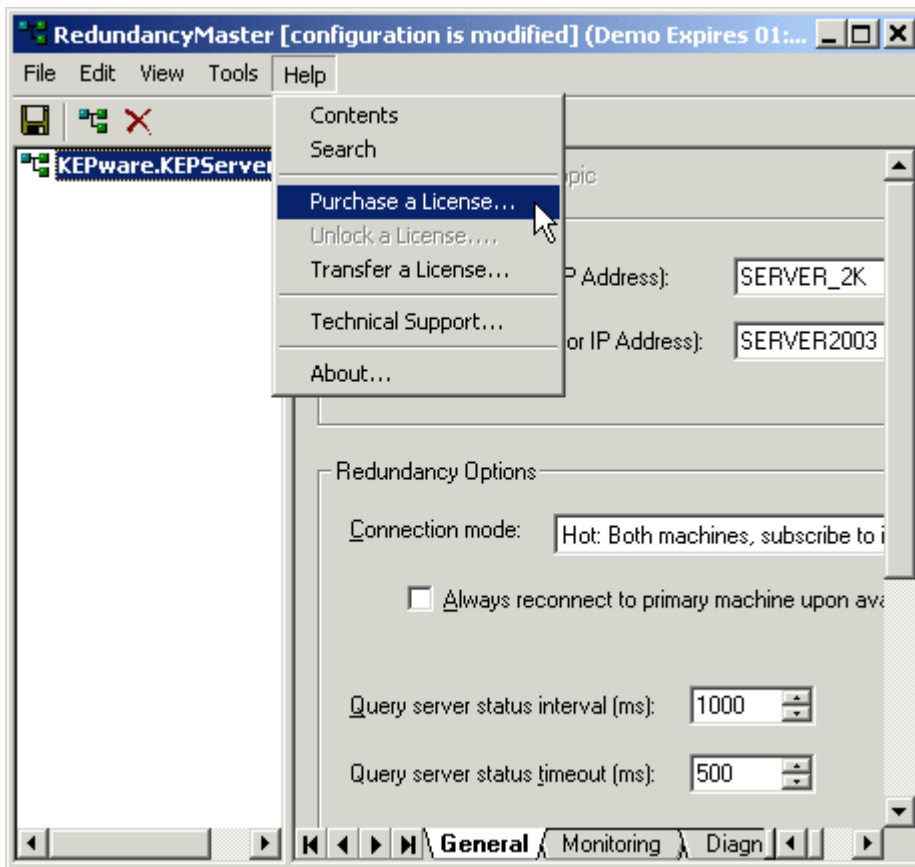
Licensing Overview

RedundancyMaster uses a software-based protection system in order to reduce unlicensed use and prevent undue hardship on legitimate users. A software-based protection system was chosen over hardware-based key protection in response to overwhelming customer feedback. Hardware-based protection systems provide the ability to move a license from one machine to another. With a hardware key, this is usually just a matter of removing the key from one machine and placing it on the new machine. Software-based systems, however, have attempted to achieve the same transportability through the use of a proprietary key disk. Many proprietary key disks are created using a specific format that prevents normal copying of the disk, however. While this may provide additional security for the software vendor, customers may experience problems when the key disk becomes corrupt.

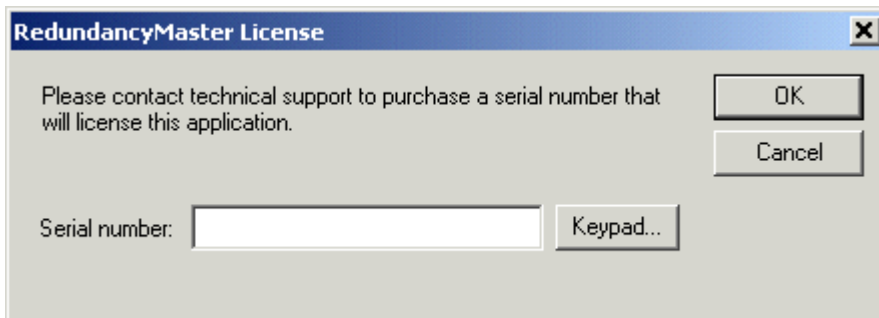
RedundancyMaster's software-based protection system provides a mechanism for transferring a license from one machine to another without the use of a proprietary key disk. Removable disk media is required to move the license from one PC to another. No special formatting is required for the disk, which may be copied if need be. This is useful for developers and integrators who need to design and test on their desktop or laptop systems before the final project is installed on the end user's machine.

Purchase a License

To purchase a license, click **Help | Purchase a License...** from the application's taskbar.

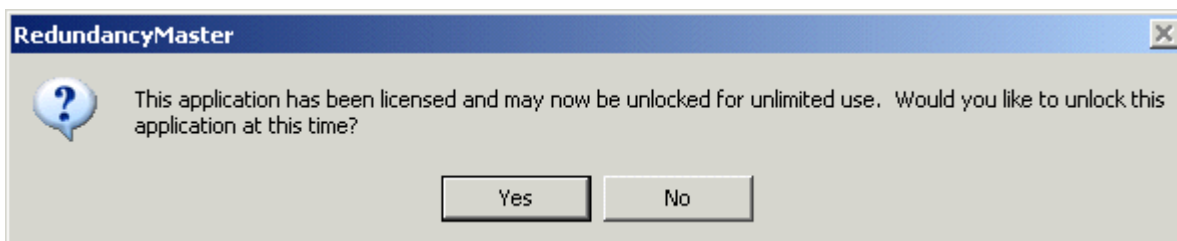


Note: The dialog should appear as shown below.



Note: A product license registration code may be obtained from Technical Support. For contact information, click **Help | Technical Support**. Once the product registration code is obtained, it can be entered in this field. If a keyboard is unavailable, use the on-screen keypad to enter the code.

The dialog below will be displayed when the application has been successfully purchased and pre-registered (licensed).

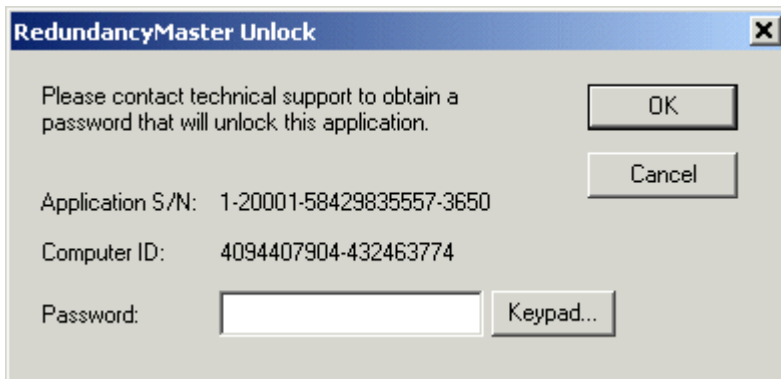


Important: Once the product registration code has been entered, the license will only continue to run for a 10-day grace period until it is unlocked. To complete the licensing sequence, users must unlock the licenses. Users have up to 10 days to contact Technical Support to unlock the application for full-unrestricted use.

See Also: [Unlock a License](#)

Unlock A License

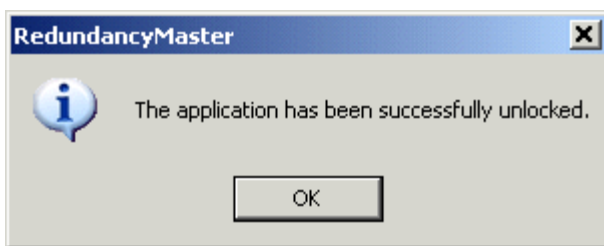
Once a license has been purchased and installed, the application can be unlocked for full, unlimited use on the Target PC. The **Help | Unlock a License...** menu option will only be enabled if a license is detected but has not been fully unlocked.



At this point, users can contact Technical Support by email, fax or phone. For contact information, click **Help | Technical Support**. The data presented on this dialog must be provided to receive a password. The **S/N** and **Computer ID** must be returned accurately to allow Technical Support to generate the appropriate password. Once the password has been obtained, it may be entered in this dialog to fully unlock the application. The on-screen keypad may be used to enter the code if a keyboard is not available.

If Technical Support cannot be contacted immediately, users may run the application in its initial 10-day licensed run mode. The 10-day installation mode is designed to allow users the time to contact Technical Support when it is convenient. If Technical Support cannot be contacted immediately, hit **Cancel** to start the 10-day installation grace period. The application's title bar will indicate that the 10-day installation grace period has begun.

When the application has been successfully unlocked, the dialog should appear as shown below.



License Transfer Instructions

RedundancyMaster's software-based protection system provides a mechanism for transferring a license from one machine to another. This is useful for developers and integrators who need to design and test on their desktop or laptop systems before the final project is installed on the end user's machine. The license transfer system does not require a proprietary key disk: any type of removable disk media can be used to move the selected license from one PC to another. No special formatting is required, and the disk may be copied if needed.

Requirements

The following is required to transfer a license:

1. Access to both the source and target systems. The source system must contain a valid license.
2. Each system should be equipped with a compatible removable media drive.

3. Formatted removable media (such as a floppy disk or USB key).

General Instructions

To transfer an existing license from one system to another, follow the instructions below.

1. Read and accept the license agreement.
2. Prepare a license disk for the target system.
3. Transfer an existing license from the source system to the license disk. This will unregister the application on the source system.
4. Transfer an existing license from the license disk to the target system. This will register the application on the target system.

License Transfer - Step 1: Agreement

To move a license from one PC to another, click **Help** and then select **Transfer a License...** Users are encouraged to re-read the software license agreement when moving a license from one machine to another. To accept the agreement, select the radio button next to **I accept the terms of this license agreement**. Then, proceed to [Step 2: Prepare Removable Media for License](#).

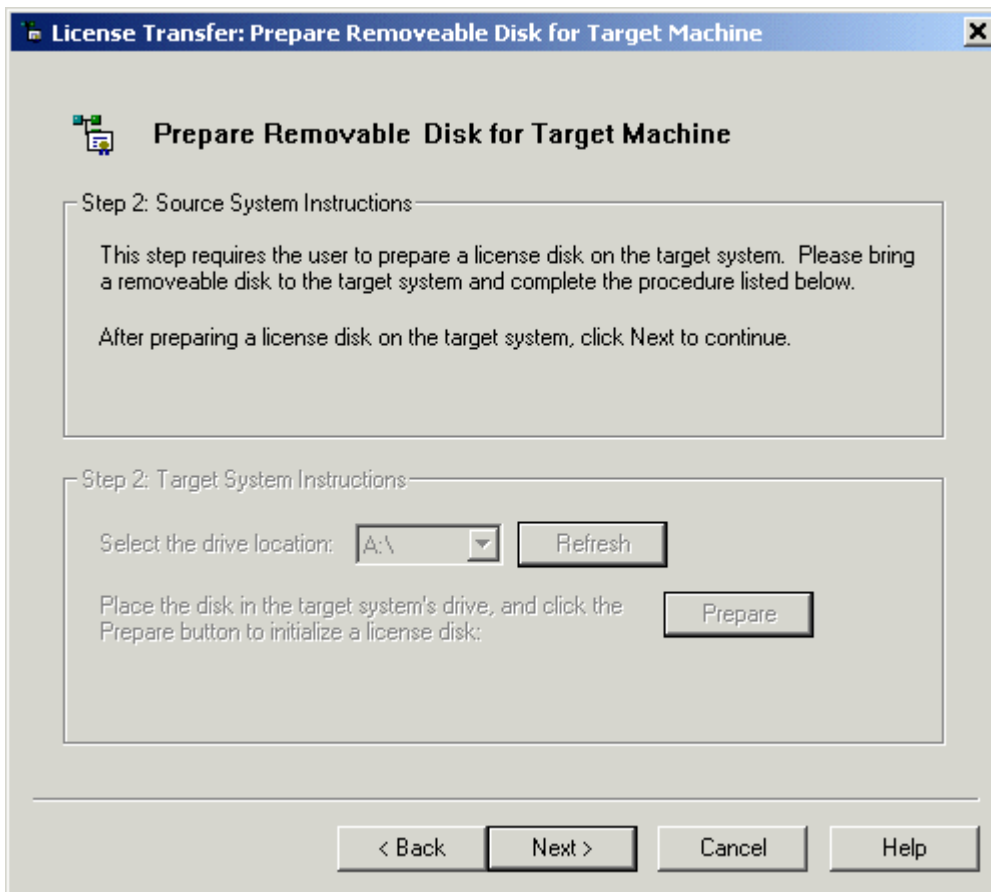
Caution: Before beginning the license transfer process, it is recommended that users have clean and newly formatted removable media ready. Once the license has been removed from the Source Machine, the process of transferring a license from one machine to another must be completed. If the process is cancelled after the license has been removed, the transfer will corrupt and users will not be able to install on either the Source or Target PC. With this in mind, ensure that the steps described above can be completed before the license transfer process begins.

License Transfer - Step 2: Prepare Removable Media for License

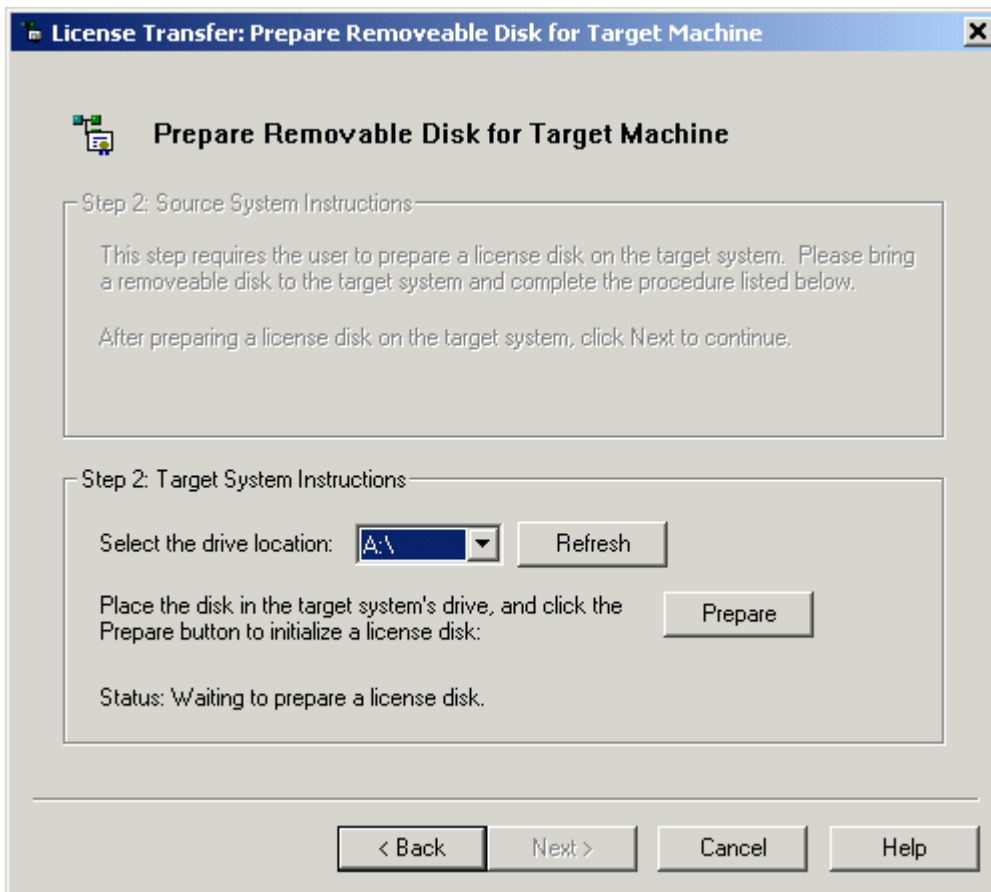
The type of dialog invoked next depends on whether the Source or Target PC is being used.

Source PC

Prepare a diskette for license transfer on the Target PC. Then, select **Prepare**.



Target PC



The diskette preparation done on the Target PC produces the media required to move the license from the Source Machine. In other words, the target system is required to produce the diskette needed to move the RedundancyMaster license from the Source Machine. If the diskette preparation is interrupted, the license will fail to transfer from the Source Machine. Once the diskette is prepared by the Target PC, users may proceed to [Step 3: Transfer License from Source PC to Removable Media](#).

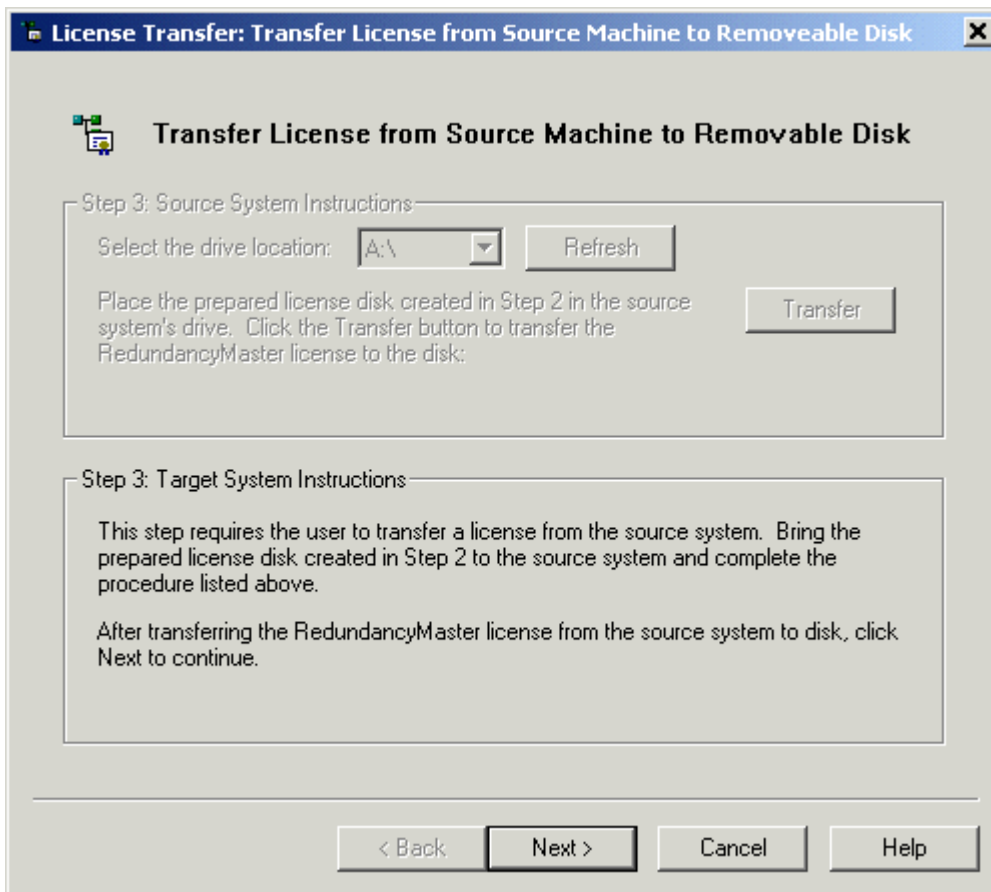
Caution: Before beginning the license transfer process, it is recommended that users have clean and newly formatted removable media ready. Once the license has been removed from the Source Machine, the process of transferring a license from one machine to another must be completed. If the process is cancelled after the license has been removed, the transfer will corrupt and users will not be able to install on either the Source or Target PC. With this in mind, ensure that the steps described above can be completed before the license transfer process begins.

License Transfer - Step 3: Transfer License from Source PC to Removable Media

Once the Target PC has prepared the license diskette, the following dialog will appear.

Target PC

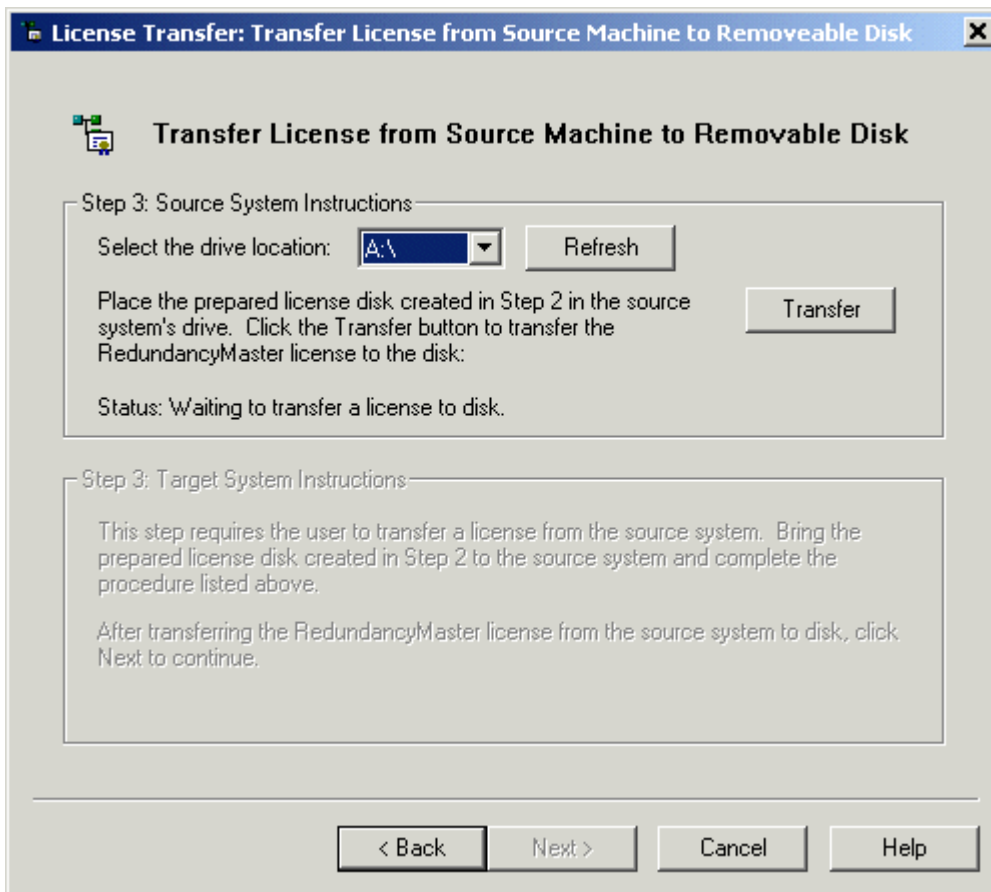
Take the prepared diskette to the Source Machine.



Note: The Source Machine's dialog should appear as shown below.

Source PC

Place the diskette prepared by the Target PC into the removable drive of the Source Machine and then press **Transfer**. The Transfer Utility will transfer the license to the diskette. When finished, proceed to [Step 4: Transfer License from Removable Media to Target PC](#).



Important: Removing the diskette from the removable drive before the disk light has been turned off may result in license information loss. Allow all disk operations to complete before attempting to remove any diskettes from the PC.

Caution: Once the license has been removed from the Source Machine, the process of transferring a license from one machine to another must be completed. If the process is cancelled after the license has been removed, the transfer will corrupt and users will not be able to install on either the Source or Target PC. With this in mind, ensure that the steps described above can be completed before the license transfer process begins.

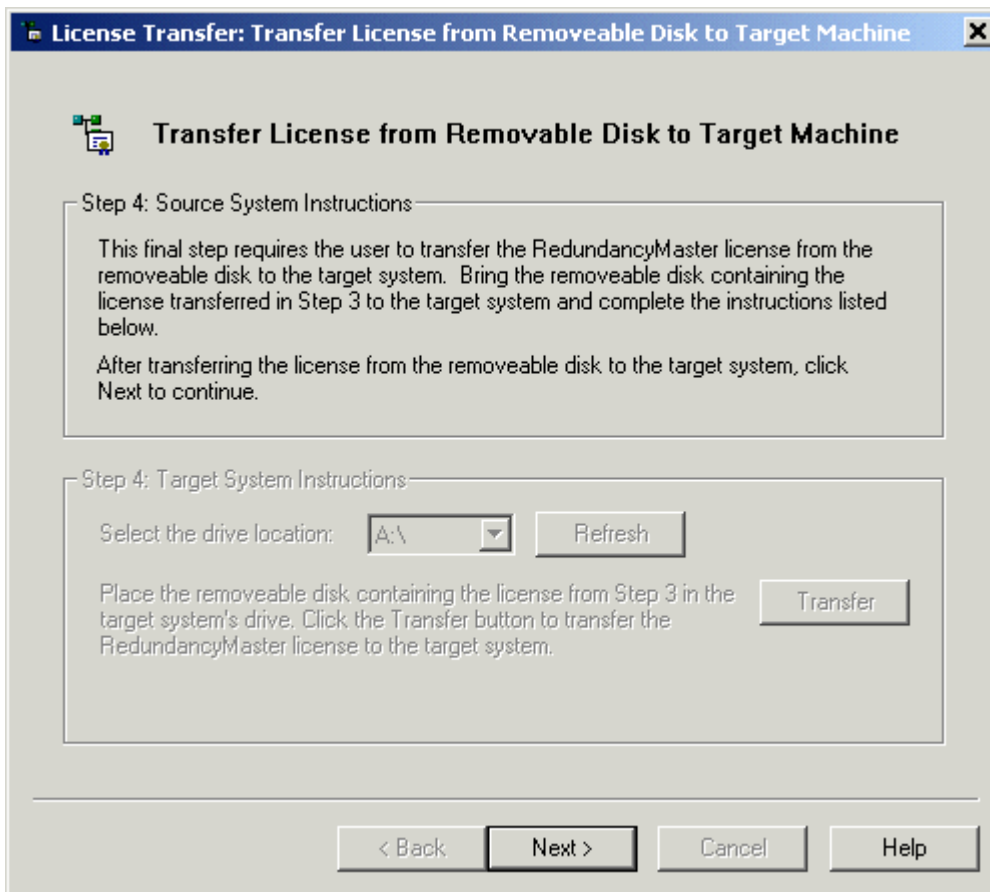
License Transfer - Step 4: Transfer License from Removable Media to Target PC

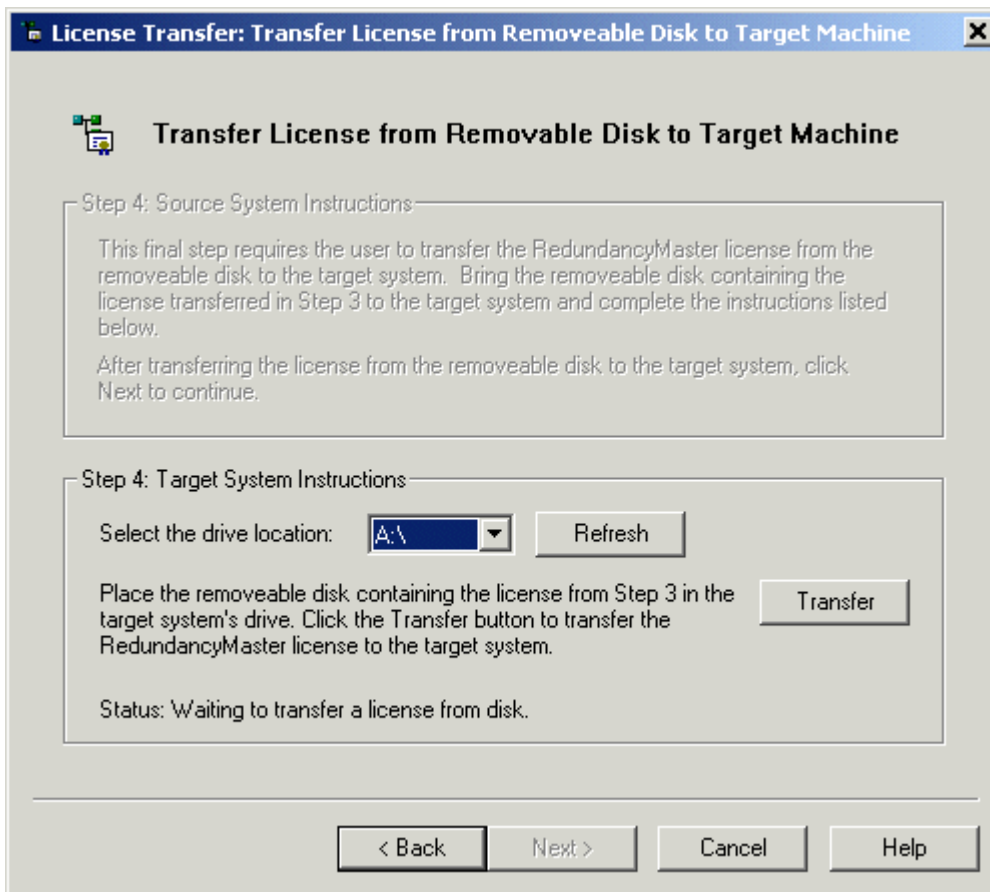
At this point, the license has been transferred to the floppy diskette from the Source PC.

Source PC

Users should now have a disk that contains a license to be installed on the target system. Place the disk in the target system's selected drive and then press **Transfer**.

Important: Do not attempt to remove the diskette from the drive, as this might cause an error that results in the loss of license information.

**Target PC**

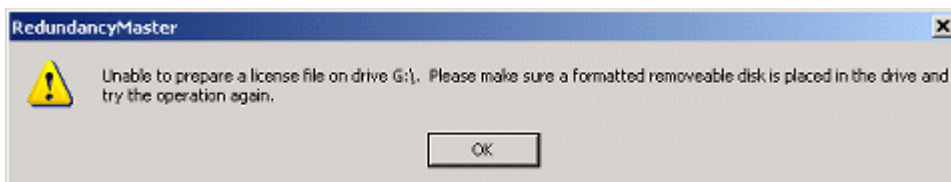


Once the license information has been transferred, click **Next>** to proceed to [Step 5: Completion](#). If the transfer is successful, the license has now been transferred to the Target Machine.

Troubleshooting License Transfer

The diskette is the most common cause of error. Although many users may just grab the nearest diskette, blow the dirt off and stick it into the drive, a clean, newly formatted diskette is recommended. As mentioned previously, removing the diskette from the drive during either the target disk preparation or actual license transfer portion on the Source Machine may cause an error that will result in the loss of license information.

Users may also encounter an instance where the transfer of license information from the Source Machine to the prepared diskette fails.

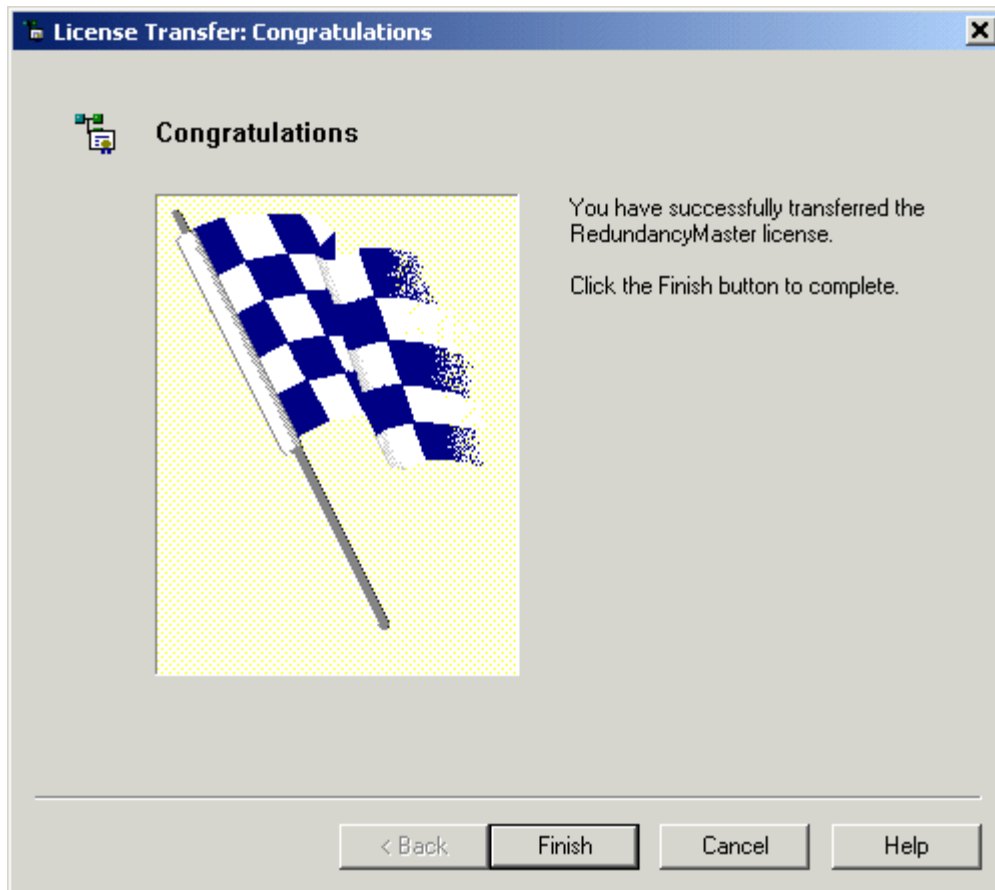


Note: Once the license transfer has been completed successfully, the diskette will no longer contain any license information. It does not need to be maintained.

Caution: The process of transferring a license from one machine to another must be completed once the license has been removed from the Source Machine. If the license transfer process is cancelled after the license has been removed, the process will become corrupt and users will not be able to install on either the source or Target PC. With this in mind, ensure that the steps above can be completed before the license transfer process begins.

License Transfer - Step 5: Completion

Congratulations! The following dialog will appear once a license has been successfully transferred.



Index

- A -

- Adding Redundancy 9
- Alias ProglID 9, 10
- Aliasing Redundancy 10
- Applying Your Settings 19
- Attempt to add monitor item '<item id>' for '<server name>' on primary machine '<machine name>' failed. This monitor item will be considered in error 22
- Attempt to add monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' failed. This monitor item will be considered in error 22

- C -

- Cold Mode 13
- Connected to '<server name>' on primary machine '<machine name>' 22
- Connected to '<server name>' on secondary machine '<machine name>' 23
- Connection Mode 13

- D -

- Deployment 21
- Diagnostics Settings 16
- Disable Redundancy 19
- Disconnected from '<server name>' on primary machine '<machine name>' 23
- Disconnected from '<server name>' on secondary machine '<machine name>' 23

- E -

- E-mail Support 17
- Enable Redundancy 19
- Enable/Disable Redundancy 19
- Errors 21

- F -

- Failed to connect to '<server name>' on primary machine '<machine name>' 23
- Failed to connect to '<server name>' on secondary machine '<machine name>' 23

- G -

- General Settings 12

- I -

- Introduction 3

- L -

- License 26, 28, 29, 31, 33, 36
- License Transfer 28, 29, 31, 33, 36
- License Transfer - Step 1: Agreement 29
- License Transfer - Step 2: Prepare Removable Media for License 29
- License Transfer - Step 3: Transfer License from Source PC to Removable Media 31
- License Transfer - Step 4: Transfer License from Removable Media to Target PC 33
- License Transfer - Step 5: Completion 36
- License Transfer Instructions 28
- Licensing Overview 26

- M -

- Monitor Item 14
- Monitor item '<item id>' for '<server name>' on primary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition 23
- Monitor item '<item id>' for '<server name>' on secondary machine '<machine name>' contains array data which is not supported for the 'specific value' trigger condition 24
- Monitor item test in error for '<server name>' on primary machine '<machine name>' 24
- Monitor item test in error for '<server name>' on secondary machine '<machine name>' 24
- Monitor Item Test Interval 14
- Monitor item test passed for '<server name>' on primary machine '<machine name>' 24

Monitor item test passed for '<server name>' on secondary machine '<machine name>' 24
Monitoring 14
Monitoring Settings 14

- N -

Notifications Settings 17
NT Service 8

- P -

Primary SMTP 17
Promoted active connection for server '<server name>' to primary machine '<machine name>' 25
Promoted active connection for server '<server name>' to secondary machine '<machine name>' 25
Purchase a RedundancyMaster License 26

- R -

Received shutdown notification from server '<server name>' on primary machine '<machine name>' 25
Received shutdown notification from server '<server name>' on secondary machine '<machine name>' 25
Removing Redundancy 20
Requirements 7
Runtime Diagnostics 16, 21

- S -

Secondary SMTP 17
Simple Mail Transport Protocol 17

- T -

Trigger Condition 14
Trigger Data 14

- U -

Unable to retrieve status for server '<server name>' on primary machine '<machine name>'. Server communications has been lost 25

Unable to retrieve status for server '<server name>' on secondary machine '<machine name>'. Server communications has been lost 26
Unalias Option 11
Unaliasing Redundancy 11
Unlock A License 28
User Interface 6