Recording HTTP, JMS, and MQ Traffic for Test Generation

This topic explains how to record HTTP, JMS, and MQ traffic for test generation purposes.

Sections include:

- Recording Overview
- Recording Traffic with SOAtest
- HTTP Recording Configuration
- JMS Recording Configuration
- MQ Recording Configuration
- Configuring Queues for Recording

Recording Overview

Recording from Parasoft Virtualize

The traffic files generated by Parasoft Virtualize (through the thin client interface of CTP) can be used for test creation as well as virtual asset creation.

About SOAtest’s Recording Proxies

If you do not have Parasoft Virtualize, you can use SOAtest’s recording proxies to monitor and capture live HTTP, JMS, or MQ traffic from a service as an application is exercised. These proxies can concurrently capture traffic that passes through multiple endpoints.

The recording proxy monitors traffic over the specified transport(s) as an application is exercised. SOAtest "listens" to traffic requests and responses, then builds a traffic file of legitimate request/response pairs. This traffic is then used to generate a test suite that represents the captured behavior in preconfigured test client tools.

JMS, MQ, HTTP, HTTPS (SSL), Basic, Digest, and Kerberos authentication are supported; NTLM is not.

HTTP chunking and continue headers are not supported.

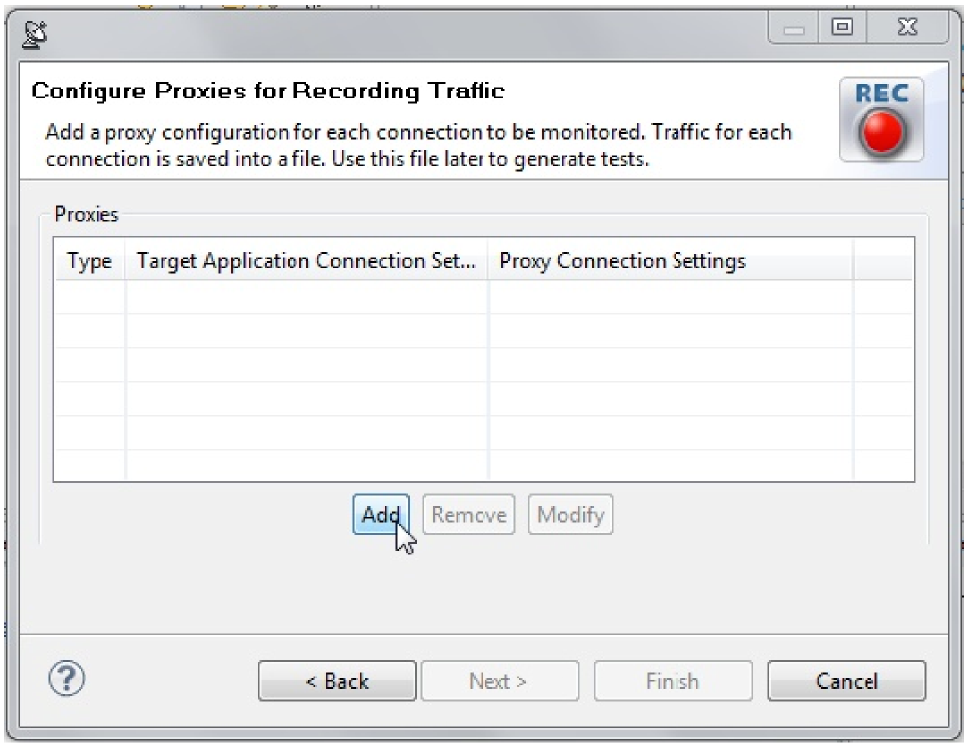
There are two main steps involved in recording and generating tests from traffic using this approach:

1. Capturing traffic in a file. You tell SOAtest how to connect and what you want it to monitor. With monitoring enabled, SOAtest builds a traffic file from the captured requests and responses.
    - For details, see below.
2. Creating tests from that traffic file.
    - For details, see Creating Parameterized Message Test Clients from Traffic and Creating Fixed Message Test Clients from Traffic.

Recording Traffic with SOAtest

To capture live traffic across one or more endpoints:

1. Choose **File> New> Other> SOAtest> Traffic> Record Traffic**.
2. For each endpoint where you want to record traffic, do the following:
    a. In the Configure Proxies for Recording Traffic dialog, click **Add**.



Configure Proxies for Recording Traffic

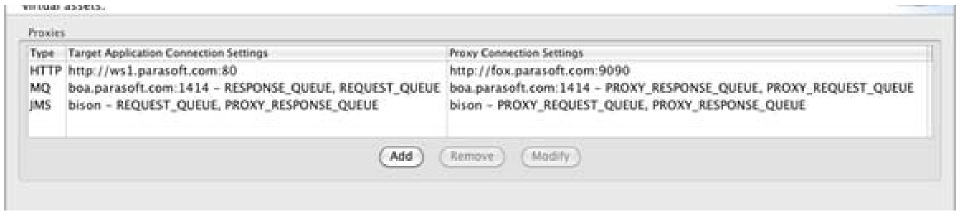
Add a proxy configuration for each connection to be monitored. Traffic for each connection is saved into a file. Use this file later to generate tests.

REC
Proxies
Type
Target Application Connection Set...
Proxy Connection Settings
Add
Remove
Modify
< Back
Next >
Finish
Cancel

The wizard that opens will be pre-populated with any connections you have already configured.

b. Under **Proxy Type**, select the desired transport (HTTP, JMS, MQ).
c. Complete the proxy settings for the selected transport. See Customizing Grouping Criteria, JMS Recording Configuration, or MQ Recording Configuration for details.
d. In the **Traffic file** field, specify where you want to save the traffic file that will be created to capture this traffic. You can later use this traffic file to generate tests that represent the live traffic captured.
e. Specify how you want traffic data recorded in traffic files:
   - **Append new session data** adds new traffic data to an existing traffic file (the one specified in the **Traffic file** field). If the specified file does not already exist, a new file will be created.
   - **Overwrite session data** overwrites the traffic data in an existing traffic file (the one specified in the **Traffic file** field). If the specified file does not already exist, a new file will be created.

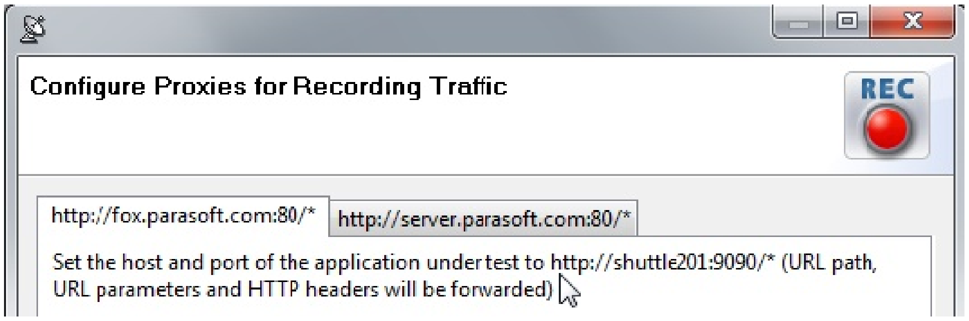
3. Click **OK**.



Proxies

Type	Target Application Connection Settings	Proxy Connection Settings
HTTP	http://ws1.parasoft.com:80	http://fox.parasoft.com:9090
MQ	boa.parasoft.com:1414 - RESPONSE_QUEUE, REQUEST_QUEUE	boa.parasoft.com:1414 - PROXY_RESPONSE_QUEUE, PROXY_REQUEST_QUEUE
JMS	bison - REQUEST_QUEUE, PROXY_RESPONSE_QUEUE	bison - PROXY_REQUEST_QUEUE, PROXY_RESPONSE_QUEUE

4. In the main wizard panel, click **Next**.
5. Set the host and port of the application under test to the location specified in the wizard.



Configure Proxies for Recording Traffic

http://fox.parasoft.com:80/*

http://server.parasoft.com:80/*

Set the host and port of the application under test to http://shuttle201:9090/* (URL path, URL parameters and HTTP headers will be forwarded)

6. From the application under test, generate the traffic that you want to record.
7. Click **Finish**.

When live traffic is being captured, you can switch between the available tabs to view the requests and responses at each endpoint.

Once traffic is captured, you have two options for generating tests:

- Creating Parameterized Message Test Clients from Traffic
- Creating Fixed Message Test Clients from Traffic

HTTP Recording Configuration

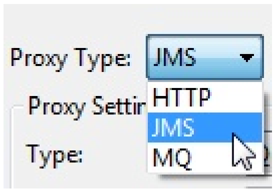
This section explains how to configure recording proxies that send and receive messages over the HTTP transport.

HTTP, HTTPS (SSL), Basic, Digest, and Kerberos authentication are supported; NTLM is not.

Specifying HTTP Settings for the Recording Proxy

In the recording proxy wizard, specify your HTTP settings as follows:

1. In the Proxy (Client) and Target Application (Server) Connection Settings dialog, choose **HTTP** for **Proxy Type**.



2. Complete the appropriate HTTP settings.
   - For server-side SSL, be sure to check **Enable server side SSL**.
   - For two-way SSL, be sure to check **Enable client side SSL** and complete the **Certificate and Private Key** settings. Enabling client-side SSL will enable server-side SSL by default.

Server-Side SSL Setup

Due to the nature of SSL, SOAtest's proxy for HTTP recording generates a dynamic server certificate signed by its own certificate authority. In order to accept this dynamic server certificate, the client gen-erating the requests over HTTPS will need to be set up to trust all certificates. In addition, SOAtest must be configured to trust the certificate presented by the service. To do this:

1. Ensure that the Server Certificate setting for SOAtest is set with **Trust all certificates** enabled (in **Parasoft> Preferences> Parasoft> Security**) or that the service's Server Certificate is properly added to the SOAtest cacerts file (see Configuring for Services Deployed Over HTTPS for more details).

Two-Way SSL Setup

Due to the nature of SSL, SOAtest's proxy for HTTP recording generates a dynamic server certificate signed by its own certificate authority. In order to accept this dynamic server certificate, the client generating the requests over HTTPS will need to be set up to trust all certificates. In addition, SOAtest must be configured to trust the certificate presented by the service as well as to send the appropriate client certificate. To do this:

1. Ensure that the Server Certificate setting for SOAtest is set with **Trust all certificates** enabled (in **Parasoft> Preferences> Parasoft> Security**) or that the service's Server Certificate is properly added to the SOAtest cacerts file (see Configuring for Services Deployed Over HTTPS for more details).
2. Make sure you have the client certificate keystore file (and— if the client certificate and private key are stored in different keystores—the private key keystore file) as well as relevant keystore passwords, keystore type information, private key password, and the name of the alias being used for the certificate/private key.

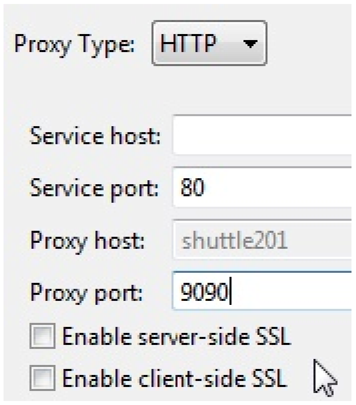
Capturing the Traffic

To generate a traffic file that captures HTTPS traffic from a service that uses server-side or two-way SSL, complete the wizard as described in Capturing the Traffic (above).

In the HTTP wizard page, be sure to enable the appropriate SSL option:

- For server-side SSL, be sure to check **Enable server-side SSL**.
  For two-way SSL, be sure to check **Enable client-side SSL** and complete the **Certificate and Private Key** settings. Enabling client-side SSL will

enable server-side SSL by default.


Proxy Type: HTTP

Service host:
Service port: 80
Proxy host: shuttle201
Proxy port: 9090
Enable server-side SSL
Enable client-side SSL

JMS Recording Configuration

This section explains how to configure recording proxies that send and receive messages over the JMS transport.

Sections include:

- JMS Prerequisites
- Specifying JMS Settings

JMS Prerequisites

JNDI

See JMS Prerequisites.

Queue Allocation

When traffic is being exchanged over queues, the assumption is that there is a client application that sends a request to a client destination queue and a server application that receives that message from the queue, then sends another (reply) message onto a second queue for the client to receive.

With that scenario in mind, Parasoft SOAtest uses a "man-in-the-middle" (a.k.a. proxy) approach between the client and the server, so two extra queues are needed on the messaging provider in order to facilitate that mediation. SOAtest proxies will receive messages from where the client places them, record the message contents (if recording is enabled), then put the message on the queue where the server will receive it from. Similarly, the server sends the message to a queue where the proxy picks it up, records it (if recording is enabled), and places it back on the queue where the client is expecting the reply response.

As a result, it is necessary to allocate two additional queues and either:

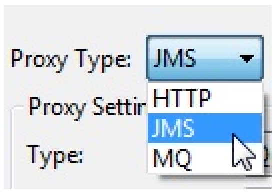
- Adjust the client application to communicate over these two new queues (with the proxy connections configured for them)
- Adjust the server application to use the two other queues.

Only one of the two application queues needs to be modified.

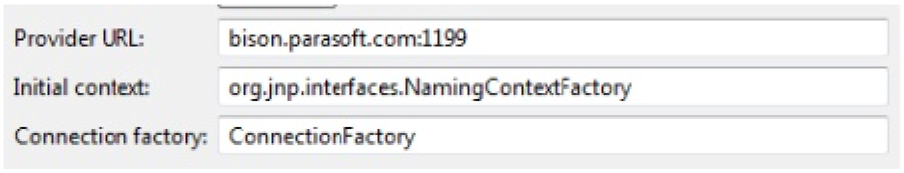
Specifying JMS Settings

Specify your JMS settings as follows:

1. In the Proxy (Client) and Target Application (Server) Connection Settings dialog, choose **JMS** for **Proxy Type**.



2. Specify whether you're using JMS Queues or Topics, the Provider URL, Initial context, and connection factory (be sure to add the related jars to the SOAtest classpath), as well as any additional initial context JNDI properties, authentication, and the queues/topics you want to monitor to collect traffic.

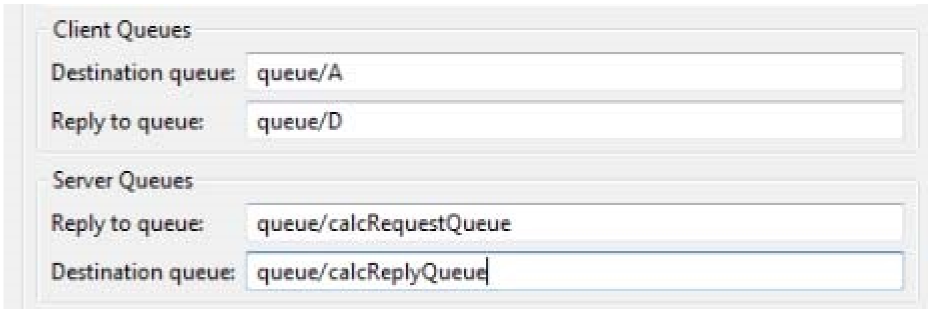


Provider URL:	bison.parasoft.com:1199
Initial context:	org.jnp.interfaces.NamingContextFactory
Connection factory:	ConnectionFactory

Use JMSReplyTo for Response

This option specifies whether to use the message's JMSReplyToQueueName header to determine where the proxy sends the response. If **Us e JMSReplyToQueueName for Response** is enabled, values from the incoming request will be used to determine where to send the response. If it is not enabled, the response will be sent to the queue specified in the UI; the values in the JMS message header will be ignored.

- **For Queues**: SOAtest will capture messages sent to the client **Destination queue** and forward them to the server **Reply to queue** for processing. The server **Destination queue** is the queue that the server will place a response message on (after proessing the request message). SOAtest will capture these messages and forward them to the client **Reply to queue**. See Configuring Queues for Recording f or more details.



Client Queues

Destination queue:	queue/A
Reply to queue:	queue/D

Server Queues

Reply to queue:	queue/calcRequestQueue
Destination queue:	queue/calcReplyQueue

- **For Topics**: SOAtest monitors incoming requests on the client **Subscribe topic** and outgoing responses on the server **Publish topic**.

MQ Recording Configuration

This section explains how to configure recording proxies that send and receive messages over the MQ transport.

Sections include:

- MQ Prerequisites
- Specifying MQ Settings

MQ Prerequisites

Jar Files

See Adding Necessary Jar Files to the Classpath.

Queue Allocation

When traffic is being exchanged over queues, the assumption is that there is a client application that puts a request to a client destination queue and a server application that gets that message from the queue then puts another (reply) message onto a second queue for the client to get.

With that scenario in mind, Parasoft SOAtest uses a "man-in-the-middle" (a.k.a. proxy) approach between the client and the server, so two extra queues are needed on the messaging provider in order to facilitate that mediation. SOAtest proxies will receive messages from where the client places them, record the message contents (if recording is enabled), then put the message on the queue where the server will receive it from. Similarly, the server sends the message to a queue where the proxy picks it up, records it (if recording is enabled), and places it back on the queue where the client is expecting the reply response.

As a result, it is necessary to allocate two additional queues and either:

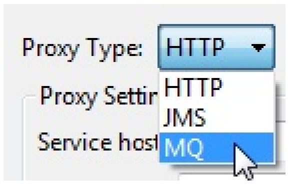
- Adjust the client application to communicate over these two new queues (with the proxy connections configured for them)
- Adjust the server application to use the two other queues.

Only one of the two application queues needs to be modified.

Specifying MQ Settings

Specify your MQ settings as follows:

1. In the Proxy (Client) and Target Application (Server) Connection Settings dialog, choose **MQ** for **Proxy Type**.



2. Specify the MQ settings in the available fields. See below for more configuration details.

Use replyToQueueName for Response

This option specifies whether to use the message’s replyToQueueName header to determine where the proxy sends the response. It impacts responses to MQ messages of type MQMT_REQUEST.

If **Use replyToQueueName for Response** is enabled, values from the incoming request will be used to determine where to send the response. If it is not enabled, the response will be sent to the queue specified in the UI. More specifically:

- If **Use replyToQueueName for Response** is enabled and values for both MQMD.replyTo-QueueManagerName and MQMD. replyToQueueName have been specified, those values will determine both the queue manager and the queue name to send the response to.
- If **Use replyToQueueName for Response** is enabled and either MQMD.replyToQueueM-anagerName or MQMD.replyToQueueName is missing from the request, the values specified in the UI will be used in place of the missing value.
- If **Use replyToQueueName for Response** is disabled, the replyToQueueName and reply-ToQueueManagerName fields in the MQ request message will be ignored. The UI settings will determine where the message is sent.

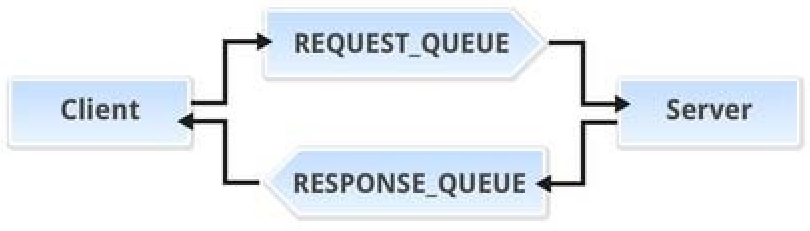
Configuring Queues for Recording

This section explains how to configure recording for WebSphere MQ and JMS point-to-point (when queues are used, not topics).

Overview

The method that Parasoft SOAtest uses for capturing/recording JMS point-to-point messaging and WebSphere MQ requires SOAtest to be a proxy— or, in other words, an intermediary / man-in-the-middle.

Assume that there is a client/consumer application (we’ll call this "Client") that sends/puts/pushes a request message on destination queue REQUEST_QUEUE and receives/gets/pulls a response/reply from RESPONSE_QUEUE. Also assume there is a server/provider application (we’ll call this "Server") that receives/gets/pulls messages from REQUEST_QUEUE and sends/puts/pushes responses to RESPONSE_QUEUE.



There are two possible configuration scenarios for recording traffic. In one, you modify the queues that are used on the client application end. In the other, you modify the queues that are used on the server.

The best approach to use depends on your ability to access and modify each end. In both cases, you will need to create two extra queues in order to enable the recording process. Here, we’ll label them as PROXY_REQUEST_QUEUE and PROXY_RESPONSE_QUEUE.

Using Topics Instead of Queues?

If a publish and subscribe scheme is used (using Topics instead of queues), then you don't need to introduce new topics into the system. That's why only two destinations need to be provided in the wizard:

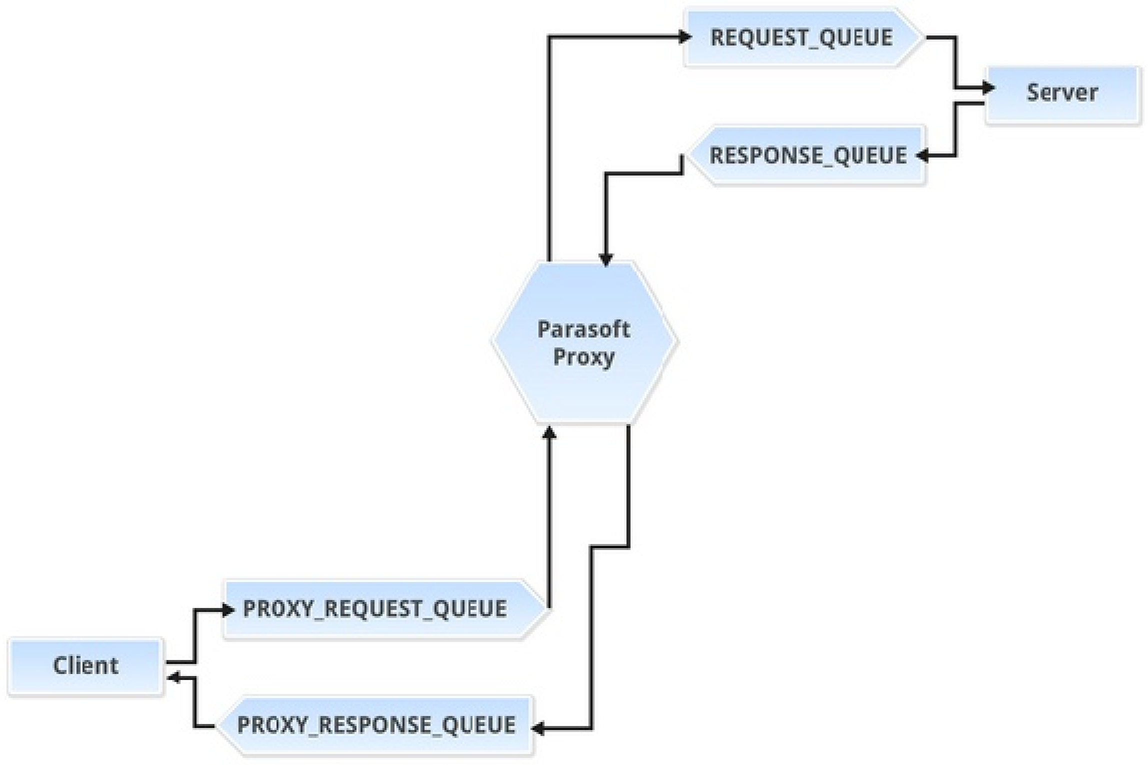
- One for the requests—labeled in the wizard as the "Client Subscribe Topic."
- One for the response—labeled in the wizard as the "Server Publish Topic."

Scenario 1: Modifying the Client Application

In this scenario, you need access to the client application deployment configuration in order to adjust the queues it uses. Modify the queues that the client uses to communicate with the server as follows:

- put/send/push messages on PROXY_REQUEST_QUEUE
- receive/get/pull messages from PROXY_RESPONSE_QUEUE

The server does not need to have its queues changed in this case.



In the MQ or JMS recording wizard, provide the queue names as follows:

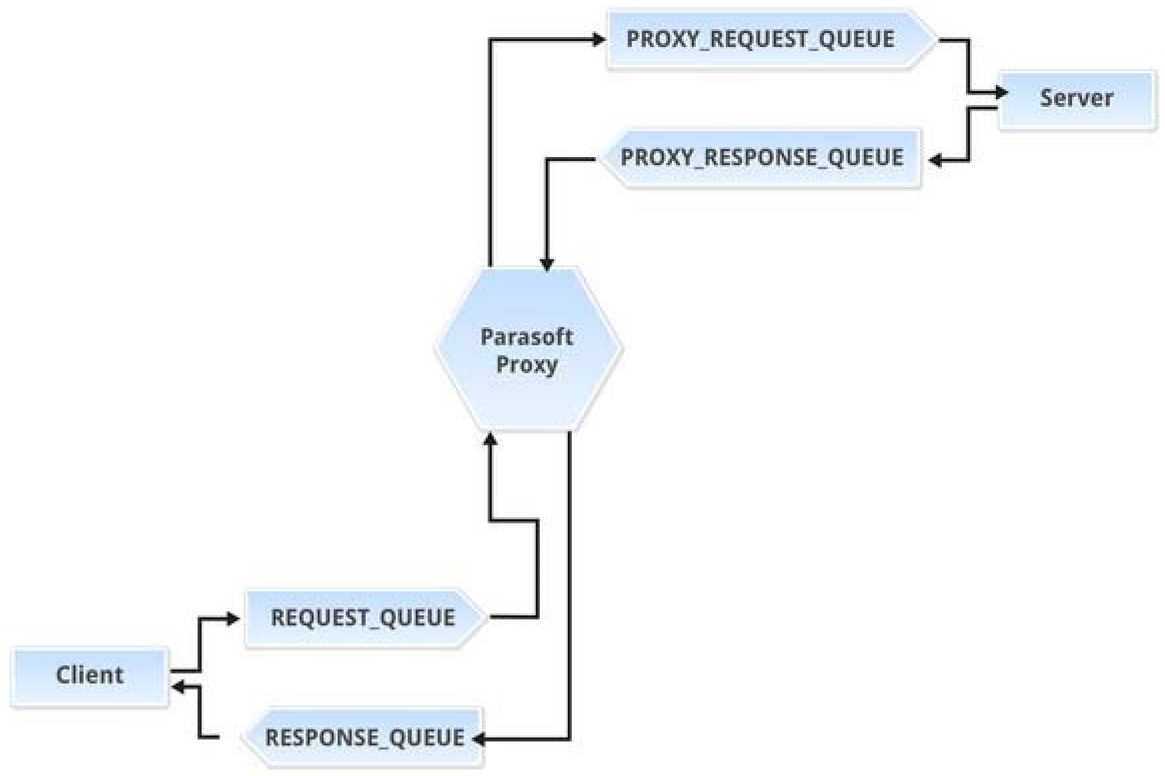
Client Queues	
Destination/Put Queue	PROXY_REQUEST_QUEUE
Reply to/Get Queue	PROXY_RESPONSE_QUEUE
Server Queue	
Reply to/Get Queue	REQUEST_QUEUE
Destination/Put Queue	RESPONSE_QUEUE

Scenario 2: Modifying the Server Application

In this scenario, you need access to the server application deployment configuration in order to adjust the queues it uses. Modify the queues that the server uses to communicate with the client as follows:

- receive/get/pull messages from PROXY_REQUEST_QUEUE
- put/send/push response messages on PROXY_RESPONSE_QUEUE

The client does not need to have its queues changed in this case.



In the MQ or JMS recording wizard, provide the queue names as follows:

Client Queues	
Destination/Put Queue	REQUEST_QUEUE
Reply to/Get Queue	RESPONSE_QUEUE
Server Queue	
Reply to/Get Queue	PROXY_REQUEST_QUEUE
Destination/Put Queue	PROXY_RESPONSE_QUEUE