XML Assertor

This topic covers the XML Assertor tool in SOAtest and Virtualize. This tool enforces the correctness of data in an XML message. This tool requires a validate license.
Sections include:

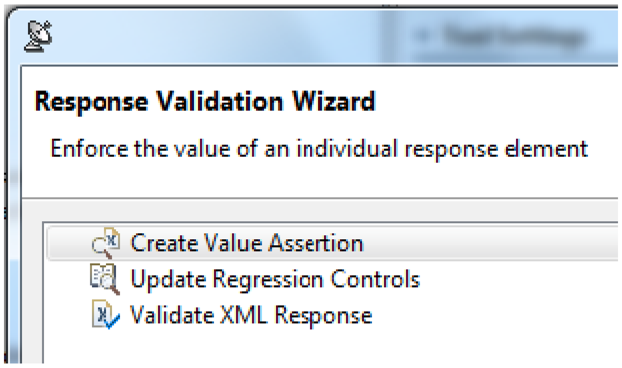
Understanding XML Assertor

This tool is most commonly connected to a SOAP Client or Messaging Client in order to verify the data returned by a service. XML Assertor provides support for complex message validation needs without the need for scripting, and allows you to easily create and maintain validation assertions on your XML messages.

When verifying results from a service, you may want to apply specific rules conformant to your business requirements. It may not be sufficient to verify that changes occurred; you may also want to define complex rules defining what constitutes an acceptable XML message.

XML Assertor is designed to provide a range of assertions which can enforce more logical and business-appropriate rules for message validity.

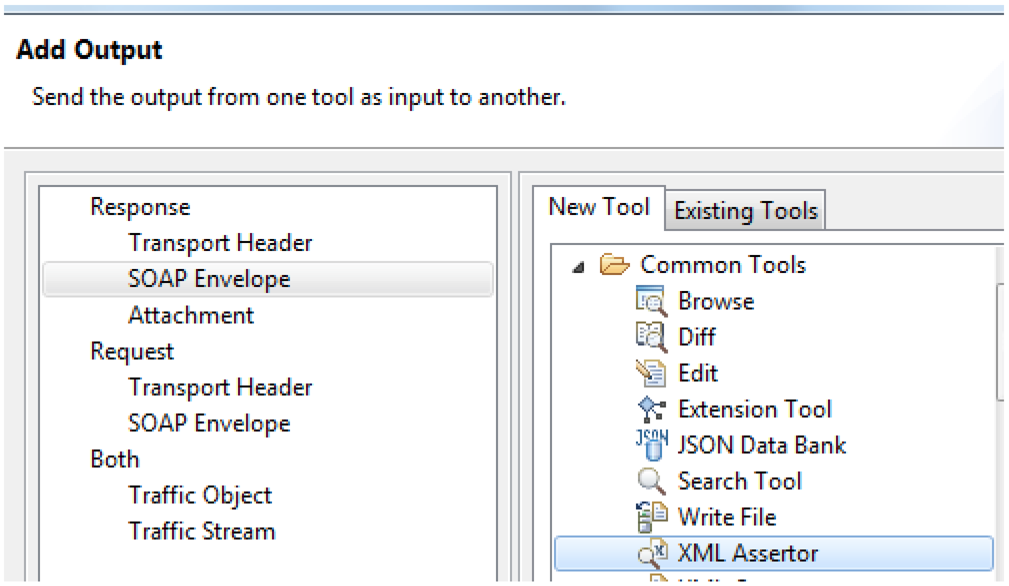
This tool is commonly added from the Create/Update Regression Control dialog for XML messages (by choosing the **Create Value Assertion** option). For details on adding XML Assertors in this manner, see Validating the Value of an Individual Response Element.



Response Validation Wizard

Enforce the value of an individual response element

It can also be added via the Add Output wizard, which is described in Adding Test Outputs.



Add Output

Send the output from one tool as input to another.

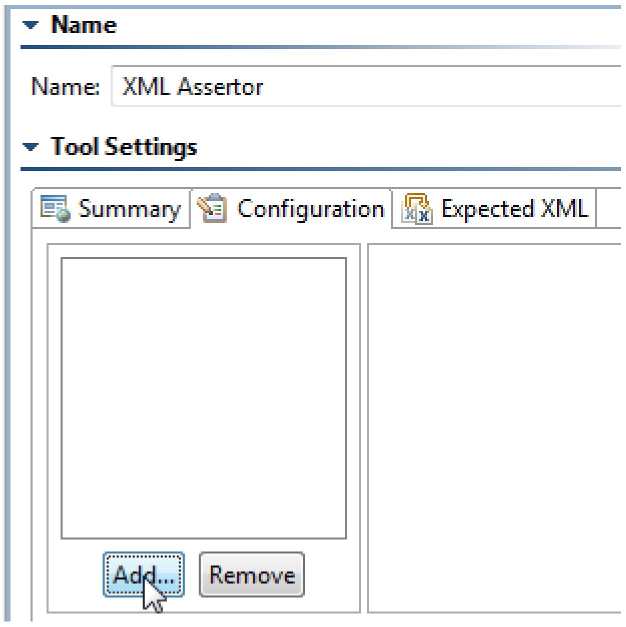
Configuring XML Assertor

XML Assertor consists of three main tabs:

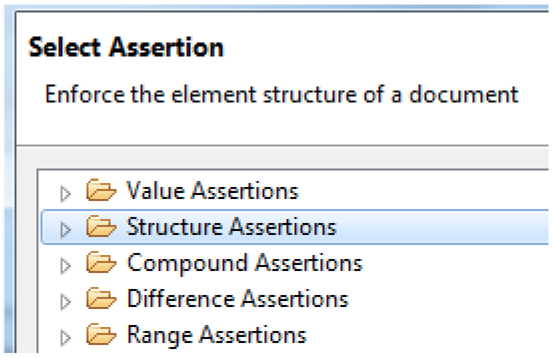
- **Summary:** This tab contains a table showing the details of the XML assertions that have been configured.
- **Configuration:** This tab is used to create and configure XML assertions.
- **Expected XML:** Specifies the expected XML response, creating a template from which you can select elements. If this tool receives a valid XML message (e.g., from traffic or as defined in the client tool it is attached to), this panel will be populated automatically. Alternatively, you can copy a sample message into the Literal or Tree tabs. Note that the expected XML does not get saved by default; if you want to save it, enable the **Save Expected XML** option.

To configure XML Assertor:

1. Click the **Add** button in the XML Assertor's Configuration tab.

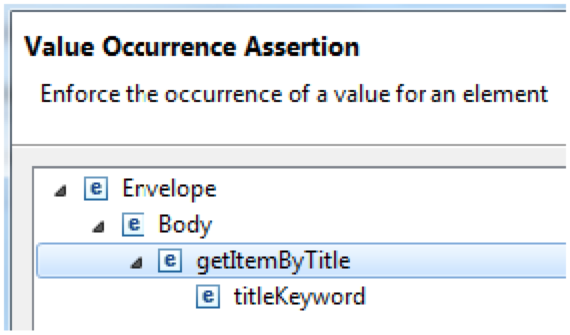
   

   The **Select Assertion** wizard displays.

2. Select an assertion type. The following is a brief summary of the available types of assertions.
   - **Value Assertions:** The following value assertions are available:
     - **Value Assertion:** Enforce the value of a particular element.
     - **Value Occurrence Assertion:** Enforce a certain number of occurrences of an element with a given value (e.g., that the document must have n matches on both the XPath selector and the specified value string).
     - **Numeric Assertion:** Enforce the numeric value of an element.
     - **String Comparison Assertion:** Enforce the value of the text content of a given element.
     - **Regular Expression Assertion:** Enforce that an element matches a regular expression.
     - **Expression Assertion:** Enforce the value of an expression composed of elements.
     - **Custom Assertion:** Enforce the value of an element by scripting custom logic.
   - **Structure Assertions:** The following structure assertions are available:
     - **Occurrence Assertion:** Enforce the number of occurrences of an element.
     - **Has Content Assertion:** Enforce that an element has text content (i.e., length of text > 0).
     - **Has Children Assertion:** Enforce that an element has one or more child elements.
   - **Compound Assertions:** The following compound assertions are available:
     - **AND Assertion:** Group multiple assertions that all must succeed.
     - **OR Assertion:** Group multiple assertions where at least one must succeed.

        - **Conditional Assertion:** Enforce an assertion only if a condition is met (where the condition is a combination of previously-defined assertions).
    - **Difference Assertions:** The following difference assertions are available:
        - **Numeric Difference Assertion:** Enforce a numeric difference on a value of a particular element. Assert that the numeric value of an element differs from a user-specified base value by a user-specified value. For example, in order to assert that a value in degrees Fahrenheit is 3 degrees below freezing, you would set the Base Value to 32 and the Difference Value to -3.
        - **Date Difference Assertion:** Enforce a date difference on a value of a particular element.
        - **DateTime Difference Assertion:** Enforce a date time difference on a value of a particular element.
    - **Range Assertions:** The following difference assertions are available:
        - **Numeric Range Assertion:** Enforce that the numeric value of an element is within the inclusive bounds of a numeric range.
        - **Date Range Assertion:** Enforce a date range on a value of a particular element
        - **DateTime Range Assertion:** Enforce a date time range on a value of a particular element
3. Click the **Next** button. A tree view displays.


Value Occurrence Assertion

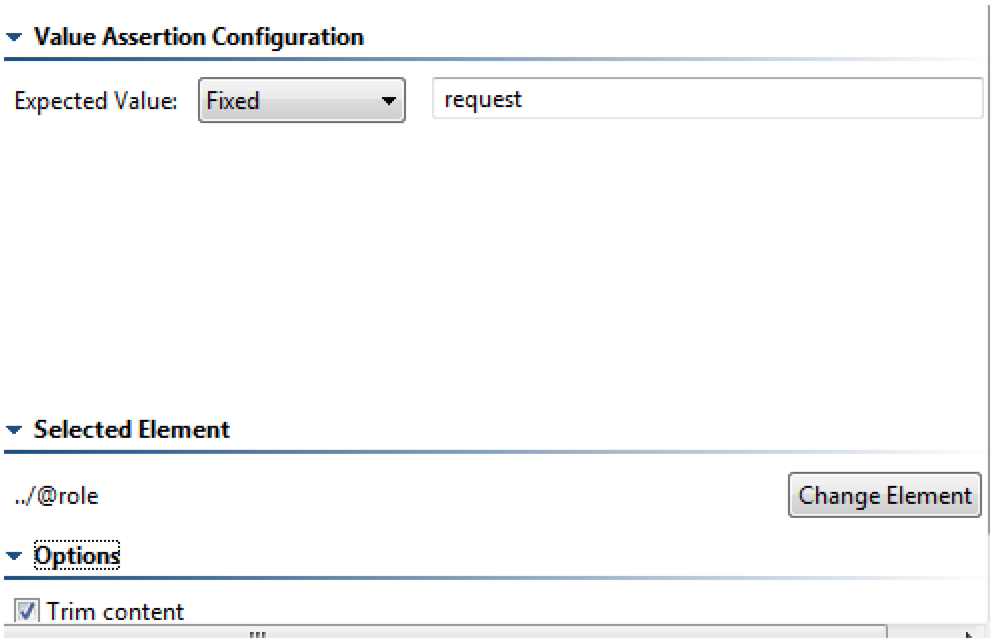
Enforce the occurrence of a value for an element

4. Select the element that you want this assertion to check, then click the **Finish** button. Note that you can edit the structure of this tree in the XML Assertor tool's **Expected XML** tab.

   By default, XPath fields are case sensitive, but you can change case sensitivity by specifying the applicable system property when starting Virtualize. See Case Sensitivity for details.

You may add additional assertions to apply to the message (such as a Numeric assertion to enforce on the price element) by clicking the **Add** button in the **Configuration** tab.

If you later want to modify the element referenced by an assertion, click **Change Element,** which is at the bottom right of the **Configuration** tab. This opens a dialog that lets you graphically or manually edit the given element. You can click the **Evaluate XPath** button to see the result of applying the XPath expression against the expected XML.


Value Assertion Configuration

Expected Value: Fixed request

Selected Element

../@role

Options

☑ Trim content

Note that the **Trim content** option will remove any white space from the start and end of the extracted string before comparing it to the expected text. For example, if " bar " was extracted (ignore quotes in all examples; they are used to show white spaces), it would become "bar"; this would match "bar" (and fail to match " bar ") if the **Trim content** option was not enabled.

If you have XML like the following

```
<foo>
bar
</foo>
```

where the new-line characters before and after the "bar" value are uninteresting and mostly a matter of formatting, you might want to enable this option so you don’t have to account for the format of the XML when creating an extraction.

Parameterizing XPaths

You can parameterize XPaths to reference test or Responder suite variables, environment variables, and data source values. The syntax to reference variables is ${myVariableName}. The syntax to reference XML Data Bank values and Data Source Values is: ${myColumnName}.

For example, if ${XPath Key} is a data source column name, you could use

/*[local-name(.)="bookstore" and namespace-uri(.)=""]/*[local-name(.)="book" and

namespace-uri(.)=""][child::node()[local-name(.)="title" and text()="${XPath Key}"]]

Video Tutorial

In this video, you'll learn how to add targeted assertions for values in XML responses.

</p></div></p><h1 id="XMLAssertor-RelatedTutorials"><span style="font-size: 20.0px;">Related Tutorials</span></h1><p>The following tutorial lessons demonstrate how to use this tool:</p><ul><li><a href="/display/SOAVIRT20211/Service+Functional+Testing">Service Functional Testing</a></li><li><a href="/display/SOAVIRT20211/Advanced+Strategies#AdvancedStrategies-ExtendingSOAtestwithScripting">Extending SOAtest with Scripting</a></li><
/ul><p /> </div> </div> </body> </html>