Creating Message Responders from Traffic Files - Overview

This topic explains how to virtualize traffic that is captured in traffic logs.

Sections include:

Introduction

You can virtualize application behavior captured in traffic logs, including:

- Logs produced by Virtualize message proxies or recording proxies (e.g., for JMS, MQ, HTTP).
- Logs produced from other sources; for example:
  - Message traces or logs captured at the network level using network sniffing tools.
  - HTTP logs obtained by having your application log its traffic.

Virtualization is achieved by using a wizard to create and deploy Message Responders within a virtual asset.

Supported Traffic Log Formats

Virtualize supports the following traffic log formats (the file extension does not matter).

HTTP Traffic Logs

For the HTTP parser type, Virtualize can recognize and parse HTTP traffic with POST, GET, and PUT methods. DELETE is not supported. Virtualize expects the log to contain each HTTP header, followed with message body content:

```
HTTP request 1 headers
HTTP request 1 body
HTTP response 1 headers
HTTP response 1 body
HTTP request 2 headers
HTTP request 2 body
HTTP response 2 headers
HTTP response 2 body
...
```

Virtualize also expects there to be two line endings (Windows or Unix) between a message body and the next HTTP header.

This format is similar to what would be captured by the Virtualize message proxy, recording wizard, or other HTTP traffic sniffing tools that can record HTTP traffic as it gets transmitted over the HTTP socket connection. HTTP chunking and traffic files that are recorded with chunked traffic are supported. However, raw HTTP traffic files that include chunked messages or "continue" headers are not supported.

Values can be encoded or decoded. For instance, if a request comes in at path /some/url?q=harry+potter, this should correlate the same as /some/url?q=harry%20potter, even if the "q" URL Parameter Correlation is set to 'harry potter'. If you are using encoded values and data source correlations, see the notes for Data Source Correlation Tab.

JMS and MQ Traffic Logs

Virtualize supports the creation of .pvas from JMS and MQ traffic files produced by Virtualize message proxies and the traffic recording wizard.

Message Formats

For parameterized messages, the traffic files can include any message format that Virtualize supports (including EDI, JSON, fixed length, and custom message formats). For fixed messages, only XML and JSON are supported.

Constructing a Traffic File Based on Sample Requests and Responses

If you want to create a virtual asset based on sample requests/responses, you can aggregate the requests/responses into a single traffic file, then create virtual assets from that traffic file.

The following sample shows the format for such a traffic file:

```
POST /SomePath HTTP/1.0
Content-Type: text/xml; charset=UTF-8

[Request 1 Message]

HTTP/1.1 200 OK
Content-Type: text/xml; charset=UTF-8

[Response 1 Message]

POST /SomePath HTTP/1.0
Content-Type: text/xml; charset=UTF-8

[Request 2 Message]

HTTP/1.1 200 OK
Content-Type: text/xml; charset=UTF-8

[Response 2 Message]
```

Specifying Depth of Cycles

When traffic has cycles in which the same element is a child of itself, Virtualize stops populating form input at the first level. But you can set the `RepositorySchemaCreator.cycle.limit` system property on startup to specify how many cycles deep to populate the form input. For example:

`-J-DRepositorySchemaCreator.cycle.limit=5`

Using Parameterized vs. Fixed Messages

Virtualize provides two options for generating Message Responders from traffic logs:

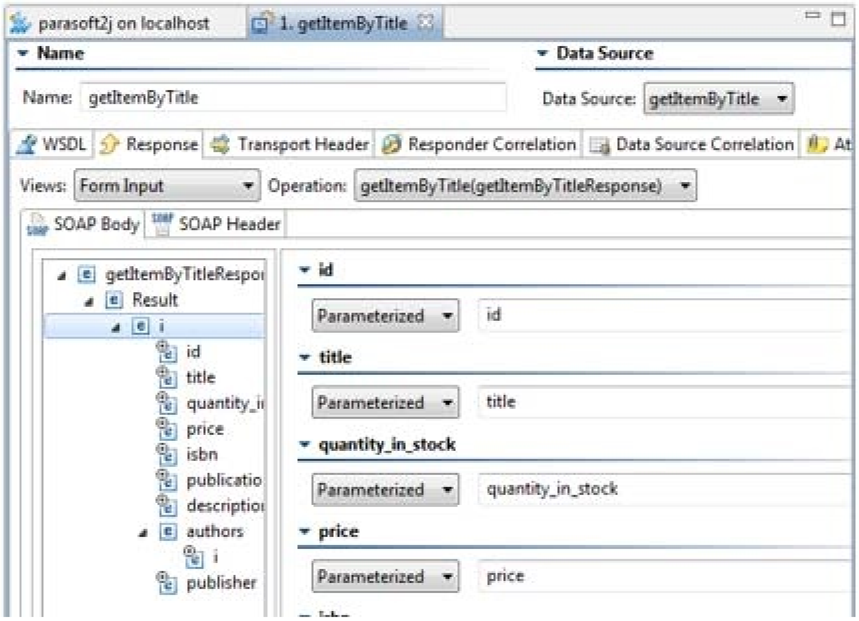
- **Generate parameterized messages** creates Message Responders that are automatically parameterized with values from a Parasoft Data Repository (which is also automatically created when you use this option). This option allows you to rapidly add and modify response message values using a graphical interface designed for manipulating large, complex data sources. Data values are stored and manipulated independently of the Message Responder and .pva. This process is described in Creating Fixed Message Responders from Traffic.
- **Generate fixed messages** creates Message Responders that specify each response literally via Multiple Response format. All response message data is stored and manipulated within the Message Responder. Response message elements can be parameterized with data extracted from incoming messages. This process is described in Creating Fixed Message Responders from Traffic.

Example

To understand the difference, consider the following example from a simple HTTP traffic file with 9 messages, including 3 messages for the getItemByTitle operation.

Parameterized Messages

With "generate parameterized messages," Virtualize configures the Message Responder’s getItemByTitle responses as follows.



Name

Name: getItemByTitle

Data Source

Data Source: getItemByTitle

Views: Form Input

Operation: getItemByTitle(getItemByTitleResponse)

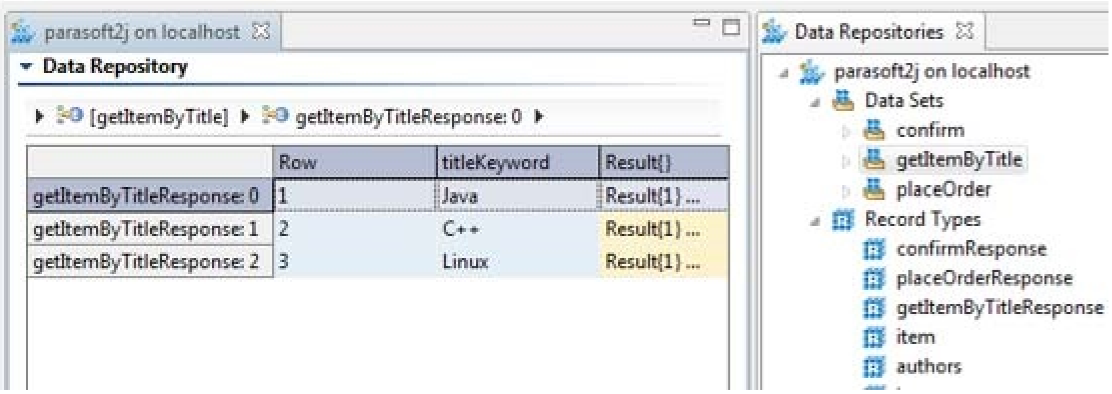
id: Parameterized — id

title: Parameterized — title

quantity_in_stock: Parameterized — quantity_in_stock

price: Parameterized — price

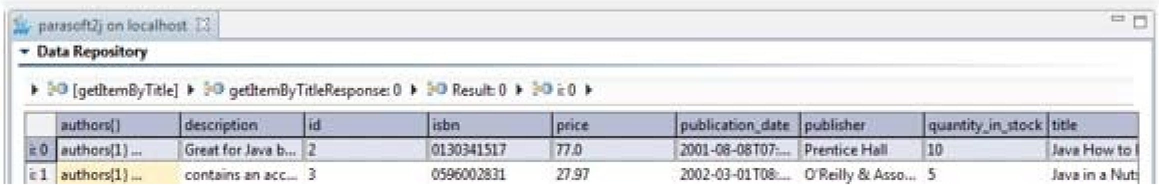
Additionally, it creates the following editable and extensible Data Repository.



Data Repository

[getItemByTitle] ▸ getItemByTitleResponse: 0 ▸

	Row	titleKeyword	Result{}
getItemByTitleResponse: 0	1	Java	Result{1} ...
getItemByTitleResponse: 1	2	C++	Result{1} ...
getItemByTitleResponse: 2	3	Linux	Result{1} ...



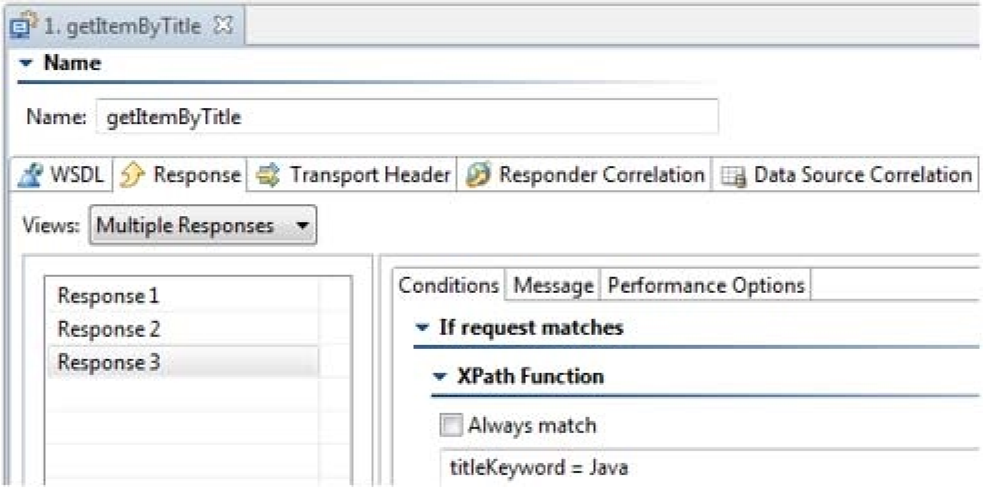
Data Repository

[getItemByTitle] ▸ getItemByTitleResponse: 0 ▸ Result: 0 ▸ i: 0 ▸

	authors{}	description	id	isbn	price	publication_date	publisher	quantity_in_stock	title
i: 0	authors{1} ...	Great for Java b...	2	0130341517	77.0	2001-08-08T07:...	Prentice Hall	10	Java How to
i: 1	authors{1} ...	contains an acc...	3	0596002831	27.97	2002-03-01T08:...	O'Reilly & Asso...	5	Java in a Nut

Fixed Messages

With "generate fixed messages," Virtualize configures the Message Responder’s getItemByTitle responses as follows.



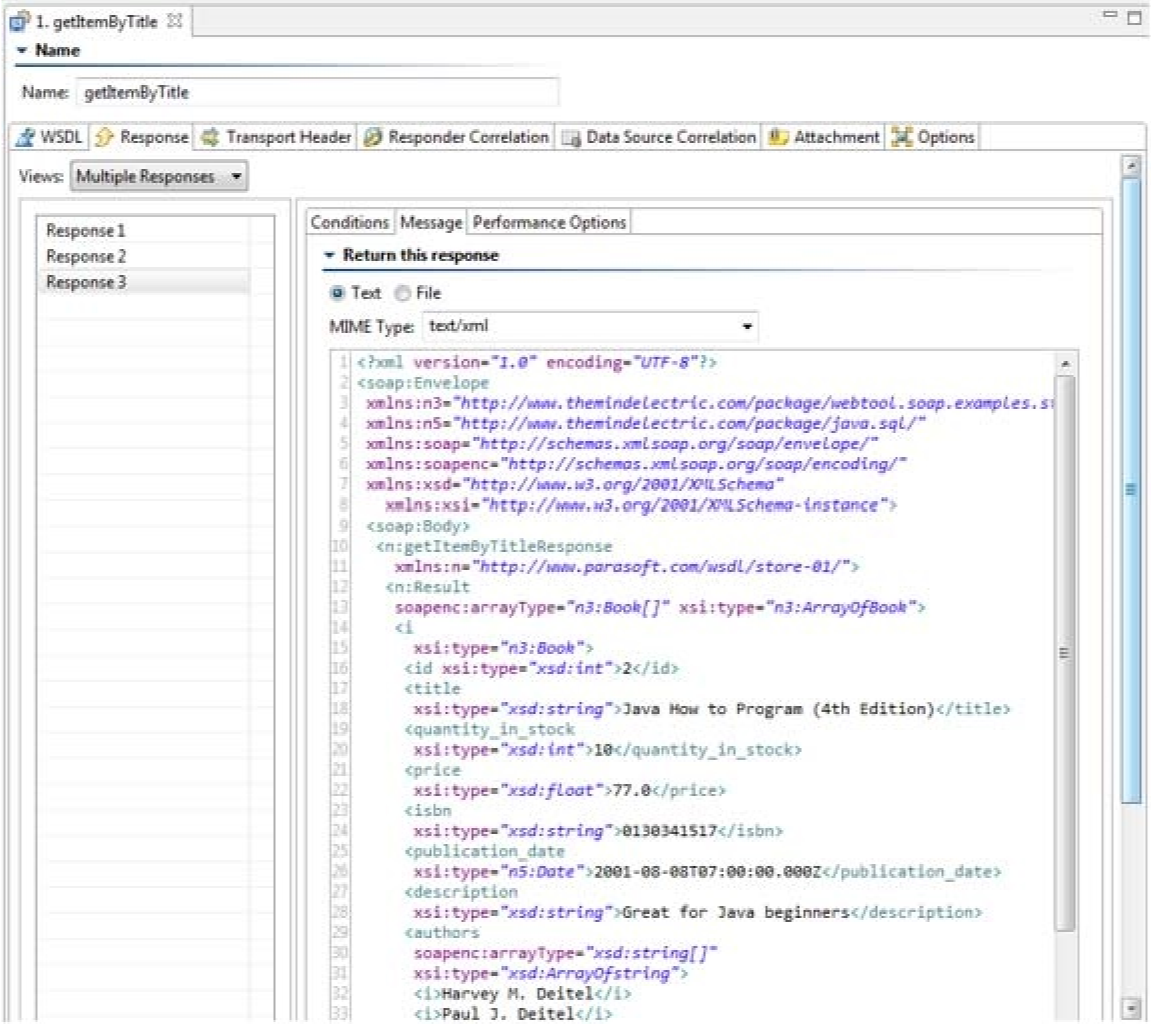
Name: getItemByTitle

Views: Multiple Responses

Response 1
Response 2
Response 3

□ Always match

titleKeyword = Java



Name: getItemByTitle

Views: Multiple Responses

Response 1
Response 2
Response 3

● Text ○ File

MIME Type: text/xml

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Envelope
 xmlns:n3="http://www.themindelectric.com/package/webtool.soap.examples.s
 xmlns:n5="http://www.themindelectric.com/package/java.sql/"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
 <soap:Body>
  <n:getItemByTitleResponse
    xmlns:n="http://www.parasoft.com/wsdl/store-01/">
   <n:Result
    soapenc:arrayType="n3:Book[]" xsi:type="n3:ArrayOfBook">
    <i
      xsi:type="n3:Book">
     <id xsi:type="xsd:int">2</id>
     <title
      xsi:type="xsd:string">Java How to Program (4th Edition)</title>
     <quantity_in_stock
      xsi:type="xsd:int">10</quantity_in_stock>
     <price
      xsi:type="xsd:float">77.0</price>
     <isbn
      xsi:type="xsd:string">0130341517</isbn>
     <publication_date
      xsi:type="n5:Date">2001-08-08T07:00:00.000Z</publication_date>
     <description
      xsi:type="xsd:string">Great for Java beginners</description>
     <authors
      soapenc:arrayType="xsd:string[]"
      xsi:type="xsd:ArrayOfstring">
      <i>Harvey M. Deitel</i>
      <i>Paul J. Deitel</i>
```

Video Tutorial: Creating Virtual Assets from Sample Request/Response Pairs

In this video you'll learn how to create a virtual asset from a sample piece of XML.

</p> </div> </div> </body> </html>