StarTeamMPX Administrator's Guide

StarTeam®

Borland®
Excellence Endures™

Borland Software Corporation
100 Enterprise Way
Scotts Valley, California 95066-3249
www.borland.com

Printed in the U.S.A.

STMADMGD 1E0R0104
0405060708-9 8 7 6 5 4 3 2 1
PDF

Contents

Chapter 1

Preface

This manual contains information for StarTeam administrators who manage MPX-enabled server configurations. It explains the basic operation and architecture of a StarTeamMPX system, and provides instructions for installing and configuring the StarTeamMPX components. For information about performing other administrative tasks on a StarTeam server configuration, see the *StarTeam Administrator's Guide*.

Important The online manuals are distributed in Adobe's Portable Document Format (.pdf), which requires version 4.0 or higher of the free Acrobat Reader to display them. An installation program for Adobe Acrobat Reader is located in the \Docs folder of the Installation CD. The reader is also available from Adobe's web site (www.adobe.com).

The documentation for the StarTeam SDK includes a StarTeamMPX Programmer's Guide. The SDK also includes the EventMonitor.jpx sample project (for Borland JBuilder^{TM}) which has some MPX examples.

Contacting Borland Developer Support

Borland Software Corporation is committed to providing world-class services in the area of consulting and technical support. We have over 15 years of experience in supporting developers and enterprise customers. Our qualified technical support engineers are prepared to handle your support needs on a case-by-case basis or in an ongoing partnership. Borland provides support worldwide, delivering timely, reliable service to ensure every customer's business success.

For more information about Borland's support services, please see our web site at http://support.borland.com.

From the Web site, you can also access many newsgroups where users exchange information, tips, and techniques. See http://info.borland.com/newsgroups/ for the latest list of free product newsgroups. Also available on the Internet is the Borland Developer Network site at http://community.borland.com. This Borland Community provides access to product specific information, articles, code examples, and news.

When contacting support, be prepared to provide complete information about your environment, the version of the product you are using, and a detailed description of the problem.

For support on third-party tools or documentation, contact the vendor of the tool.

Documentation Conventions

The documentation uses the following conventions.

Select File > Exit	Indicates a menu selection followed by a submenu selection. The greater-than symbol (>) separates the commands to be selected from subsequent menus. In this case, select File from the menu bar, then select Exit from the drop-down menu.
`Fixed-Space Font`	Text appearing in Courier font represents information that you need to type and messages from the system.
italics	Information that you replace with the names of your files, child folders, etc.
Bold	Information that you must use exactly as shown (if you use it).
[ ]	Square brackets surround optional syntax.
|	A vertical bar separates mutually exclusive choices.

Note Identifies supplemental information.

Tip Identifies information on alternative procedures or other helpful but nonessential information.

Important Identifies information that is essential to the completion of a task.

Caution Identifies actions that may result in loss of data or procedures that must be followed to ensure that data is *not* lost.

Chapter

2

Overview

StarTeamMPX is a framework for publish/subscribe messaging. It contains both common and application-specific components that together provide advanced messaging capabilities for StarTeam Enterprise Advantage.

In StarTeam Enterprise Advantage, StarTeamMPX improves the performance of the clients and extends the scalability of server configurations. When the term client is used in this guide, it refers to any clients that can take advantage of one or more StarTeamMPX features.

Changes to the server configuration's repository are broadcast in an encrypted format to StarTeam clients and Cache Agents through a publish/subscribe channel. For example, the Event Transmitter broadcasts encrypted messages about changes to objects, such as change requests, and the File Transmitter broadcasts archive files.

Caching modules automatically capture events that a client subscribes to. This reduces the client's need to send refresh requests to the server and improves client response times for the user.

Cache Agents can be installed and configured to cache files in a network-near location to speed up check-out operations. They reduce the distance that the data travels at the time of the client check-out operation. While Cache Agents are StarTeamMPX clients that rely on messages and files transmitted by the Event Transmitter and File Transmitter, they also serve other StarTeamMPX clients as they check out files.

The StarTeamMPX technology offers a number of key benefits, including:

- **Bandwidth multiplication:** Every request by a client that is fulfilled from a cache is a request that the server does not have to fulfill. As a result, a single server configuration can support more clients.

In certain environments, requests for item refresh may constitute up to 50% of client-to-server traffic. This percentage increases with project size and when the “All Descendants” option is used.

For the person who builds software products, check-out operations are a very high percentage of the client-to-server traffic. Using the Bulk CheckOut (bco) command-line utility with a Cache Agent can speed up the time it takes to create a build.

- **Performance acceleration:** Because the event caching modules reside on the same computer as the client and the Cache Agents are on computers that are network-near the supported clients, requests filled from the caches are faster than those requiring a round-trip to the server.
- **Burst control:** As the number of clients increases, the likelihood of burst periods increases, during which time a server configuration can become deluged and less responsive. The caching modules even-out the demand on a server configuration.

The following table lists the MPX features that specific clients can take advantage of.

StarTeam client:	Can take advantage of:
StarTeam Windows client	Event caching
StarTeam Cross-Platform client	Event and file caching
Bulk CheckOut (bco) command-line utility	File caching
IDEs based on Cross-Platform and .NET components (such as the Eclipse integration and the Visual Studio .NET integration)	Event and file caching
Web Edition	Event caching (if the default client profile is for a Message Broker that is accessible from the computer running the IIS)
StarDisk	Event caching (if the default client profile is for a Message Broker that is accessible from the computer running StarDisk)

The StarTeamMPX Framework and Architecture

The key components within the StarTeamMPX framework are messaging engine components, publisher components, and subscriber components. Some components both subscribe and publish.

- Messaging engine components

  Message Broker is the messaging engine and Multicast Service can be used to enhance Message Broker’s capabilities.
- Publisher components

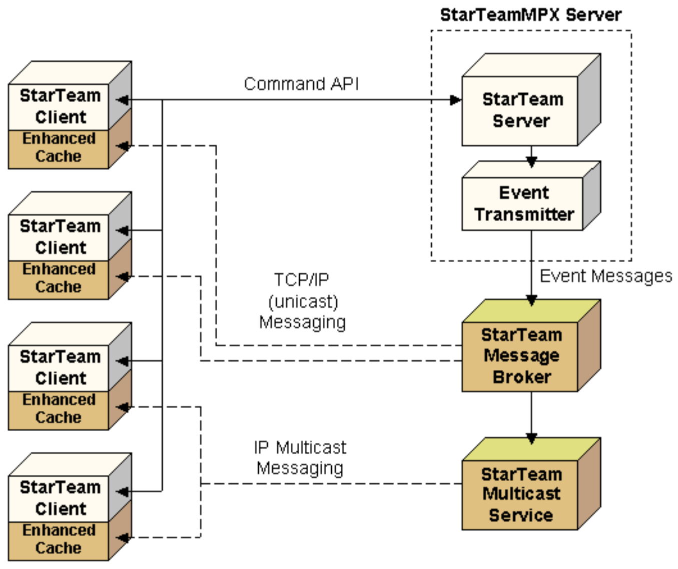
  The publisher components send messages to messaging engines, which forward those messages to the appropriate subscriber components. For example, the Event Transmitter and File Transmitter are publishing components.

- Subscriber components

  The subscriber components receive only those messages to which they have subscribed. Subscriber components can receive messages via TCP/IP from the Message Broker, or via IP multicast from the Multicast Service. For example, the StarTeam clients can receive and cache event messages.

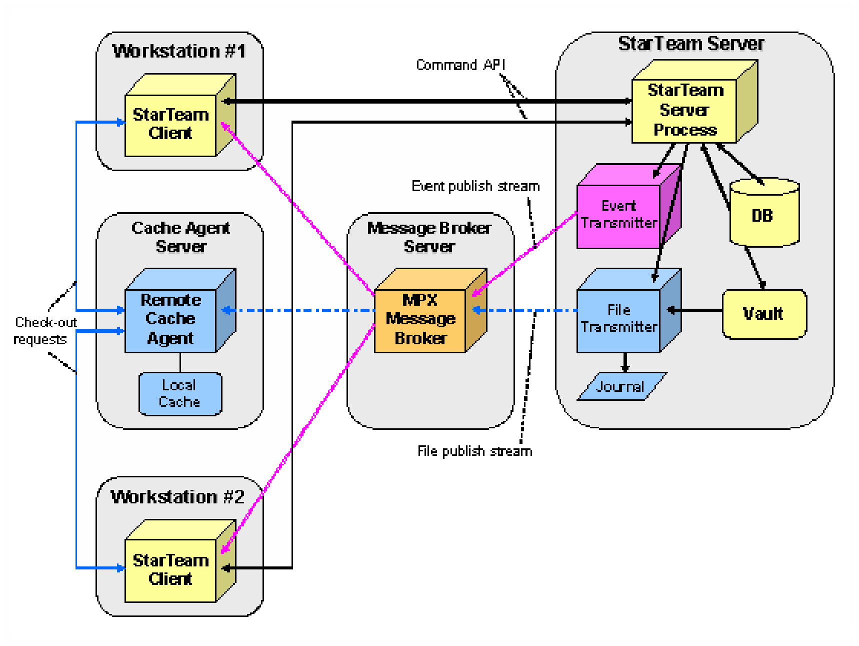
Figure 1 presents an overview of the StarTeamMPX system architecture for event messaging. Depending upon the individual needs of a particular site or facility, there may be several MPX-enabled server configurations, Message Brokers, and Multicast Services serving many clients.

Figure 2.1 StarTeamMPX System Architecture for Event Transmission



Figures 2 and 3 present an overview of the StarTeamMPX system architecture for file transmission. In Figure 2, a single Cache Agent is operating on its own server, servicing check-out requests for two Cache Agent-aware clients. Depending upon the individual needs of a particular site or facility, there may be several MPX-enabled server configurations, Message Brokers, Multicast Services, Root Cache Agents, and Remote Cache Agents serving many clients.

Figure 2.2 StarTeamMPX System Architecture for File Transmission

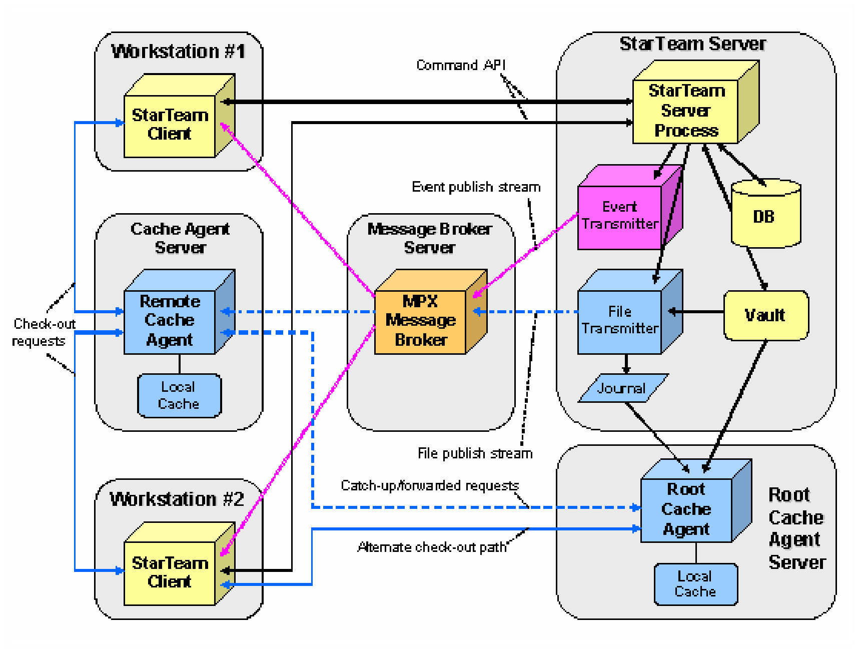


Cache Agents can be organized hierarchically or “tiered” to support distributed teams and improve performance over slow or unreliable network connections. The tiered Cache Agent capability allows Cache Agent caches to be “trickled charged” with content. It also allows Cache Agents to forward request “misses” and to “catch-up” with content that was missed during network or process outages. An example of tiered Cache Agents is depicted in Figure 3.

The tiered caching capability requires the operation of a specially-configured Cache Agent known as the Root Cache Agent. The Root Cache Agent operates directly on a server configuration’s vault. It can execute on the same computer as the server process or on another computer, as long as it has direct access to the vault. The Root Cache Agent also requires access to the Journal file maintained by the File Transmitter. The journal file provides the information needed by the Root Cache Agent to access individual file revisions.

In Figure 3, one Remote Cache Agent is operating on its own computer and is being used by two clients. While most file revision transmissions are initiated by the File Transmitter, the fact that the Remote Cache Agent is “tiered” to the Root Cache Agent allows cache misses and catch-up requests to be forwarded to the Root Cache Agent from the Remote Cache Agent.

Figure 2.3 StarTeamMPX System Architecture for File Transmission



A Cache Agent can operate on the same computer as a client. This is especially useful for check-out intensive clients such as build applications, because the presence of a local Cache Agent provides maximum performance for major check-out operations, especially those done with the Bulk Checkout (bco) command-line utility. See the *StarTeam Administrator's Guide* for more information on this utility which improves check-out speeds both with and without Cache Agent's help.

Component Descriptions

This section provides a short description for each StarTeamMPX component.

StarTeam Server

A StarTeam Server can support a number of StarTeam server configuration, any or all of which can be MPX-enabled. An MPX-enabled server configuration initiates both the Event Transmitter and the File Transmitter. For example, it notifies the Event Transmitter each time a subscribed event occurs, and sends it relevant details about the event.

Event Transmitter

The Event Transmitter subscribes to events of interest to clients. The Event Transmitter formats the event information it receives into XML messages, encrypts them, and publishes them to a Message Broker. Messages are assigned topics so that they can be distributed to clients interested in the accompanying content (project/view, item type, event type, etc.). The Event Transmitter is installed on the same computer as StarTeam Server. It must have the same build number as the server.

File Transmitter

The File Transmitter broadcasts file contents to one or more Remote Cache Agents via a Message Broker. Like the Event Transmitter, the File Transmitter must be installed on the same computer as StarTeam Server and have the same build number as the server.

Message Broker

The Message Broker is a publish/subscribe messaging engine that broadcasts messages to subscriber components on a topic basis. It is a stand-alone process that can run on a separate computer to offload network processing overhead in high-volume environments. The Message Broker broadcasts messages to each of its recipients using TCP/IP (unicast) messaging.

The Message Broker receives encrypted XML messages from the Event Transmitter or encrypted files from the File Transmitter, and forwards them to the appropriate clients. Information is sent from a Message Broker directly to clients that have subscribed to that Message Broker via a unicast (TCP/IP) connection profile. For clients that have subscribed using a multicast connection profile, the Message Broker forwards the information to the appropriate Multicast Service, which then relays it to the client.

Multicast Service

The Multicast Service is an optional process that gives the Message Broker additional broadcast capabilities. However, only the StarTeam Windows client and Remote Cache Agents can use Multicast Service at this time.

The Multicast Service delivers the same information as the Message Broker, but it uses IP multicast messaging instead of TCP/IP unicast communication. This means that the information is sent only once, regardless of the number of clients that have subscribed. As a result, network traffic is kept relatively constant even as the number of clients grows, reducing network congestion and improving scalability in large user communities.

Cache Agents

Cache Agents add persistent file caching. Each MPX-enabled server configuration can have one Root Cache Agent. One or more Remote Cache Agents can be distributed throughout the enterprise.

A Root Cache Agent operates directly on the server configuration’s vault. They store files in their own caches.

A Root Cache Agent handles requests forwarded from Remote Cache Agents for missing files and provides “file-catch-up” assistance for Remote Caches after network or process outages.

Cache Agent-aware StarTeam clients can fetch files from any available Cache Agent.

By using “network-near” Cache Agents, clients can improve file check-out performance and reduce their demands on the StarTeam Server. This frees server resources for additional tasks and users.

StarTeam Clients—Event Transmission

When a client connects to an MPX-enabled server configuration, a connection profile determines which Message Broker or Multicast Service the client uses to receive event messages.

Clients benefit from event messages through an enhanced internal cache. This cache subscribes to specific caching message topics, keeping its cached objects up-to-date with respect to the projects and views that the client uses. As a result, several types of object fetching (most notably item refresh) no longer require round-trips to the server.

The client cache has no persistence mechanism, and cache contents are not shared among multiple client processes. However, while the client is running, the cache remains updated with changes made to the client’s open views, thereby speeding-up its operation.

A client session provides the keys required for performing MPX functions and ensures that each access is verified for applicable security context.

StarTeam Clients—File Transmission

A logged-on user can use the StarTeam Cross-Platform client, an IDE based on StarTeam Cross-Platform or .NET components, or Bulk CheckOut (bco) command-line utility to retrieve files from a Cache Agent. The Cache Agent’s file caching is independent of client processes because the Cache Agent operates as a separate process. Consequently, a client can fetch files that were broadcast while it was not operational.

StarTeamMPX Security

The StarTeamMPX security features include:

- Data encryption
- User authentication and access rights

Each of these features is described in greater detail in the following sections.

Data Encryption

A client receives data from an MPX-enabled server configuration over one of two paths:

- directly from the server
- indirectly from transmitters and Cache Agents via a Message Broker or Multicast Service.

The encryption level for data sent directly from the server is specified on the Server Properties dialog for each individual server configuration; it is possible to have no encryption set for this data path. See the *StarTeam Administrator's Guide* for more information on setting encryption levels for a server configuration.

All data sent by the transmitters or Cache Agents is encrypted. Each Event Transmitter has its own encryption key. When the server configuration starts an Event Transmitter, it creates a unique encryption key for that instance of the Event Transmitter. When a client opens a project, the server configuration sends the client the Event Transmitter encryption key directly. The client will have one encryption key for each MPX-enabled server configuration it is accessing.

All files sent by the File Transmitter is encrypted. The files are stored in encrypted format by Cache Agents and decrypted only “at the last moment” within the client process.

User Authentication and Access Rights

As users log on to a server configuration, they are identified individually by their user names and as members of the groups to which they belong. This information is stored as an access token for each user. Based on a user's access rights, the server configuration determines which objects a user can see and which operations that user can perform on those objects.

The caching module in the client enforces the same user access rights set. When a client receives a message from a Message Broker or Multicast Service, it verifies whether the user is authorized to view the data in the message. If the user has the necessary access rights, the message is stored in the client cache. Otherwise, that object will not be cached.

In a StarTeam client, you can control detailed access rights for a file: the ability to see the file, see history, check-out, check-in, etc. For example, you can give someone the "see item and its properties" right but deny the "check-out" right.

However, with the Cache Agent, granting someone the "see item and its properties" right implicitly virtually grants them a "Cache Agent check-out" right. This is because the client can get a file's MD5, which is all it is needed to request a Cache Agent check-out. For environments in which this difference in security "interpretation" matters, you should not deploy Cache Agent or deny the "see item and its properties" right for users who should not check-out the corresponding files.

Message Broker and Multicast Service Profiles

MPX profiles that specify an appropriate Message Broker or Multicast Service are specified in the Event Transmitter's configuration file and used by the server configuration, clients, and Cache Agents. It is best to use a network-near broker or service.

The StarTeam administrator must create MPX profiles to fit the needs of each general location in distributed groups. For example, all users using the same Remote Cache Agent may be network-near the same Message Broker and, therefore, use that Message Broker's profile.

For more information, see "Understanding Connection Profiles" on page 33.

Dependencies

The startup order of MPX components is important:

- MPX-enabled server configurations and Cache Agents are dependent on Message Brokers and Multicast Services. These should be running before the applications that are dependent on them.

  For example, if a Root Cache Agent is installed as an auto-start service and uses a Message Broker on the same computer, the Root Cache Agent may start more quickly than the Message Broker. In this case, the Root Cache Agent fails. The fix is to make the Root Cache Agent service depend on the Message Broker service.
- The MPX-enabled StarTeam server configuration must be run at least once before the Cache Agents.

  The Cache Agents are dependent on the existence of a CacheJournal.dat file for each server configuration whose files are being transmitted. You must run each MPX-enabled server configuration once so that its File Transmitter can create the initial journal.
- The MPX-enabled StarTeam server configuration must be running before the Cache Agents or clients can access it.
- If you are using the Native-I Cache Auditor (because the server configuration was created prior to StarTeam Server 2005 and still contain Native-I archive files), you must run the Auditor before you run the Root Cache Agent so that the Native-I data files will already have been created.
- The Cache Agents must be running before the clients can retrieve files from them.

Chapter

3

Managing Message Engines

Message Broker is the messaging engine and Multicast Service can be used to enhance Message Broker's capabilities. All clients and Cache Agents must use a profile for either a Message Broker or Multicast Service. Only the StarTeam Windows client and Remote Cache Agents can use a Multicast Service at this time.

Depending on the needs of your environment, you may decide to install several Message Brokers on multiple systems. The section named "Understanding Clouds" will help you decide.

This chapter also provides sections on configuring Message Brokers and Multicast Services.

Installation instructions can be found in Chapter 8, "Installing MPX Components on Windows" and Chapter 9, "Installing MPX Components on Solaris."

Planning for Message Brokers

Planning considerations for the Message Broker include:

- At least one Message Broker is required for StarTeamMPX operation.
- When the total number of clients is relatively low (less than 100 users) and the overall activity is low to moderate (up to 1,000 updates per hour), a single Message Broker can be installed on the same computer as the StarTeam Server.

- A single Message Broker can fulfill the messaging requirements for multiple configurations (deployed on the same computer or on multiple computers) if the total update volume is low to moderate.
- To use the same Message Broker for multiple configurations, the connection profiles in each transmitter XML file should point to the same Message Broker (see Chapter 4, "Managing the Transmitters").
- Clients should use a network-near Message Broker if possible.

  The StarTeam administrator must create MPX profiles to fit the needs of each general location in a distributed group and set one of them as the default client profile. Some clients can override the default profile and should do that to select a network-near Message Broker.
- Because you can use no more than 10 Message Brokers, your company may need to use the same Message Broker for a large area; for example, California rather than Los Angeles County.
- As described in "Configuring a Multicast Service" on page 23, WANs, firewalls, and other factors may require you to install multiple Message Brokers, connected together as a cloud.

Planning for Multicast Services

Key planning considerations for the Multicast Service include:

- The Multicast Service is optional; it can be installed to reduce network traffic when many users are often connected to the same configuration simultaneously.
- Only StarTeam Windows clients and Remote Cache Agents can use Multicast Services.
- The Multicast Service and Message Broker must be installed on the same computer.
- If the overall update traffic is low to moderate, the Message Broker and Multicast Service can operate on the same computer as the StarTeam Server.
- In environments where update traffic is moderate to high (greater than 1,000 updates per hour), the Message Broker and Multicast Service can be installed together, but on a separate computer from the StarTeam Server. This offloads CPU processing and network traffic for improved distributed computing.
- If multiple Message Brokers are deployed, typically only one Multicast Service is needed in each local network community.
- Multicast Services should generally be used on high-speed networks, because IP multicast does not have TCP/IP's automatic flow control.

- In some situations, using a Multicast Service is impossible or not recommended. For example:
    - StarTeam clients connecting over dial-up or other slow connection types should use unicast messaging by connecting to a Message Broker instead of a Multicast Service
    - Many firewalls and Internet Service Providers (ISPs) do not forward IP multicast packets. Consequently, users accessing a StarTeamMPX through a firewall or over an ISP should use also use a unicast profile.

Understanding Clouds

When MPX-enabled server configurations publish messages to different Message Brokers, the Message Brokers may need to be configured to forward messages to one another. Such a configuration is called a “Message Broker cloud”.

The cloud allows a client to use StarTeamMPX features with more server configurations than would otherwise be possible. A client is restricted to receiving messages from only one Message Broker or Multicast Service per session. So message need to get to that Message Broker or Multicast Service from every server configuration the client wants to access.

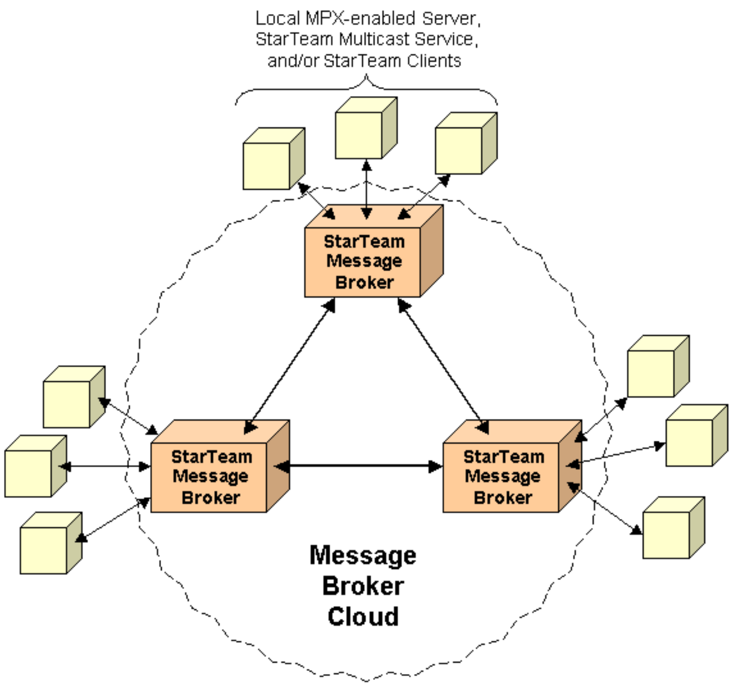
For example, when a client opens a project on one MPX-enabled server configuration, it establishes a connection to a specific Message Broker or Multicast Service. Later, the client attempts to opens a project on an MPX-enabled server configuration that uses a different Message Broker. In the process, the client verifies the message path to the new configuration. If messages can travel from the new configuration’s Event Transmitter to the client’s current Message Broker or Multicast Service (because they are the same or in the same cloud), the client uses StarTeamMPX to access the second server configuration. If the message path cannot be verified, the client displays an error message, and accesses the second configuration as if it were not MPX-enabled. The client performs all functions properly, but it does not receive the performance and instant notification benefits provided by StarTeamMPX.

When a Message Broker receives a forwarded message from another Message Broker, it:

- delivers the messages to the clients that are connected directly to it
- forwards messages to a connected Multicast Service (if any) for subsequent delivery to clients using multicast connections
- forwards the messages to other Message Brokers

Figure 1 shows an example of a Message Broker cloud. Some components in the StarTeamMPX architecture are omitted for simplicity.

Figure 3.1 Example Message Broker Cloud



How Communication Is Established Between Message Brokers

A Message Broker can establish communication with another Message Broker by:

- initiating the communication with another Message Broker during its startup procedure
- being contacted by another Message Broker when that Message Broker starts

Each Message Broker has its own STMessageBroker64.ini configuration file located in the same folder as the executable STMessageBroker64.exe. The optional `server_names` parameter in the STMessageBroker64.ini file identifies other Message Brokers in the Message Broker cloud. When a Message Broker first starts, it reads the STMessageBroker64.ini file and attempts to connect with each of the Message Brokers listed in `server_names`. If any of these Message Brokers is not running or cannot be reached when contact is attempted, communication is not established with that Message Broker. However, each Message Broker will retry the connection periodically.

Note Message Brokers broadcast their presence periodically using UDP broadcast messages. If there are multiple Messages Brokers on your local network, they may be able to establish communication and form a cloud on their own. However, if there are Message Brokers residing in different network segments that need to be formed into a cloud, you must explicitly connect them using STMessageBroker64.ini file parameters.

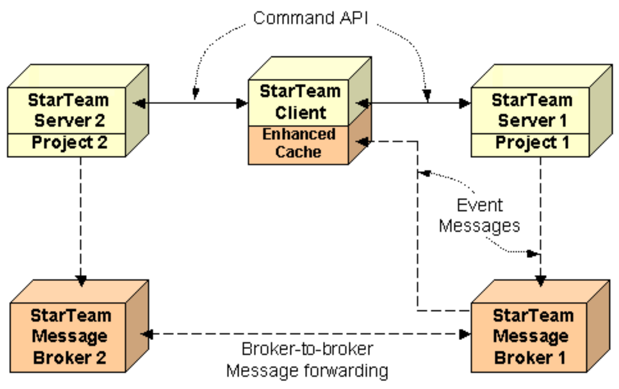
Message Routing in Message Broker Clouds

When a client first connects to an MPX-enabled server configuration, it uses a connection profile (described in "Understanding Connection Profiles" on page 33) to determine which Message Broker or Multicast Service to use. Because each client can use only one Message Broker or Multicast Service, the client continues to use the same messaging service for all projects that it opens. Even when a client opens projects from different MPX-enabled server configurations, it receives messages from the first messaging service that it chooses. Each message broadcast by an MPX-enabled server configuration is automatically forwarded to every Message Broker and Multicast Service that has a connected client interested in that message.

For example, in Figure 2, the client has two open projects that are located on two different MPX-enabled server configurations. When a user opens Project 1 on StarTeam Server 1, the StarTeam client chooses a connection profile for Message Broker 1.

When the user opens Project 2 on StarTeam Server 2, the client sends a subscribe message to Message Broker 1 registering its interest in messages regarding Project 2. Thereafter, StarTeam Server 2 sends all messages pertaining to Project 2 to Message Broker 2, which forwards them to Message Broker 1, and finally to the client. Even if the user closes Project 1 and, therefore, its connection to StarTeam Server 1, it continues to receive messages via Message Broker 1 during the client session.

Figure 3.2 Client with projects open on two MPX-enabled Servers


Command API
StarTeam Server 2 Project 2
StarTeam Client
Enhanced Cache
StarTeam Server 1 Project 1
Event Messages
StarTeam Message Broker 2
StarTeam Message Broker 1
Broker-to-broker Message forwarding

Routing in Unconnected Message Broker Clouds

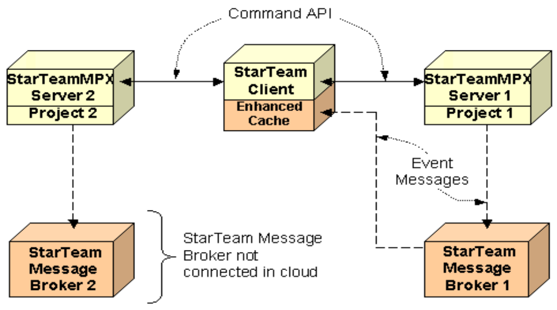
In a properly configured Message Broker cloud, messages from all server configurations to which the client is connected are routed to the appropriate Message Broker or Multicast Service for that client. However, in unconnected clouds, not all Message Brokers are aware of other Message Brokers and cannot forward messages appropriately.

You might have unconnected clouds because:

- There is a limit of 10 Message Brokers per cloud.
- The clouds are not properly configured.
- The clouds are intentionally configured this way, perhaps for increased security.

Suppose a client accesses two StarTeam server configurations. Suppose that each server configuration uses a different Message Broker, and that these two Message Brokers are in unconnected clouds (see Figure 3). In this case, the client receives messages for only the server configuration in the first cloud. The client treats the server configuration in the second cloud as if it did not support StarTeamMPX. The client performs all functions properly for such "disjoint" server configurations, but it does not receive the performance and instant notification benefits provided by StarTeamMPX.

Figure 3.3 Message routing in an unconnected Message Broker cloud


Command API
StarTeamMPX Server 2 Project 2
StarTeam Client
Enhanced Cache
StarTeamMPX Server 1 Project 1
Event Messages
StarTeam Message Broker 2
StarTeam Message Broker not connected in cloud
StarTeam Message Broker 1

Volume Considerations

In many situations, a single Message Broker is sufficient to handle the message processing of one or more MPX-enabled server configurations, depending on the average update volume of your environment. Similarly, a single Multicast Service is often sufficient even when supporting multiple server configurations.

In environments with a low to moderate number of updates (up to a few thousand updates per hour), the Message Broker and Multicast Service can be installed on the same computer as the StarTeam Server. In this case, the Message Broker and Multicast Service can handle the message broadcasting for MPX-enabled server configurations on the same computer, as well as for configurations operating on other computers located on the same local area network.

In moderate to high volume update environments (several thousands of updates per hour or more), you should consider operating the Message Broker and Multicast Service together, but on a separate computer from the StarTeam Server. This will offload CPU and network demand from the computer hosting the StarTeam server configuration, providing an opportunity for additional distributed processing.

In some cases, you should consider installing multiple Message Brokers, each on their own computer, and connecting them together into a cloud of up to ten Message Brokers. The common scenarios in which multiple Message Brokers are advantageous are listed below.

- Large number of simultaneous users

  In the StarTeamMPX architecture, each client using a unicast connection profile creates a TCP/IP connection to a Message Broker. The Event Transmitter and File Transmitter create additional TCP/IP connections. The Multicast Service also requires TCP/IP connections to a Message Broker. A single Message Broker can support between 500 and 1,000 simultaneous connections, depending upon message load. The Message Broker has a limit of 1,000 simultaneous connections and will refuse new connection requests beyond that number. If you expect the number of simultaneous connections to exceed this range and/or if you expect very high update rates, you should consider installing multiple Message Brokers.

- Fault-Tolerant Operation

  Multiple Message Brokers can be installed on separate computers and operate in parallel as “peers”. The transmitters can subsequently be configured to use one Message Broker but, if it is not available, to use a second Message Broker (or a third, and so forth). Some clients can automatically select an alternate Message Broker capability. The configurations settings needed for selecting these options are discussed in Chapter 4, “Managing the Transmitters.”

- Wide Area Networks (WANs)

  If the network topology on which your application community operates contains multiple subnets or if the subnets are geographically distributed, you may want to install a Message Broker to serve each “local” user community and connect the Message Brokers into a cloud. This reduces message traffic over the greater network, because messages are transferred between Message Brokers only once, and then replicated locally to directly-attached clients.

- External Users

  If you have users who need to access a StarTeam server configuration over the public Internet, you may wish to operate a Message Broker that is inside the “DMZ” or completely outside of your corporate firewall. Such a Message Broker would then be connected to other internal Message Brokers, providing external users with StarTeamMPX functionality. Issues for operating Message Brokers external to a firewall are discussed in "Configuring a Multicast Service" on page 23.

Using Message Brokers with a Firewall

In some cases, you may have users who need to access an MPX-enabled server configuration over the public Internet without using a Virtual Private Network (VPN). A common technique for providing access to a StarTeamMPX is to install the StarTeam Server on a computer in the "DMZ" area of the corporate firewall (while hosting persistent data such as the database on a separate system behind the firewall).

Typically, that computer has two IP addresses and host names: an internal address/host name used by inside users, and an external address/host name used by outside users. In this scenario, a Message Broker can be operated on the same computer as StarTeam Server and be accessed by both internal and external users.

Alternatively, you could operate a Message Broker on a separate computer, also within the "DMZ", and therefore also accessible to both internal and external users. However, in some cases (such as when corporate policy seeks to minimize the number of applications operating within the "DMZ"), you may wish to operate one or more internal Message Brokers behind the firewall and perhaps one Message Broker outside of the firewall. When the Message Brokers are formed into a cloud, both internal and external users receive the appropriate messages for the server configurations to which they are connected.

To connect an external Message Broker into a Message Broker cloud, it is best to modify the STMessageBroker64.ini file of one or more internal Message Brokers to point out to the external Message Broker. That is, modify the internal Message Broker's `server_names` parameter to include the address of the external Message Broker. This technique is preferred because the firewall may not allow outside-in connections, thereby preventing the cloud from being formed in the opposite direction.

From a security perspective, a Message Broker can operate safely within the "DMZ" or completely external to a firewall for two reasons:

- The Message Broker is a communications server only and stores no persistent data that could become the target of a security attack.
- The cache messages are encrypted with a key dynamically generated by each Event Transmitter session. Only clients who are successfully authenticated via the logon sequence receive the key required to decipher the cache messages. Consequently, packet snooping and other eavesdropping techniques aimed at Message Broker traffic will not produce any meaningful data.

Configuring a Message Broker

Each Message Broker has a STMessageBroker64.ini file that contains startup parameters. This file must be located in the same folder as the Message Broker executable STMessageBroker64.exe.

This folder is typically "C:\Program Files\Borland\Message Broker 6.4" on Windows platforms.

On Solaris platforms, it is "/opt/MessageBroker64". If the

STMessageBroker64.ini file is missing, the Message Broker uses predefined default values for all of the parameters.

Parameters that may need to be changed based on your needs are described in the next sections.

Configuring a Message Broker Cloud

If you plan to run more than one Message Broker, they probably should be configured into a cloud.

To create a Message Broker cloud:

1 If you have not already done so, install the Message Brokers. (See "Installing the Message Broker and Multicast Service" on page 66 for instructions.)

2 For each Message Broker you want to include in the cloud, open the associated STMessageBroker64.ini file in a text editor.

3 Add or edit the following line to list all the Message Brokers you want to include in the cloud:

```
setopt server_names tcp:servername:endpoint
```

where

- `servername` is the TCP/IP address or computer name where the remote Message Broker is installed
- `endpoint` is the port number on which the remote Message Broker is listening

Separate multiple addresses by a comma (,). For example, the following line creates a cloud consisting of four Message Brokers: the current Message Broker and the three Message Brokers listed below.

```
setopt server_names tcp:ProdServer1:5110,
tcp:ProdServer2:5120,tcp:ProdServer3:5130
```

Tip If a Message Broker operating outside of a firewall must participate with Message Brokers behind a firewall, an inside Message Broker should be directed to establish contact with the outside Message Broker, because the firewall may prohibit the establishment of the connection in the reverse direction.

4 Save and close the file.

The next time you start the Message Broker, it will establish connections with the Message Brokers listed in the `server_names` parameter.

Changing the Endpoint of a Message Broker

The endpoint (port number) of a Message Broker is specified when you install it. If you later want to configure a Message Broker to use a different endpoint, you must edit the STMessageBroker64.ini file.

To change the endpoint number of a Message Broker:

1. Make the change to the STMessageBroker64.ini file for that Message Broker:
   - a. Open the STMessageBroker64.ini file in a text editor.
   - b. Add or edit the following line:

     ```
     setopt conn_names tcp:servername:endpoint
     ```

     where

     - `servername` is the TCP/IP address or name of the computer where the Message Broker runs; you can use the keyword "_node" to designate the local host independent of its current IP address or host name
     - `endpoint` is the new endpoint you want the Message Broker to use; the default is 5101

     For example:

     ```
     setopt conn_names tcp:_node:5523
     ```
   - c. Save and close the file.

   Your changes will take effect the next time you start the Message Broker.
2. If this Message Broker is part of a Message Broker cloud, be sure to edit the STMessageBroker64.ini files associated with the other Message Brokers in the cloud.
3. For each server configuration that has one or more profiles that use this Message Broker, correct each profile using one of the following methods:
   - Use the Server Administration utility to edit the affected profiles:
     - a. From Server Administration, select the server configuration.
     - b. Select Tools > Administration > Configure Server.
     - c. From the Event Handlers tab, select StarTeamMPX Transmitter as the event handler.
     - d. From the Profile list box, select an affected profile.
     - e. Click Modify to open the Event Handlers Profile Properties dialog.
     - f. In the Profile Properties list box, double-click server_names.
     - g. In the Event Handler Property dialog, edit the appropriate endpoints in the Property Value text box.

     Your changes take effect the next time you start the server configuration.

- Edit the associated MPXEventTransmitter.xml file directly:
  - **a** Open the MPXEventTransmitter.xml file in a text editor.
  - **b** Edit the `server_names` parameter in each affected multicast profile to reflect the new endpoint.
  - **c** Save and close the file.

  Your changes take effect the next time you start the server configuration.

4 For each Cache Agent that accesses this Message Broker, edit the MessageBroker group to change the appropriate `server_names` parameter.

- **a** Open the Cache Agent's XML file in a text editor.
- **b** Find the `server_names` parameter for the correct Message Broker. For example, it might look like the following:

```
<server_names>tcp:12.34.56.78:5101</server_names>
```

- **c** Change the endpoint/port number portion of that line.

Configuring a Multicast Service

When a Multicast Service starts, it reads a file called STMulticastService64.ini to obtain its initialization parameters. This file must be located in the same directory as the Multicast Service executable STMulticastService64.exe. Both files are normally stored in the same location as the Message Broker executable (because of a shared registry key).

- On Windows platforms, this folder is typically "C:\Program Files\Borland\Message Broker 6.4\".
- On Solaris, this folder is typically "/opt/MessageBroker64".

Parameters that may need to be changed based on your needs are described in the next sections.

Changing the Endpoint of a Multicast Service

To use a Multicast Service for receiving messages, application clients first connect to the Multicast Service with a TCP/IP connection. The TCP/IP connection enables the clients to exchange initialization information with the Multicast Service, including the multicast address that will be used for message broadcasts.

The endpoint (port number) of a Multicast Service is specified when you install it. If you later want to configure a Multicast Service to use a different endpoint, you must edit the STMulticastService64.ini file.

To change the endpoint number of a Multicast Service:

1 Make the change to the STMulticastService64.ini file for that Multicast Service.

2 For each server configuration that has one or more profiles that use this Multicast Service, do one of the following:
   - Use the Server Administration utility to edit the affected profiles
   - Edit the associated MPXEventTransmitter.xml file directly.
3 For each Cache Agent that accesses this Multicast Server, edit the MessageBroker group to change the appropriate `server_names` parameter.

For more details about making these changes, see the similar instructions in "Changing the Endpoint of a Message Broker" on page 22.

Changing the Message Broker used by a Multicast Service

Each Multicast Service must communicate with a Message Broker.
The Multicast Service installer asks for a Message Broker address at installation time, and writes this address to the STMulticastService64.ini file. You can change this address, for example, to instruct the Multicast Service to connect to a Message Broker with a different endpoint.

To change the Message Broker address used by the Multicast Service:

1 Open the STMulticastService64.ini file in a text editor.
2 Add or edit the following line:

```
setopt server_names tcp:servername:endpoint
```

   where
   - `servername` is the TCP/IP address or name of the computer where the Message Broker which the Multicast Service is to use runs; you can use the keyword "_node" to designate the local host independent of its current IP address or host name
   - `endpoint` is the port number with which the Message Broker accepts connections; the default is 5101

   For example:

```
setopt server_names tcp:_node:5101
```

3 Save and close the file.

   Your changes will take effect the next time you start the Multicast Service.

Changing the Multicast Packet Time-to-live

Like all IP packets, multicast packets broadcast by the Multicast Service include a time-to-live or TTL value. The TTL value controls how far a packet will travel within a network. Essentially, the TTL value is the number of routers a packet will travel. Each router decrements the TTL value in a packet and when the value reaches zero, it is not forwarded any further. With multicast packets, when the TTL value reaches zero, the router simply drops the packet.

By default, the Multicast Service uses a TTL value of 2, allowing multicast packets to traverse up to two routers. However, if clients using a Multicast

Service are connected to the network such that more than two routers must be traversed by multicast packets, you will need to increase the TTL value.

Unlike most other Multicast Service parameters, the TTL value is not specified in the STMulticastService64.ini file. Instead, it is defined in a file called mcast.cm, which resides in the “standard” subfolder of the Multicast Service executable.

- On a Windows platform, the typical path name of this file is “C:\Program Files\Borland\Message Broker 6.4\Standard\mcast.cm”.
- On Solaris, the typical path name of this file is “/opt/MessageBroker64/standard/mcast.cm”.

To change the multicast TTL value:

1. Open the mcast.cm file in a text editor.
2. Edit the following line:

   ```
   setopt pgm_source_group_ttl 2
   ```

   Change the value from “2” to the maximum number of routers that multicast packets may need to traverse on your network.
3. Save and close the file.

Changing the Multicast Maximum Transmission Rate

TCP/IP or unicast communication uses a one-to-one connection model. This means that for each connection, there is one process at each end of the connection. Each process must acknowledge (ack) every packet that it receives. Packet acknowledgement is important for both message reliability and flow control. Flow control forces each process to automatically slow down, when necessary, to match the processing speed of the other process.

In contrast, multicast uses a one-to-many or “broadcast” style of communication. Each receiving process does not acknowledge packets to the sender. Instead, message reliability is guaranteed through other techniques such as message sequencing and “negative acknowledgement” (nak) messages. The lack of packet-level acknowledgements means that multicast does not have automatic flow control. Consequently, the transmission rate of multicast messaging must be controlled using other means.

For the Multicast Service, the maximum multicast transmission rate is controlled by a configuration parameter in the mcast.cm file. This file resides in the “standard” subfolder of the Multicast Service executable.

- On a Windows platform, the typical path name of this file is “C:\Program Files\Borland\Message Broker 6.4\Standard\mcast.cm”.
- On Solaris, the typical path name of this file is “/opt/MessageBroker64/standard/mcast.cm”.

By default, the maximum transmission rate is set to 4,000,000 or four megabits per second. This value is sufficient for most environments, but in high-volume environments, you may want to increase this value.

To change the multicast maximum transmission rate:

1 Open the mcast.cm file in a text editor.

2 Change the value of the “pgm_source_max_trans_rate” parameter. For example, to set a maximum transmission rate of 10 megabits per second, edit the line containing this parameter to read:

```
setopt pgm_source_max_trans_rate 10000000
```

3 Save and close the file.

   The new transmission rate will take effect the next time you start the Multicast Service.

Chapter

4

Managing the Transmitters

When you install the transmitters, the Event Transmitter and File Transmitter files are placed in the StarTeam Server's installation folder. This chapter explains how to configure them.

Configuration-specific Transmitter XML Files

Each server configuration uses its own event and file transmitter XML files. This means that each configuration has its own set of profiles that define connection alternatives for accessing the Message Broker cloud in which the configuration operates.

During the StarTeamMPX installation, template files names MPXEventTransmitterTemplate.xml and FileTransmitterTemplate.xml files are installed on the same computer as StarTeam Server. Their default locations depend on the operating system type, as shown in the following table.

Operating System	Default location for MPXEventTransmitterTemplate.xml
Windows NT, 2000, or 2003 Server	C:\Program Files\Borland\StarTeam Server 2005 R2\EventServices\
Sun Solaris (SunOS 5.9 and 5.10)	/opt/starteamserver2005r2/EventServices

The EventServices folder is relative to the installation path of the StarTeam Server. The default installation path is shown, but another path may be used.

Initially, the MPXEventTransmitterTemplate.xml file contains two sample profiles: one unicast profile and one multicast profile. The File TransmitterTemplate.xml file does not use profiles.

The configuration-specific XML files are named MPXEventTransmitter.xml and FileTransmitter.xml. They are usually created for you automatically based on the template files in the EventServices folder. For more information about creating these XML files, see "Generating Transmitter XML Files" on page 65.

The configuration-specific XML files are stored in subfolders of the EventServices folder. These subfolders have the same names as the server configurations.

Because the template XML files are used to create the configuration-specific XML files for new StarTeam server configurations, a newly edited template file becomes the configuration-specific XML file for future server configurations. The format of the template and configuration-specific XML files is the same.

Any template file or configuration-specific XML file can be edited manually in a text editor. However, to edit connection profiles in a configuration-specific MPXEventTransmitter.xml file, it is best to use the Server Administration utility. See the *StarTeam Administrator's Guide* for more details about configuring Event Handlers. The section entitled "Changing the Endpoint of a Message Broker" on page 22 also offers some information about changing profile data in MPXEventTransmitter.xml file.

Note You only configure XML files for the File Transmitter if you are using Cache Agent.

Understanding the Event Transmitter

The next few sections cover the Event Transmitter and its XML file.

Startup

When a StarTeam server configuration starts, it determines if it is an MPX-enabled configuration by locating a configuration-specific Event Transmitter XML file. The server attempts to load this file, identify the server default profile, and load the Event Transmitter with that profile. All startup messages and errors for the server configuration and the Event Transmitter are recorded in the configuration's server log file.

On most systems, the server's log file is located in the root folder of the server configuration's repository. See "Server Log Entries" on page 79 for examples of typical startup messages.

If the configuration-specific XML file loads successfully, the Event Transmitter establishes a connection with the Message Broker identified in the server default profile.

If the Event Transmitter fails to connect with a Message Broker, the StarTeam Server terminates the Event Transmitter and starts the server configuration in non-MPX mode.

When a client opens a project on a server configuration, it determines whether or not the server configuration is MPX-enabled. If the configuration is MPX-enabled, the server sends the client the encryption key needed to decrypt messages from the Event Transmitter. It also sends the client the connection profiles defined in the configuration-specific Event Transmitter XML file.

Some clients have settings that allow the server's default connection profile to be overridden. It the user can select a specific profile and has done so, the client uses the profile set by the user. Otherwise, the client uses the profile marked as the client default profile. The first profile selected by the client becomes the profile used for all projects opened during that session, even if the projects are managed by different server configurations with different default profiles.

If the client is unable to connect to the designated Message Broker or Multicast Service, it displays an error dialog and proceeds to open the project in non-MPX mode.

Note Because a client can communicate with only one Message Broker or Multicast Service per session, systems running multiple Message Brokers should be configured to forward messages to one another using Message Broker clouds.

The XML File's General Format

The general format of the Event Transmitter XML file is as follows:

```
<?xml version="1.0"?>
<StExternHandlerModule>
   ...module identification...
   <HandlerSet>
      <Handler>
         ...handler general information...
         <ServerDefault>...</ServerDefault>
         <ClientDefault>...</ClientDefault>
         <ProfileSet>
            <Profile>
               ...profile #1 information...
            </Profile>
            <Profile>
               ...profile #2 information...
            </Profile>
            ...more profiles, if any...
         </ProfileSet>
      </Handler>
   </HandlerSet>
</StExternHandlerModule>
```

Note that in this example, the ellipses (…) denote information where some details have been left out. Some XML parameters in the Event Transmitter XML file are used to identify the Event Transmitter library to the server configuration; the presence and order of these parameters should not be modified.

Table 4.1 lists the parameters that can be modified to tailor the connection profiles within the XML file. The parameters are listed in the order in which they appear in the XML file.

Unless otherwise stated, all integer parameters have very large positive ranges. For example, all time values have a range of 1 to 2,147,483,647. In most cases, you should consider the default value the minimum setting. Use a practical upper limit based on common sense.

Table 4.1 Event Transmitter Parameters

Parameter	Description
ServerDefault	Defines the server default profile. Its value must match the name of a Profile within the Profile Set group in this XML file. The server default profile is used by the Event Transmitter to establish a Message Broker connection. For configuration-specific XML files, you can use the Server Administration utility to set a profile as the server default.
ClientDefault	Defines the client default profile. Its value must match the name of a Profile within the Profile Set group in this XML file. The client default profile is used by clients as the default definition to establish a Message Broker or Multicast Service connection.For configuration-specific XML files, you can use the Server Administration utility to set a profile as the client default.
Profile	Can be repeated. It is both a group for parameters and a member of the Profile Set group. Each profile has a name, a brief textual description, and a set of properties. The initial profiles in the Event Transmitter XML file are specified during the transmitter's installation. In most cases, the default values are sufficient and do not need to be changed. However, if you need to add, move, or remove Message Brokers or Multicast Services, or if you want to customize any connection profile settings, you can edit the XML file. **`ProfileName`** Defines the name of the profile. The name should provide a useful tip as to the purpose of the profile (such as "West-coast on-site"). If the profile is a server or client default profile, whose value should match the value of the corresponding `ServerDefault` or `ClientDefault` parameter. **`ProfileDescription`** Provides a short textual message describing the profile. It should contain a value that helps users and administrators understand when to use the parameter (such as "A multicast profile for users residing in the west coast office LAN"). Along with the profile name, the profile description appears in both the client and Server Administration connection profile dialogs.

Table 4.1 Event Transmitter Parameters

Parameter	Description
	`Properties` This is a group inside the Profile group. Its inner parameters define the connection details of the profile. The parameters within this group determine whether the profile is a unicast or a multicast profile, and they specify the connection details of the profile. The most common options that can be used within the `Properties` group and their usage are described below. In special cases, some other options which are not described below can also be used. However, these options should only be used under the advice of Borland developer support services (www.borland.com/devsupport/). The parameters in the Properties group are listed in table 4.2.

Table 4.2 Parameters for the Properties Group Inside the Profile Group

Parameter	Description

server_names	Designates the protocol, address, and port number of one or more Message Brokers or Multicast Services. The general syntax for this parameter is:
	protocol:*servername*:*endpoint*
	where
	`protocol` is either tcp or pgm. The value “tcp” designates a unicast connection to a Message Broker, while the value “pgm” designates a multicast connection to a Multicast Service.
	`servername` is the TCP/IP address or computer name of the computer running a Message Broker (if the `protocol` is “tcp”) or a Multicast Service (if the `protocol` is “pgm”).
	`endpoint` is the port number on which the Message Broker accepts connections (if the `protocol` is “tcp”), or the value tcp.endpoint where endpoint is the port number on which the Multicast Service accepts connections (if the `protocol` is “pgm”). The default endpoint for a Message Broker connection is 5101, while the default endpoint for a Multicast Service is 5104.
	Examples:
	For a Message Broker running on a computer with the TCP/IP address of 12.34.56.78 and port number 5320, the `server_names` value would be:
	`<server_names>tcp:12.34.56.78:5320</server_names>`
	To specify a Multicast Service located on the host Star1 using the default port 5104, the `server_names` value would be:
	`<server_names>pgm:Star1:tcp.5104</server_names>`
	Any process using the profile can communicate with only one Message Broker or Multicast Service. However, you can list multiple Message Brokers or Multicast Services (each separated by a comma) to provide alternatives:
	`<server_names>tcp:HostA:5101,tcp:HostB:5101`
	`</server_names>`

Table 4.2 Parameters for the Properties Group Inside the Profile Group

Parameter	Description
	The Event Transmitter and any clients using a connection profile with this `server_names` value will first attempt to connect to the Message Broker on the computer named HostA, using port 5101. If a connection to that Message Broker is unsuccessful, the process will attempt to connect to the Message Broker on HostB using port number 5101.
project	Specifies a publish/subscribe "universe" name. The default value is StarTeam and should not be changed.
enable_control_msgs	Specifies the types of control messages that the StarTeamMPX process (and application clients) will honor. Only the echo control message should normally be enabled, so this value defaults to echo and normally should not be modified.
server_keep_alive_timeout	integer; number of seconds. The default is 10. Both the Event Transmitter and application clients send an occasional "keep alive" message (analogous to a ping) to the Message Broker or Multicast Service to ensure that it is still responding. This parameter specifies the time (in seconds) during which the keep alive response must be received. If the keep alive response is not received within the specified amount of time, the Message Broker or Multicast Service connection is severed, and a reconnect sequence is initiated. Minimum value is zero, which disables the feature.
server_read_timeout	integer; number of seconds. The default is 10. Specifies the time (in seconds) that the Event Transmitter or client using this profile will wait for a response when performing a read operation. If no data is received from the Message Broker or Multicast Service within the specified amount of time, a timeout occurs, and a keep alive operation is performed. Minimum value is zero, which disables the feature.
server_write_timeout	integer; number of seconds. The default is 10. Specifies the time (in seconds) that the Event Transmitter or client will wait on a write operation. If a write request is not accepted by the Message Broker or Multicast Service within the specified amount of time, a write timeout occurs. Unlike read timeouts, which trigger a keep alive sequence, write timeouts cause the Message Broker or Multicast Service connection to be immediately severed, followed by a reconnect sequence. Minimum value is zero, which disables the feature.
server_start_max_tries	Integer. The default is 1. Specifies how many times the Event Transmitter or application client will traverse the `server_names` list during a connect sequence, before giving up and deciding that no messaging service is available. Minimum value is zero, which disables the feature.
server_start_delay	Integer; number of seconds. The default is 10. Specifies how long (in seconds) to wait between traversals of the `server_names` list. If an attempt to reconnect to a Message Broker or Multicast Service fails, the Event Transmitter or application client will wait for up to the specified number of seconds before attempting the reconnect again. Minimum value is zero, which disables the feature.

Here is an example unicast profile:

```
<Profile>
  <ProfileName>Unicast Off-site</ProfileName>
  <ProfileDescription>The best profile for clients
    connecting over the Internet.
    </ProfileDescription>
  <Properties>
    <server_names>tcp:123.45.67.89:5101
      </server_names>
    <project>Product1</project>
    <enable_control_msgs>echo</enable_control_msgs>
    <server_keep_alive_timeout>10
      </server_keep_alive_timeout>
    <server_max_reconnect_delay>10
      </server_max_reconnect_delay>
    <server_read_timeout>10</server_read_timeout>
    <server_write_timeout>10</server_write_timeout>
    <server_start_max_tries>1
      </server_start_max_tries>
    <server_start_delay>10</server_start_delay>
  </Properties>
</Profile>
```

If you manually edit an Event Transmitter configuration-specific XML file, the changes will take effect after you save the updated file and restart the corresponding configuration. If you use the Server Administration utility, changes can only be made to the profiles, but they go into effect immediately.

If you edit an Event Transmitter template XML file, you can only edit it manually. The updated file will be used for new configurations that you create using the Server Administration utility.

Understanding Connection Profiles

Event Transmitters use a connection profile to determine which Message Broker or Multicast Service to use. File Transmitters do not use profiles because this is already determined by the Event Transmitters.

A connection profile defines connection and usage parameters for a single Message Broker or a single Multicast Service. A **unicast profile** defines a Message Broker connection, while a **multicast profile** defines a Multicast Service connection.

One or more connection profiles are defined in the MPXEventTransmitter.xml file. Multiple connection profiles can be defined when multiple Message Brokers or Multicast Services have been deployed, or when multiple profiles are desired with different connection parameters.

Within MPXEventTransmitter.xml, one connection profile is designated as the **server default** profile. This profile is used by the transmitter when it initializes. The server default profile should always be a unicast profile, because the transmitters are “pure publisher” applications that receive no benefit from using multicast messaging.

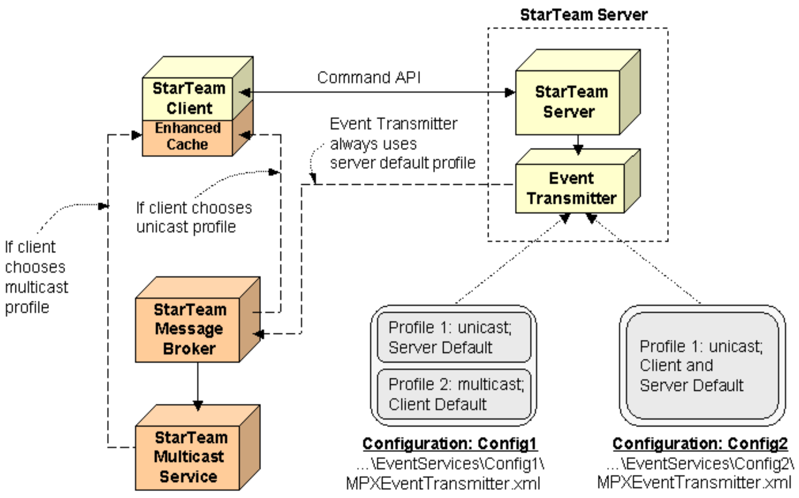
One connection profile is also designated as the **client default** profile, causing that profile to be the default profile used by StarTeam clients and Cache Agents. Only StarTeam Windows clients and Cache Agents can use multicast profiles. However, they are not restricted to multicast profiles. They can use either type of profile.

In environments that use a single Message Broker and no Multicast Service, only a single unicast connection profile is needed. That profile is designated as both the server and client default, and the specified Message Broker is used by all. This scenario is viable for small to moderate user communities having less than 100 users.

Using Profiles with Multiple Connections

In Figure 4, a sample StarTeam Server is shown with two MPX-enabled server configurations (Config1 and Config2). Each configuration has its own MPXEventTransmitter.xml file. In the XML file for Config1, two profiles are defined: a unicast profile which is the server default profile, and a multicast profile which is the client default profile. In the XML file for Config2, a single unicast profile is defined, and it is both the server and client default profile.

Figure 4.1 Example Configuration-specific Transmitter XML Files


StarTeam Server
StarTeam Client
Enhanced Cache
Command API
StarTeam Server
Event Transmitter always uses server default profile
Event Transmitter
If client chooses unicast profile
If client chooses multicast profile
StarTeam Message Broker
StarTeam Multicast Service
Profile 1: unicast; Server Default
Profile 2: multicast; Client Default
Profile 1: unicast; Client and Server Default
Configuration: Config1
...\EventServices\Config1\ MPXEventTransmitter.xml
Configuration: Config2
...\EventServices\Config2\ MPXEventTransmitter.xml

When an Event Transmitter for a specific configuration is successfully initialized, one profile is designated as the default client profile. StarTeam clients can open projects in MPX mode using that default profile or by choosing an alternate connection profile:

- When a client first logs onto a server configuration, it queries the server to determine if it is operating in MPX mode. If the server configuration is operating in MPX mode, it returns the list of connection profiles defined in its Event Transmitter XML file.
- If the user has locally chosen a specific connection profile to use for the configuration, the details of the chosen profile are loaded. Otherwise, the details of the client default profile are loaded.
- The client uses the chosen connection profile to connect to the corresponding Message Broker or Multicast Service. If a connection cannot be established, the client displays an error message, and continues accessing the configuration as if it were not MPX-enabled.
- After the client establishes a Message Broker or Multicast Service connection, it continues to use that connection even as other projects and configurations are opened.
- When a Message Broker or Multicast Service connection is established and a new project is opened on a different, MPX-enabled server configuration, the client verifies the message path to the new configuration. If messages can travel from the new configuration's Event Transmitter to the client's current Message Broker or Multicast Service (for example, because they are the same or in the same cloud), the client uses StarTeamMPX to access the second server configuration. If the message path cannot be verified, the client displays an error message, and accesses the second configuration as if it were not MPX-enabled.

Note that the client can also define custom connection properties that are stored locally on the client workstation. Client configuration options are described in Chapter 6, "Configuring Clients."

Understanding the File Transmitter

The File Transmitter obtains information about new and updated Native-II files by scanning the vault cache when it first runs and from events it receives from the Event Transmitter. It logs this information into CacheJournal.dat. It also trims the CacheJournal.dat file to keep the file's size manageable.

Root Cache Agent uses this information and takes files directly from the Native-II Vault.

Startup

The first time a server configuration uses the File Transmitter, the File Transmitter scans the server configuration's existing Native-II vault cache and generates a new CacheJournal.dat file. The initialization process may take up to an hour or more as it computes the MD5 of each file in the cache; however, the server is available as the file is built. File content transmission begins when the cache scan is complete. If any errors occur, they are reported in the server's log file.

The XML File

The general format of the Event Transmitter XML file is as follows:

```
<?xml version="1.0" ?>
<StExternHandlerModule>
    <Module>
        <ModuleName>filexmitt.dll</ModuleName>
        <ModuleLanguage>C++</ModuleLanguage>
    </Module>
    <HandlerSet>
        <Handler>
         <HandlerName>StarTeamMPX File Transmitter</HandlerName>
          <HandlerDescription>File transmitter for the MPX Cache
Agent.(Thiseventhandlerdoesnotuseprofiles.)</HandlerDescription>
            <HandlerType>async</HandlerType>
            <EventID>-1</EventID>
            <ClassName />
            <HandlerOrder>1</HandlerOrder>
<JournalPath>C:\ProdServer\CacheJournal.dat</JournalPath>
<MaxJournalAge>180</MaxJournalAge>
<JournalTrimInterval>24</JournalTrimInterval>
        </Handler>
    </HandlerSet>
</StExternHandlerModule>
```

All File Transmitter-specific parameters are optional. By default, the File Transmitter gets its mandatory parameters from the server.

The parameters in Table 4.3 should rarely need to be overridden. The parameters are listed in the order in which they appear in the XML file.

Table 4.3 File Transmitter Parameters

Parameter	Description
JournalPath	Specifies the location of the cache journal file, which is created and maintained by the File Transmitter. The default is 'CacheJournal.dat' in the configuration's 'repository path' folder.
MaxJournalAge	Integer; number of days. The default is 180. Specifies the maximum age in days of journal records within the cache journal file. The minimum value is 1.

Table 4.3 File Transmitter Parameters

Parameter	Description
JournalTrimInterval	Integer; number of hours. The default is 24. Specifies how often (in hours) the File Transmitter trims the cache journal file of records that are older than the configured age. The cache journal is trimmed of aged records when the server first starts and every trim-interval hours. This parameter ensures that the CacheJournal.dat file will be periodically trimmed if the server runs for a long time without restarting. After initialization, the File Transmitter scans the vault cache only for the files created after the time stamp of the last journal file entry and adds new journal records for them.This scan serves to keep the journal file compact, allowing records older than MaxJournalAge days to remain “trimmed.” It also speeds up the scan time. Minimum value is zero, which disables the feature.

Chapter

5

Managing Cache Agents

Cache Agent adds persistent file caching to StarTeamMPX. It allows the contents of new files to be broadcast in encrypted form via the MPX framework. One or more Cache Agents can be distributed throughout the enterprise, providing local file caches for diverse groups.

Cache Agent-aware clients can fetch files from any available Cache Agent. By using “network-near” Cache Agents, clients receive faster file check-out performance. This reduces the demands on the StarTeam Server and frees resources for additional tasks and users.

With a tier of Cache Agents, there is less overall network traffic because files are usually broadcast only once. They can add themselves dynamically to the “cloud”.

The Root Cache Agent provides several important services:

- It supports the same fetch requests that all Cache Agents provide. Because of this, it provides an alternate “pathway” to the StarTeam server configuration’s vault and relieves the server of file fetch requests. In this capacity, the Root Cache Agent can serve clients directly, although it shares I/O access, and – if it is running on the same computer – network access with the StarTeam Server process.
- The Root Cache Agent also acts as an “upstream” Cache Agent, providing “downstream” Remote Cache Agents with a place to forward cache misses. This increases the “hit rate” of Remote Cache Agents and improves their effectiveness.
- The Root Cache Agent provides a “catch up” API, allowing Remote Cache Agents to proactively fetch files that they missed while they were not running. Because the Journal is time-sequenced, a Remote Cache Agent can request all content newer than the last known cache time stamp and “trickle charge” with that content before clients request it. This feature reduces cache misses and allows Remote Cache Agents to be effective even over unreliable network connections.

Although a Root Cache Agent can be assigned to only one server configuration, a Remote Cache Agent can be assigned to several.

A Remote Cache Agent can be configured to:

- Refine its subscription to a server configuration such that it receives files only in specified projects.
- Forward request misses to either a Root Cache Agent or another “upstream” Cache Agent

- Use a different upstream Cache Agent for the file contents of each server configuration it is tracking
- Automatically find the Root Cache Agent via MPX poll messages. This allows a Remote Cache Agent to be installed with a minimal configuration: all it needs is connection parameters to a Message Broker, the GUIDs of the server configurations it will track, and the names of the projects it wants to track within each server configuration. When first started, a Remote Cache Agent automatically finds the Root Cache Agent for each server configuration it is tracking and will trickle charge itself with the latest content that it is interested in. Even while it is trickle charging, a Remote Cache Agent can be used immediately, since cache misses will be forwarded on demand.

Planning for the Cache Agents

Issues to be considered as you do Cache Agent planning include:

- On Windows, a Cache Agent can run as a service or as a console application. More than one Cache Agent can be run as a service on each computer if appropriate.
- A Root Cache Agent can manage only one server configuration. It uses the server configuration's archives and cached files, rather than restoring these in a cache of its own.
- The Root Cache Agent must use a Message Broker, but the Remote Cache Agents can use either Message Brokers or Multicast Services.
- The Root Cache Agent requires access to the vault for the one server configuration that it services. Consequently, it can be installed on the same computer as the StarTeam Server. Alternatively, if it can be installed on a separate computer to prevent the Root Cache Agent from competing for CPU or network I/O with the corresponding server configuration. However, this requires it to access the archive files and the CacheJournal.dat via a shared network drive, so use this option only when a high-speed network file system is in place.
- There is no limit to the number of Cache Agents that can be installed throughout an enterprise. However, keep in mind that each Cache Agent requires access to a Message Broker, for which there is a maximum of 10 within a single messaging "cloud".
- Remote Cache Agents should be installed in each geographic location that can benefit from improved file check-out performance. One approach is to install a Cache Agent in each network environment in which local users can access it over a high-speed LAN. (Example: Install two Remote Cache Agents at headquarters, one each for the engineering and quality assurance teams, a third Remote Cache Agent at the Chicago office, and a fourth at the London office.)
- Installing a Remote Cache Agent on a computer dedicated to a check-out intensive application such as a build utility can be very beneficial. If that computer is sufficiently "network-near" to the StarTeam Server's computer, you could deploy a Root Cache Agent on the build computer as long as that computer has access to the server configuration's vault. This reduces check-out demands on the StarTeam Server, but it doesn't reduce I/O to the vault.
- A Remote Cache Agent can receive broadcasts and store files from multiple server configurations. It stores all new files for the specified server configurations or for the specified projects within the server configurations. However, each unique file is stored only once, regardless of the number of times it is used in different folders, projects, or servers. The file's uniqueness is determined by its contents, not its name or location.

- Cached files are stored individually within a folder tree, which has a configurable root folder. They are stored in encrypted format and decrypted only “at the last moment” within the client process.
- The maximum total size of a Remote Cache Agent’s cache is configurable.
- The cache for a Remote Cache Agent does not have to be backed up and can be deleted, if necessary, when the Remote Cache Agent is not running. (A Root Cache Agent uses the corresponding server configuration’s vault.)
- Clients can be configured to use a specific Cache Agent by specifying that Cache Agent’s host name (or IP address) and port number. Alternatively, some clients can be configured to locate an appropriate Cache Agent automatically. If multiple Cache Agents are available, the client automatically chooses the “network-nearest” Cache Agent. This feature keeps administrative overhead to a minimum and allows the automatic detection of new Cache Agents by clients.
- You can use a single Remote Cache Agent without deploying a Root Cache Agent. It will receive files via MPX broadcasts. If the Message Broker is running most of the time, it will receive most files and could be useful to a person whose job is building software applications. (Without a Root Cache Agent there is no file-catch-up or request forwarding.)

Understanding the Native-I Cache Auditor

This section pertains only to server configurations that were created prior to StarTeam Server 2005 and do not have their vaults completely converted from the Native-I to the Native-II format.

Normally, the Root Cache Agent receives information about new and updated files from the File Transmitter via information that it logs into CacheJournal.dat file records. It obtains this information by scanning the vault cache when it first runs and from events it receives from the StarTeam Server’s Event Transmitter.

The Native-I Cache Auditor application improves the effectiveness of a Root Cache Agent by monitoring the Native-I vault cache for new files and by maintaining a “cache audit” file. When a new Native-I file is added to the vault cache, it adds information to the cache audit file that the Root Cache Agent can track, making it aware of the file. Consequently, it can return files for which update events are not generated by StarTeam Server. Run Cache Auditor before you run the Root Cache Agent so that the appropriate .dat files are created.

The Native-I Cache Auditor is an optional console application that runs completely independently of the StarTeam Server and Root Cache Agent processes. The installer creates a batch script called RunV1CacheAudit.bat to run the application. You must manually run this batch file from a console window. You might choose to add a shortcut to your Program Files > Startup folder for this batch script so that the script starts each time you log on to Windows.

The Native-I Cache Auditor can monitor **all** the server configurations for a StarTeam Server. However, its application **must** be installed on the same computer as the server configuration caches that it will monitor. That is, each Native-I vault cache folder must be on a disk that is local to the V1CacheAudit.exe program.

For example, if you run:

```
RunV1CacheAudit C:\Prod1\Vault\Cache C:\Prod2\Vault\Cache C:\Prod3\Vault\Cache
```

Native-I Cache Auditor will create three CacheAudit.dat files simultaneously and keep them up to date. The .dat files are found in the repository path.

You can install the Native-I Cache Auditor along with the Root Cache Agent, by selecting the “Install optional Native-I Cache Audit application” during the installation process.

Configuring a Root Cache Agent

By default, the name of the Root Cache Agent configuration file is RootCAConfig.xml. If you run only one instance of Cache Agent on a given computer, you should probably keep the default name. Otherwise you create multiple configuration files (whether for Root or Remote Cache Agents) and run them as either services or console applications—whatever is appropriate for your environment. In general practice, you are unlikely to have more than one Cache Agent per computer. See "Running Cache Agent" on page 58 for more details.

An example of the configuration file to establish a Root Cache Agent is shown below. The root group for the Cache Agent configuration file must be `MPXCacheAgent`. The XML parameters are summarized in tables 5.1 and 5.3.

```
<?xml version="1.0" ?>
<MPXCacheAgent>
    <RootCacheAgent>
        <RootRepositoryPath>C:\ProdServer</RootRepositoryPath>
        <RootHiveIndexPath>C:\ProdServer\HiveIndex\hive-index.xml
           </RootHiveIndexPath>
        <RootCachePath>C:\ProdServer1\Vault\Cache</RootCachePath>
   <RootJournalPath>C:\ProdServer1\CacheJournal.dat</RootJournalPath>
        <RootAuditPath>C:\ProdServer1\Vault\Cache\CacheAudit.dat
           </RootAuditPath>
    </RootCacheAgent>

    <MessageBroker>
        <server_names>tcp:12.35.58.71:5101</server_names>
       <enable_control_msgs>echo</enable_control_msgs>
       <start_server_delay>10</start_server_delay>
       <socket_connect_timeout>10</socket_connect_timeout>
</MessageBroker>

    <RequestPort>5201</RequestPort>
    <MaxConnections>100</MaxConnections>
    <MaxCatchupSize>100000000</MaxCacheSize>
    <SharePolicy>Public</SharePolicy>
    <CachePath>C:\.MPXCacheAgent\Cache</CachePath>
    <MaxCacheSize>100000000</MaxCacheSize>
</MPXCacheAgent>
```

Configuring a Remote Cache Agent

Remote Cache Agent can cache content for many StarTeam server configurations. They can be cache files for all the projects managed by the server configuration or be set up to filter for specific projects within each server configuration.

An example of a configuration file that defines a Remote Cache Agent is shown below. The absence of a `RootCacheAgent` parameter denotes the configuration for a Remote Cache Agent.

```
<?xml version="1.0" ?>
<MPXCacheAgent>
    <MessageBroker>
        <server_names>tcp:12.35.58.71:5101</server_names>
    </MessageBroker>
```

```
        <RequestPort>5201</RequestPort>
        <MaxConnections>100</MaxConnections>
        <MaxCatchupSize>100000000</MaxCatchupSize>
        <SharePolicy>Public</SharePolicy>
        <CachePath>C:\.MPXCacheAgent\Cache</CachePath>
        <MaxCacheSize>100000000</MaxCacheSize>

        <ContentSource>
            <ServerGUID>be5ee3b0-c719-49c6-a1a1-f493764a03f5</ServerGUID>
            <UpstreamCache>
                <UpstreamHost>ProdServer1</UpstreamHost>
                <UpstreamPort>1123</UpstreamPort>
            </UpstreamCache>
        </ContentSource>

        <ContentSource>
            <ServerGUID>79408139-1768-4031-9ddd-7f1b095c94e7</ServerGUID>
           <Projects>
              <Project>FelixTools</Project>
              <Project>Bank*</Project>
              <Project>Insurance*West*</Project>
           </Projects>
            <UpstreamCache>
                <AutoLocate/>
            </UpstreamCache>
        </ContentSource>
    </MPXCacheAgent>
```

The XML parameters for both Root and Remote Cache Agents are summarized in tables 5.1 through 5.3.

Unless otherwise stated, all integer parameters have very large positive ranges. For example, all time values have a range of 1 to 2,147,483,647. In most cases, you should consider the default value the minimum setting. Use a practical upper limit based on common sense. For example, if you set the CacheCheckInterval at 10,000,000, the cache will only be checked once every 4 months or so, which makes cache management ineffective.

Table 5.1 Parameters Used by Any Cache Agent

Parameter	Description
CacheCheckInterval	Integer; number of seconds. The default is 60. The frequency with which the Cache Agent compares its cache size to the configured cache limit (MaxCacheSize). When the total cache size exceeds the configured limit, least-recently-used files are removed from the cache until the cache size is under the configured limit. Root Cache Agents do not add to their local cache after the Native-I vault has been converted to Native-II format. Minimum value is zero, which disables the feature.
CachePath	Path. The default is "/MPXCacheAgent/Cache". The root folder of the Cache Agent's local cache. Cached files are stored in compressed, encrypted format within subfolders of this path. For the Root Cache Agent, the local cache is used only to store compressed and encrypted copies of files read from the Native-I vault cache. As files are requested from the Root Cache Agent either by “downstream” Cache Agents or by clients, they are compressed, encrypted, and stored in a folder tree rooted at the specified directory. The local cache makes secondary file access faster, and it removes I/O contention with files in the server configuration’s vault. Root Cache Agents do not add to their local cache after the server configuration’s Native-I vault has been converted to Native-II format.

Table 5.1 Parameters Used by Any Cache Agent

Parameter	Description
InitialRequestThreads	Integer; number of connections. The default is 10. The range is from 1 to MaxConnections. The initial number of request handler threads launched when the Cache Agent starts. Additional request handler threads are launched, up to MaxConnections, as needed when all current threads are dedicated to active connections.
MaxCacheSize	Integer; number of bytes. The default is100MB. The range is 0 to approximately 8 exabytes (264 characters). The maximum size of the Cache Agent's cache in bytes. If this value is zero, the cache size will not be constrained. Otherwise, files are periodically deleted on a least-recently-used basis to maintain the specified size. The actual total size of the cache may rise above this value momentarily while new files are received.
MaxCatchupSize	Integer; number of bytes. The default is 100MB. The range is 0 to 2^{63}-1. For a Root Cache Agent, this parameter constrains the maximum number of files returned in a catch-up request. For a Remote Cache Agent, this parameter constraints the maximum number of files requested in a catch-up request. In both cases, the value is the total size in bytes of the files in the catch-up operation. If this value is 0, the catch-up request is unconstrained. In a given catch-up operation, the smaller of the requester's and the requestee's MaxCatchupSize is used to constrain the operation.
MaxConnections	Integer; number of connections. The default is 100. The range is 1 to 1000. The maximum number of simultaneous connections that the Cache Agent will accept. This value controls the maximum number of request handler threads used by the Cache Agent. If all request handler threads have been started and are in use when a new connection is received, it is queued until a request handler becomes available. A larger value than the default could be used in highly concurrent environments, at a cost of more memory and potentially more demand on the Cache Agent. A smaller number is generally unnecessary. The maximum is limited by OS/process issues.
MessageBroker	Required group. A group for specifying Message Broker parameters. Minimally, a Cache Agent's MessageBroker group should contain a value for the server_names parameter. A Remote Cache Agent uses a Message Broker connection to receive file content messages from the content source(s) that it is monitoring. It also uses the Message Broker connection if is a "public" Cache Agent that will respond to poll messages. (See SharePolicy.) **server_names** Defines the publish/subscribe messaging service to be used by the Cache Agent in the format *protocol*:*host*:*port*. The default is the value "_node" which is equivalent to "localhost". To use the default, the Message Broker must be on the same computer as the Cache Agent. To use unicast messaging, the *protocol* should be "tcp", *host* must be the name or IP address of the Message Broker computer, and *port* must be the port number with which the Message Broker is receiving connections (5101 by default). Example: tcp:MBServer1:5101. To use multicast messaging, *protocol* should be "pgm", *host* must be the name or IP address of the Multicast Service computer, and *port* should be "tcp.5104" (or whatever port number the Multicast Service is listening on). Example: pgm:12.34.56.78:tcp.5104. A Root Cache Agent should use unicast messaging, which means it connects directly to a Message Broker.

Table 5.1 Parameters Used by Any Cache Agent

Parameter	Description
	enable_control_msgs Specifies the types of control messages that the StarTeamMPX process (and application clients) will honor. Only the echo control message should normally be enabled, so this value defaults to echo and normally should not be modified. **server_start_delay** Integer. Number of seconds. The default is 10. Specifies the interval between attempts to reconnect to the Message Broker. **socket_connect_timeout** Integer. Number of seconds. The default is 5. Controls how long in seconds the Event Transmitter, Cache Agent, or StarTeam client waits to connect to the Message Broker before giving up. The default is good for most environments. Minimum value is zero, which disables the feature.
RequestPort	Integer; port number. The default is 5201. The range is 1 to 65535. The port number with which the Cache Agent receives requests. The port number can not be in use by any other process on the same host.
RequestReadTimeout	Integer; number of seconds. The default is 30. Specifies the time which a Cache Agent will wait to read the next request from a client before passively closing the corresponding connection. When the connection is closed, the request handler thread is freed-up to service other connections. Minimum value is zero, which disables the feature.
SharePolicy	Public or Private. The default is Public. Indicates whether or not this Cache Agent advertises the server GUIDs for which it is caching data. A cache-aware MPX client can broadcast a "poll" request to look for Cache Agents that are caching data for a specific StarTeam server GUID. Public Cache Agents will respond to such requests when they are caching the requested server's content, but private Cache Agents will not.

Table 5.2 Parameters Used by Remote Cache Agent Only

Parameter	Description
CatchupCheckInterval	Integer; number of seconds. The default is 300. The interval in which a remote Cache Agent will check to see if any of its cache sources require catch-up because the Message Broker connection was lost. When catch-up is required, catch-up cycles continue to be performed until normal Message Broker connectivity has been resumed. The Remote Cache Agent "catch-up" function is still based on the "whole server" scope, not project-specific scope. For example, if a Remote Cache Agent that is tracking by project is stopped or is disconnected from the network at time T1 and then reestablishes a connection at time T2, it will request a catch-up of files added since time T1. However, it will receive a list of all files added since T1, not just those added for the projects it tracks. Hence, catch-up may add more files than actually needed based on a Remote Cache Agent's projects. Similarly, a Cache Agent's "pre-charge" (initial catch-up) uses ‘whole server’ scope instead of project scope. Minimum value is 1.

Table 5.2 Parameters Used by Remote Cache Agent Only

Parameter	Description
ContentSource	Each ContentSource group specifies a StarTeam server configuration that the Remote Cache Agent will monitor for new file content via MPX messages. Optionally, this group specifies an "upstream" Cache Agent that will be contacted for catch-up and forwarding requests. A Remote Cache Agent can have one or more ContentSource groups. It can contain the following parameters: **ServerGUID** This value must be specified within each ContentSource group. It must be the GUID of the StarTeam Server that this remote Cache Agent will track. It is used to establish the publish/subscribe “subjects” used to monitor new file content. If the GUID is specified incorrectly, the Cache Agent will not receive new file content. To locate the GUID for a specific server configuration: 1 From the Server Administration dialog, select the server. 2 Click the Properties icon on the toolbar or select Server > Server Properties from the menu. These actions display the Properties dialog. 3 Copy the GUID from the server configuration’s Properties dialog and paste it into the .xml file. **Projects** If specified, this group value indicates that not all content for the corresponding StarTeam server is to be tracked. Instead, only content for each Project parameter within the Projects group will be tracked and stored. It contains one or more Project parameters. **Project** Each Project parameter specifies one project name or name pattern. All files checked into a project whose name matches the specified name or pattern are cached by the Cache Agent. A pattern is a name that contains one or more asterisk (*) wildcard characters. For caching purposes, each file revision "belongs" to the project it is checked in to. In practice, a file could be shared among multiple projects and a given check-in may cause the corresponding file revision to appear in multiple projects. Because the file content is "broadcast" with the project name it was checked into, only Remote Cache Agents tracking that project will store the file. For example, suppose a file is shared between projects P1 and P2, and a new revision of the file is checked into project P1. Only Remote Cache Agents tracking project P1 store the new file revision. A Remote Cache Agent tracking only project P2 will not receive the broadcast. However, "pull through" caching, explained more below, allows the Remote Cache Agent that is tracking only project P2 to obtain the file revision anyway. When a Remote Cache Agent is configured to track specific projects, it still accepts requests for any file. If a requested file is not in its local cache, a Remote Cache Agent forwards the request to the appropriate Root Cache Agent and stores it locally for future requests, regardless of which projects the file belongs to. In other words, tracking-by-project limits files visible to Remote Cache Agents by "push" caching, but it does not limit the files a Remote Cache Agent can store via "pull through" (request forwarding) caching. **UpstreamCache** This group specifies that an upstream Cache Agent will be contacted for catch-up and forwarding requests for content related to the corresponding StarTeam server. The upstream Cache Agent can be automatically located or explicitly configured as specified by the group’s parameters: AutoLocate, UpstreamHost, and UpstreamPort. If a ContentSource parameter does not contain an UpstreamCache parameter, catch-up and miss forwarding will not occur for the corresponding server configuration. When used, UpstreamCache must contain either an UpstreamHost or an AutoLocate parameter. The two are mutually exclusive. UpstreamHost (along with UpstreamPort) explicitly specifies the upstream Cache Agent, while AutoLocate requests polling. **AutoLocate** Indicates that the Root Cache Agent for the corresponding StarTeam server configuration is to be automatically located and used as the upstream Cache Agent.Remote Cache Agent to continue periodic polls for a Root Cache Agent until one is found. It is an empty tag and mutually exclusive with UpstreamHost.

Table 5.2 Parameters Used by Remote Cache Agent Only

Parameter	Description
	UpstreamHost Explicitly identifies the host name or IP address of the upstream Cache Agent to be used for content related to the corresponding StarTeam server. This parameter is mutually exclusive with AutoLocate. **UpstreamPort** Integer. The default is 5201. The range is from 1 to 65535. Only meaningful if UpstreamHost is specified, this parameter specifies the port number of the upstream Cache Agent.
PrechargeSize	Integer. The default is MaxCatchupSize. Specifies the maximum size of a Remote Cache Agent's first catch-up operation, which is used to precharge its cache. It defaults to the Cache Agent's MaxCatchupSize, but it can be specified larger or smaller than that value. If this value is zero, the Remote Cache Agent does not perform an initial precharge operation.

Table 5.3 Parameters Used by Root Cache Agent Only

Parameter	Description
RootCacheAgent	Group for Root Cache Agent parameters as defined below. **RootRepositoryPath** Required. The full path name of the StarTeam server configuration's repository folder. **RootHiveIndexPath** The full path of the StarTeam Server's Native-II hive index file. The default is *RootRepositoryPath*/HiveIndex/hive-index.xml. **RootCachePath** Not meaningful after a pre-2005 Native-I vault has been converted to Native-II format. The full path name of the StarTeam Server's Native-I vault cache folder. The default is *RootRepositoryPath*/Vault/Cache. **RootJournalPath** The full path name of the StarTeam server configuration's cache journal file, which is created by the File Transmitter. The default is *RootRepositoryPath*/CacheJournal.dat. **RootAuditPath** Not meaningful after a pre-2005 Native-I vault has been converted to Native-II format. Both enables the Native-I Cache Audit feature and specifies where the audit file resides. The V1CacheAudit application must be installed and executed at least one, creating the corresponding audit file, before the Root Cache Agent can be started with this parameter. The usual name and location for this file is *RootCachePath*/CacheAudit.dat".

Reviewing Status and Log Information

You can review status and log information for each Cache Agent in your browser by using the Cache Agent’s port number and the name (or IP address) for the computer on which the Cache Agent is located. The status information for a Root Cache Agent and a Remote Cache Agent are slightly different.

To check the status of an active Cache Agent:

1 Open your browser.

2 Entering the appropriate URL using the following syntax.

```
http://host:port/status
```

host is the Cache Agent’s host name or IP address

port is the inbound port on which it is listening

Similarly, you can use the syntax:

http://*host*:*port*/log

to view the Cache Agent's log file from a browser.

If verbose mode was used when the Cache Agent was started (see "Running Cache Agent as a Service" on page 59 and "Running Cache Agent As a Console Application" on page 61 for more details), you can use the following syntax:

http://*host*:*port*/debuglog

to display a more detailed log.

Using Cache Agent with the Clients

Starting with StarTeam 2005, the Cross-Platform client, IDEs based on StarTeam Cross-Platform or .NET components, and the Bulk Checkout (bco) command-line utility can perform check-out operations using Cache Agent.

See "Using Cache Agent from the Cross-Platform Client and IDEs" on page 53 for information about using Cache Agent when checking out files with this client.

See the *StarTeam User's Guide* for more information about bco.

Be aware the keyword expansion is not available for files checked out using Cache Agent.

Chapter

6

Configuring Clients

Any client can connect to an MPX-enabled StarTeam Server, but not all of them can take advantage of MPX-features.

Note Unless otherwise stated, this chapter's references to client refer to only the StarTeam Windows and Cross-Platform clients.

Using MPX from a Client

To configure support for StarTeamMPX on your workstation:

1. Start the client.
2. Select the Tools > Personal Options command from the menu bar.
3. In the resulting *Personal Options* dialog, select the StarTeamMPX tab.
4. Select the "Enable StarTeamMPX" check box to use StarTeamMPX with any MPX-enabled StarTeam Server connected to by the client.
5. Do one of the following:
   - To refresh manually (*Shift+F5*), clear the "Automatic refresh with" check box.
   - To refresh automatically:
     1. Select the "Automatic refresh with" check box.
     2. Set a minimum number of seconds between refreshes in the "Minimum delay of ___ seconds" text box. The default is 5 seconds.

3 Set a maximum number of seconds between refreshes in the “Maximum delay of ___ seconds” text box. The default is 30 seconds.

After every cache update, the application waits a minimum number of seconds before refreshing. This means that if cache updates are infrequent, the application performs a refresh almost immediately. However, if cache updates are frequent, the minimum refresh timer is constantly being reset and never reaches the number of seconds set for a refresh. In such cases, the next refresh occurs when the maximum number of seconds between refreshes forces a refresh.

6 Click OK.

Your changes will take effect for all projects you open from this point on. Note that any projects that are currently open will be unaffected by your changes.

To stop using StarTeamMPX:

- Clear the “Enable StarTeamMPX” check box to stop support for any MPX-enabled StarTeam Server connected to by the client.

Displaying MPX Status

When you open a project, the client’s status bar displays the type of server configuration in use, the autorefresh setting, and (for StarTeamMPX configurations) whether support for StarTeamMPX is enabled.

Table 6.1 shows the icons and words that provide information about StarTeamMPX when they appear on the status bar.

Table 6.1 MPX Information in the Client’s Status Bar

Status Bar Information	Description
Yellow lightning bolt	Indicates that StarTeamMPX is available and enabled for the currently selected project view. If Web Edition can use the default client profile for an MPX-enabled server and, therefore, take advantage of MPX, this icon appears in front of the server configuration’s name in the browser window.
Gray lightning bolt	Indicates that StarTeamMPX is available for the currently selected project view but that it has not been enabled in the client.
Red circle with a slash beside a yellow lightning bolt	Indicates that StarTeamMPX is enabled for the currently selected project view, but something happened to break the connection. For example, the Message Broker may be stopped.
(no icon)	Indicates that StarTeamMPX is not available for the currently selected project view.
Instant	Indicates that StarTeamMPX’s autorefresh is turned on.

Table 6.1 MPX Information in the Client's Status Bar

Status Bar Information	Description
Auto	Indicates that your workstation's autorefresh is turned on—but that StarTeamMPX's autorefresh is either turned off or unavailable. (Your workstation's autorefresh option is on the Workspace tab of the *Personal Options* dialog.)
Manual	Indicates that your workstation's autorefresh is turned off and that StarTeamMPX's autorefresh is either turned off or unavailable. You must manually refresh the current project view by pressing *F5*.

Choosing a Non-default Connection Profile

In some cases, the client default profile for a given configuration may not be appropriate for every client. For example, a multicast profile will not work for application clients whose connection paths do not forward multicast packets (for example, those accessing the configuration through a firewall). In those cases, the user can choose a profile other than the default.

To choose a StarTeamMPX connection profile other than the client default profile:

1. In the client, display the *Open Project Wizard* by selecting the Project > Open command.
2. Select the server configuration for which you wish to select a non-default client profile, and click Server Properties.
3. On the resulting *Server* dialog, click the "MPX Profiles" button.

   The resulting dialog lists each profile defined in the Event Transmitter XML file for this server configuration.
4. To examine the details of any profile, select the profile and click the Properties button.
5. Select the alternate profile that you wish to use and click Set. (If you wish to restore the client default profile for this configuration, click Restore Default instead.)
6. Click Close on this dialog, and then click OK on the *Server Properties* dialog.

When you open a project on a configuration for which you have chosen a non-default client profile, that profile will be used. Note that after a Message Broker or Multicast Service connection has been established, the client continues to use that messaging service even when it opens projects from other configurations.

Logging MPX Information in the Client Log

The StarTeam Windows client can create a client log named StarTeam.log.

To record MPX information in this log:

1. Start the client.
2. Select the Tools > Personal Options command from the menu bar.
3. In the resulting *Personal Options* dialog, select the Workspace tab.
4. Select the Log StarTeamMPX Events check box.
5. Click OK.

To see the StarTeam log:

- Select the Tools > StarTeam Log command from the menu bar.

Working with the StarTeamMPXCommands.txt File

In some cases, you may want a specific client to use different Message Broker or Multicast Service properties than any of those available in the Event Transmitter's XML file. One reason for wanting to do this is for external clients operating outside of the corporate firewall when no profile has been defined for external users. For such users, you may require that they use an external Message Broker that is not defined in any of the Event Transmitter's profiles.

To direct a client to use an external Message Broker, you need to set the `server_names` parameter for the appropriate Message Broker. This parameter, as well as other options, can be overridden by creating a file called StarTeamMPXCommands.txt and placing it in the same folder as the client executable (normally "C:\Program Files\Borland\StarTeam Client 2005 R2\" for the Windows client, or "C:\Program Files\Borland\StarTeam Cross-Platform Client 2005 R2\ for the Cross-Platform client). This file must contain one parameter per line in the format:

`setopt` *`optionName optionValue`*

Valid parameter names and values are the same as the XML parameter names (without the angle-brackets) used within the Profile group of the MPXEventTransmitter.xml file. See the tables named "Event Transmitter Parameters" on page 30 and "Parameters for the Properties Group Inside the Profile Group" on page 31.

Note that the StarTeamMPXCommands.txt file uses one setopt command per line, whereas the MPXEventTransmitter.xml file uses an XML format. As an example, to override the `server_names` parameter for a client, the StarTeamMPXCommands.txt file would have to contain a line such as:

```
setopt server_names tcp:ExtHostA:5112
```

This parameter will set the `server_names` parameter that the client receives from the StarTeam Server, instructing it to use the Message Broker on host ExtHostA at port number 5112. Note that when a client opens a project on an

MPX-enabled server configuration, it always pings the Event Transmitter to ensure that it is visible within the Message Broker cloud. This ensures that if the client is using a different Message Broker than the Event Transmitter, they are properly connected into a cloud and visible to each other. If a client cannot successfully ping the Event Transmitter for a new project being opened, it will display an error dialog and proceed in non-StarTeamMPX mode.

The StarTeamMPXCommands.txt file can be used to override other Message Broker options as well, but only in unusual circumstances should anything other than the `server_names` parameter be overridden.

Using Cache Agent from the Cross-Platform Client and IDEs

If StarTeamMPX and Cache Agent have been installed and configured, you can use Cache Agent from your Cross-Platform client or an IDE based on StarTeam Cross-Platform or .NET components. Using a "network-near" cache agent should provide faster check-out operations.

Enabling Cache Agent Use

The StarTeam Server to which you connect must be MPX-enabled. However, the Cross-Platform client does not have to enable StarTeamMPX to take advantage of Cache Agent. StarTeamMPX provides properties about files and other items. Cache Agent provides file contents. Using both would be typical but is not mandatory.

To enable Cache Agent use:

1. Start the Cross-Platform client.
2. Select Tools > Personal Options from the menu bar.
3. From the resulting Personal Options dialog, select the StarTeamMPX tab.
4. Select the Enable StarTeamMPX Cache Agent check box.
5. You can specify a specific cache agent to use or allow the client to locate the nearest cache agent. Do one of the following:
   - Select the Use Cache Agent At option button, providing both an address and port. The address can be the computer name or an IP address.
   - Select the Automatically Locate the Closest Cache Agent for Each StarTeamMPX option button and let the client do the work.
6. You can change the number of threads in the Maximum Request Threads text box, but the default should be adequate for most users needs.
7. Click OK.

Note From some IDEs, such as the Eclipse and Visual Studio .NET integrations, you can set StarTeam Personal Options although the menu command used to reach the options dialog may be different.

Checking out Files with Cache Agent

The visible advantage to using Cache Agent is the improved speed of file check-out operations. The more files you check out, the more advantage you will gain from Cache Agent. Over time, more and more of the files will come from Cache Agent, reducing the strain on StarTeam Server. As a result, the check-out speed should continue to improve until all files are available from Cache Agent.

For a particular check-out operation, you can see how many files are being sent by StarTeam Server directly and how many are being sent by Cache Agent, by displaying the check-out statistics.

To monitor check-out statistics using Cache Agent:

1. Select the files to be checked out.
2. Select File > Check Out from the menu bar.
3. From the resulting Check Out dialog, select the Show Checkout Statistics check box.
4. Choose any other option settings that are appropriate to your check-out operation.
5. Click OK.
6. During the check out process, you will see a dialog, which indicates the file names and the location from which they are coming (StarTeam Server or Cache Agent).

   After the operation completes, the Checkout Statistics dialog provides a summary.

Note From some IDEs you can Show Checkout Statistics, too.

Using Cache Agent with bco

You can use Cache Agent with the Bulk Checkout (bco) utility. For more information about this utility see the *StarTeam User's Guide*. Be aware that you cannot use keyword expansion with bco.

Chapter

7

Running MPX Components

Most of the StarTeamMPX start-up and shut down routines are performed automatically by the operating system and StarTeam Server. This chapter describes how to manually start and stop the Message Broker and Multicast Services on Windows and Solaris platforms. It also explains how to start and stop Cache Agent on those platforms.

Running Message Broker and Multicast Service on Windows

On Windows platforms, both the Message Broker and the Multicast Service are installed as Windows services. When each of these services is installed, the installer asks whether you wish to create an automatic or a manual service.

- If you choose **automatic**, the corresponding service will be started by Windows each time the system is initialized.
- If you choose **manual**, you must manually start the service each time the system is initialized.

Both automatic and manual services are automatically stopped when the system is shutdown. Both automatic and manual services can be manually started and stopped as needed. The procedures for starting and stopping the Message Broker and the Multicast Service are provided in the following topics.

Starting a Message Broker

To manually start a Message Broker:

1. On the computer where the Message broker is installed, choose Start > Settings > Control Panel > Administrative Tools > Services.
2. Select the service named “StarTeam Message Broker 6.4”.
3. Click Start.

When the Message Broker starts, it reads the STMessageBroker64.ini configuration file. On Windows systems, this file is typically located in the “C:\Program Files\Borland\Message Broker 6.4” folder. Options in this file tell the Message Broker what TCP/IP port (end point) to accept connections on, and which other Message Brokers (if any) to establish communication with to form a Message Broker cloud. See “Configuring a Message Broker Cloud” on page 21 for details on the contents of the STMessageBroker64.ini file.

Stopping a Message Broker

Under most conditions, a Message Broker runs continuously; however, occasionally it may be necessary to stop a Message Broker. For example, you may choose to move a Message Broker to a different computer, or remove a Message Broker from a Message Broker cloud. If the Message Broker that you stop is serving clients, those clients continue to access the server configurations, but without using StarTeamMPX.

To manually start a Message Broker:

1. You should first notify users that StarTeamMPX will be unavailable.
2. On the computer where the Message broker is installed, choose Start > Settings > Control Panel > Administrative Tools > Services.
3. In the Service list box, select the service named “StarTeam Message Broker 6.4”.
4. Click Stop.

Starting a Multicast Service

Starting a Multicast Service is similar to the process used to start a Message Broker. However, because a Multicast Service requires communication with a Message Broker, the Message Broker to which the Multicast Service will connect should be started first.

To manually start the Multicast Service:

1. On the computer where the Message broker is installed, choose Start > Settings > Control Panel > Administrative Tools > Services.
2. Select the service named “StarTeam Multicast Service 6.4”.
3. Click Start.

When the Multicast Service starts, it reads the STMulticastService64.ini configuration file. On Windows systems, this file is typically located in the "C:\Program Files\Borland\Multicast Service 6.4" folder. Options in this file tell the Multicast Service which Message Broker to connect to, and what TCP/IP port (endpoint) to accept client connections on. See "Configuring a Multicast Service" on page 23 for details on the contents of the STMulticastService64.ini file.

Stopping a Multicast Service

The process for stopping a Multicast Service is similar to that for a Message Broker.

To manually stop a Multicast Service:

1. Existing users of the Multicast Service should be notified before stopping the service.
2. On the computer where the Message broker is installed, choose Start > Settings > Control Panel > Administrative Tools > Services.
3. In the Service list box, select the service named "StarTeam Multicast Service 6.4".
4. Click Stop.

Running Message Broker and Multicast Service on Solaris

On Solaris platforms, the Message Broker and Multicast Service run as daemons and are incorporated into the init.d initialization and termination process. This means that the following files are placed in the /etc/init.d/ directory by the installer:

- STMB64: This file is the initialization/termination script for the Message Broker.
- STMS64: This file is the initialization/termination script for the Multicast Service.

These script files are linked to the run level 3 multi-user state by placing the following files in the /etc/rc3.d/ directory:

- S98stmb64: This file is a symbolic link to "/etc/init.d/STMB64", which is the Message Broker script. Because the file prefix is "S", the OS calls this script with the parameter "start" when the system is entering run level 3. When called with this parameter, the script will start the Message Broker. The number "98" is the startup sequence number, which controls the order in which scripts are executed. (The sequence number can be changed to another value by renaming the file; see the man pages for "init.d" for more information.)
- S99stms64: This file is a symbolic link to "/etc/init.d/STMS64", which is the Multicast Service script. Because the file prefix is "S", the OS calls this

script with the parameter “start” when the system is entering run level 3. When called with this parameter, the script will start the Multicast Service. The sequence number “99” ensures that the Multicast Service starts after the Message Broker.

- K98stmb64: This file is a symbolic link to “/etc/init.d/STMB64”, which is the Message Broker script. Because the file prefix is “K”, the OS calls this script with the parameter “stop” when the system is exiting run level 3. When called with this parameter, the script will terminate the Message Broker. During a shutdown, scripts are called in the order of descending sequence number. Hence, this script is normally called after the Multicast Service is terminated.
- K99stms64: This file is a symbolic link to “/etc/init.d/STMS64”, which is the Multicast Service script. Because the file prefix is “K”, the OS calls this script with the parameter “stop” when the system is exiting run level 3. When called with this parameter, the script will terminate the Multicast Service. During a shutdown, scripts are called in the order of descending sequence number. Hence, this script is normally called before the Message Broker is terminated.

As a result of these scripts, the Message Broker and Multicast Service are normally started and stopped automatically as the system initializes and shuts down.

If you need to manually start or stop the Message Broker or Multicast Service, you can do so by performing the following steps:

1. Log onto an account with root privileges.
2. Change directories to the folder containing the initialization/termination scripts. For example:

   ```
   cd /etc/init.d
   ```

3. Run the appropriate script with the parameter “start” to start the corresponding daemon, or “stop” to stop an existing daemon.
   - To start the Message Broker: STMB64 start
   - To stop the Message Broker: STMB64 stop
   - To start the Multicast Service: STMS64 start
   - To stop the Multicast Service: STMS64 stop

   Borland Software Corporation recommends starting the Message Broker before starting the Multicast Service, and stopping the Multicast Service before stopping the Message Broker.

Running Cache Agent

To run the Cache Agents successfully, start the applications you have installed in the following order: StarTeam Server (the server configuration starts the transmitters), StarTeamMPX Services, Root Cache Agent, and Remote Cache Agent.

Running Cache Agent On Windows

Cache Agents can be run as service or console applications. Up to 25 Root and Remote Cache Agents can run as services on the same computer. However, only the first one installed on a computer will be registered as a service. It is registered as manual service with the display name StarTeamMPX Cache Agent and an internal name of CacheAgentService. You will need to register the others.

Cache Agent is stopped automatically when the computer is shut down.

Running Cache Agent as a Service

If a Cache Agent is running as a service, you can start and stop it manually using the Windows Control Panel Services utility. Select Start > Settings > Control Panel > Administrative Tools > Services on Windows 2000 and 2003 to see the list of services.

Using the same utility, you can change the Cache Agent service to run automatically when you start Windows.

You can also control the Cache Agent service from the command line by running the CacheAgentService.exe. Use any of the following syntaxes:

```
CacheAgentService -start [ configFile ] [ -log logFile ] [ -verbose ]

CacheAgentService -register [ Manual | Auto] [ configFile ]
[ -dependson list ] [ -log logFile ] [ -name serviceName ] [ -verbose ]

CacheAgentService -unregister
```

Parameter	Description
configFile	The default configuration file is CacheAgentConfig.xml. If you use multiple configuration files, each one must specify unique values for CachePath ad RequestPort so that Cache Agent services do not interfere with each other’s operation.
-dependson *list*	Specifies service dependencies for the new Cache Agent service. The *list* must be a quoted, space-separated list of internal (not display) names of the services on which the Cache Agent service will depend. (A service’s internal name is its registry key within the Windows registry.) The most common dependency for a Cache Agent is to make it dependent on the Message Broker service running on the same machine. The Message Broker service’s display name is typically “StarTeam Message Broker 6.4”, but its internal name is typically “StarTeamMessageBroker6.4”. Consequently, to make a Cache Agent depend on the Message Broker service on the same computer you would use: -dependson "StarTeamMessageBroker6.4

Parameter	Description
-log *logFile*	Specifies a log file name other than the default, which is CacheAgentService.log. For more details about log file naming conventions, see “Log File” on page 60.
-name *serviceName*	Specifies the display name for the service in the Control Panel’s Services utility. The default is StarTeamMPX Cache Agent. This is **not** the internal name which defaults to CacheAgentService (or, if that is already in use, CacheAgentService2 through CacheAgentService25).
-register	Register the service with the specified start mode (Manual or Auto), optionally with a specific configuration file at startup.
-start	Starts the service, optionally with a specific configuration file.
-unregister	Removes a service. For example, if you change the Cache Agent from Manual to Auto, you would unregister it and reregister it. A service must be stopped before it can be unregistered.
-verbose	Causes another more detailed log to be generated. It defaults to CacheAgentService-debug.log. For more details about log file naming conventions, see “Log File” on page 60.

Examples

Below is an example command-line to register an auto-start Cache Agent service with the name “Prod1 Root CA”, dependent on the MPX Message Broker, with the default log file name, and verbose logging enabled:

```
CacheAgentService-registerAutoc:\CAConfigs\Prod1RootCA.xml-dependson
"StarTeamMessageBroker6.4" -name "Prod1 Root CA" -verbose
```

As with the register command, the default service name is “StarTeamMPX Cache Agent”. That means that the “name” parameter must be used when you unregister a service that has a non-default name. For example, to un-register the service used in the last example:

```
CacheAgentService -unregister -name "Prod1 Root CA"
```

Log File

When the Cache Agent service runs, it creates a log file in its installation folder. The default file name for the log is CacheAgentService.*locale*.log where *locale* is something like “en-US”.

If a file with that name already exists, it is renamed to include a time stamp:

```
CacheAgentService_YYYYMMDDhhmmss.location.log
```

Its log can be viewed from a browser by entering a URL using the following syntax:

```
http://host:port/log
```

host identifies the computer on which the Cache Agent is running

port provides its configured port number.

Running Cache Agent As a Console Application

CacheAgentApp.exe can be used instead of the service application. Use it when multiple Cache Agents operate on the same machine against different StarTeam server configurations.

For this scenario, each Cache Agent uses a different request port and different local cache paths.

running the CacheAgentApp.exe. Use the following syntax:

```
CacheAgentApp [ -c configFile ] [ -l logFile ] [ -v off | on ]
```

Parameter	Description
-c *configFile*	Starts the Cache Agent as a console application, optionally with a specific configuration file. The default configuration file is RootCAConfig.xml or RemoteCAConfig.xml, depending on the type of the Cache Agent.
-l *logFile*	Specifies a log file name other than the default, which is CacheAgentApp.log.
-v	Add more detail to the log. The settings are off or on. Off is the default.

Log File

When the Cache Agent service runs, it creates a log file in its installation folder. The default file name for the log is CacheAgentApp.*locale*.log where *locale* is something like “en-US”.

If a file with that name already exists, it is renamed to include a time stamp:

```
CacheAgentService_YYYYMMDDhhmmss.location.log
```

Its log can be viewed from a browser by entering:

```
http://host:port/log
```

host identifies the computer on which the Cache Agent is running

port provides its configured port number.

Running Cache Agent on Solaris

To run a Cache Agent on a Solaris system, use the following start command:

```
cacheagentapp -c RootCacheAgentConfig.xml -d start
```

To stop running a Cache Agent on a Solaris system, use the following stop command:

```
cacheagentapp -c RootCacheAgentConfig.xml -d stop
```

Chapter 8

Installing MPX Components on Windows

This chapter explains how to install the StarTeamMPX components on supported Windows systems. (See Appendix C, "StarTeamMPX System Requirements" for detailed platform requirements.) Each component can be separately installed using the steps described in the following sections.

Installing the Transmitters

The Event Transmitter and File Transmitter must be installed after the StarTeam Server has been installed. They must be installed on the same computer as the StarTeam Server. A single Event Transmitter and a single File Transmitter provide StarTeamMPX capabilities for all server configurations on the same computer.

Notes StarTeamMPX components are installed separately. Be sure to install the StarTeam Server prior to installing StarTeamMPX transmitters. The transmitters must be the same release and build number as the server. Directions for installing the StarTeam Server can be found in the *StarTeam Installation Guide*.

To install the transmitters:

1 On the computer where you wish to install the transmitters, log on as the administrator or as a user with administrative privileges.

2 Insert the Installation disc into your CD-ROM drive.

The *Borland StarTeam 2005 Release 2 CD Launcher* window should appear automatically.

3 If the window does not appear, use the following steps to display it:

   a Run Windows Explorer.

   b Click the drive letter corresponding to your CD-ROM drive.

   c Start the Setup program shown in the right panel (for example, double-click "setup" or "setup.exe").

4 Click the Install Products button.

   The window's content changes to list all the products on the CD.

5 Click the entry for Borland StarTeamMPX. The window's content changes to list all the MPX components.

6 Click the entry for Borland StarTeamMPX Transmitters.

7 Follow the instructions on the following screens until you reach the *MPX Options* dialog.

   If this is the first time the transmitters have been installed on this computer, the installer creates both Event Transmitter XML and File Transmitter template files using information from the *MPX Options* dialog. These files are stored in a subfolder of the installation folder named EventServices.

8 (Optional) Use the *MPX Options* dialog to make changes to the MPX profiles used by your server configurations.

   The local host name and IP address are provided at the top of this dialog for reference purposes.

   By default, both the Message Broker connection address for the unicast profile and the Multicast Service connection address for the multicast profile uses your local computer's IP address.

   Also by default, the initial profile to be used by clients is the unicast profile.

   a To specify a different Message Broker for Event Transmitter (and any clients that will use the unicast profile), change the proposed Message Broker connection address.

      This address must use the format:

      `tcp:`*`host`*`:`*`port`*

      `host` is the host name or IP address on which the Message Broker operates

      `port` is the TCP/IP port number (endpoint) with which the Message Broker accepts connections (5101 is the default).

Important When installing to a computer that has more than one NIC or multiple IP addresses, do not accept the default server address settings. You must manually enter the correct server IP address. When the installer detects multiple network addresses on the host computer, it sets up default profiles that contain a server-address property whose syntax is invalid (because it contains multiple addresses).

Tip You can enter the connection information for more than one Message Broker. In the event that the Event Transmitter (or a client using the unicast profile) is unable to connect with the first Message Broker in the list, it will attempt to connect with the next Message Broker listed. Separate the address for each Message Broker with a comma (,). For example:

```
tcp:HostA:5101,tcp:HostB:4999
```

b To specify a different Multicast Service, change the proposed Multicast Service connection address. Use the following format:

`pgm:`*`host`***`:tcp.`***`port`*

`host` is the host name or IP address on which the Multicast Service operates

`port` is the TCP/IP port number (endpoint) with which the Multicast Service accepts connections (5104 is the default).

Note that this is not the port with which multicast broadcasts occur; the multicast address is dynamically assigned by the Multicast Service.

If you do not intend to use a Multicast Service, you can ignore the Multicast Service connection address and delete the multicast profile after installation.

c To change the default client profile from a Message Broker (Unicast) profile to a Multicast Service (Multicast) profile, select the Multicast option button.

d Click Install and continue to follow the directions on the screen.

See Chapter 4, "Managing the Transmitters" for more information on creating and editing connection profiles.

Generating Transmitter XML Files

When the transmitters are installed on a StarTeam Server, the transmitter template files (MPXEventTransmitterTemplate.xml and FileTransmitterTemplate.xml) are installed in the EventServices folder, a subfolder of the server's installation folder.

When existing configurations are in place at the time of the installation, a configuration-specific set of transmitter XML files is created automatically for each existing configuration. The configuration-specific XML files (MPXEventTransmitter.xml and FileTransmitter.xml) are created by copying the XML template file to a configuration-specific subfolder of the EventServices folder. For example, the StarDraw sample server configuration's existence results in a subfolder of the EventServices folder named StarDraw.

When a new configuration is defined, a set of configuration-specific XML files may be generated automatically depending on how the configuration is created:

- If you create a new configuration by using the Server Administration utility, a configuration-specific MPXEventTransmitter.xml and FileTransmitter.xml will be created automatically. The utility does this by copying the current XML template files to the appropriate configuration-specific subfolder of the EventServices folder and removing "Template" from their names.
- If you create a new configuration by using StarTeam Server's command-line interface, no configuration-specific XML files are not created. If you want the new configuration to be MPX-enabled, you need to create the configuration-specific subfolder of the EventServices folder. The configuration subfolder's name must be the same as the configuration name (with the same casing on Solaris). Then manually copy the XML template files to the appropriate configuration-specific subfolder of the EventServices folder and remove "Template" from their names. Edit the new XML files as needed.

See the section "Understanding Connection Profiles" on page 33 for more information on configuration-specific and the XML template file.

Upgrading from Previous Releases

The general process for upgrading an existing transmitters to the new release is as follows:

1. Install the new release of StarTeam Server, as described in the *StarTeam Installation Guide*.
2. Upgrade each of your existing server configurations, as described in the *StarTeam Installation Guide*. (This will upgrade each server configuration's repository to the format for the new release.)
3. Install the StarTeamMPX Transmitters, as described earlier in this chapter.
4. Check the readme files for any upgrade issues.

Installing the Message Broker and Multicast Service

The Message Broker can be installed on the same computer where the StarTeam Server is installed, or it can be installed on a different computer. At least one Message Broker must be installed in your environment to provide unicast messaging services for an MPX-enabled StarTeam Server. As described in "Understanding Clouds" on page 15, you may want to install more than one Message Broker to meet your needs.

The Multicast Service is an optional service that can be used to reduce network demands in larger environments. If you decide to use the Multicast Service, you need to install only one Multicast Service in each local network community.

The Multicast Service must communicate with only one Message Broker, and you would normally install the Multicast Service on the same computer as that Message Broker. The Multicast Service must be connected to a Message Broker from StarTeamMPX 2005 or StarTeamMPX 2005 Release 2.

To install the Message Broker and Multicast Service on Windows:

1. On the computer where you wish to install the Message Broker, log on as the administrator or as a user with administrative privileges.
2. Insert the Installation CD into your CD-ROM drive.

   The *Borland StarTeam 2005 Release 2 CD Launcher* window should appear automatically.
3. If the window does not appear, use the following steps to display it:
   a. Run Windows Explorer.
   b. Click the drive letter corresponding to your CD-ROM drive.
   c. Start the Setup program shown in the right panel (for example, double-click “setup” or “setup.exe”).
4. Click the Install Products button.

   The window’s content changes to list all the products on the CD.
5. Click the entry for Borland StarTeamMPX.

   The window’s content changes to list all the MPX components.
6. Click the entry for Borland StarTeamMPX Services.

   This selection installs both the Message Broker and the Multicast Service, in the same directory.
7. Follow the instructions on the screen.

Because the Message Broker and the Multicast Service are installed as services, you can choose whether to install them as automatic or manual services.

Note that the Setup program assumes the Message Broker and the Multicast Service will use the default port numbers 5101 and 5104, respectively. If you want to use a different endpoint, edit the STMessageBroker64.ini and STMulticastService64.ini files after the setup program completes. For more information, see “Changing the Endpoint of a Message Broker” on page 22 and “Changing the Endpoint of a Multicast Service” on page 23.

Installing a Cache Agent

The following sections cover pre-installation issues and explain how to install a Cache Agent.

You need to install Cache Agent only once on any computer that will use one or more Cache Agents. You run as many instances of Cache Agent as you need, each with a different and appropriate XML file.

The default XML file for a Root Cache Agent is RootCAConfig.xml. The default XML file for a Remote Cache Agent is RemoteCAConfig.xml. As you install Cache Agent, you also set up an initial Cache Agent configuration file, so you must select either a Root or Remote Cache Agent during the installation. This does not stop you from creating another configuration file for the same or a different type of Cache Agent later on that same computer.

Before Installing a Cache Agent

You need to install the StarTeam Server, StarTeamMPX Transmitters, and StarTeamMPX Services before installing Cache Agents.

The StarTeam Server and StarTeamMPX Transmitters must have the same build numbers.

StarTeamMPX Services are the Message Broker and Multicast Service. The Root Cache Agent must have a Message Broker, but the Remote Cache Agents can use either Message Brokers or Multicast Services.

The Remote Cache Agents can use the same Message Broker as the Root Cache Agent, but multiple Message Brokers are suggested for distributed teams, especially over distances. In practice, Remote Cache Agents usually use remote Message Brokers. Because the Multicast Service requires greater configuration and network competence, the use of a Multicast Service is considered an advanced Cache Agent configuration.

Before you install a Cache Agent:

1. Install the Server. See the *StarTeam Installation Guide* for instructions.
2. Install the StarTeamMPX Transmitters.
3. After installing the transmitters, start each server configuration that you will use with Cache Agent. Doing this causes the File Transmitter to generate a CacheJournal.dat file for each configuration.

   The first time a server configuration uses the File Transmitter, the File Transmitter scans the server configuration’s vault cache and generates a new CacheJournal.dat file. This may take up to an hour or more as it computes the MD5 of each file in the cache; however, the server will be available as the file is built. File content transmission can begin when the cache scan is complete. If any errors occur, they are reported in the server’s log file.
4. Optional. Modify the MaxJournalAge parameter’s value in the FileTransmitter.xml file to represent the maximum number of days for which records will be kept within the Journal file. The pre-configured value is 180 days. Note that the Journal file is “trimmed” of expired records (those whose age exceeds the MaxJournalAge value) only when the Server process is started for the associated configuration.
5. Use StarTeamMPX Services to install appropriate Message Brokers (using the unicast option) and optionally Multicast Services (using the multicast option).

Installing a Cache Agent

Whether a Cache Agent operates as a Root Cache Agent or a Remote Cache Agent is determined solely by its configuration. The installation of the Cache Agent is similar in both cases except for consideration of the computer on which to install it:

- The Root Cache Agent requires access to the vault for the server configuration that it services. Consequently, it can be installed on the same computer as the StarTeam Server. Alternatively, if it can be installed on a separate computer to prevent the Root Cache Agent from competing for CPU or network I/O with the corresponding server configuration. However, this requires it to access the vault files and the CacheJournal.dat file via a shared network drive, so use this option only when a high-speed network file system is in place.
- Remote Cache Agents should be installed in each geographic location that can benefit from improved file check-out performance. One approach is to install a Cache Agent in each network environment that local users can access over a high-speed LAN. (Example: Install two Remote Cache Agents at headquarters, one each for engineering and QA, a third Remote Cache Agent at the Chicago office, and a fourth at the London office.) Another beneficial use of the Cache Agent is to install an instance on a computer dedicated to a check-out intensive application such as a build utility. There is no limit on the number of Cache Agents that can be installed throughout an enterprise. However, keep in mind that each Cache Agent requires access to a Message Broker, for which there is a maximum of 10 within a single messaging cloud. For more information about clouds, see “Understanding Clouds” on page 15.

To install a Root Cache Agent or Remote Cache Agent:

1 Run the Cache Agent installer and choose an installation folder. The default installation path is C:\Program Files\Borland\StarTeamMPX Cache Agent 2005 R2. Then choose Root Cache Agent or Remote Cache Agent installation.

2 For a Root Cache Agent installation, a dialog prompts you for the following information:

Message Broker Address: Enter the host name or address of the MPX Message Broker that the Root Cache Agent will use to communicate with other Cache Agents and clients. This value can optionally include the protocol prefix (“tcp:”) and port number suffix (“:1234”). The port number must be provided if the Message Broker is not using the default port of 5101. For example:

```
tcp:MBServer1:5123
```

Note that the Root Cache Agent should use a Message Broker for publish/subscribe communication, not a Multicast Service.

StarTeam Server Cache folder: Enter (or browse to) the folder path that contains the vault cache of the server configuration that the Cache Agent will serve. This path must be relative to the computer on which the Root Cache Agent is being installed.

Install optional Native-I Cache Auditor: Check this box is you would like to install the Native-I Cache Auditor. For a description of this application and a discussion of its importance, see “Understanding the Native-I Cache Auditor” on page 41.

For new installations of StarTeam Server, files will never be in Native-I format. Therefore, in this situation it is not necessary to install the Native-I Cache Auditor. However, if you are upgrading from a pre-2005 release of StarTeam Server and do not want to convert your archive files to Native-II format at the present time, you can select this option. Then the Native-I Cache Auditor can be used to monitor the Native-I cache for files that are added after Cache Agent is installed.

Native-I Cache Audit Installation Folder: If you check the box above, enter or browse for the location in which you wish to place the Native-I Cache Audit.

The default installation directory for the Native-I Cache Auditor is C:\Program Files\Borland\V1CacheAudit. You can accept this default or enter or browse for an alternate install directory.

3 For a Remote Cache Agent installation, you will be prompted for the following information:

Message Broker Address: Enter the host name or IP address of the MPX Message Broker that the Remote Cache Agent will use to receive file content messages and to communicate with other Cache Agents and clients. To use unicast messaging, a Remote Cache Agent should be configured to use a Message Broker and consequently the “tcp” style address. Alternatively, a Remote Cache Agent can use multicast messaging by entering in this field the address of a Multicast Service. In that case, a “pgm” style address should be entered. For example:

```
pgm:12.34.56.78:tcp.5104
```

This option is described in more detail in “Configuring a Remote Cache Agent” on page 42.

After Installing Cache Agent

Depending upon your environment, you may have to perform additional steps to use the Cache Agent:

For any Cache Agent:

- The Cache Agent installer installs both CacheAgentService.exe and CacheAgentApp.exe so you can run Cache Agent as either a service or a console application.
- CacheAgentService is configured as a “Manual” start service.You may want to change the start type to “Auto” so that the Cache Agent will start automatically thereafter. To change the start type to “Auto”, use the Control

Panel Services interface or unregister and re-register the service. See "Running Cache Agent On Windows" on page 59 for more information.

If you installed a Root Cache Agent:

- The default configuration file for a Root Cache Agent is RootCAConfig.xml. It may need to some modification. See "Configuring a Root Cache Agent" on page 42.
- The Native-I Cache Auditor (if installed and used) must be run as a console application. The installer will create a batch script called RunV1CacheAudit.bat to run the application. You must manually run this batch file from a console window. Add a shortcut to this batch script to your Program Files > Startup folder so that it starts each time you log on to Windows.

If you installed a Remote Cache Agent:

- The default configuration file for a Remote Cache Agent is RemoteCAConfig.xml. It needs some modification for your environment. It is configured with a sample configuration file that "tiers" the Cache Agent to the Root Cache Agent for the StarDraw sample configuration (even if it does not actually exist). See "Configuring a Remote Cache Agent" on page 42.

Uninstalling MPX Components

Each time you install a StarTeamMPX component, an Uninstall program shortcut is created for that component. The Uninstall programs enable you to remove the transmitters, the Message Broker, and/or Multicast Service from your computer. Your StarTeam Server installation, server configurations, and repositories are unaffected.

The uninstall program for the transmitters leaves the XML template files and each configuration's XML files intact. If you start the corresponding server configuration without deleting, moving, or renaming its MPX XML files, you see an error message ("Failed to load external event handler module…") in the configuration's log, indicating that the configuration is starting in non-MPX mode. If you leave an XML template or configuration-specific file in place, it will not be replaced by a subsequent installation of the transmitters.

The StarTeamMPX Services Uninstall programs leaves the STMessageBroker64.ini and STMulticastService64.ini files intact. If you later reinstall these services, these files will not be replaced by the new installation.

Uninstalling the Transmitters

To uninstall the transmitters:

1. From a computer on which the Server is installed, select Start > Programs > StarTeam > StarTeam Server *x* > StarTeam Server.
2. If any server configurations are running, shut them down.

3 Uninstall the transmitter by selecting Start > Programs > StarTeam > StarTeamMPX Transmitters 2005 Release 2 > Uninstall.
4 In the resulting confirmation dialog, click Yes.

Uninstalling the transmitter does not affect StarTeam Server. All server components remain installed.

Uninstalling the Message Broker and Multicast Service

To uninstall a Message Broker and Multicast Service:

1 From a computer on which the Server is installed, select Start > Programs > StarTeam > StarTeam Server *x* > StarTeam Server.
2 If any server configurations are running, shut them down. You can do this by clicking the Shut Down Local Server icon or selecting Actions > Shut down Server from the menu.
3 If the Message Broker or the Multicast Service is running, stop it using the Windows Control Panel Services utility. Select Start > Settings > Control Panel > Administrative Tools > Services on Windows 2000 and 2003 to see the list of services.
4 Uninstall the Message Broker and the Multicast Service by selecting Start > Programs > StarTeam > StarTeamMPX Services > Uninstall.
5 In the resulting confirmation dialog, click Yes to uninstall the Message Broker and the Multicast Service.

Uninstalling Cache Agent

To uninstall a Cache Agent:

1 If the Cache Agent is running, stop it. See "Running Cache Agent" on page 58 for details.
2 Uninstall the Cache Agent by selecting Start > Programs > StarTeam > StarTeamMPX Cache Agent 2005 Release 2 > Uninstall.
3 In the resulting confirmation dialog, click Yes.

Chapter

9

Installing MPX Components on Solaris

This chapter explains how to install the StarTeamMPX components on Sun Solaris (SunOS 5.8 or 5.9). See Appendix C, “StarTeamMPX System Requirements” for details on Solaris operating system requirements including required patches.

Planning considerations for StarTeamMPX components on Solaris are the same as those for Windows platforms. Installation steps, however, follow an analogous but different process for Solaris, as defined in the following sections.

Important If the installer for the Event Transmitter detects multiple network addresses on the host computer, the default profiles it sets up contain a server-address property whose syntax is invalid (multiple addresses). When installing to a computer that has more than one NIC or multiple IP addresses, do not accept the default server address settings. You must manually enter the correct server IP address.

Installing StarTeamMPX

StarTeamMPX component installations begin by running a setup script. You should install StarTeamMPX components immediately after installing StarTeam Server as the post-installation script should be run after the two have been installed.

To install StarTeamMPX:

1 Change directory to the location of your CD or downloaded files.

2 Then change directory to ./MPX.

3 Execute the initial setup script named “setup”.

```
# ./setup
```

The Setup script performs the pre-installation steps such as displaying the Borland license agreement.

4 The Setup script asks whether you wish to proceed with the command line (“cmd”) installation or the graphical user interface (“gui”) installation.

- To install StarTeamMPX Server components using the command line installation enter “cmd”.

  This starts the Solaris pkgadd utility, which allows you to install StarTeamMPX Server components: Event Transmitter, Message Broker, and Multicast Service. The Event Transmitter and File Transmitter are installed in the “/opt/starteamserver2005” directory and Message Broker, and Multicast Service are installed in “/opt/MessageBroker64” directory. Proceed to “Finalizing the StarTeam Server and MPX Installations” on page 76.

- To install the StarTeamMPX Server components using the Java installer, enter “gui”. The Java installer installs one or more StarTeamMPX Server components in a single process. Go to step 5.

5 On the *Welcome* screen, click Next.

The *Select Type of Install* screen opens.

6 Select the desired type of installation, and click Next.

Typical is recommended. Use it to install all the StarTeamMPX Server components (Event Transmitter, Message Broker, Multicast Service, and File Transmitter) on the local computer.

7 When the *Component Validation* screen is displayed, click Next.

8 After the selected components pass the compatibility tests, click Go To Next Panel in the resulting confirmation dialog.

9 On the resulting *Ready to Install* screen, click Install Now.

10 If you are installing the Multicast Service, the installer asks for the address of the Message Broker to which the Multicast Service will connect. This address is pre-filled with the IP address of the local computer and the default Message Broker port number (5101). Change this address if the Multicast Service will be using a Message Broker with a different address and/or port number, and click Next.

11 If you are installing the Event Transmitter, the installer displays a screen for configuring the unicast and multicast profiles that will be added to the XML template and configuration-specific files.

The choices on this screen are:

- Unicast profile address

  By default, this address is initialized with the address of the local computer and the default Message Broker port number (5101). Change this address if the unicast profile, which is used by the Event Transmitter and unicast StarTeam clients, should use a Message Broker with a different address and/or port number.

- Multicast profile address

  By default, this address is initialized with the address of the local computer and the default Multicast Service port number (5104). Change this address if the multicast profile, which is used by multicast StarTeam clients, should use a Multicast Service with a different address and/or port number.

- Default Client Profile

  Select the default profile that StarTeam clients should use: Unicast or Multicast.

  After specifying this information, click Next.

Note If the installer for the Event Transmitter detects multiple network addresses on the host computer, the default profiles it sets up will contain a server-address property whose syntax is invalid (multiple addresses). When installing to a computer that has more than one NIC or multiple IP addresses, do not accept the default server address settings. You must manually enter the correct server IP address.

12 On the *Important Information* screen, click Next.

Remember to execute the “post_install” script later, to completely finish the installation.

13 On the final *Installation Summary* screen, click Exit.

StarTeamMPX components are installed in the following locations:

- Event Transmitter and File Transmitter: The transmitters are always stored in the StarTeam Server installation directory (“/opt/starteamserver2005r2”).
- Event Transmitter and File Transmitter XML Template Files: These files are named MPXEventTransmitterTemplate.xml and FileTransmitterTemplate.xml. They are placed in the directory “/opt/starteamserver2005r2/EventServices”.
- Configuration-specific XML Files: Configuration-specific XML files are named MPXEventTransmitter.xml and FileTransmitter.xml. They are placed in a subfolder of the EventServices folder. The name of the subfolder matches the name of the configuration. So, for example, the MPXEventTransmitter.xml for a configuration called “Config1” is placed in the folder “/opt/starteamserver2005r2/EventServices/Config1”.

Finalizing the StarTeam Server and MPX Installations

After installing either or both StarTeam Server and MPX components, you must run a post installation script.

To complete the installation:

1 After the setup script exits, you must execute the “post_install” script:

```
# ./post_install
```

This script:

- Installs StarTeam Extensions into a subdirectory of the StarTeam Server installation directory.

  For details about StarTeam Extensions, see the *StarTeam Extensions User’s Guide*.
- Expands the StarDraw sample server configuration
- Installs the Sun JRE in a subdirectory of the StarTeam Server installation directory.
- Changes ownership of the installed StarTeam Server and StarTeamMPX components to the user account selected during setup.

  After the post_install script exits, the installation process is complete.

2 If you installed the server under an existing account, please source your .profile or .cshrc files, or log out and log in again to your account.

Installing StarTeamMPX Cache Agent

You install one Root Cache Agent per server configuration. Usually you install the Root Cache Agent on the same computer as the StarTeam Server. You install Remote Cache Agents where appropriate, so that data can be made more accessible to people at remote locations. Usually the people at the remote locations use the same MPX profile, but a different profile than used by local StarTeamMPX users. This section covers installing Cache Agent.

To install Cache Agent:

1 Change directory to the location of your CD or downloaded files.

2 Then change directory to ./CA.

3 Execute the initial setup script named “setup”.

```
# ./setup
```

The Setup script performs the pre-installation steps such as displaying the Borland license agreement.

4 The Setup script asks you to select a user account to be used to administer StarTeamMPX Cache Agent.

The default user name is cagent. Whatever you choose, this name should be different than the one selected earlier for StarTeam Server.

5 On the *Welcome* screen, click Next.

6 On the *License Agreement* screen, click the “I accept....” option button and then click Next.

7 On the *Choose Install Folder* screen:

 a Do one of the following:
  - Accept the default location for Cache Agent.
  - Click Choose... to browse for another location.

 b Click Next.

8 On the *Choose Link Folder* screen, you specify where the uninstall information for Cache Agent will go.

 a Do one of the following:
  - Accept the default location for link.
  - Click Choose... to browse for another location.

 b Click Next.

9 On the *Pre-Installation Summary* screen, review the summary and click Install.

10 When the *Install Complete* screen opens, click Done.

11 After the setup script exits, you must execute the “post_install” script for Cache Agent:

```
# ./post_install
```

Uninstalling StarTeamMPX Components

When the StarTeamMPX installer has completed successfully, it will create an Uninstall java class for uninstalling one or more StarTeamMPX components. This is called “uninstall_StarTeamMPX_Server.class”, and it is placed in the installation folder (for example, “/opt”).

To uninstall one or more StarTeamMPX components (other than Cache Agents):

1 Log on with root privileges and run the Uninstall program from the java command line. For example:

```
cd installationFolder
java -cp . uninstall_StarTeamMPX_Server
```

After the *Welcome* dialog, the *Ready to Uninstall* dialog appears.

2 Click Uninstall Now.

The Uninstaller removes all the StarTeamMPX components.

Note When the StarTeamMPX components are uninstalled, some files are deliberately not removed. For example, the STMessageBroker64.ini, STMulticastService64.ini, and transmitter XML files are not deleted. If you plan to delete all files in the /opt/starteamserver2005r2 and the /opt/MessageBroker64 folders (for example, by using the “rm -rf /opt/starteamserver2005r2” and “rm -rf /opt/MessageBroker64” commands), you should first back-up these files in case you want to reuse them later after a re-install.

To uninstall a Cache Agent:

1. Log on with root privileges and run the Uninstall program from the java command line. For example:

   ```
   cd installationFolder
   java -cp . Uninstall
   ```

   After the *Welcome* dialog, the *Ready to Uninstall* dialog appears.

2. Click Uninstall Now.

   The Uninstaller removes all the Cache Agent.

Appendix

A

Server Log Entries

Each time you start a server configuration, it creates a new server log file to record all activity. If a server log file already exists, the existing file is renamed before the new file is created.

The server log file has been renamed to reflect your language environment. For example, a Server.en-US.log file indicates the US English language. Note that even if your language is not US English, a Server.en-US.log file will always be created for use by Borland support personnel.

This section shows sample StarTeamMPX-related messages that may appear in a server log file. These messages are typically prefixed with “StarTeamMPX”.

Note Each message written to the server log file includes a line number, and a date-time stamp. This information has been omitted from the following sample for the purposes of clarity.

Start-Up Messages

When you start a server configuration, the system records all start-up messages in the server log file. When you start a StarTeamMPX configuration, the server log file also records start-up information for the Event Transmitter. The following sample shows the typical entries that are made in the server log file when you start an MPX-enabled server configuration. (The StarTeamMPX-specific messages appear in boldface type.)

```
*********Starting boot log*********
StarTeam Server version: X.X.XXX
```

```
Microsoft Windows 2000 Service Pack 4 (Build 2195)
ODBC driver version: SQL Server 03.81.9041
Server database opened.
Server configuration database opened
The security manager loaded.
Server component loaded: AuditComponent
Server component loaded: ChangeRequestComponent
Server component loaded: FileComponent
Server component loaded: RequirementComponent
Server component loaded: TaskComponent
Server component loaded: TopicComponent
The project manager loaded.
The logon manager loaded.
StarTeamMPX: Connecting to project <starteam> on <tcp:YOURSERVER:5101>
RTserver
StarTeamMPX: Using tcp protocol
StarTeamMPX: Message from RTserver: Connection established.
StarTeamMPX: Start subscribing to subject
</StarTeamServer_be5ee3b0-c719-49c6-a1a1-f493764a03f5>
Loaded asynchronous handler StarTeamMPX Transmitter in Module C:\Program
Files\Borland\StarTeam Server X\
xmitt.dll.
5 named user licenses will be granted.
TCP/IP (Socket) protocol registered with the endpoint 49201
Server successfully initialized.
```

Reconnect Messages

If an MPX-enabled server configuration loses the connection to its Message Broker, it records that event in the server log file. In most circumstances, the server configuration attempts to re-establish communication with the Message Broker. The number of retries made and the time between retries depends on the parameters set in the Event Transmitter XML file's server default profile.

The following sample shows the typical entries that are made in the server log file when a StarTeamMPX configuration loses and then reconnects to its Message Broker.

```
StarTeamMPX: Lost connection to RTserver: error code = 10
StarTeamMPX: Waiting before reconnecting
StarTeamMPX: Attempting to reconnect to RTserver
StarTeamMPX:Connectingtoproject<YourProject>on<tcp:YOURSERVER:5101>
RTserver
StarTeamMPX: Using tcp protocol
StarTeamMPX: Could not connect to <tcp:YOURSERVER:5101> RTserver
StarTeamMPX: Skipping starting <tcp:YOURSERVER:5101> RTserver
StarTeamMPX: Cannot reconnect to RTserver
   .
   .
   .
```

```
StarTeamMPX: Attempting to reconnect to RTserver
StarTeamMPX:Connectingtoproject<YourProject>on<tcp:YOURSERVER:5101>
RTserver
StarTeamMPX: Using tcp protocol
StarTeamMPX: Message from RTserver: Connection established.
StarTeamMPX: Start subscribing to subject
</StarTeamServer_be5ee3b0-c719-49c6-a1a1-f493764a03f5> again
```

Appendix

B

Troubleshooting StarTeamMPX

To determine whether your StarTeamMPX system is operating correctly, you can perform the following steps. This chapter assumes you are troubleshooting on a Windows operating system.

To check your StarTeamMPX system:

1. Review the following configuration files to ensure that the server addresses and endpoints are correct:
   - MPXEventTransmitter.xml (for the Event Transmitter installed for each StarTeam Server)
   - FileTransmitter.xml (for the File Transmitter installed for each StarTeam Server)
   - STMessageBroker64.ini (for each Message Broker)
   - STMulticastService64.ini (for each Multicast Service)
   - RootCAConfig.xml (for each Root Cache Agent)
   - RemoteCAConfig.xml (for each Remote Cache Agent)
2. If they are not already running, start the Message Brokers and Multicast Services.
3. For each Message Broker you start, start an MPX-enabled server configuration that communicates with that Message Broker.
4. For each server configuration you start, review its server log file.

   If the Event Transmitter has any problems connecting to the Message Broker, the error messages will be written to the server log file, which is located in the root folder of the server configuration's repository.

The server log file is located in the server configuration's repository folder.

5. Start a client and ensure that support for the StarTeamMPX is enabled for your workstation. See Chapter 6, "Configuring Clients" for details.
6. Enable client StarTeamMPX options:
   a. In your client, select the Tools > Personal Options command.
   b. In the resulting *Personal Options* dialog, select appropriate options on the *Workspace* and *StarTeamMPX* tabs.

   Only the StarTeam Windows client has MPX options on the *Workspace* tab. This option allows the StarTeam log file to include MPX information. The lot file can be viewed at any time by selecting the Tools > StarTeam Log command.

   Both the Windows and Cross-Platform clients have settings for enabling and disabling MPX.

   Only the Cross-Platform client and IDEs based on StarTeam Cross-Platform and .NET components have options for Cache Agent.

   c. Click OK.

   Test these settings.
7. In your client, open a view from an MPX-enabled server configurations. If MPX is enabled in both the client and the server configuration, a yellow lightning bolt appears in the status bar.

Diagnosing Message Broker and Multicast Service

To assist with diagnosing problems, both the Message Broker and the Multicast Service can generate logging information. To enable diagnostic tracing for the Message Broker, add the following lines to the STMessageBroker64.ini file:

```
setopt trace_file c:\temp\mbtrace.txt
setopt trace_flags timestamp
setopt trace_level debug
```

To enable diagnostic tracing for the Multicast Service, add similar lines to the STMulticastService64.ini file, but use a different file name for the trace_file option.

After adding these lines and saving the corresponding file, restart the corresponding service to begin generation of the diagnostic messages. The messages are written to the specified trace file, prefixed with time stamp information.

Note that there may be some delay between the time a trace message is generated and when it is written to the log file. Consequently, the most recent trace messages may not appear until the corresponding service is stopped, allowing the file to be closed.

Appendix

C

StarTeamMPX System Requirements

StarTeamMPX transmitters have the same system requirements as the StarTeam Server because they are part of that system. For details, see the *StarTeam Installation Guide*. This appendix covers Message Broker, Multicast Service, and Cache Agent.

Message Broker and Multicast Service System Requirements

Operating System	Hardware	Other Software
One of the following: ■ Microsoft Windows 2000 with Service Pack 3 or higher ■ Microsoft Windows 2003 Server	Computer with a 400 MHz Pentium Pro processor and 128 MB of RAM	■ Adobe Acrobat Reader 4.0 or higher
Sun Solaris (SunOS 5.8 or 5.9)	Sun Ultra-250 (Sun Ultra-4 recommended) with: ■ 512 MB memory ■ 8 GB disk space ■ 2 GB swap space	■ TCP/IP (Sockets) ■ SunOS Patches needed for StarTeam Server for Solaris ■ Adobe Acrobat Reader 4.0 or higher

Cache Agent System Requirements

These requirements should be sufficient for teams that have 50 to 100 members. See the paragraphs below the table for suggestions for different-sized teams.

Operating System	Hardware	Other Software
One of the following: ■ Microsoft Windows 2000 with Service Pack 3 or higher ■ Microsoft Windows 2003 Server	■ 1 CPU P4 1Ghz or better ■ 256MB memory or better ■ Disk appropriate for size of cache; fast disk preferable but not mandatory ■ 100Mbit NIC or better	■ Adobe Acrobat Reader 4.0 or higher
Sun Solaris (SunOS 5.8 or 5.9)	■ Dual CPU UltraSpark III ■ 800 MHz ■ 512 MB memory ■ Disk appropriate for size of cache; fast disk preferable but not mandatory ■ 100Mbit NIC or better	■ TCP/IP (Sockets) ■ SunOS Patches needed for StarTeam Server for Solaris ■ Adobe Acrobat Reader 4.0 or higher

On a Windows system, a smaller computer can be used for smaller teams. For example, fewer than 50 developers on a Windows system can get by with 128MB memory and a slower CPU. A faster CPU (maybe even dual CPUs) should be used for larger teams (more than 200 users).

On a Solaris system, a smaller computer can be used for smaller teams. For example, fewer than 50 developers on a Solaris system can get by with 256MB memory and a slower CPU. A faster CPU (maybe even quad CPUs) should be used for larger teams (more than 200 users).

Index