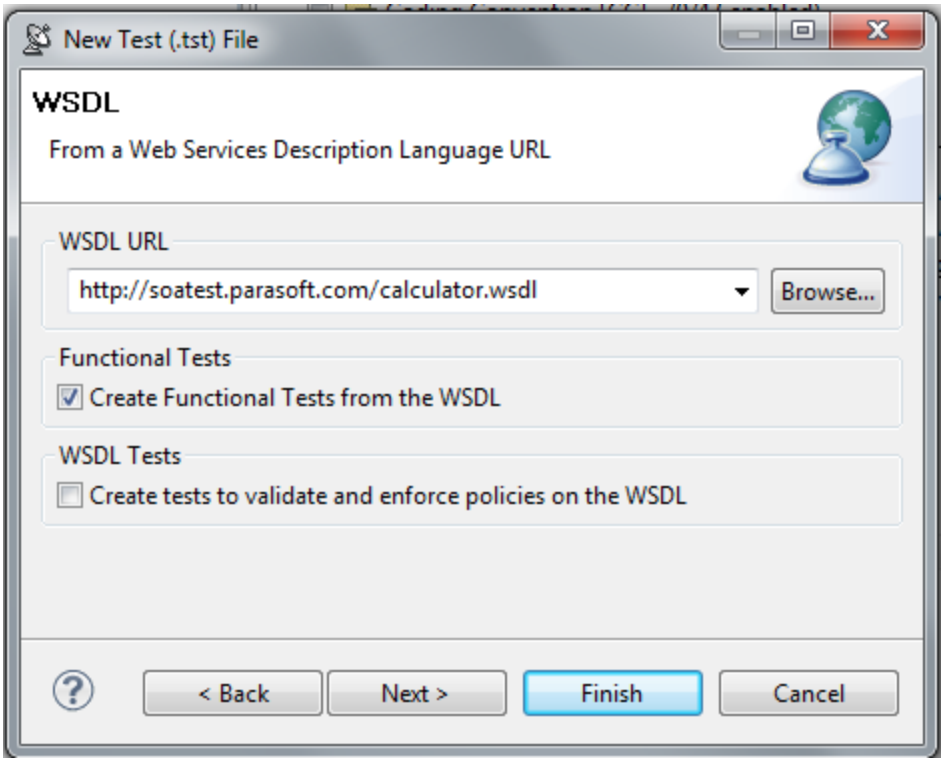
Creating Tests From a WSDL

This wizard can create:

- Functional tests for each operation defined in the WSDL.
- Comprehensive WSDL tests to ensure that the WSDL conforms to the schema and passes XML validation tests.

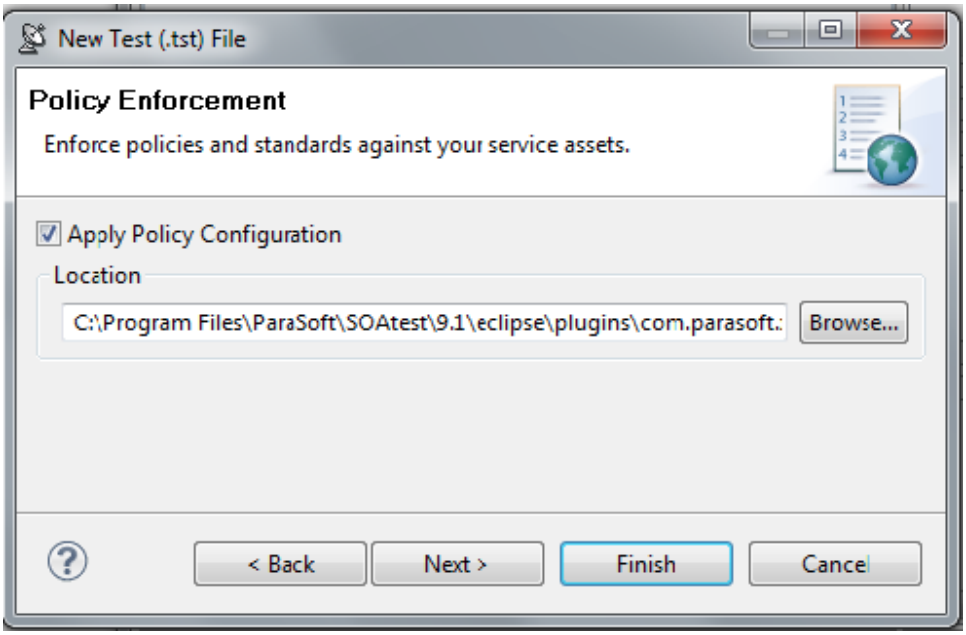
To automatically create a test suite from a valid WSDL document, complete the following:

1. Choose the **SOA> WSDL** option in one of the available test creation wizards. For details on accessing the wizards, see:
    - Adding a New .tst File to an Existing Project
    - Adding a New Test Suite
2. In the wizard's WSDL page, enter a valid WSDL URL in the **WSDL URL** field, or click the **Browse** button to locate a WSDL file on the local file system.

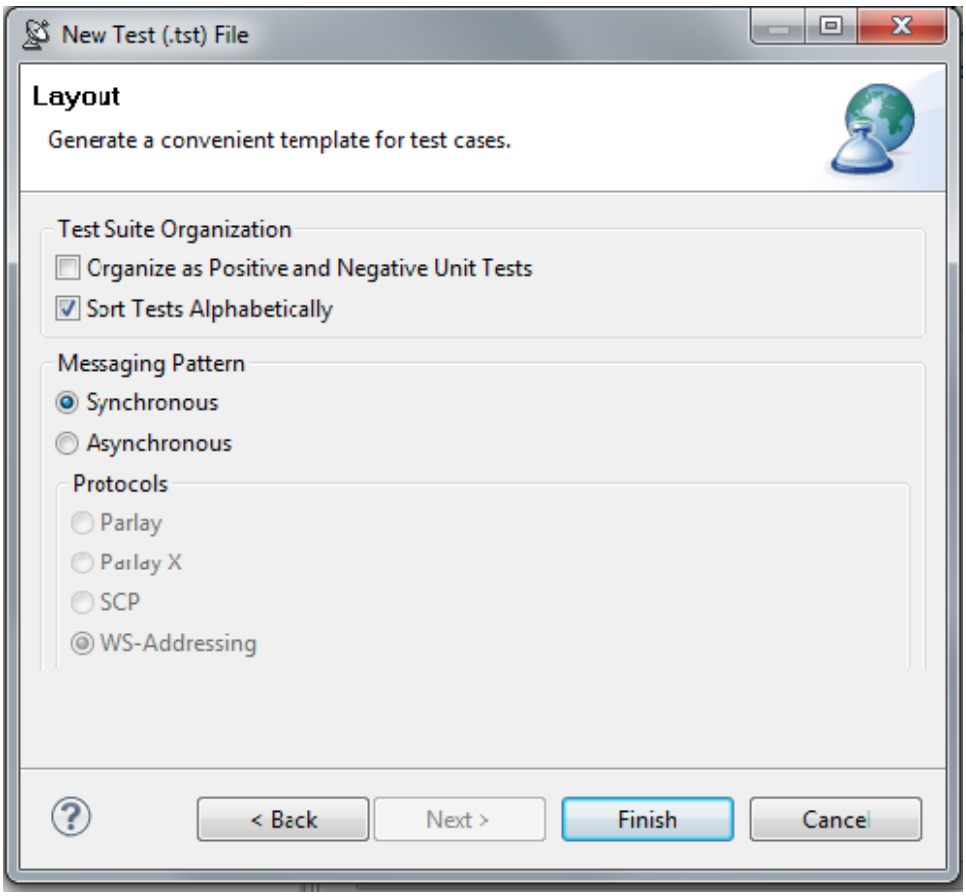


> **Note**
>
> The remaining steps are optional. Once you enter a valid WSDL URL, you can go ahead and click the **Finish** button and SOAtest will generate a suite of test cases that test every object associated with the WSDL you entered. If you would like to configure the test suite further, continue to the next step.

3. Select the **Create Functional Tests from the WSDL** checkbox.
4. If you want to create a separate test suite that generates a series of tests to verify the WSDL (Schema Validity, Semantic Validity, WS-I Interoperability, and WSDL Regression), select the **Create tests to validate and enforce policies on the WSDL** checkbox.
5. Click **Next**. The **Interoperability** dialog opens.
6. Select whether you would like to create **SOAtest (Java) Clients** or **.NET WCF Clients**.
    - If you are using .NET WCF clients, see NET WCF SOAP Clients: FAQ for additional information.
7. Click **Next**. The **Create Environment** dialog opens.
8. Specify whether you want to reference an existing environment or create a new one.
    - To create a new environment:
        a. Select the **Create a new environment for your project** checkbox
        b. Enter an **Environment Name** and **Variable Prefix**
        c. Select whether you want to create environment variables for **WSDL URI Fields, Client Endpoints,** or **Both**.
    - To reference an existing environment, select **Reference an existing environment** then specify the appropriate environment file. SOAtest will look for the WDSL and endpoint URLs inside the referenced environment variable values. If a match is found, SOAtest will replace the relevant portions of the URL in the SOAP or Messaging Client with the environment variable name. If no match is found, then the environment will be referenced and added to the project, but the WSDL and Endpoint URLs will be untouched.
    - For more information on environments, see Configuring Testing in Different Environments.
9. Click **Next**. The **PolicyEnforcement** dialog opens.



10. Select the **Apply Policy Configuration** check box. This will create WSDL and functional tests that will enforce the assertions defined in the specified policy configuration.
    - The default policy configuration, `soa.policy`, is a collection of industry-wide best practices. To use a custom policy configuration, you can either use the **Browse** button to select a policy configuration or the policy configuration's path can be entered in the text field. For details on policy enforcement, see SOA Policy Enforcement: Overview.
11. Click the **Next** button to advance to the **Layout** dialog.



12. (Optional) Select the **Organize as Positive and Negative Unit Tests** checkbox to create both positive and negative tests for each operation since it is important to test situations where we send expected data as well as unexpected data to the server. The default value is configured to **So rt Tests Alphabetically**.
13. (Optional) Select the **Asynchronous** radio button and choose **Parlay, Parlay X, SCP, or WS-Addressing** to create asynchronous test suites. For more information on asynchronous testing, see Creating Asynchronous Tests.

14. Click the **Finish** button.

SOAtest will generate a suite of test cases that test every operation defined in the WSDL you entered.

If you enabled **Create tests to validate and enforce policies on the WSDL**, SOAtest automatically creates the following WSDL tests:

- **Test 1: Schema Validity:** Runs XML validation on the WSDL against WSDL schemas from W3C.
- **Test 2: Semantic Validity:** Checks the correctness of the WSDL by parsing and consuming it like an actual service consumer would, but with stricter adherence to standards.
- **Test 3: WS-I Interoperability:** Checks WSDL conformance to the WS-I Basic Profile 2.0 (for SOAP 1.2), 1.2 (for SOAP 1.1) or 1.1 (also for SOAP 1.1).
- **Test 4: WSDL Regression:** Creates a regression control for the WSDL so that changes in the WSDL document can be detected.

Video Tutorial

In this video, you'll learn how to automatically generate tests for the operations defined in a WSDL.

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