

IBM WebSphere InterChange Server



# Benchmarking Guide

*Version 4.3.0*



IBM WebSphere InterChange Server



# Benchmarking Guide

*Version 4.3.0*

**Note!**

Before using this information and the product it supports, read the information in Chapter 6, "Notices," on page 41.

**30September2004**

This edition of this document applies to IBM WebSphere InterChange Server (5724-178), version 4.3.0, and to all subsequent releases and modifications until otherwise indicated in new editions.

To send us your comments about this document, e-mail [doc-comments@us.ibm.com](mailto:doc-comments@us.ibm.com). We look forward to hearing from you.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© Copyright International Business Machines Corporation 2002, 2004. All rights reserved.

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

# Contents

<b>About this document</b> . . . . .	<b>v</b>
Audience . . . . .	v
Related documents . . . . .	v
Typographic conventions . . . . .	vi
<b>New in This Release</b> . . . . .	<b>vii</b>
New in Release 4.3 . . . . .	vii
New in Release 4.2.2 . . . . .	vii
New in Release 4.1.1 . . . . .	vii
New in Release 4.1.0 . . . . .	vii
<b>Chapter 1. Overview of benchmarking IBM WebSphere InterChange Server.</b> . . . . .	<b>1</b>
About benchmarks . . . . .	1
About IBM WebSphere InterChange Server benchmarks . . . . .	1
<b>Chapter 2. Working with benchmark definitions.</b> . . . . .	<b>7</b>
Starting the benchmark wizard . . . . .	7
Creating a new benchmark . . . . .	8
Editing a benchmark definition . . . . .	14
Deleting a benchmark definition . . . . .	15
Generating workload to a file . . . . .	16
Viewing benchmark results . . . . .	17
<b>Chapter 3. Performing IBM WebSphere InterChange Server benchmarks</b> . . . . .	<b>19</b>
Back up the IBM WebSphere InterChange Server system . . . . .	19
Modifying IBM WebSphere InterChange Server content if necessary . . . . .	19
Prepare sample data . . . . .	21
Preparing a benchmark definition . . . . .	24
Creating client emulator startup scripts . . . . .	25
Perform a benchmark . . . . .	28
Clear the system. . . . .	30
Restore the backed-up repository . . . . .	33
Hints on benchmarking . . . . .	33
<b>Chapter 4. Examples.</b> . . . . .	<b>35</b>
Using repos_copy . . . . .	35
Using the jar program . . . . .	36
Import file system components . . . . .	36
<b>Chapter 5. Troubleshooting IBM WebSphere InterChange Server benchmarks</b> . . . . .	<b>39</b>
File of input data not in location specified . . . . .	39
File of bad input data . . . . .	39
<b>Chapter 6. Notices.</b> . . . . .	<b>41</b>
Programming interface information . . . . .	42
Trademarks and service marks . . . . .	43



---

## About this document

IBM<sup>R</sup> WebSphere<sup>R</sup> InterChange Server and its associated toolset are used with IBM WebSphere Business Integration adapters to provide business process integration and connectivity among leading e-business technologies and enterprise applications.

This document describes the tools and tasks used to benchmark the IBM WebSphere InterChange Server system. It is designed to explain the benchmark user interface and functionality, but it does not cover how to interpret a benchmark and modify IBM WebSphere InterChange Server content in response to such an interpretation.

---

## Audience

This document is designed for IBM WebSphere InterChange Server developers who need to benchmark the system. To use the benchmarking tools, you should be very experienced in both developing and administering IBM WebSphere InterChange Server systems.

---

## Related documents

The complete set of documentation available with this product describes the features and components common to all WebSphere InterChange Server installations, and includes reference material on specific components.

You can install the documentation from the following sites:

- For InterChange Server documentation:  
<http://www.ibm.com/websphere/integration/wicserver/infocenter>
- For collaboration documentation:  
<http://www.ibm.com/websphere/integration/wbicollaborations/infocenter>
- For WebSphere Business Integration Adapters documentation:  
<http://www.ibm.com/websphere/integration/wbiadapters/infocenter>

These sites contain simple directions for downloading, installing, and viewing the documentation.

**Note:** Important information about this product might be available in Technical Support Technotes and Flashes issued after this document was published. These can be found on the WebSphere Business Integration Support Web site, <http://www.ibm.com/software/integration/websphere/support/>. Select the component area of interest and browse the Technotes and Flashes sections. Additional information might also be available in IBM Redbooks at <http://www.redbooks.ibm.com/>.

Before using this document, you should have the deep technical knowledge that comes with years of implementing and administering IBM WebSphere InterChange Server systems. You should have read virtually every document in the IBM WebSphere InterChange Server documentation set, with the exception of some connector and collaboration-specific documents. This document directly references

the *Technical Introduction to IBM WebSphere InterChange Server*, the *System Installation Guide for Windows*, the *System Installation Guide for UNIX*, and the *System Administration Guide*.

---

## Typographic conventions

This document uses the following conventions:

---

<code>courier font</code>	Indicates a literal value, such as a command name, file name, information that you type, or information that the system prints on the screen.
<b>bold</b>	Indicates a GUI element.
<i>italic, italic</i>	Indicates a new term the first time that it appears, a variable name, or a cross-reference.
<i>blue text</i>	Blue text, which is visible only when you view the manual online, indicates a cross-reference hyperlink. Click any blue text to jump to the object of the reference.
{ }	In a syntax line, curly braces surround a set of options from which you must choose one and only one.
[ ]	In a syntax line, square brackets surround an optional parameter.
...	In a syntax line, ellipses indicate a repetition of the previous parameter. For example, <code>option[,...]</code> means that you can enter multiple, comma-separated options.
< >	In a naming convention, angle brackets surround individual elements of a name to distinguish them from each other, as in <code>&lt;server_name&gt;&lt;connector_name&gt;tmp.log</code> .
/, \	In this document, backslashes (\) are used as the convention for directory paths. For UNIX installations, substitute slashes (/) for backslashes. All IBM WebSphere InterChange Server product path names are relative to the directory where the IBM WebSphere InterChange Server product is installed on your system.
<b>UNIX:/Windows:</b>	Paragraphs beginning with either of these indicate notes listing operating system differences.
<code>%text%</code> and <code>\$text</code>	Text within percent (%) signs indicates the value of the Windows text system variable or user variable. The equivalent notation in a UNIX environment is <code>\$text</code> , indicating the value of the <code>text</code> UNIX environment variable.

---



---

## **New in This Release**

This chapter describes the following new features of IBM WebSphere InterChange Server, which are covered in this document:

---

### **New in Release 4.3**

Screen captures and step instructions were updated in this manual for WebSphere InterChange Server version 4.3 release, to reflect changes to the user interface.

---

### **New in Release 4.2.2**

Terminology, product names, and copyright information were updated in this manual for WebSphere InterChange Server version 4.2.2 release.

---

### **New in Release 4.1.1**

Updated in March, 2003. The “CrossWorlds” name is no longer used to describe an entire system or to modify the names of components or tools, which are otherwise mostly the same as before. For example “CrossWorlds System Manager” is now “System Manager,” and “CrossWorlds InterChange Server” is now “WebSphere InterChange Server.”

This product has been internationalized.

---

### **New in Release 4.1.0**

The changes made in IBM WebSphere InterChange Server 4.1.0 do not affect the content of this document.



---

# Chapter 1. Overview of benchmarking IBM WebSphere InterChange Server

This chapter addresses some of the fundamental concepts of benchmarking IBM WebSphere InterChange Server. It discusses why you might want to run a benchmark, what happens when a benchmark runs, and what the results of running a benchmark are.

Topics included in this chapter are:

- “About benchmarks”
- “About IBM WebSphere InterChange Server benchmarks”

---

## About benchmarks

A benchmark is a test that measures the performance of a system or subsystem on a well-defined task or set of tasks. A benchmark for a graphics card, for instance, might test how many frames the card can draw per second; a benchmark for a Central Processing Unit (CPU) might test how many computations the CPU can perform per second.

Software benchmarks typically test how many transactions an application can process in a specified time frame. The Transaction Processing Council’s TPC-C benchmark, for instance, is a common test used to calculate the number of orders processed by a database in a minute.

---

## About IBM WebSphere InterChange Server benchmarks

The IBM WebSphere InterChange Server benchmarking tool enables you to test various WebSphere InterChange Server components, interfaces, and systems to measure their throughput. Although the results of a benchmark cannot precisely predict the throughput in a production environment, they can give a useful estimate.

**Important:** The benchmarking feature is designed to be used in a development or testing environment only. Running a benchmark adds data to cross-reference tables, and performing the required clean-up actions eliminates data from both work-in-progress tables and persistent messaging queues. It is critical that neither of those things happen in a production environment, and that the benchmarking feature therefore be used only in development or testing environments.

## Benchmark terminology

In order for a benchmark to be useful, the terms used to describe it and the task or set of tasks that it measures must be very well defined. Although the terms and tasks involved in benchmarking some technologies (such as graphics cards, CPUs, and databases) are well-defined because of the maturity of those markets and because of established standards, there are no universally accepted terms and tasks within the business process integration market. The following sections establish a set of definitions that are central to benchmarking the system.

## **Unit of work**

A unit of work is a basic, complete and countable piece of work. The different types of WebSphere InterChange Server benchmarks evaluate different types of interfaces or components, so the definition of a unit of work varies with the type of benchmark. For one type of benchmark, a unit of work may consist of the completion of an entire business process where a business object request is submitted by a source connector, processed by a collaboration, processed by any destination connectors, and returned to the collaboration, with any requisite transformations and other operations in between. For another type of benchmark, a unit of work may consist only of the processing of a business object by a connector. The different types of benchmarks and their units of work are described in the section “Types of benchmarks” on page 2.

## **Transaction**

A transaction is one execution of a unit of work.

## **Response time**

Response time is the amount of time it takes during a benchmark for the component or interface to complete a transaction.

## **Throughput**

Throughput is the number of transactions completed in a unit of time. This figure is typically expressed in terms of the number of business objects processed per unit of time (such as second, minute, or hour).

## **Workload**

The service demand placed on InterChange Server by the connector agents and access framework in the form of business object requests.

## **Interface**

A set of software components that work together to automate a business process. The components may be IBM WebSphere InterChange Server components, such as connectors, maps, collaborations, and so forth, and may be external, such as servlets, triggers, and scripts.

# **Types of benchmarks**

There are six types of WebSphere InterChange Server benchmarks available to profile the performance of particular components, interfaces, or systems.

## **Business Object Throughput**

The Business Object Throughput benchmark measures the throughput of a connector as it manages business objects within the system.

A unit of work for this benchmark consists of the transmission of the business object between the connector controller and connector agent across the transport protocol and the transformation of the business object through mapping in one direction.

## **Agent**

The Agent benchmark measures the interaction between a connector agent and the application it communicates with.

A unit of work for this benchmark comprises the agent posting a request to the application (whereupon the application performs whatever action is dictated by its

design and the metadata of the event, such as invoking an API to respond to operations such as create, update, delete, retrieve, and so forth) and receiving the response from the application.

### **Collaboration Throughput**

The Collaboration Throughput benchmark measures the number of objects processed within a unit of time by a collaboration.

A unit of work for this benchmark begins when a business object is delivered asynchronously to a collaboration and ends when the collaboration processes the response business object returned by the destination application.

### **Access Throughput**

The Access Throughput benchmark measures the system throughput for an IBM interface that is triggered by an external process (such as a web servlet) making a synchronous direct call.

A unit of work for this benchmark is identical to that for a Collaboration Throughput benchmark, except in that the collaboration is invoked through the Server Access Interface rather than through a connector controller.

### **Access Response Time**

The Access Response Time benchmark measures the throughput for an external process to make a request of a collaboration and receive the response returned by it.

A unit of work for this benchmark is identical to that for an Access Throughput benchmark, though it also includes the time taken to return the response to the access client.

### **Business Process Throughput**

The Business Process Throughput benchmark measures the throughput of the entire system; it might include any number of connectors, collaborations, and collaboration groups.

## **How benchmarks can be useful**

Benchmarks can be very useful to compare the throughput of two systems that are identical except for a single variable. A CPU benchmark, for instance, might compare the performance of CPUs from competing manufacturers by installing the CPUs on identical computers, giving them the same kind of workload, and testing how many instructions each can process in the same amount of time. Table 1 shows the various variables that figure in IBM's benchmarks, and how they change to achieve different measurements.

*Table 1. Benchmark variables*

<b>Product</b>	<b>Release</b>	<b>Workload</b>	<b>Number of interfaces</b>	<b>Available resources</b>
Different	N/A	Same	Same	Same
Same	Different	Same	Same	Same
Same	Same	Different	Same	Same
Same	Same	Same	Different	Same
Same	Same	Same	Same	Different

By changing a single variable and keeping the others the same, an IBM benchmark can be used to:

- Compare the throughput of systems running competing products with the same workload across the same interfaces and using the same available resources  
A benchmark is useful in this case to compare WebSphere InterChange Server with integration software developed by other software vendors.
- Compare the throughput of systems running different releases of the same product with the same workload across the same interfaces and using the same available resources  
A benchmark is useful in this case to determine the gain in throughput achieved by upgrading from one version of WebSphere InterChange Server to another.
- Compare the throughput of systems running the same release of the same product with a different workload across the same interfaces and using the same available resources  
A benchmark is useful in this case to determine the impact on the throughput of the system if the number of transactions processed by an interface increases.
- Compare the throughput of systems running the same release of the same product with the same workload across existing interfaces and the same available resources, while introducing additional interfaces with their own workloads  
A benchmark is useful in this case to determine if there is an impact on the throughput of the system if new interfaces are added to those that are already running.
- Compare the throughput of systems running the same release of the same product with the same workload across the same interfaces while increasing the available resources  
A benchmark is useful in this case to determine if the throughput of the system improves significantly by tuning performance or investing in new hardware.

## Benchmark core concepts

The basic concepts covered in this section must be understood to configure and run IBM WebSphere InterChange Server benchmarks.

### Benchmark component roles

Different WebSphere InterChange Server components (such as collaborations, connectors, and so on) fill roles within a benchmark. The roles are described in the following subsections.

**Participant:** A participant is any component that is required in a benchmark.

You add components to a benchmark as participants when you define the benchmark. Table 2 on page 5 shows which components are participants in each benchmark type.

A benchmark executes after it has been defined, InterChange Server has been restarted, and all of its participants have started up. Some participants are started automatically when InterChange Server starts (such as collaborations), but others must be started by you (such as connectors or the access client).

A component, besides being a participant in a benchmark, may also serve in other roles.

**Coordinator:** A coordinator is a component that keeps track of how many benchmark participants have started up so that it can initiate the benchmark, and shuts down all of the participants when the benchmark completes.

One benchmark participant is automatically chosen by the system to be a coordinator when a benchmark is created. Table 2 shows which type of components can be the coordinator for each benchmark type.

**Sample provider:** A sample provider periodically profiles characteristics of the system throughput during the execution of a benchmark. You can specify how long the sample provider waits during the benchmark before it starts taking the samples, and you can specify how many samples are taken during the course of the benchmark. When the benchmark finishes, the sample provider analyzes the collection of samples it took and presents an overall profile of the benchmark performance.

One benchmark participant is automatically chosen by the system to be a sample provider when a benchmark is created. Table 2 shows which type of components can be a sample provider for each benchmark type.

**Workload generator:** A workload generator provides the sample business objects that are processed during the benchmark.

You configure components as workload generators when you define the benchmark. The data used by the workload generator to provide the sample business objects can be generated by the system or supplied by you. Table 2 shows which type of components can be workload generators for each benchmark type.

*Table 2. Benchmark components*

Benchmark Type	Participants	Coordinator	Sample provider	Workload generator
Business Object Throughput (AppToGeneric directionality)	All user-selected connectors (agents and controllers)	One connector controller	Controllers of all participating connectors	All participant connectors
Business Object Throughput (GenericToApp directionality)	All user-selected connectors (agents and controllers)	One connector controller	Agents of all participating connectors	All participant connectors
Agent	One connector (controller and agent)	Participating connector controller	Participating connector agent	Participating connector agent
Collaboration Throughput	One collaboration and all connectors bound to it	Collaboration	Collaboration	One connector bound to a triggering port of the collaboration
Access Throughput	All user-selected collaborations, all connectors bound to them, and the access client	Collaboration	Access client	Access client
Access Response Time	All user-selected collaborations, all connectors bound to them, and the access client	Collaboration	Access client	Access client

Table 2. Benchmark components (continued)

Benchmark Type	Participants	Coordinator	Sample provider	Workload generator
Business Process Throughput	All user-selected collaborations and all connectors bound to them	Collaboration	All participating collaborations	All source connectors

### Exclusivity in benchmark participants

Although multiple benchmarks can be defined, no IBM component can be a participant in multiple benchmarks at the same time. If a component must participate in multiple benchmarks, then one benchmark must be defined, performed, and then deleted so that the next benchmark can go through the same series of steps.

### Benchmark statistics

A benchmark produces the following statistics:

- Maximum Throughput  
The highest observed throughput for the benchmark
  - Minimum Throughput  
The minimum observed throughput for the benchmark
- Note:** If the Java Virtual Machine (JVM) performs garbage collection during execution of the benchmark, it might prevent other activity from occurring in the system, and cause the minimum throughput to be reported as 0.
- 90th Percentile  
The 90th percentile figure is the time in milliseconds that 90% of the transactions surpassed
  - Average  
The arithmetic mean of all the samples collected throughout the benchmark
  - Sampling Interval  
The amount of time between taking samples



---

## Chapter 2. Working with benchmark definitions

The task of performing a benchmark includes many steps, and is covered in Chapter 3, “Performing IBM WebSphere InterChange Server benchmarks,” on page 19. One of the major subtasks is configuring a definition for the benchmark you want to perform. This chapter details the tasks involved in preparing benchmark definitions, and describes the benchmark wizard that you use to work with them.

The benchmark wizard uses a series of screens to collect information about the components or interfaces that you want to benchmark. The screens are mostly the same for all six types of benchmarks and for all four main benchmark configuration actions: creating a new benchmark, editing an existing benchmark, loading a benchmark from a file, and generating workload to a file.

All of the individual wizard screens are described in detail in the section “Creating a new benchmark” on page 8. For the other sections, such as editing or deleting benchmark definitions, where steps are identical to those for creating a new benchmark, you are directed to “Creating a new benchmark” on page 8. Screens that are unique to the other tasks or are different for the other tasks, however, are described in the particular section for that task.

Topics included in this chapter include:

- “Starting the benchmark wizard”
- “Creating a new benchmark” on page 8
- “Editing a benchmark definition” on page 14
- “Starting the benchmark wizard”
- “Deleting a benchmark definition” on page 15
- “Generating workload to a file” on page 16
- “Viewing benchmark results” on page 17

---

### Starting the benchmark wizard

#### Launching the benchmark wizard

To launch the benchmark wizard, right click **Benchmark** under **Integration Component Libraries** -> **Samples\_ICL** in the IBM System Manager, and select **Benchmark setup**. Select either **Configure a new benchmark**, **Edit an existing benchmark** or **Generate a workload to a file**.

The wizard starts and presents the Benchmark key code screen, where you can enter a key code to continue after reading the warning.

#### Entering the key code

The first screen displayed by the benchmark wizard presents a warning that performing a benchmark can result in loss or corruption of event management and relationship data, and that you should only perform a benchmark when you understand those facts and have taken the necessary backup steps as described in this document. It requires you to contact Technical Support to obtain a key code in order to proceed further in the wizard.

Enter the code in the **Key Code** field and click **Next**.

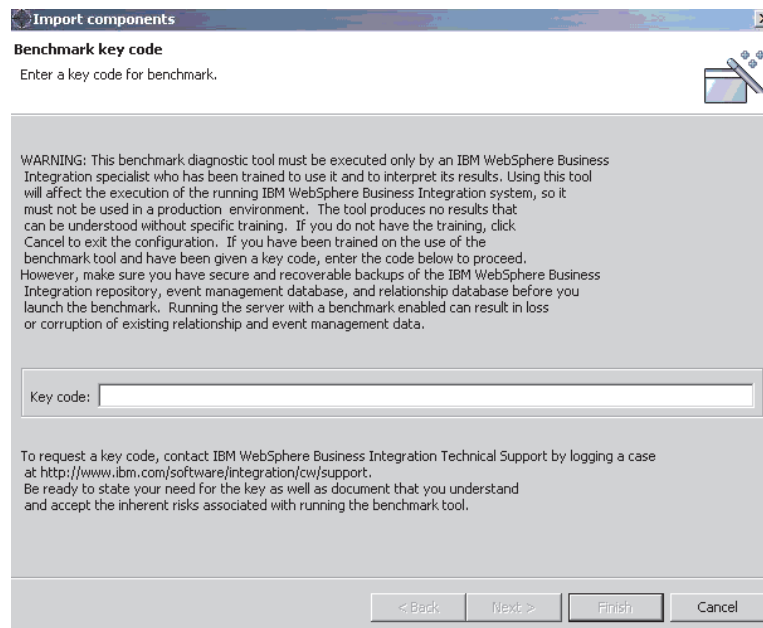


Figure 1. Warning screen

If you choose to configure a new benchmark, the wizard presents the General settings screen. If you choose to edit a benchmark or generate a workload to a file, the wizard presents the Select a benchmark screen, where you can select a benchmark to edit or a benchmark to use to generate a workload to a file.

---

## Creating a new benchmark

Launch the benchmark wizard as described in “Starting the benchmark wizard” on page 7, and choose **Configure a new benchmark**. The General settings screen is displayed.

### Configuring general settings

In the General Settings screen you configure the basic properties of the benchmark definition.

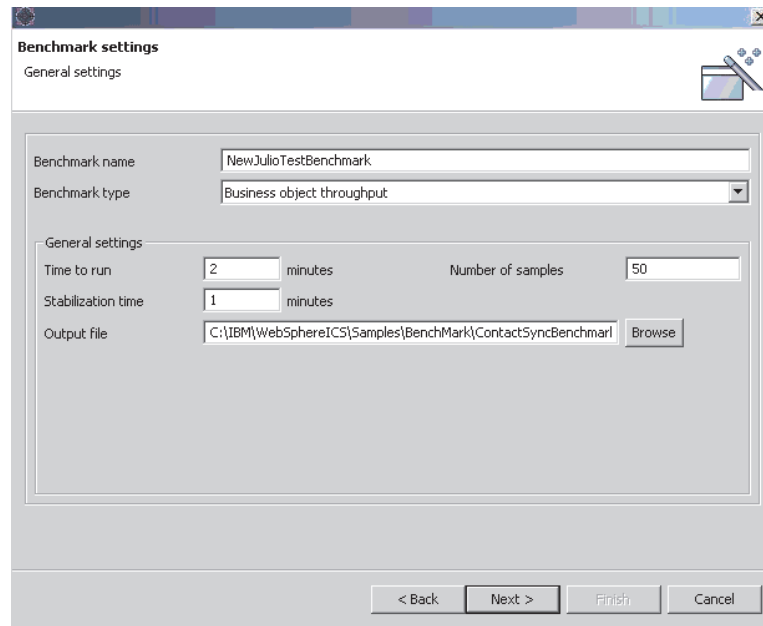


Figure 2. General Settings screen

1. Type a value in the **Benchmark Name** field.
2. Click the desired value from the **Benchmark type** list.
3. Type a number in the **Time to run** field to specify the number of minutes the benchmark executes.
4. Type a number in the **Number of samples** field to specify how many samples should be taken during the course of the benchmark's execution.
5. Type a number in the **Stabilization time** field to specify the number of minutes the sample provider waits before it begins to take samples.
6. (optional) If you want the results of the benchmark execution to be written to a file, enter the path and name of a file in the **Output File** field, or click the browse button to the right of the field to navigate to a file. If the field is left blank, then the results are written to the logging destination of WebSphere InterChange Server.
7. Click **Next** after specifying the proper values to advance to the next screen.

## Configuring benchmark components

In the Benchmark Components screen, you add components as participants to the benchmark definition and configure their behavior during the benchmark's execution.

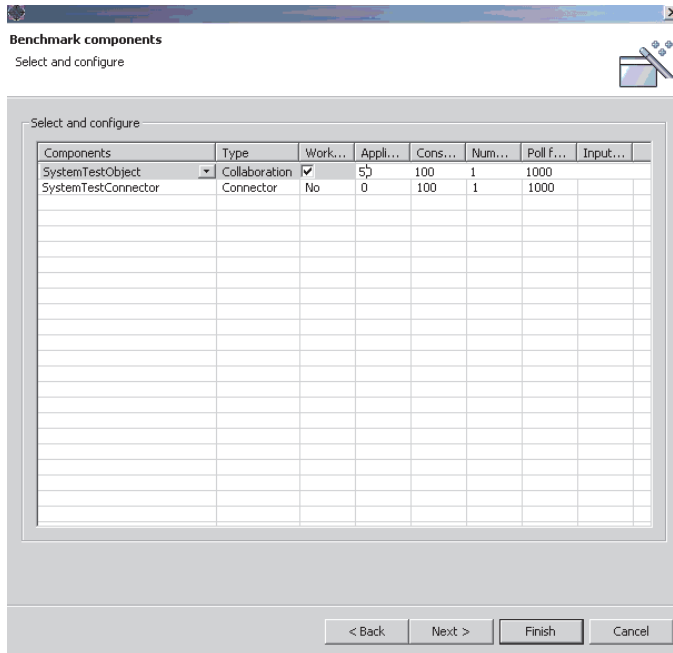


Figure 3. Benchmark Components screen

1. In the Select and configure screen, right click and select **Add**. In the new row that is inserted, click on the empty component box, then click on the down arrow. Select a component from the list.

If you add a component that depends on other components, then those other components are automatically added as well; for instance, if you add a collaboration object, then any connectors or grouped collaboration objects or the access client that it depends on are automatically added. The wizard does not allow you to add components that are not valid participants for the benchmark type; for instance, you cannot add a map definition to the Benchmark Components screen.

When components are added to the pane, the **Components** column lists the names of the components and the **Type** column lists the type of component.

2. Select the **Work generator** check box to specify a component as a workload generator in the benchmark. Table 2 on page 5 specifies the valid workload generators are for each type of benchmark. Typically these are the clients that are the source of business objects in the benchmark setup—for instance, the source connector for a collaboration throughput benchmark, or the access client for an access throughput benchmark.
3. (optional) Type a value in the **Application response time** column for a component to specify the number of milliseconds it waits before replying to a service call request. This value can be used to simulate anticipated application latency.

It is recommended that you perform tests with the assistance of application experts at the site to determine the average latency for the application to respond to business object requests that are sent to it by the connector. Use that average value for the simulated latency to obtain a more realistic set of numbers while still benefitting from the simplified setup of simulated connectors.

4. (optional) Type a value between 1 and 100 in the **Consume success rate** column for a component to specify the percentage of requests it should process successfully. This value can be used to simulate the average ratio of successful

flows to failed flows. The default value is 100, which means that the simulated connector responds with success for 100 percent of the business objects requests its processes (provided that the flow does not fail for other reasons, such as mapping problems).

Failed flows sometimes involve more collaboration processing than successful ones do, depending on the business requirements. Many collaborations are designed with logic that responds to an initial failure by resending the business object with a different verb (this logic is typically identifiable by use of the CONVERT\_CREATE and CONVERT\_UPDATE properties). Other collaborations have error-handling routines that perform transactional or administrative actions in response to a failure. These execution paths affect performance, so performing a benchmark that accurately simulates them is important.

It is recommended that you perform some tests with the assistance of IBM WebSphere InterChange Server development team to determine the average percentage of successfully processed flows. Then specify the average value for the consume success rate to determine the impact of failures on throughput.

5. (optional) Type a value in the **Number of objects per poll** column for a connector component to specify the number of events that it picks up with each poll call. This value can be used to simulate the common connector-specific capability of polling multiple objects with each poll call.
6. (optional) Type a value in the **Poll frequency** column for a connector component to specify the number of milliseconds between poll calls. This value can be used to simulate the standard connector capability of polling with variable frequency.

It is recommended that you perform some initial tests with the connector that is being simulated to determine a good initial set of values for the **Number of objects per poll** column and the **Poll frequency** column. The behaviors of these common capabilities are closely related and they affect throughput. Perform a benchmark with these values, then modify them and perform the benchmark again. By testing a number of combinations you can determine which combination provides the optimal throughput.

7. (optional) Type the path and name of a file in the **Input file** column for a component that has been marked as a workload generator. The file must contain sample data for the workload generator and be in the standard IBM WebSphere InterChange Server business object format (that is, the one in which business objects are written out during system tracing operations, or are saved from the Test Connector tool).

An input file of sample data can be produced by choosing **Generate workload to a file** in the Action screen of the benchmark wizard; this option is discussed in the section “Generating workload to a file” on page 16.

**Note:** The input file must reside on the same computer where the connector agent runs; if the connector agent is distributed on a computer other than the one where WebSphere InterChange Server runs, then the input file must be distributed with the agent.

8. Click Next.

## Configuring object properties

In the Object Properties screen, you configure business objects for the benchmark execution.

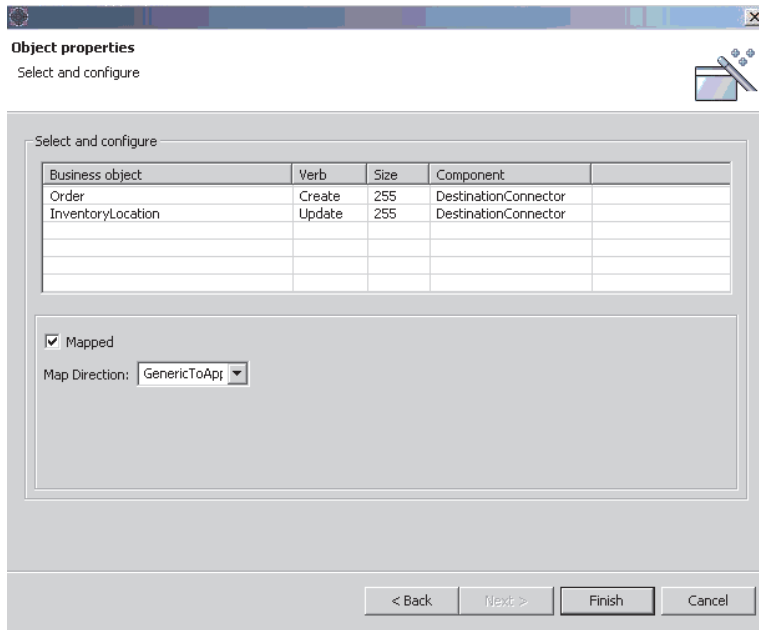


Figure 4. Object Properties screen

1. In the Select and configure screen, right click and select **Add**.
2. In the **Business Object** column, click to select a business object definition for the benchmark.
3. In the **Component** column, select a benchmark participant that is associated with the business object definition.
4. In the **Verb** column click the verb with which you want the sample business objects to be submitted.
5. (optional) In the **Size** column, type the size in bytes you want for the sample business objects.

It is recommended that you perform tests with the assistance of the application experts and IBM WebSphere InterChange Server development team at the site to determine the average size of business objects for the transaction. The recommended procedure for doing this is:

- a. Set the AgentTraceLevel property of the connector to a level at which it outputs the entire contents of the business objects it processes.
  - b. Generate a number of events that represent production data as closely as possible.
  - c. Start the actual application connector and have it poll and then process the events.
  - d. Extract the output of several business objects to individual text files without including any of the other tracing messages.
  - e. Open the individual text files containing the extracted business objects in a text editor that is can report the size of its contents in bytes and record the values.
  - f. Add the sizes of all the files together and divide the sum by the number of files to calculate the average business object size, then type that value in the field of the **Size** column.
6. Click **Finish** to complete the wizard.

## Properties specific to Business Object Throughput benchmarks

If you are performing a Business Object Throughput benchmark, an additional property is exposed at the Object Properties screen. The **Mapped** property lets you specify mapping with the benchmark execution; if mapping is not included, then only the business object and its transmission across the transport protocol are benchmarked. If you want mapping to be included in the benchmark, do the following:

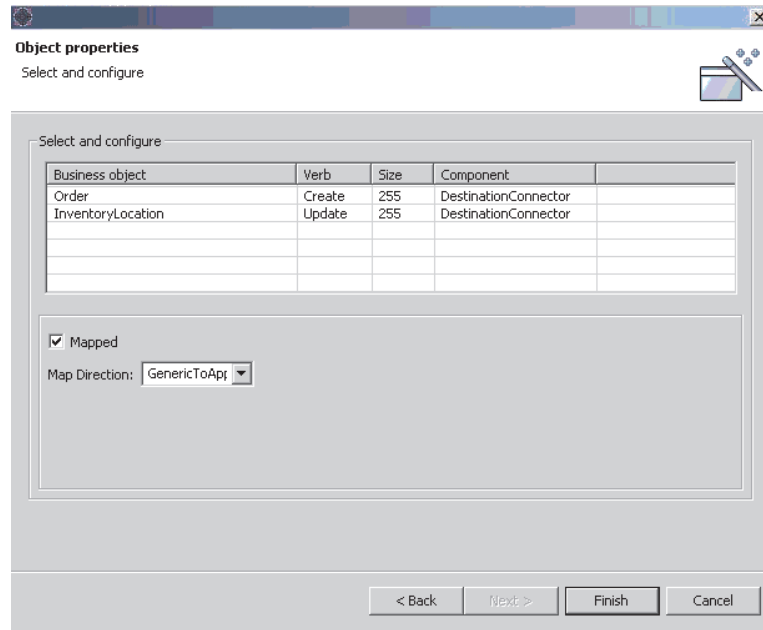


Figure 5. Object Properties screen—Business Object Throughput-specific

1. Select the **Mapped** check box.
2. In the **Map Direction** list click the appropriate value based on the context you want the benchmark to test.

If the direction is set to the value **GenericToApp**, then generic business objects are generated and mapped to the application-specific business objects with a calling context of `SERVICE_CALL_REQUEST`; the application-specific business objects are then placed on the transport protocol.

If the direction is set to the value **AppToGeneric**, then application-specific business objects are generated and mapped to generic objects with a calling context of `EVENT_DELIVERY`; the generic objects are then placed on the transport protocol.

## Properties specific to benchmarks for synchronous interfaces

The benchmark types dedicated to synchronous types of interfaces—the Access Throughput and Access Response Time benchmarks—do not have the **Component** and **Type** columns in the Object Properties screen, but have unique properties.

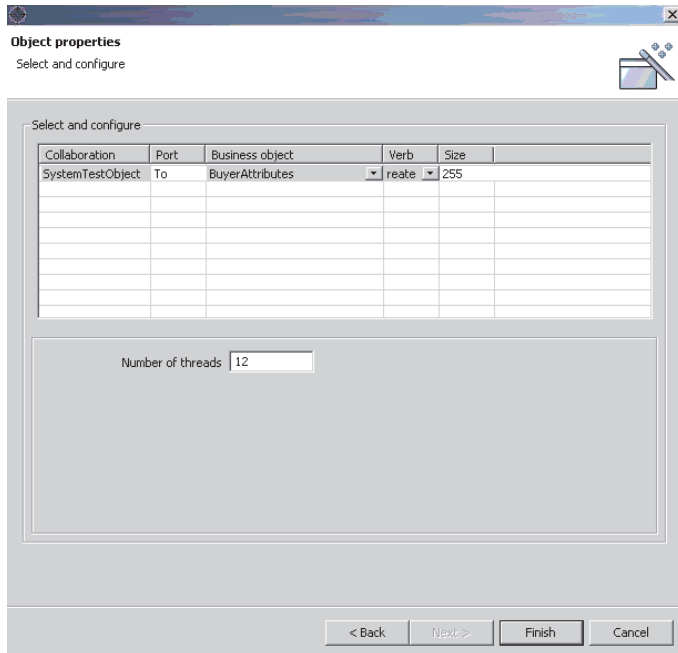


Figure 6. Object Properties screen—Synchronous Interface-specific

To set the object properties for benchmarks of synchronous interfaces, do the following:

1. In the list in the **Port** column, click the name of the port on the collaboration object specified in the **Collaboration** column to which direct calls are made by access clients.
2. Type a number in the **Number of threads** field to specify how many threads are created to make direct calls to the collaboration object.

---

## Editing a benchmark definition

To edit a benchmark definition do the following:

1. Launch the benchmark wizard as described in “Starting the benchmark wizard” on page 7, and choose **Edit an existing benchmark**.
2. In the Benchmark State screen, highlight the benchmark you wish to edit in the **Benchmarks** column, and click **Next**.



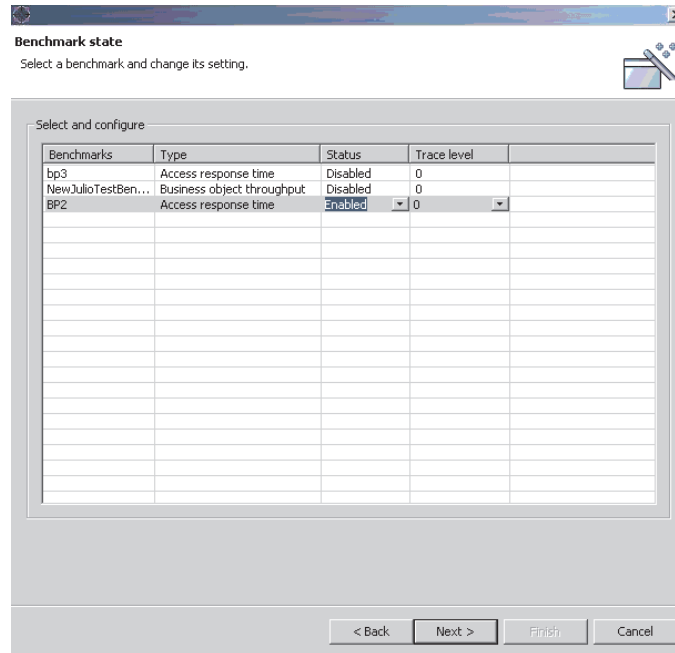


Figure 7. Benchmark State screen

- (optional) To disable a benchmark, click **Disabled** from the list in the **Status** column. Disabled benchmarks do not execute when WebSphere InterChange Server restarts, enabling you to keep them defined in the system without executing, but while still performing tasks that require shutting down and restarting WebSphere InterChange Server.

**Important:** Do not disable any benchmark while it is running. The system does not know if a benchmark is running, so there are no safeguards to prevent you from disabling a running benchmark. It is your responsibility to ensure that you do not disable a running benchmark. Disabling a running benchmark might produce unwanted results.

- (optional) Click a desired value from the list in the **Trace level** column.

**Note:** Tracing affects performance and therefore skews the results of a benchmark, but may be helpful for troubleshooting if needed.

- Click **Next** to advance through the wizard and then proceed through the remaining screens as described in the sections “Configuring general settings” on page 8, “Configuring benchmark components” on page 9, and “Configuring object properties” on page 11.

## Deleting a benchmark definition

Only one benchmark at a time may reference a component, as described in “Exclusivity in benchmark participants” on page 6.. If a component is required in multiple benchmarks then each benchmark must be created, performed, and then deleted before the next one may go through the same process.

To delete a benchmark, in the IBM System Manager, right click on the benchmark that you want to delete, under **Integration Component Libraries -> Samples\_ICL -> Benchmark**. Select **Delete** from the list of options, and then click **OK** to confirm the deletion.

## Generating workload to a file

The benchmark wizard can generate an input file of test data to use during a benchmark. It does the following:

- Determines the configured size of the business object
- Generates as many child objects as specified by the `BenchNumContainedObjs` parameter in the business object-level application-specific information for each child object
- Generates a numeric value for each attribute marked as key
- Compares the size of the data that would be generated based on the number of child objects instantiated and the size of the data generated for key attributes with the size specified by the user in the **Size** column in the Object Properties screen in the wizard
- Fills String attributes with placeholder text as necessary to make up the difference in size between the user-requested size and the size based on child object instances and key values
- Produces a file of the proper format from the generated data.

To generate workload to a file, do the following:

1. Launch the benchmark wizard as described in “Starting the benchmark wizard” on page 7, and choose **Generate a workload to a file**.
2. In the Select a benchmark screen, highlight a defined benchmark under the **Benchmarks** column and then click **Next**.

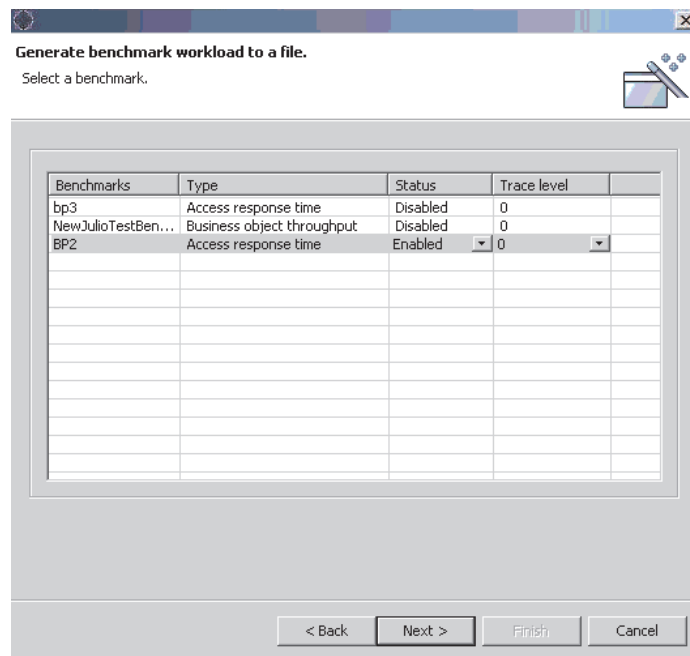


Figure 8. Benchmark Workload screen

3. In the Benchmark Workload screen, specify the path and name of the file of workload data you want to be created.

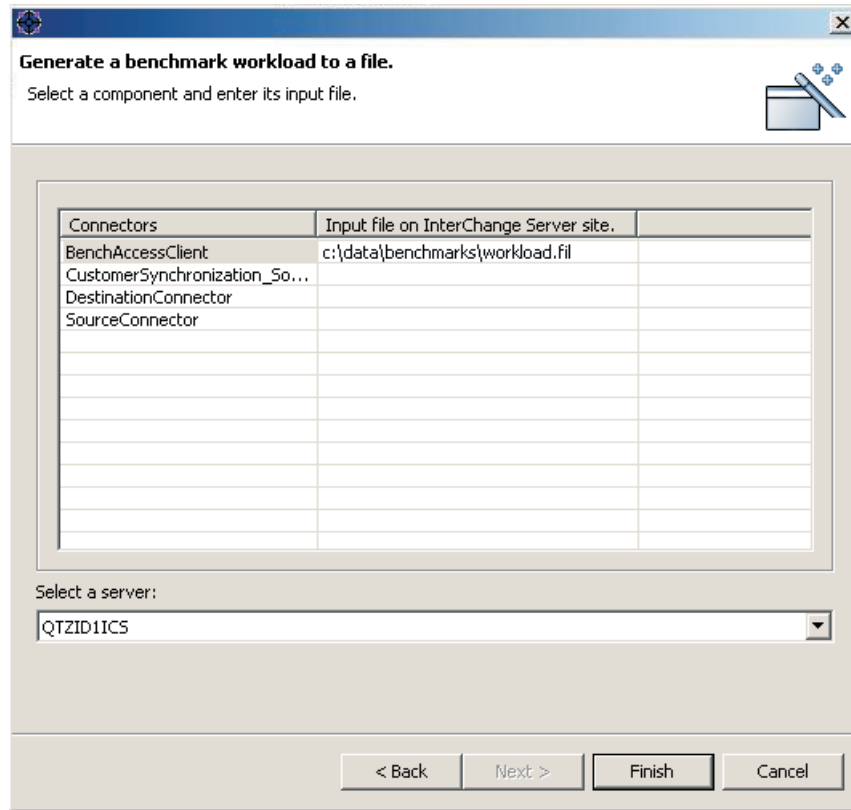


Figure 9. Benchmark Workload screen

4. Click **Finish**. When the wizard exits, the system creates the workload file in the location specified.

## Viewing benchmark results

In the General Settings screen, if you configured the benchmark definition to direct the results of a benchmark to a file, you can view the results through the benchmark wizard after the benchmark has executed. Follow these steps to view benchmark results:

1. Launch the benchmark wizard as described in “Starting the benchmark wizard” on page 7. and choose **Edit existing benchmark**
2. In the Benchmark State screen, highlight the benchmark whose results you want to view and then click **Next**.
3. In the General Settings screen, click **View Benchmark Running Results**. The Benchmark Results screen appears (Figure 10).

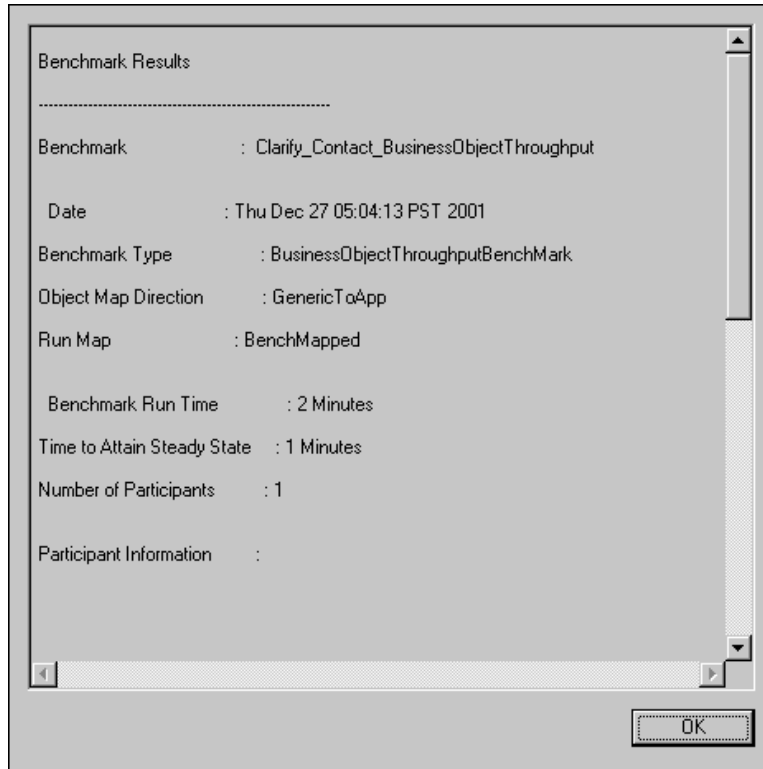


Figure 10. Benchmark Results screen

4. When you are finished viewing the results, click **OK**.

---

## Chapter 3. Performing IBM WebSphere InterChange Server benchmarks

This chapter describes the process of performing benchmarks.

**Important:** The benchmarking feature is designed to be used in a development or testing environment only. Running a benchmark adds data to cross-reference tables, and performing the required clean-up actions eliminates data from both work-in-progress tables and persistent messaging queues. It is critical that neither of these actions occur in a production environment.

Topics in this chapter include:

- “Back up the IBM WebSphere InterChange Server system”
- “Modifying IBM WebSphere InterChange Server content if necessary”
- “Prepare sample data” on page 21
- “Preparing a benchmark definition” on page 24
- “Creating client emulator startup scripts” on page 25
- “Perform a benchmark” on page 28
- “Clear the system” on page 30
- “Restore the backed-up repository” on page 33
- “Hints on benchmarking” on page 33

---

### Back up the IBM WebSphere InterChange Server system

For the following reasons, it is strongly recommended that you back up the entire IBM WebSphere InterChange Server system before you create or run IBM WebSphere InterChange Server benchmarks:

- It may be necessary or desirable to modify some of the components specifically for participation in the benchmark (see “Modifying IBM WebSphere InterChange Server content if necessary” on page 19). The changes should not remain in the system after the benchmarking, however, and it is easier to restore a system that has been backed up than it is to undo the changes.
- When a benchmark runs, it performs actions that might produce undesirable results. For instance, a benchmark generates event data that is stored in the event management database and the MQSeries queues, and a benchmark might affect cross-reference data and application data. For this reason, you might not want the results affecting even a test environment permanently. A backup provides the opportunity to restore the environment to its state before the execution of the benchmark.

To back up the system, follow the instructions in the section titled “Backing Up WebSphere InterChange Server Components,” in Chapter 3, Operational Tasks, in the System Administration Guide.

---

### Modifying IBM WebSphere InterChange Server content if necessary

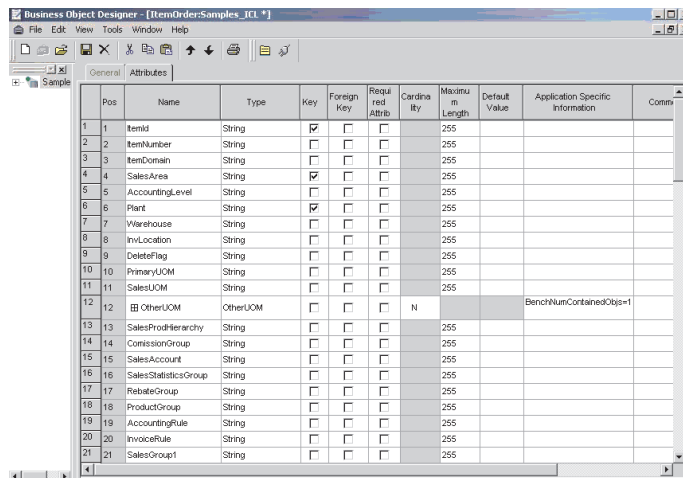
Some benchmarks may require that you modify IBM WebSphere InterChange Server content. This section describes those situations and how to modify the content.

## Specifying the number of generated child objects

You can specify the number of child object instances that are generated for a top-level source object when the benchmark executes. This is an optional technique that results in greater accuracy in the simulation than using a default number of instances generated for each child object. Follow these steps to implement this approach:

1. Interview persons at the site who are experts in the source application to find out the average number of each contained entity within the top-level source object.
2. Use Business Object Designer (BOD) to modify the business object-level application-specific information of the child business object definitions of the source top-level object. Create a parameter named `BenchNumContainedObjs` and set its value to the number of child object instances that you ascertained in step 1.

Figure 11 shows how to add the `BenchNumContainedObjs` parameter to a business object definition.



Pos	Name	Type	Key	Foreign Key	Required	Cardinality	Maximum Length	Default Value	Application Specific Information	Comments
1	ItemId	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
2	ItemNumber	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
3	ItemDomain	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
4	SalesArea	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
5	AccountingLevel	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
6	Plant	String	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
7	Warehouse	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
8	InvLocation	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
9	DeleteFlag	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
10	PrimaryUOM	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
11	SalesUOM	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
12	OtherUOM	OtherUOM	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	N			BenchNumContainedObjs=1	
13	SalesProdHierarchy	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
14	CommissionGroup	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
15	SalesAccount	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
16	SalesStatisticsGroup	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
17	RebateGroup	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
18	ProductGroup	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
19	AccountingRule	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
20	InvoiceRule	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			
21	SalesGroup1	String	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>		255			

Figure 11. `BenchNumContainedObjs` parameter

## Special formatting operations

Another reason you may need to modify IBM WebSphere InterChange Server content in preparation for a benchmark is to handle the presence of any special formatting operations in the components. Some examples of this are:

- The content expects a value belonging to a specific set of values

For instance, a transformation that implements a lookup relationship requires that the value in an attribute be equal to a value in the database tables of the lookup relationship. A lookup relationship might be used to perform static cross-referencing between an application that uses text and another that uses numeric codes for country names.

This sort of transformation is typically achieved with a lookup relationship, but can also be done by evaluating the values in Java code.

- The content expects a value that appears in a particular format

In this situation, an operation expects that a value in an attribute is formatted in a particular way, for instance, a series of tokens separated by a particular order of delimiters, or a specific date format.

Either of these situations typically occurs in mapping but can occur anywhere depending on the business requirements and how the interface is designed. Furthermore, a violation of any such transformation rule typically results in a failure of the flow, although this behavior also depends on the business requirements and how the interface is designed.

Special formatting operations such as these are likely to result in failed flows when performing a benchmark because the system cannot know to generate sample data from a specific set of values, or sample data that conforms to a specific format. A benchmark that measures the throughput of a system where 100% of the flows fail does not present an accurate measure of the system's throughput capabilities, so it is important to avoid this problem.

You can take one of two approaches to handle these requirements:

- Modify the content

With this approach you change the behavior of the interface so that the special formatting operations do not apply.

If the operation is performed in a map, then you can comment out the operation and recompile the map. If the operation is performed in a collaboration, then you must comment out the operation in the collaboration template, compile the template, and restart any collaboration objects that are based on the template. In both cases, you must make sure that no operations reference or depend on the value derived from any operations that are commented-out.

This is the less helpful of the two approaches because special operations (lookup relationships in particular) affect the throughput of an interface, and the benchmark should be as accurate as possible. If the transformations are not executed as part of the benchmark, then the benchmark is an inaccurate portrait of the system behavior. There may be occasions, however, where the only alternative is to modify the content in this way.

- Modify the sample data

With this approach, you prepare and modify a custom file of sample data that has values that belong to the required subset or conform to the required format.

This is the more helpful of the two approaches because it allows the benchmark to emulate "real" transactions as accurately as possible. This may not always be a viable approach, however, in which case you may have to modify the content as a workaround.

The ways you can generate sample data and modify it are covered in greater detail in the section "Prepare sample data."

---

## Prepare sample data

There are several ways to prepare sample data to be used for generated business objects in a benchmark:

---

Using Test Connector	page 21
Using Map Designer	page 22
Using tracing output	page 22
Generating workload to a file	page 23

---

## Using Test Connector

You can save a business object to a file from Test Connector and use it as an input file of sample data for a benchmark.

To implement this approach, see Chapter 9, Test Connector Tool in the *Collaboration Development Guide*.

The advantages of this approach are:

- You may already have some test data files saved from when the component or interface was tested during development, allowing you to use existing work.
- The graphical interface of Test Connector makes it easy to supply specific values so that you can conveniently handle the situations described in the section “Special formatting operations” on page 20.
- You run the benchmark wizard only once. If you use the technique “Generating workload to a file” on page 23, then you must run the benchmark wizard once to create the benchmark definition and a second time to create the file of sample data for it.

The disadvantages of this approach are:

- You must create all the test data by hand. If you use the technique “Generating workload to a file” on page 23, then the process is automated.
- Test Connector cannot bind to a connector agent that is a participant in an enabled benchmark definition so you must either prepare the test data before defining the benchmark, or you must disable the benchmark if it already exists.

## Using Map Designer

You can save a business object to a file from Map Designer and use it as an input file of sample data for a benchmark.

To implement this approach, see Chapter 4, Compiling and Testing Maps in the *Map Development Guide*.

The advantages of this approach are:

- You can use Map Designer to generate test data even if the benchmark definition already exists and is enabled. Although Map Designer and Test Connector share many advantages, this is one respect in which using Map Designer for generating test data is more convenient.
- You may already have some test data files saved from when the component or interface was tested during development, allowing you to use existing work.
- The graphical interface of Map Designer makes it easy to supply specific values so that you can conveniently handle the situations described in the section “Special formatting operations” on page 20.
- You run the benchmark wizard only once. If you use the technique “Generating workload to a file” on page 23, then you must run the benchmark wizard once to create the benchmark definition and a second time to create the file of sample data for it.

The disadvantage of this approach is that you must create all the test data by hand. If you use the technique “Generating workload to a file” on page 23, then the process is automated.

## Using tracing output

You can copy business object data that is output to the tracing destination of a connector at higher trace levels and use it as an input file of sample data for a benchmark.

To implement this approach, do the following:



1. Set the trace level for the connector to a level at which it outputs the contents of the business objects it processes.
2. Trigger a business object of the appropriate type and have the connector process it.
3. Copy the business object contents from the tracing destination into a file.  
Business object data is recorded in a kind of markup language: elements like attributes, child objects, and the entire top-level business object are demarcated by pairs of tags. Each business object begins with a set of StartHeader and EndHeader tags, which is followed by a StartB0 tag that includes the type of the top-level business object. Below is an example:

```
<StartHeader>
<Version = 3.0>
<EndHeader>
<StartB0:BusinessObjectType>
```

The output for a business object ends with an EndB0 tag that includes the type of the top-level business object just as the StartB0 tag does. Below is an example:

```
<EndB0:BusinessObjectType>
```

Copy all of the output from the StartHeader tag to the EndB0 tag.

**Important:** The StartHeader tag might appear on a line that also contains other text (such as information about the tracing subsystem), and the EndB0 tag typically appears on a line that ends with a right bracket (]). Be sure to include only the important tags; do not copy unnecessary characters, or the system cannot use the file as input.

4. Save the file.

The advantages of this approach are:

- You run the benchmark wizard only once. If you use the technique “Generating workload to a file” on page 23, then you must run the benchmark wizard once to create the benchmark definition and a second time to create the file of sample data for it.
- You can obtain production-quality test data with a minimum of effort—the data can be taken from production environment tracing logs and be an accurate image of the typical business entity. You do not have to type the values by hand.

The disadvantages of this approach are:

- This technique is not as easy to use as other approaches.
- If you do not copy the business object contents properly, then the system cannot use it to generate business objects.

## Generating workload to a file

You can use a feature of the benchmark wizard to generate a file of sample data.

To implement this approach, do the following:

1. Determine the location and name that you want for the file of sample data to be generated.

**Note:** If you plan ahead this way then you only have to run the benchmark wizard two times—one time to create the benchmark definition and a second time to generate the input file. If you do not plan ahead then you have to run the benchmark wizard a third time to reconfigure the benchmark definition to use the input file.

2. Prepare a benchmark definition. Follow the instructions in the section “Preparing a benchmark definition” on page 24.
  - a. When you are presented with the Benchmark Components screen, specify the file path and name you determined in step 1, as described in step 7 of the section “Configuring benchmark components” on page 9.
  - b. Proceed through the Object Properties screen and finish the benchmark definition.
3. Follow the instructions in the section “Generating workload to a file” on page 16 to create a file of sample data for the benchmark definition created in step 2.

The advantage of this approach is: that you can conveniently generate a large amount of sample data through configuration and metadata.

The disadvantages of this approach are:

- You must run the benchmark wizard once to create the benchmark definition and then a second time to generate sample data for it.
- In order to customize the sample data to handle situations like those described in the section “Special formatting operations” on page 20, you must edit the text file by hand, replacing all the generated values with values that meet the required subset or format.

---

## Preparing a benchmark definition

With the content backed up and modified in preparation for benchmarking, you can create the definitions of the benchmarks you want to perform.

1. Use the section titled “Types of benchmarks” on page 2 to determine what type of benchmark you want to configure.
2. Determine which components to add to the benchmark definition and which ones to configure as workload generators. Use the section Table 2 on page 5, with your knowledge of the site-specific content to make these determinations.
3. Consult with the application experts and IBM WebSphere InterChange Server developers at the site to determine the appropriate values for the variable properties that you configure when preparing a benchmark definition. The “Variable” column in Table 3 lists these variables, and the “Reference” column lists the sections in this document where the variables are described in detail. Consult with the appropriate experts to determine the proper values for these variables.

*Table 3. Benchmark configuration variables*

Variable	Reference
Application response time	step 3 in “Configuring benchmark components” on page 9
Consume success rate	step 4 in “Configuring benchmark components” on page 9
Number of objects per poll	step 5 “Configuring benchmark components” on page 9
Poll frequency	step 6 “Configuring benchmark components” on page 9
Size in bytes of business object	step 5 “Configuring object properties” on page 11

4. Use the instructions in Chapter 2, “Working with benchmark definitions,” on page 7 to create, edit, or load a benchmark definition and then configure it with the information you derived in steps 1 through 3.

---

## Creating client emulator startup scripts

Although many types of benchmarks can be performed with real clients, it is often desirable to remove the complexity associated with having to prepare the applications and configure the clients. To satisfy this requirement, you can use libraries provided by IBM WebSphere InterChange Server that emulate connector agents.

### About starting IBM WebSphere InterChange Server clients

This section discusses some of the concepts and implementations for starting IBM WebSphere InterChange Server clients.

For an IBM WebSphere InterChange Server client to start up, it must start a Java program and do the following:

- Load supporting classes
- Invoke the appropriate wrapper class
- Use the appropriate dynamically loadable library (DLL) for C++ connectors, or the proper Java class for Java connectors
- Specify the name of the connector (for a connector client) or the name of the collaboration to which the client sends objects (for an access client)
- Specify the name of IBM WebSphere InterChange Server

These actions are accomplished through startup scripts, which are similar in function across Windows and UNIX-based systems, though there are slight differences in implementation.

### Creating benchmark connector- emulator startup scripts

To create a benchmark connector-emulator startup script for a particular connector, you must create a connector-emulator startup script that invokes the following Java class for a connector emulator:

`CxCommon.BenchConnector.BenchMarkConnector.`

The steps to create this connector-emulator startup script depend on your operating-system environment.

### Connector-emulator startup scripts for Windows

To create a benchmark connector-emulator startup script on a Windows system, take the following steps:

1. Make a copy of the connector startup script template, which is located in the following directory:

`ProductDir\templates\start_connName.bat.`

Give the copy a name that suggests its role in emulating a connector for the purpose of benchmarking (such as `start_benchmark.bat`) and put it in the `bin` subdirectory of the product directory.

2. Edit the copied startup script so that it invokes the benchmark connector emulator's Java class.

For the Windows environment, the line in the startup-script template to invoke a Java connector class appears as follows:

```
call startup_adapter.bat -nconnName -serverName  
-lconnectorSpecificClasses
```

where:

- `connName` is the name to assign to the connector emulator.
- `serverName` is the name of your InterChange Server instance.

- *connectorSpecificClasses* specifies the name of the Java connector class.

In this line, change the name of the Java-class (*connectorSpecificClasses*) to the name of the Java class for the connector emulator, so that the line appears as follows:

```
call startup_adapter.bat -nconnName -sserverName
-lCxCommon.BenchConnector.BenchMarkConnector
```

3. Save the benchmark-connector-emulator startup script.
4. Configure the startup mechanism so that the name of the connector is passed as the value to the *-n* parameter (*connName*) and the name of the InterChange Server instance (*serverName*) is passed as the value to the *-s* parameter. Implement one of these three approaches to create a solution that allows you to emulate each connector you are benchmarking:

- Hard-code the names of the connector and InterChange Server instance into the connector-emulator startup script by replacing *connName* and *serverName* with their appropriate values.

The advantages of this approach are that you can ensure that the case and spelling of the values is correct, and you avoid the need to type them at the command line each time you start the connector emulator. The disadvantage to this approach is that it is customized to a particular connector; if you have multiple connectors to emulate, then you must customize a startup script for each one.

- Run the connector-emulator startup script from the command line and type the names of the connector and InterChange Server instance each time. For example, if your connector-emulator startup script is called *start\_Benchmark.bat*, you can start any connector emulator as follows:

```
start_Benchmark connName serverName
```

In this case, you must also edit your connector-emulator startup script to change the names of the connector and InterChange Server instance to command-line parameters, so that the line to invoke *start\_adapter.bat* appears as follows:

```
call start_adapter.bat -n%1 -s%2
-lCxCommon.BenchConnector.BenchMarkConnector
```

The *start\_adapter.bat* file is installed when the Adapter Framework is installed. The advantage to this approach is that you can typically use a single connector-emulator startup script for multiple connector emulators. The disadvantage of this approach is that you must be careful to correctly specify the names of the connector and InterChange Server instance on the command line.

- Create a Windows shortcut for the benchmark connector-emulator startup script.

In this case, you must edit the connector-emulator startup script so that it accepts the names of the connector and InterChange Server instance as command-line parameters (as described above) and then include the actual names in the target of a Windows shortcut for the connector emulator. Use this shortcut to invoke the connector emulator.

This approach combines the advantages of the previous two approaches—you do not have to maintain multiple scripts for each connector emulator, and you do not have to manually enter the names of the connector and InterChange Server instance on the command line each time you start the connector emulator. The disadvantage to this approach is that it is limited to Windows systems.

## Connector-emulator startup scripts for UNIX

To create a benchmark connector-emulator startup script on a UNIX-based system, take the following steps:

1. Make a copy of the low-level connector startup-script template, which is located in the following directory:

```
ProductDir\templates\start_connName.sh
```

Give the copy a name that suggests its role in emulating a connector for the purpose of benchmarking (such as `start_Benchmark.sh`) and put it in the `bin` subdirectory of the product directory.

2. Edit the copied startup script so that it invokes the benchmark connector emulator's Java class.

For UNIX-based environments, the line in the startup-script template to invoke a Java connector class appears as follows:

```
exec ${ADAPTER_RUNTIME}/bin/start_adapter.sh -nconnName -sserverName  
-lconnSpecificClasses -f... -p... -c... ...
```

where:

- *connName* is the name to assign to the connector emulator.
- *serverName* is the name of your InterChange Server instance.
- *connectorSpecificClasses* specifies the name of the Java connector class.

In this line, change the name of the Java-class (*connSpecificClasses*) to the name of the Java class for the connector emulator, so that the line appears as follows:

```
exec ${ADAPTER_RUNTIME}/bin/start_adapter.sh -nconnName -sserverName  
-lCxCommon.Bench.Connector.BenchMarkConnector -f... -p... -c... ....
```

The “-f... -p... -c... ..” string indicates the position for other command-line parameters that the `start_adapter.sh` script supports. You must decide if any of these command-line parameters are useful. For more information on these parameters, see the *System Administration Guide*.

3. Save the benchmark-connector-emulator startup script.
4. Configure the startup mechanism in one of the following ways so that the name of the connector is passed as the value to the `-n` parameter (*connName*) and the name of the InterChange Server instance (*serverName*) is passed as the value to the `-s` parameter.

Implement one of these three approaches to create a solution that allows you to emulate each connector you are benchmarking:

- Hard-code the names of the connector and InterChange Server instance into the connector-emulator startup script by replacing *connName* and *serverName* with their appropriate values.  
The advantages of this approach are that you can ensure that the case and spelling of the values is correct, and you avoid the need to type them at the command line each time you start the connector emulator. The disadvantage to this approach is that it is customized to a particular connector; if you have multiple connectors to emulate, then you must customize a startup script for each one.

- Run the connector-emulator startup script from the command line and type the names of the connector and InterChange Server instance each time. For example, if your connector-emulator startup script is called `start_Benchmark.sh`, you can start any connector emulator as follows:  
`start_Benchmark connName serverName`

In this case, you must also edit your connector-emulator startup script and make the names of the connector and InterChange Server instance command-line parameters, so that the line to invoke `start_adapter.sh` appears as follows:

```
exec ${ADAPTER_RUNTIME}/bin/start_adapter.sh -n$1 -s$2
```

-lCxCommon.BenchConnector.BenchMarkConnector

The advantage to this approach is that you can typically use a single connector-emulator startup script for multiple connector emulators. The disadvantage of this approach is that you must be careful to correctly specify the names of the connector and InterChange Server instance on the command line.

---

## Perform a benchmark

With the system backed up, the content modified as necessary, and the benchmark defined and configured, you can perform the benchmark test.

### Restart InterChange Server

**Note:** IBM WebSphere InterChange Server must be restarted in order to perform a benchmark.

1. Shut down WebSphere InterChange Server by following the instructions in the section "Shutting Down InterChange Server," in Chapter 3 of the System Administration Guide.
2. Restart WebSphere InterChange Server by following the instructions in the section "Starting InterChange Server," in Chapter 3 of the System Administration Guide.

As WebSphere InterChange Server starts up, it first loads all business object definitions in the repository and then initializes all maps.

WebSphere InterChange Server then starts the coordinators for all defined benchmarks that are enabled. For each coordinator that starts, an entry similar to the one below is written to the logging destination (with the name of the benchmark definition in place of the text *BenchmarkName*):

```
System: Server] [Thread: main (#8316668)] [Msg: BENCH : BenchmarkName : Waiting for Participants ]
[Time: 2001/12/26 09:22:58.102] [System: Server] [Thread: main (#8316668)]
[Msg: BenchMark ***** BenchmarkName : Starting Coordinator ***** ]
```

WebSphere InterChange Server next initializes all the connectors in the repository, and then initializes all the collaborations in the repository.

WebSphere InterChange Server indicates its readiness by writing an entry such as the following to the logging destination:

```
[Time: 2001/12/26 05:40:55.975] [System: Server] [Thread: main (#8316668)] [Type: Info]
[MsgID: 20] [Msg: InterChange Server "CW400DEV" is ready.]
```

When WebSphere InterChange Server reaches a ready state, any benchmark participants synchronize with the coordinator of the benchmarks they participate in. Because collaborations initialize while WebSphere InterChange Server is starting, any collaborations that participate in benchmarks synchronize with their coordinator components at this time. For each participant that synchronizes with a coordinator component, an entry such as the one below is written to the logging destination (with *X* being replaced by the number of participants that have synchronized and *N* being the total number of participants):

```
[Time: 2001/12/26 05:40:55.935] [System: Server] [Thread: Thread-5 (#5138078)]
[Msg: BenchMarkBenchmarkName5: Participant ParticipantName : X out of N has synchronized ]
```

### Ensure components are in the proper state

You should examine the states of the components that participate in benchmarks once WebSphere InterChange Server has been restarted. Follow the steps below to do so.

1. Connect to WebSphere InterChange Server using System Manager.
2. Click **Server --> System View** from the menu bar of System Manager.
3. When the Monitor tool appears, click **View --> Show Maps and Relationships** from the menu bar.
4. Investigate all of the components that are part of the interface to be benchmarked in the System Monitor and ensure that each has a green traffic light icon next to it (indicating that they have started successfully).

If any of the components are not active then the load of events generated by the system during the benchmark will fail, resulting in a poor measurement of system throughput and increased cleanup responsibilities, so it is best to check their states first. If any component involved in a benchmark is not active, right-click the component and click the proper menu item to start the component. If a component remains inactive even after an attempt to start it, there may be issues with the content that need to be resolved.

## Start benchmark client participants

Once you have made sure that all of the components involved in the benchmark are active, you should start the client participants.

To start clients that are not being emulated you should take the normal steps, such as using the `connector_manager.sh` script in a Solaris environment, using a shortcut or service in a Windows environment, or using the Object Activation Daemon. To start clients that are being emulated you should use the appropriate client emulator created in the section “Creating client emulator startup scripts” on page 25.

When you start up client participants they synchronize with the coordinator component. For each client participant that synchronizes with a coordinator component, an entry such as the one below is written to the logging destination (with *X* being replaced by the number of participants that have synchronized and *N* being the total number of participants).

```
[Time: 2001/12/26 05:40:55.935] [System: Server] [Thread: Thread-5 (#5138078)]
[Msg: BenchMarkBenchmarkName5: Participant CollaborationObjectName : X out of N has synchronized ]
```

After all the required participants have synchronized an entry such as the one below is written to the logging destination, indicating that the benchmark has begun:

```
[Time: 2001/12/26 08:43:02.042] [System: Server] [Thread: VBJ ThreadPool Worker (#6231419)]
[Msg: BenchMark : BenchmarkName ***** START BENCHMARK *****]
```

Benchmarks execute for as long as they are configured to. During the execution you may use the System Monitor or view the logging destination to observe its progress. When the benchmark finishes it automatically shuts down the client participants.

## View benchmark results

After a benchmark finishes executing the results are output to either the location specified in the benchmark definition, or to the logging destination of WebSphere InterChange Server if no location is configured. If the results were output to a dedicated file then you can view them by opening the file or by following the instructions in the section titled “Viewing benchmark results” on page 17. If the results were output to the logging destination of WebSphere InterChange Server then you must view them in that location.

---

## Clear the system

Running a benchmark has the potential to affect several areas of the product that may require clearing afterwards.

### Clear the relationship tables

The cross-referencing tables for any identity relationships that are used by benchmarked interfaces contain data after a benchmark, and should be cleared. The two ways you can clear the cross-referencing tables for relationships are covered in the following sections.

#### Using Relationship Manager

Relationship Manager is a graphical tool that provides a database vendor-independent way to manage relationship data.

To implement this approach, reference the section “Deleting a Relationship Instance in Chapter 6, Using Maps and Relationships” in the *System Implementation Guide*. Note that you can highlight one relationship instance in the Relationship Manager, hold down the Shift key, and highlight another relationship instance to select all the relationship instances between them, making it easier to delete multiple instances at once.

The advantages of this approach are:

- Relationship Manager is database vendor-independent, so you can use it to manage relationships in SQL Server or Oracle without having to be familiar with specific client tools.
- Relationship Manager does not require knowledge of the Structured Query Language (SQL).

The disadvantages of this approach are:

- Relationship Manager only runs on the Windows platform.
- WebSphere InterChange Server must be running to use Relationship Manager, so you cannot use it to clear the relationship tables if WebSphere InterChange Server has been shut down.
- To delete relationship instances, you must first have the Relationship Manager retrieve them. This can take some time if there are many instances, and there is a limit to how many instances can be retrieved at a single time, so you might have to perform this task several times to completely clear the relationship tables.

#### Using SQL queries

You can use SQL queries to truncate the relationship tables and thereby clear them of data.

To implement this approach, do the following:

1. Determine the table names for each participant in each identity relationship that is used in each interface affected by the benchmark. Reference the section titled “Advanced Settings for Participant Definitions” in Chapter 7, Creating Relationship Definitions in the *Map Development Guide* to determine how to find out the table names for participants by looking up their advanced settings.
2. Execute the following statement in the appropriate database query tool for every participant as necessary, replacing the text *ParticipantTableName* with the name of the participant table:

```
truncate table ParticipantTableName ;
```



3. Execute the queries to truncate the participant tables.

The advantages of this approach are:

- WebSphere InterChange Server does not have to be running to use this approach.
- The queries can be saved, making it easy to run a benchmark multiple times and just execute the query each time.
- You do not have to retrieve the participant instances before deleting them, as you do with Relationship Manager, and there is no restriction on how many records may be deleted at the same time.
- The database query tool may not have the platform restrictions that Relationship Manager does.

The disadvantages of this approach are:

- You must be familiar enough with the database query tool to use it effectively.
- You must be familiar enough with SQL to construct the statements, and you must construct one for each participant table.

## Shut down InterChange Server

In order to perform the task “Clear the MQSeries queues” on page 32 that is described in the following section, you must make sure that WebSphere InterChange Server and any of its connector clients that communicate with MQSeries are shut down. Follow the instructions in the section titled “Shutting Down InterChange Server” in Chapter 3 Operational Tasks in the *System Administration Guide* to shut down WebSphere InterChange Server.

## Clear the persistent stores

The IBM WebSphere InterChange Server system stores event data persistently by using MQSeries to record images of initiating events as files on hard disk, and by keeping pointers to those files in work-in-progress tables in a database. Even during a benchmark simulation these messages and database entries are created in order to reproduce production-environment conditions as accurately as possible.

After a benchmark finishes executing there may still be some residual event data in the tables and queues (if some events were to fail, for instance). The data is not important in the way that production data is, and it can present problems if left in place (for instance, if the interfaces that those events referenced are deleted, then the events entries are left dangling). It is therefore recommended that you clear the system after performing a benchmark. The following sections describe how to do so.

**Important:** The following sections describe actions that eliminate event data from the IBM WebSphere InterChange Server system. As mentioned before, the benchmarking activities described in this document should never be performed in a production environment, but only in a development or test environment. The cleanup techniques described below should likewise only be performed in a development or test environment after a benchmarking simulation, and never in a production environment. Furthermore, it is important that these actions only be taken when the server is in a quiescent state; if the tables or queues are cleared while events are still being processed then WebSphere InterChange Server tries to access information it needs but which no longer exists, and crashes as a result.

## Clear the MQSeries queues

If MQSeries is configured as the delivery transport for the source connector in the interface being benchmarked, then event data is stored in MQSeries queues. You must clear the queues as part of the task of clearing the events from the system.

The steps below provide instructions on how to clear the queues on a Windows system:

1. Click **Start --> Programs --> IBM MQSeries --> MQSeries Explorer**.
2. Double-click the name of the proper queue manager.
3. Double-click the **Queues** folder to explore it.
4. Right-click the proper input queue and click **All Tasks --> Clear Messages**.
5. When presented with a confirmation prompt, click **Yes**.

**Note:** At this point you may be presented with a prompt that reads “Object is open. An attempt was made to delete or change an object that was in use. Wait until the object is not in use and retry. (AMQ4070)”. If you are, click **OK** and then terminate any processes that might be using the queue, such as WebSphere InterChange Server or any connectors.

6. When presented with an information prompt that the queue has been cleared of messages, click **OK**.
7. The **Current Depth** column indicates the number of messages in each queue; repeat the steps above for all queues that have benchmark event messages in them.

## Clear the work-in-progress tables

The IBM WebSphere InterChange Server system keeps references to in-progress and failed events in several database tables. To clear the events tables of all entries use a database query tool such as SQL Plus, TOAD, or SQL Server Query Analyzer to truncate the tables. Execute the following commands in the query tool:

```
truncate table cxwipobjects ;  
  
truncate table cxwipblobs ;  
  
truncate table cxpbusobjstate ;  
  
truncate table cxpbusobjmessage ;  
  
truncate table cxfailedeventkeys ;
```

If any of the interfaces being benchmarked use the transactional services provided by the IBM WebSphere InterChange Server system, then there may also be data in the transaction tables. To clear the transaction tables execute the following commands in the query tool:

```
truncate table cxastatebusobjs ;  
  
truncate table cxcompbusobjs ;  
  
truncate table cxstatebusobjs ;  
  
truncate table cxtransblobs ;
```

---

## Restore the backed-up repository

IBM recommends that you restore the system backup that you prepared as part of the task “Back up the IBM WebSphere InterChange Server system” on page 19. This restores any content that needed to be changed to accommodate the benchmark to its previous state, and prevents benchmark definitions from being accidentally migrated into a production environment.

---

## Hints on benchmarking

You do not have to adhere to the following hints to perform a benchmark, but you may find it more convenient if you do so:

- Create a folder with just the shortcuts to your benchmark client emulator scripts, WebSphere InterChange Server, System Manager, System Monitor, and other tools that you use frequently when benchmarking, such as MQSeries Explorer and database query tool. Benchmarking involves a great deal of repeatedly starting and stopping different tools and programs, and having them all conveniently accessible makes the process more efficient.
- Create a directory named benchmarking within the IBM WebSphere InterChange Server directory. Within the benchmarking directory create the following subdirectories:

---

Subdirectory name	Purpose
results	Follow the instructions in step 6 of the section “Configuring general settings” on page 8 to output the results of the benchmark to a file. This makes it easier to view the results without having to search through the other entries in the WebSphere InterChange Server logging destination, and makes it easier to compare the results of different executions of the same benchmark against one another.
sampledata	Save all input files of sample data in this directory so that you (or anyone else you might be working with that has to perform benchmarking as well) can reliably locate good test data to use.
scripts	Save the different scripts that you might use in support of your benchmarking activities in this directory. These might be SQL scripts to clear relationship tables, or scripts to stop the MQSeries queue manager and clear the queues, and so forth.

---

- Come up with a system for recording and easily comparing and contrasting the results of the benchmarks you perform. A spreadsheet is a very good tool for this purpose.
- Benchmarking is frequently performed in a continuous session; you will probably perform one benchmark multiple times, then perform another benchmark multiple times, and so forth. You will probably find it helpful to keep an instance of your database query tool open with scripts loaded to do the following:
  - Select all rows from the work-in-progress tables and transaction tables
  - Truncate the work-in-progress tables and transaction tables
  - Select all rows from the impacted relationship tables

- Truncate the impacted relationship tables

You will probably also find it helpful to keep a tool or script handy for clearing the MQSeries queues, as this task is performed frequently when benchmarking.

---

## Chapter 4. Examples

---

### Using repos\_copy

**Note:** IBM WebSphere adapters, IBM WebSphere InterChange Server tools, and the repos\_copy program are clients of WebSphere InterChange Server and cannot interact with it if it is not running.

1. Launch a command line interface tool.

For example, to open a command prompt in Windows follow the appropriate guide below:

```
Windows 2000
Click Start > Programs > Accessories > Command Prompt.
End of Windows 2000
```

```
Windows
Click Start > Programs > Command Prompt.
End of Windows
```

2. Run the repos\_copy program with the -i switch to import the repository file, the -s switch to specify the case-sensitive name of IBM WebSphere InterChange Server, the -u switch to specify the user name, the -p switch to specify the password, and the -ar switch to replace any duplicate components.

**Note:** Other switches may be necessary depending on the environment; the repos\_copy program is documented in the *System Administration Guide*.

An example is provided below for an environment where the arguments for the various switches are set to the values specified in the table in the section "Hints on benchmarking" on page 33 and where the input file named contactsyncrepos.in is in the Samples\\BenchMark\\ContactSyncBenchmark subdirectory of the IBM WebSphere InterChange Server directory. The variables that must be changed for specific environments are the directory in which IBM WebSphere InterChange Server is installed, the name of WebSphere InterChange Server, and the path to the input file. Press the Enter key after providing the proper values.

```
D:\CrossWorlds400Dev\Samples\\BenchMark\\ContactSyncBenchmark>repos_copy
-icontactsyncrepos.in -sCW400DEV -uadmin -pnull -ar
```

The repos\_copy program reads the component definitions from the file and imports them into the repository database. When the import is complete in Windows, repos\_copy presents a prompt with the text "Press any key to continue", so do so at that time.

---

## Using the jar program

1. Launch a command line interface tool to invoke the jar program. For example, to open a command prompt in Windows follow the appropriate guide below:

**Windows 2000**

Choose Start > Programs > Accessories > Command Prompt.

**End of Windows 2000**

**Windows**

Choose Start > Programs > Command Prompt.

**End of Windows**

2. Navigate to the directory in which IBM WebSphere InterChange Server is installed.

In Windows, use of the cd (change directory) command in conjunction with the %CROSSWORLDS% environment variable automatically navigates to the IBM WebSphere InterChange Server directory if the prompt currently resides somewhere on the same drive or partition where IBM WebSphere InterChange Server is installed. The two lines below illustrate how the sequence of prompts should appear if the cd command were used as described above:

```
D:\>cd %CROSSWORLDS%
```

```
D:\\WBIDev>
```

3. Use the jar program to extract the contactsynccollabs.jar file in the Samples\BenchMark>ContactSyncBenchmark subdirectory of the WebSphere InterChange Server directory into the WebSphere directory itself. The command below, for instance, provides the desired results in a Windows system. Press the Enter key after typing the command:

```
jar -xvf  
%CROSSWORLDS%\samples\benchmark\contactsyncbenchmark\contactsynccollabs.jar
```

The jar program inflates the compressed archive, creates the proper subdirectories as necessary, and copies the required files into the subdirectories.

4. Repeat step 3 above with the file named contactsyncmaps.jar file in the same directory.

---

## Import file system components

1. Launch a command line interface tool to invoke the jar program. For example, to open a Command Prompt in Windows follow the appropriate guide below:

**Windows 2000**

Choose Start > Programs > Accessories > Command Prompt.

**End of Windows 2000**

**Windows**

Choose Start > Programs > Command Prompt.

**End of Windows**

2. Navigate to the `DevelopmentKits\edk\ServerAccessInterfaces\AccessSample` subdirectory of the WebSphere directory.

In Windows, use of the `cd` (change directory command) in conjunction with the `%CROSSWORLDS%` environment variable automatically navigates to the WebSphere directory if the prompt currently resides somewhere on the same drive or partition where WebSphere InterChange Server is installed.

The two lines below illustrate how the sequence of prompts should appear if the `cd` command were used as described above:

```
D:\>cd %CROSSWORLDS%\DevelopmentKits\edk\ServerAccessInterfaces\AccessSample
```

```
D:\\CrossWorlDs400Dev\\DevelopmentKits\\edk\\ServerAccessInterfaces\\AccessSample>
```

3. Copy the contents of the `DevelopmentKits\edk\ServerAccessInterfaces\AccessSample\collaborations` subdirectory of the WebSphere InterChange Server installation into the `collaborations` subdirectory of the WebSphere installation.

The command below, for instance, would provide the desired results in a Windows system; press the Enter key after typing the command:

```
D:\\CrossWorlDs400Dev\\DevelopmentKits\\edk\\ServerAccessInterfaces  
  \\AccessSample>xcopy collaborations %CROSSWORLDS%\collaborations /s
```

4. Copy the contents of the `DevelopmentKits\edk\ServerAccessInterfaces\AccessSample\DLMS` subdirectory of the WebSphere InterChange Server installation into the `DLMS` subdirectory of the WebSphere installation.

The command below, for instance, would provide the desired results in a Windows system; press the Enter key after typing the command:

```
D:\\CrossWorlDs400Dev\\DevelopmentKits\\edk\\ServerAccessInterfaces  
  \\AccessSample>xcopy DLMS %CROSSWORLDS%\DLMS /s
```





---

## Chapter 5. Troubleshooting IBM WebSphere InterChange Server benchmarks

---

### File of input data not in location specified

```
[Time: 2002/03/14 18:05:14.800] [System: Server] [Thread: main (#5184781)]  
[Type: Fatal Error] [MsgID: 45063] [Mesg: A fatal error occurred attempting  
to open the file FileName for benchmark input data.  
The error message was: FileName (The system cannot find the file specified)]
```

---

### File of bad input data

```
[Time: 2002/03/14 19:09:30.084] [System: Server] [Thread: appPolling  
(#2458703)] [Type: Error] [MsgID: 45064] [Mesg: An error occurred while  
attempting to read business objects as input data for the benchmark.  
The error message was: {2}]
```



---

## Chapter 6. Notices

IBM may not offer the products, services, or features discussed in this document in all countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Burlingame Laboratory Director  
IBM Burlingame Laboratory  
577 Airport Blvd., Suite 800

Burlingame, CA 94010  
U.S.A

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement, or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurement may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not necessarily tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information may contain examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples may include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

#### COPYRIGHT LICENSE

This information may contain sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

---

## Programming interface information

Programming interface information, if provided, is intended to help you create application software using this program.

General-use programming interfaces allow you to write application software that obtain the services of this program's tools.

However, this information may also contain diagnosis, modification, and tuning information. Diagnosis, modification and tuning information is provided to help you debug your application software.

**Warning:** Do not use this diagnosis, modification, and tuning information as a programming interface because it is subject to change.

---

## Trademarks and service marks

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States or other countries, or both:

IBM  
the IBM logo  
AIX  
CICS  
CrossWorlds  
DB2  
DB2 Universal Database  
Domino  
IMS  
Informix  
iSeries  
Lotus  
Lotus Notes  
MQIntegrator  
MQSeries  
MVS  
OS/400  
Passport Advantage  
SupportPac  
WebSphere  
z/OS

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

MMX, Pentium, and ProShare are trademarks or registered trademarks of Intel Corporation in the United States, other countries, or both.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product or service names may be trademarks or service marks of others.







Printed in USA