Groovy スクリプトによる API の自動化

内容

概要

このドキュメントでは、soapUI アプリケーション プログラミング インターフェイス (API) 要求の作成方法、Quantum Policy Suite (QPS) への API 要求を自動化するテスト手順で繰り返される soapUI テスト ケースの作成方法について説明します。

この記事の soapUI テスト ケースの例では、サブスクライバ ID のファイルを読み取り、querySubscriberRequst を作成して QPS に送信するテスト手順が実装されています。

soapUI プロジェクトの作成

この手順を開始する前に、soapUI アプリケーションをデスクトップにインストールしてください。www.soapui.org から soapUI インストール実行可能ファイルをダウンロードできます。

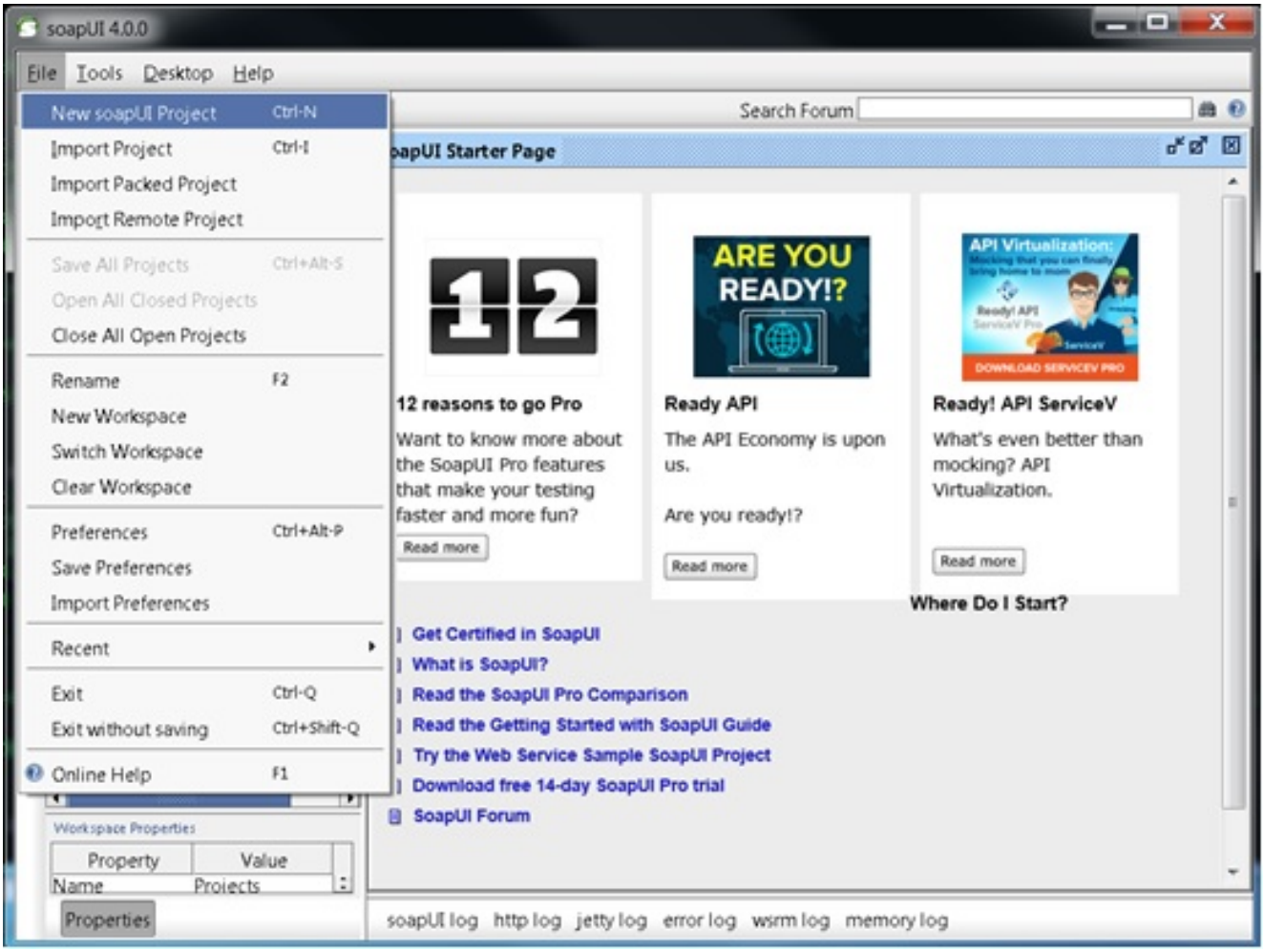
API 要求またはテスト ケースを作成するには、その前に soapUI プロジェクトを作成する必要があります。プロジェクトを作成するには、Web サービス記述言語 (WSDL) ファイルと XML スキーマ定義 (XSD) ファイルが必要です。WSDL は、サポートされる API を指定します。ロードバランサ (LB) から次のコマンドを実行すると、通常 QPS から WSDL と XSD を取得できます。

- **wget http://lbvip01:8080/ua/wsdl/UnifiedApi.wsdl**
- **wget http://lbvip01:8080/ua/wsdl/UnifiedApi.xsd**

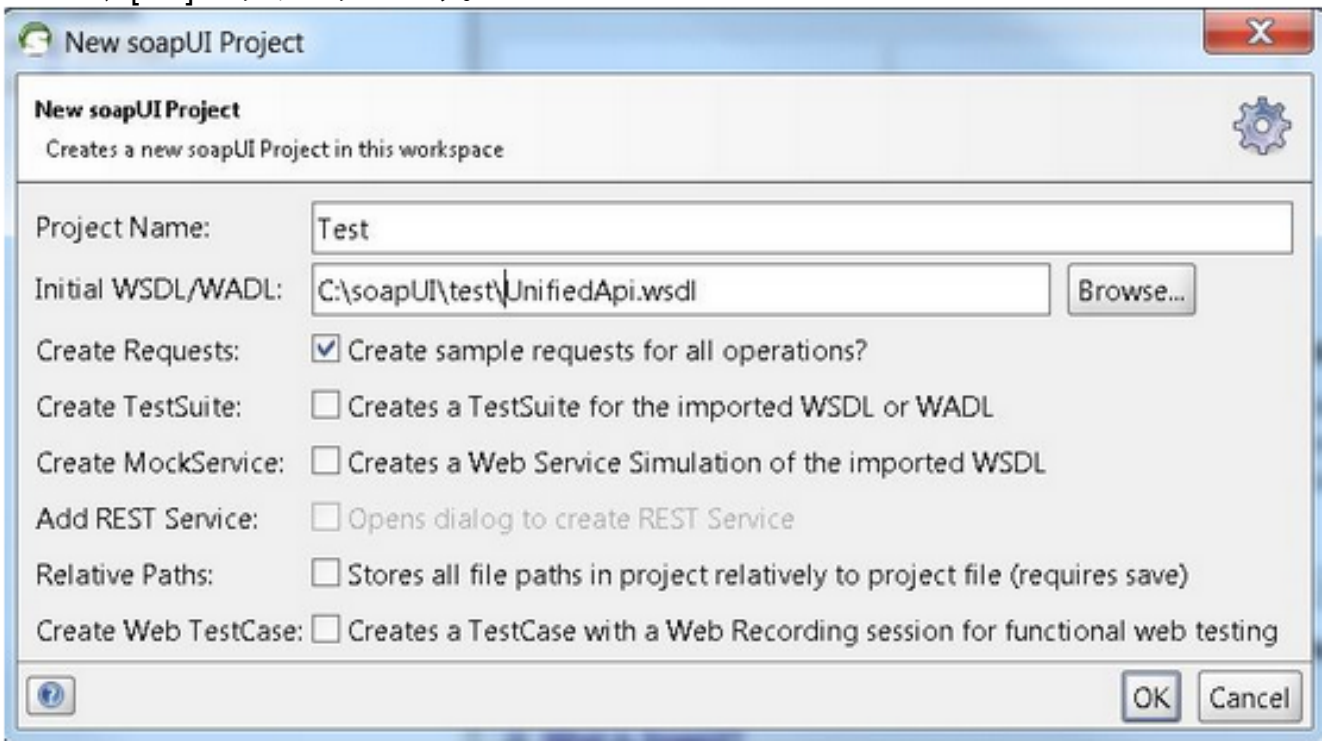
soapUI アプリケーションを実行するデスクトップの同じディレクトリに WSDL と XSD を保存します。

次の手順を実行して、soapUI プロジェクトを作成します。

1. soapUI ウィンドウから [File] > [New soapUI Project] の順に選択します。



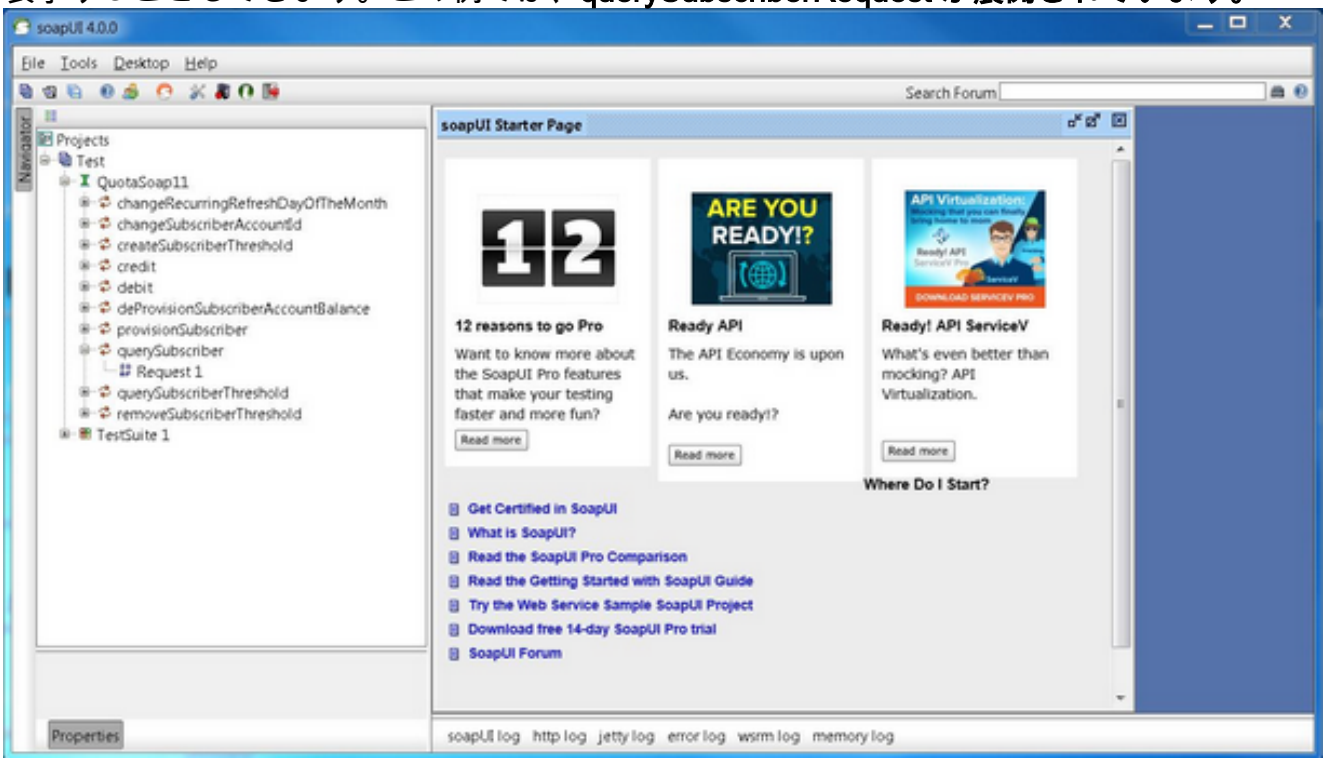
2. [New soapUI Project] ウィンドウの [Project Name] フィールドにプロジェクトの名前を入力し、WSDL ファイルを保存する場所を [Initial WSDL/WADL] フィールドに入力します。完了したら、[OK] をクリックします。



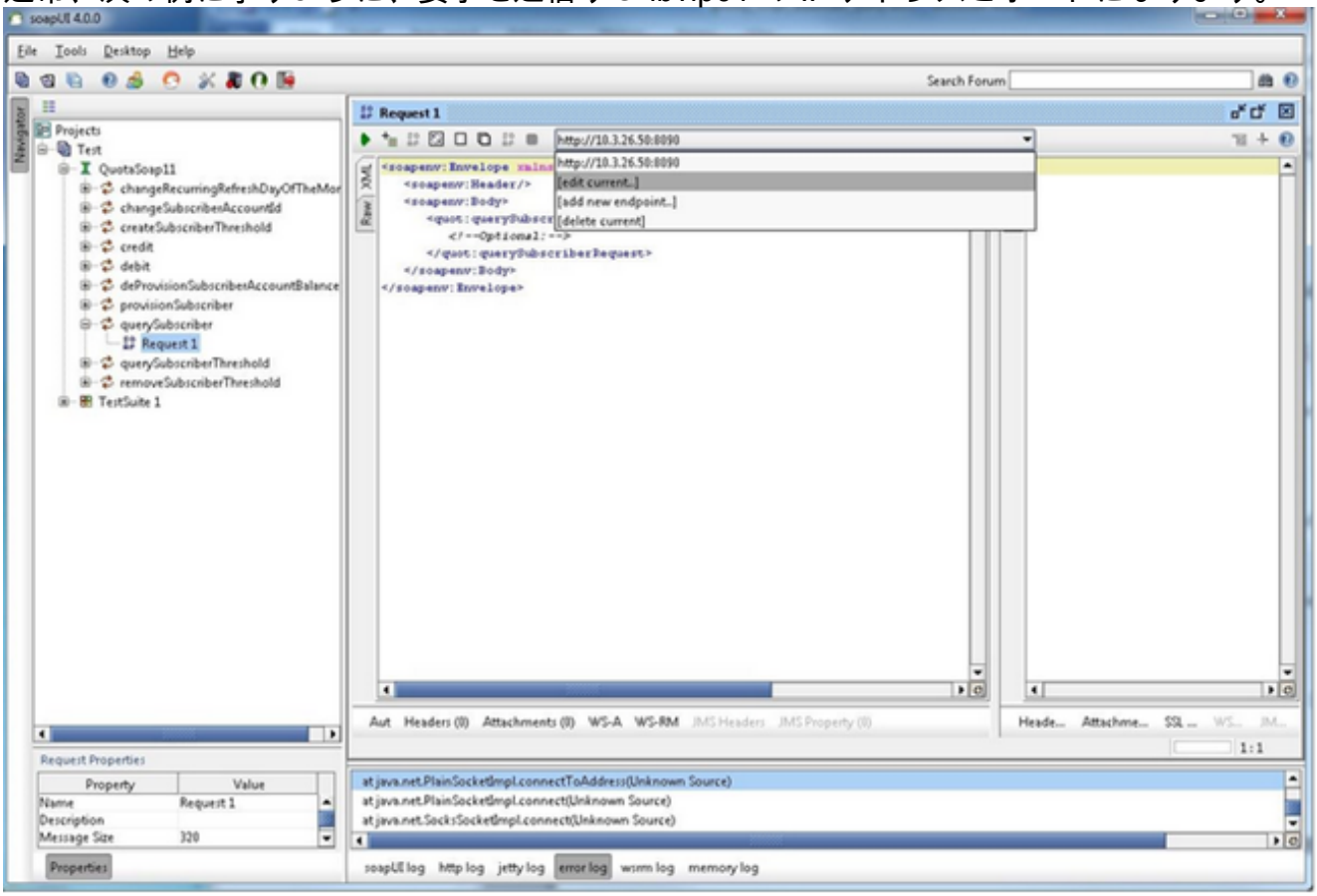
soapUI API 要求の作成

soapUI API 要求を作成するには、次の手順に従います。

1. 作成した soapUI プロジェクトを展開し、API を表示します。API の 1 つを展開して要求を表示することもできます。この例では、**querySubscriberRequest が展開されています。**



2. 要求を開くと、クエリーを形成する XML と [Request] ウィンドウが表示されます。[Request] ウィンドウで http:// の IP アドレスを IP アドレスとポートに編集します。これは通常、次の例に示すように、要求を送信する lbvip01 の IP アドレスとポートになります。



3. 要求で送信するデータを使用して XML のフィールドを変更します。この例では、要求は querySubscriberRequest です。クエリーするサブスクライバのサブスクライバ ID を変更し、showDetailedInformatin を [false] に設定します。

4. [Request] ウィンドウの上部にある緑色の実行ボタンをクリックしてクエリーを実行します。

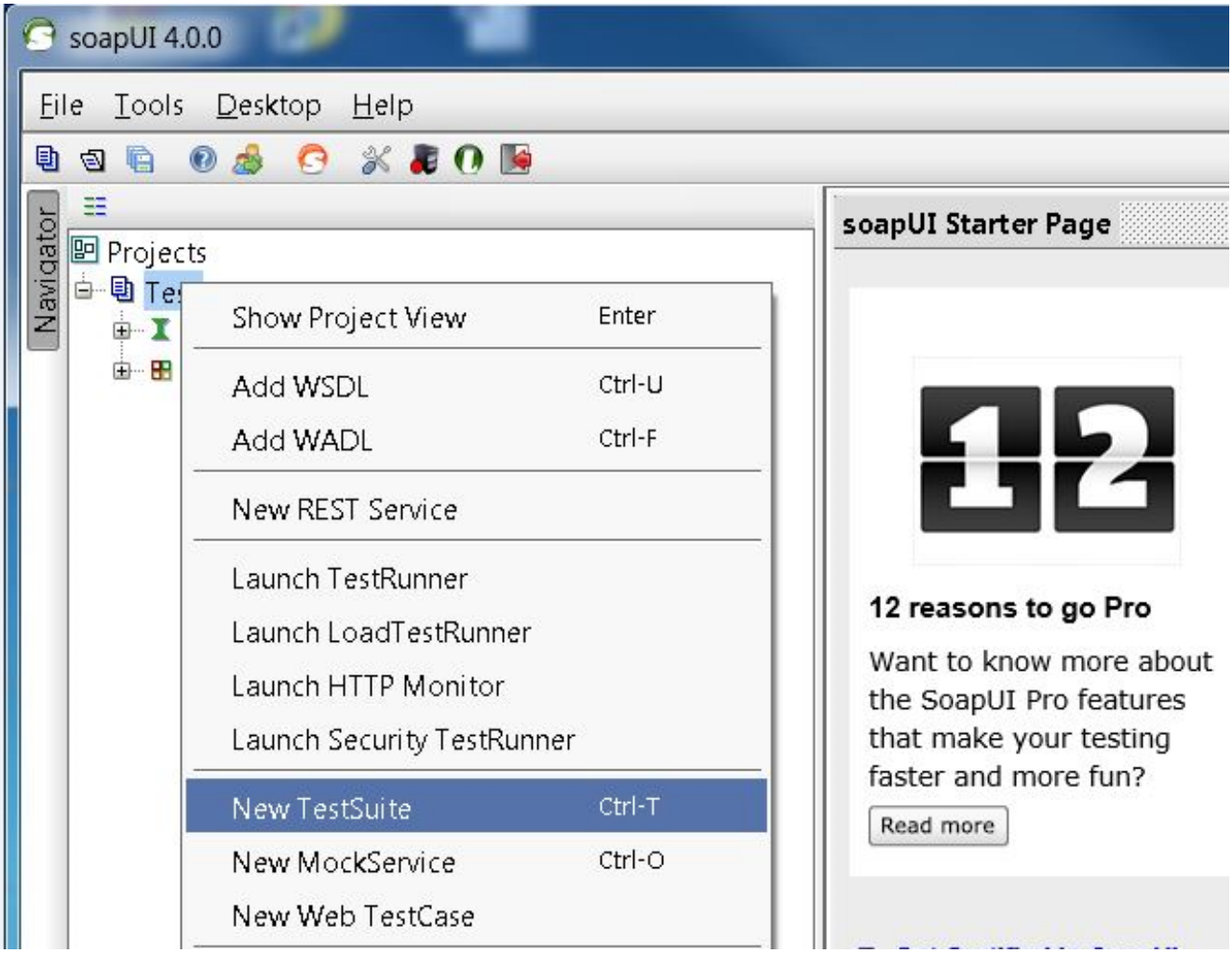
soapUI テスト ケースの作成

この手順では、API の QPS への送信時に自動的に実行できるテスト スイートを作成する方法について説明します。

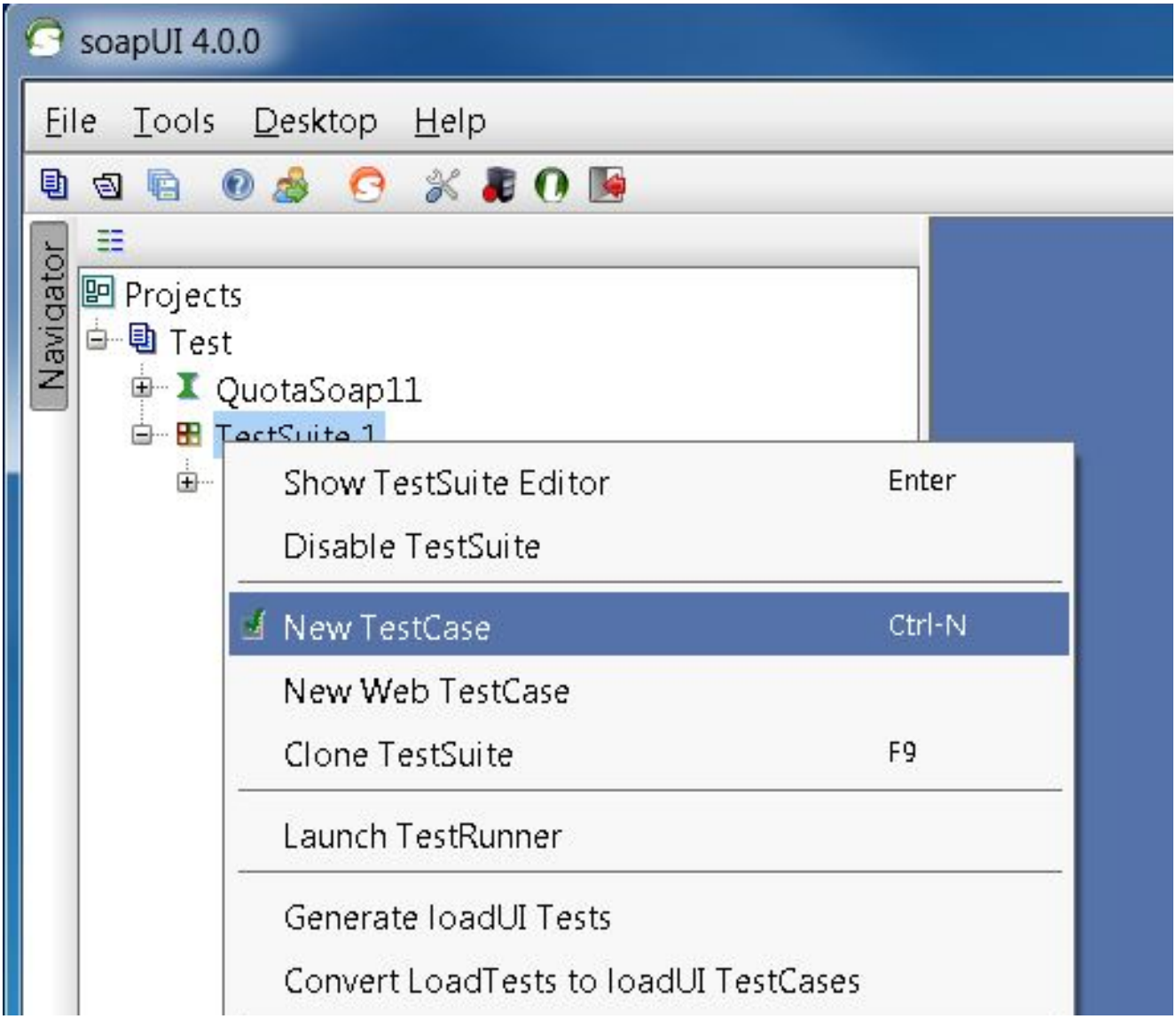
この例の手順では、テスト スイートはサブスクライバ ID のリストをループし、QPS に送信する querySubscriberRequest でそれらのサブスクライバ ID を使用します。サブスクライバ ID は、**subid.txt** というテキスト ファイルに 1 行ずつリストされます。

次の手順を実行して、テスト スイートを作成します。

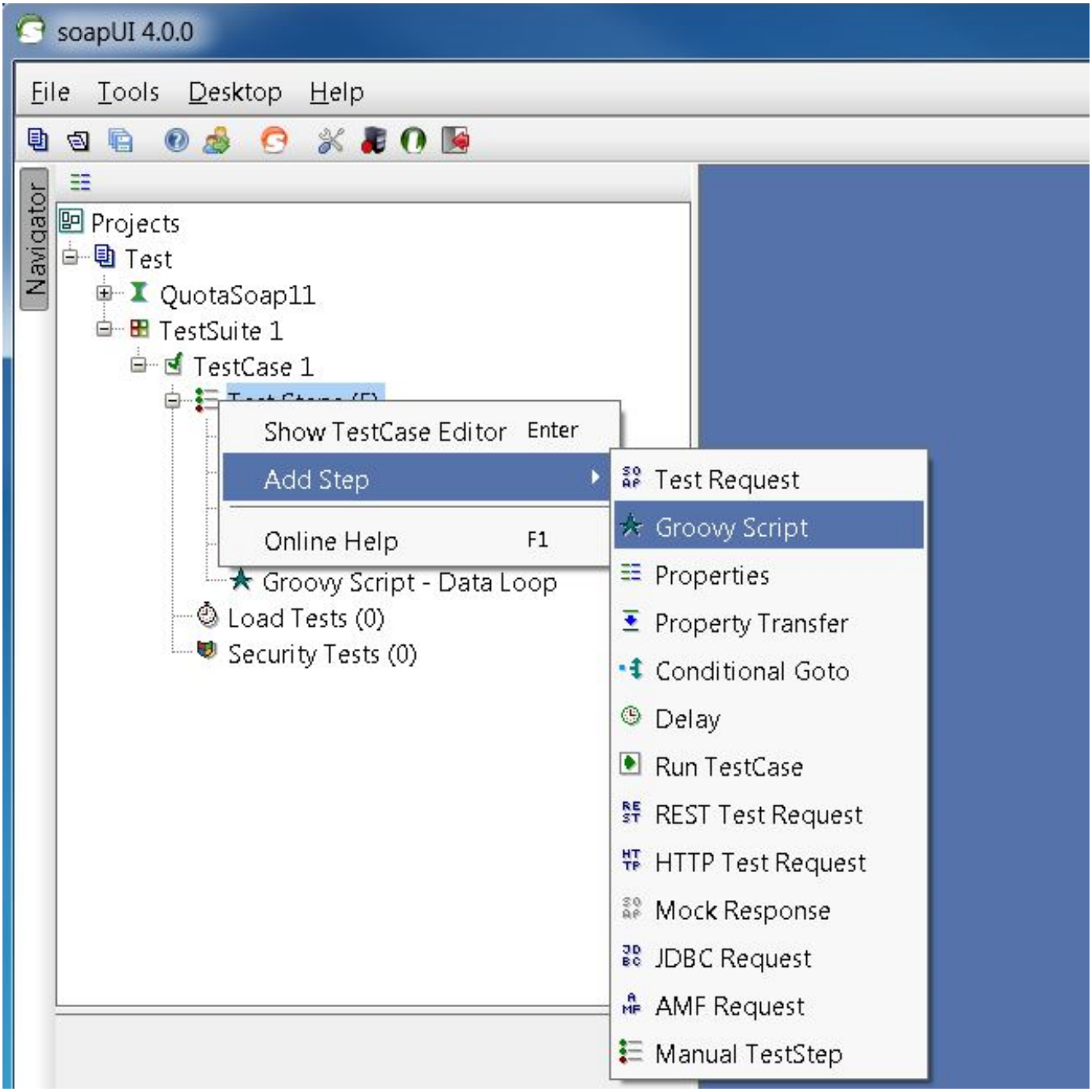
1. 作成した soapUI プロジェクトに、新しいテスト スイートを作成します。soapUI を右クリックして、[New TestSuite] を選択します。



2. テスト スイートを右クリックし、[New] > [TestCase] を選択します。



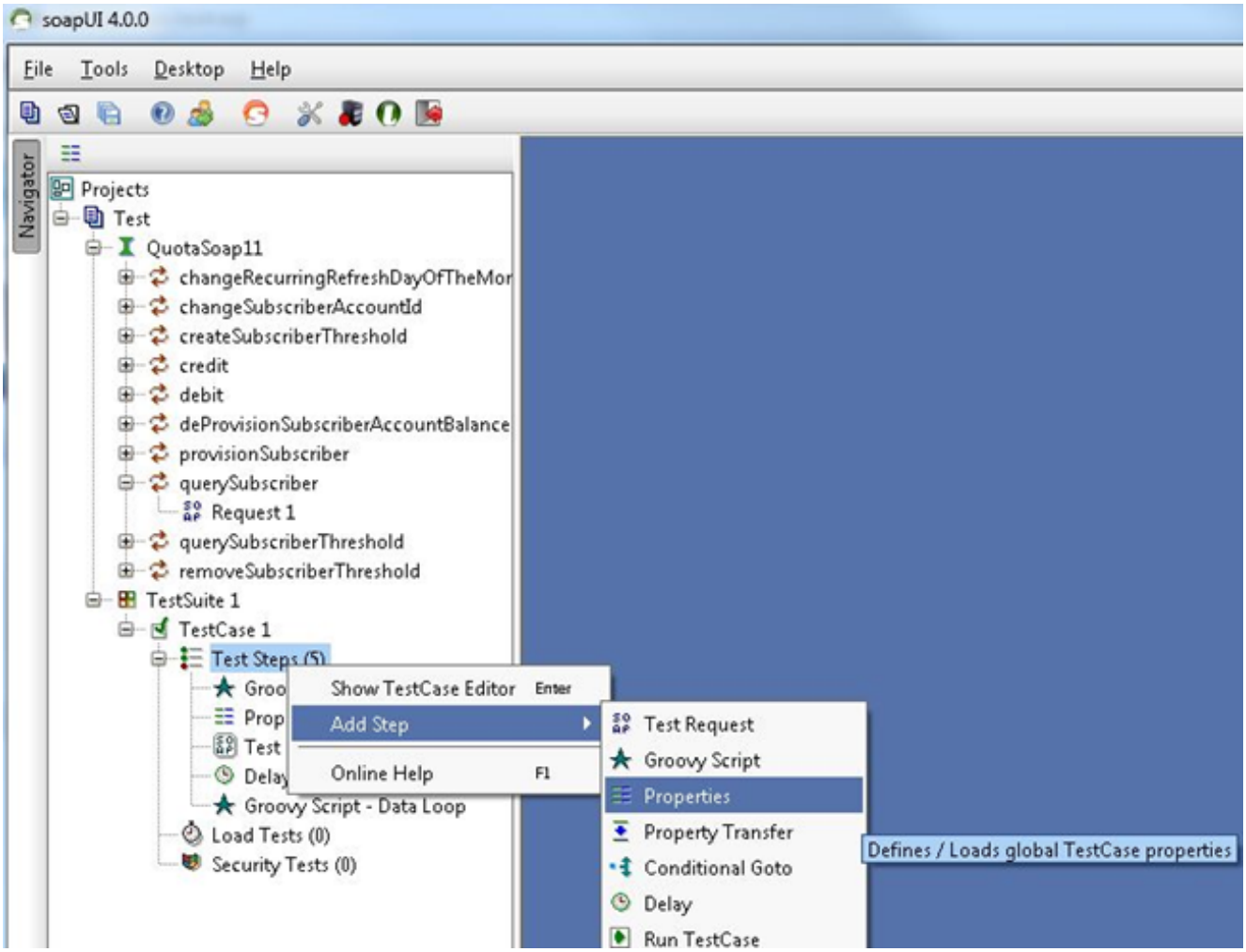
3. テスト ケースを右クリックし、[Add Step] > [Groovy Script] の順に選択して、Groovy Script のテスト手順を追加します。これに「Data Source」という名前を付けます。



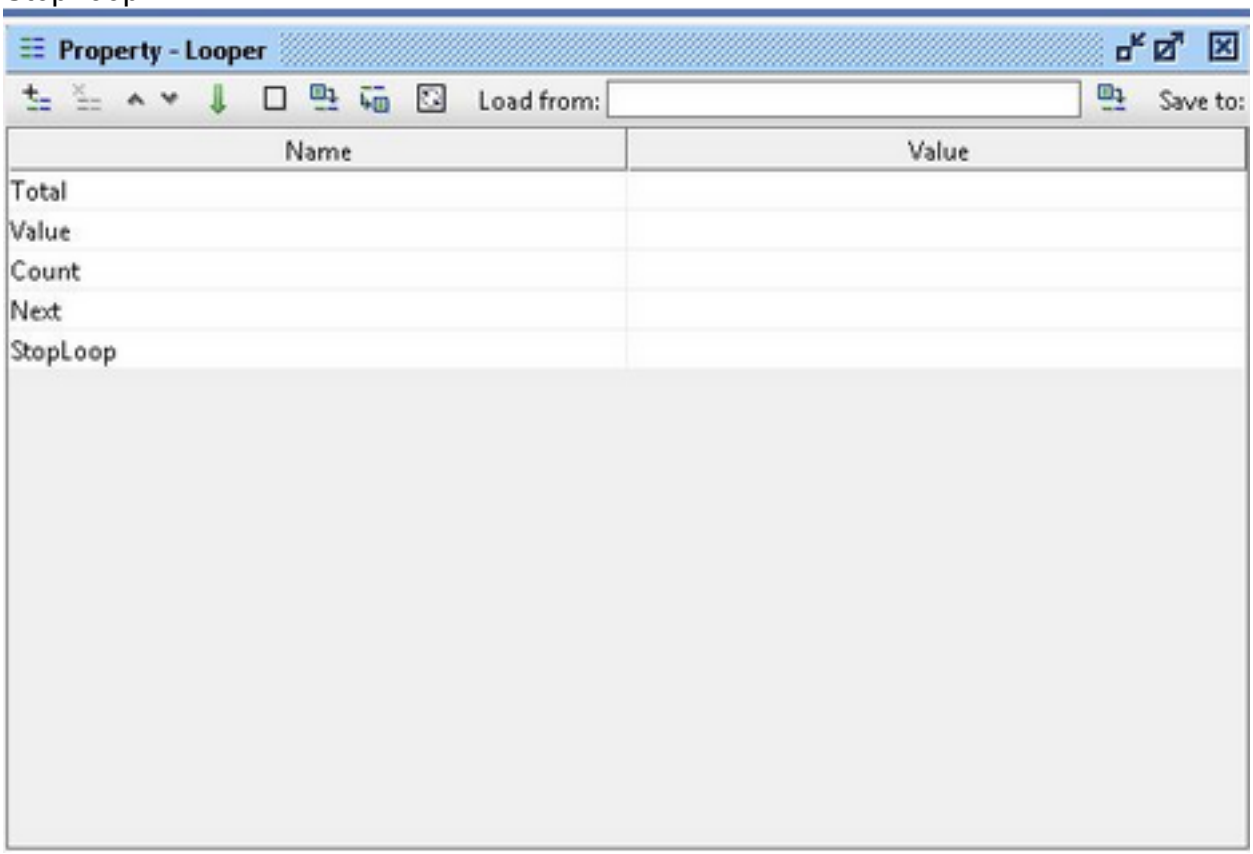
4. Data Source ファイルに以下のコードを貼り付けます。次のコードは、各行にサブスクライバーIDを含むファイルC:/subid.txtを読み取ります。

```
import com.eviware.soapui.support.XmlHolder def myTestCase = context.testCase
def counter,next,previous,sizeFile tickerEnumFile = new File("C:/subid.txt") //subscriber
IDs sepearted by new line (CR). List lines = tickerEnumFile.readLines() size =
lines.size.toInteger() propTestStep = myTestCase.getTestStepByName("Property - Looper")
// get the Property TestStep propTestStep.setPropertyValue("Total", size.toString())
counter = propTestStep.getPropertyValue("Count").toString() counter= counter.toInteger()
next = (counter > size-2? 0: counter+1) tempValue = lines[counter]
propTestStep.setPropertyValue("Value", tempValue) propTestStep.setPropertyValue
("Count", next.toString()) next++ log.info "Reading line : ${(counter+1)} /
$lines.size"propTestStep.setPropertyValue("Next", next.toString()) log.info
"Value '$tempValue' -- updated in $propTestStep.name" if (counter == size-1) {
propTestStep.setPropertyValue("StopLoop", "T") log.info "Setting the stoploop property
now..."}
else if (counter==0) { def runner = new
com.eviware.soapui.impl.wsdl.testcase.WsdlTestCaseRunner
(testRunner.testCase, null) propTestStep.setPropertyValue("StopLoop", "F") } else{
propTestStep.setPropertyValue("StopLoop", "F") }
```

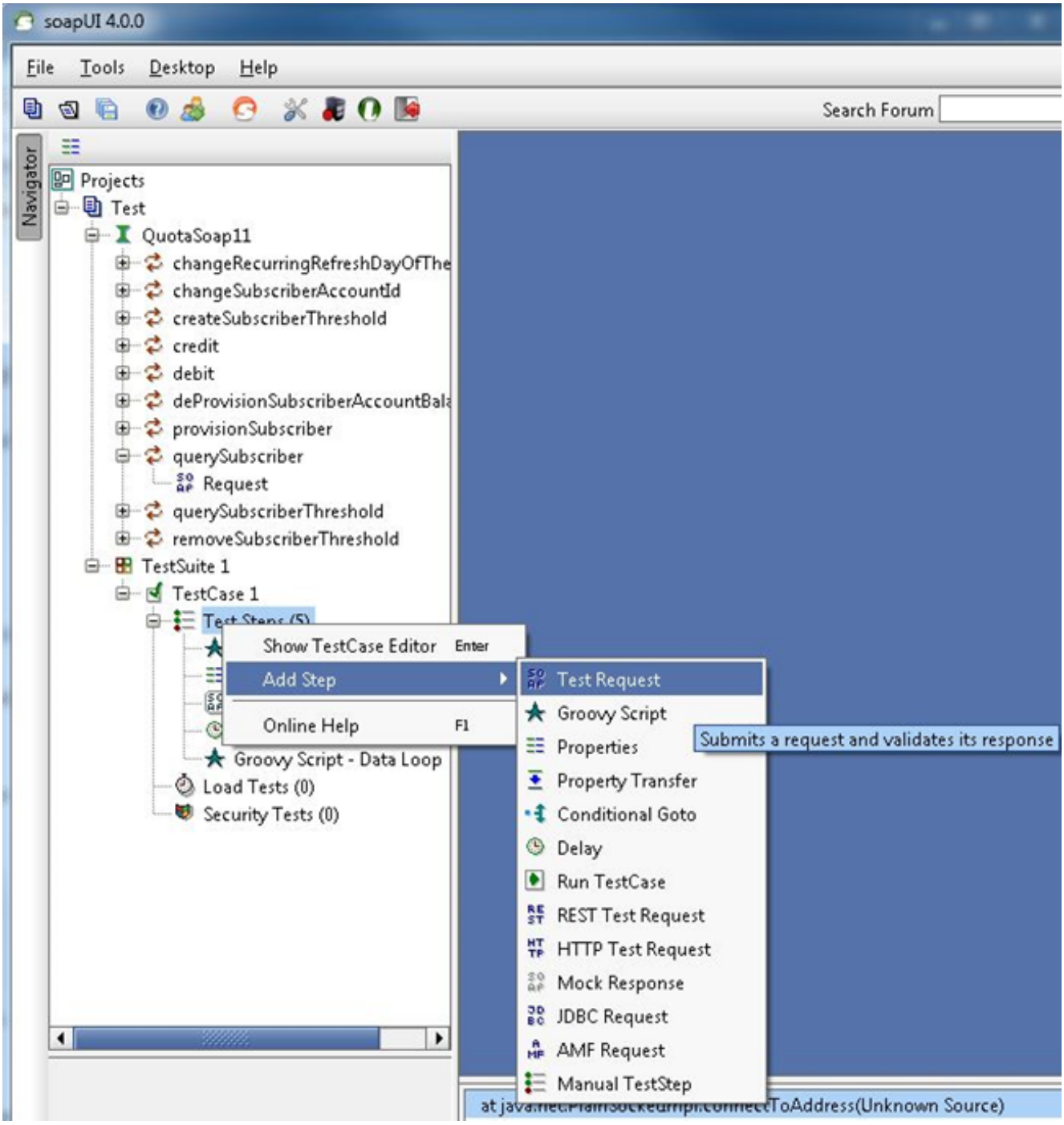
5. [Test Step] を右クリックし、[Add Step] > [Properties] の順に選択して、Property のテスト手順を追加します。これに「Property - Looper」という名前を付けます。



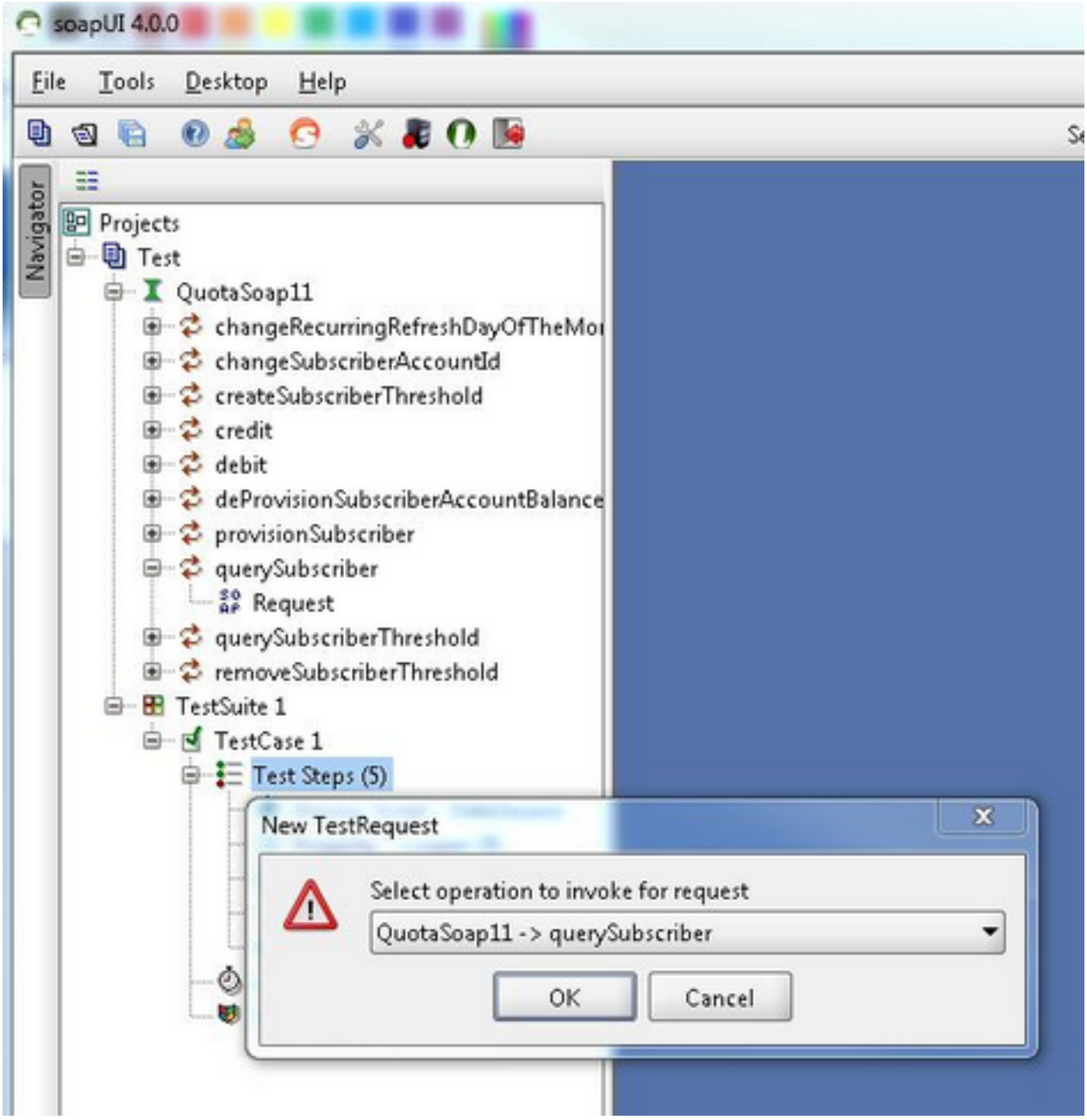
6. 以下のユーザを Looper テスト手順の定義済みプロパティに追加します。 合計Value (この例では、これにサブスクライバ ID のファイルから読み取られたサブスクライバ ID が保持されます) [Count]次
StopLoop



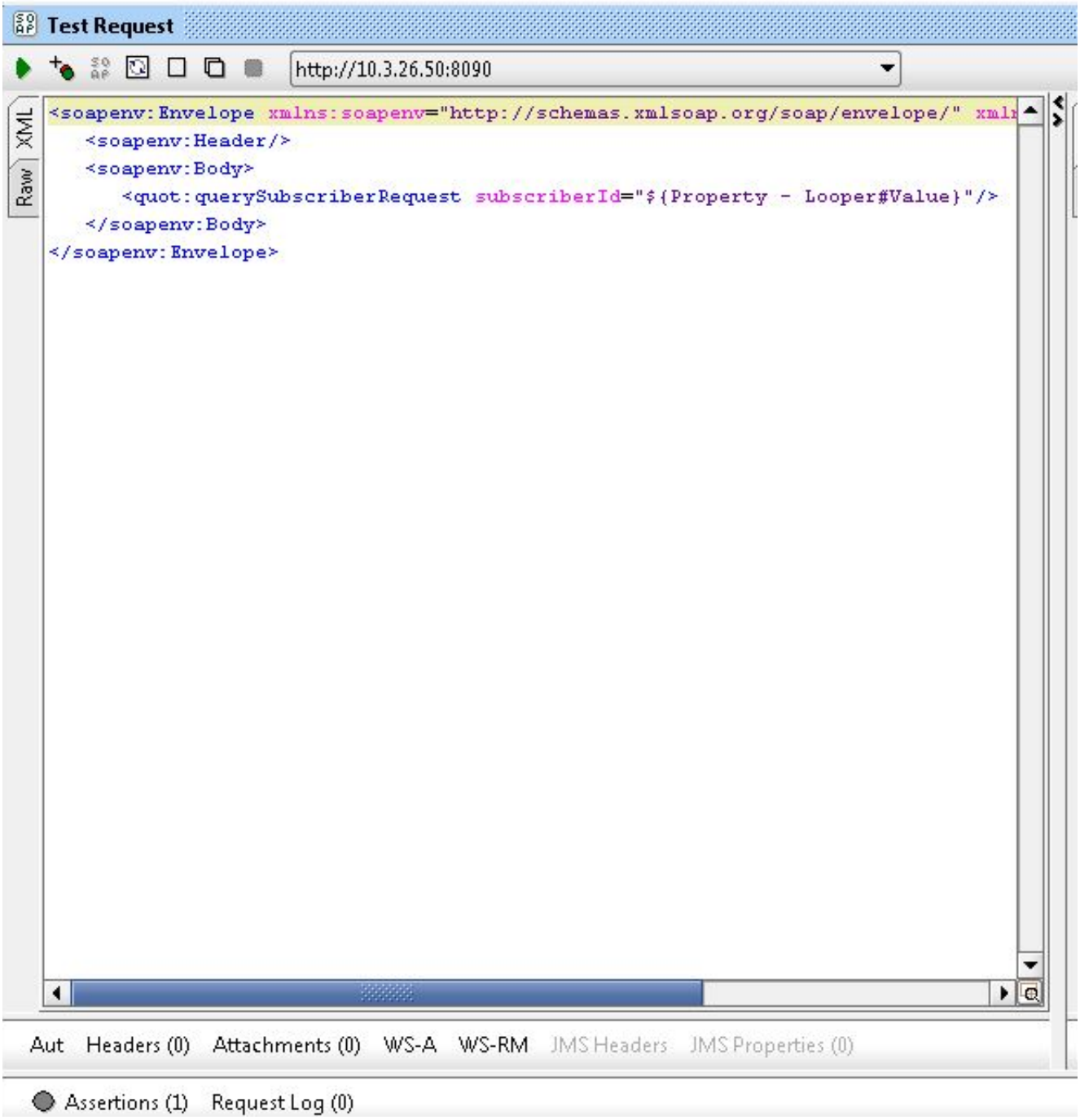
7. [Test Step] を右クリックし、[Add Step] > [TestRequest] の順に選択して、Test Request のテスト手順を追加します。実行したい要求を選択します。



この例では、querySubscriberRequest が使用されます。

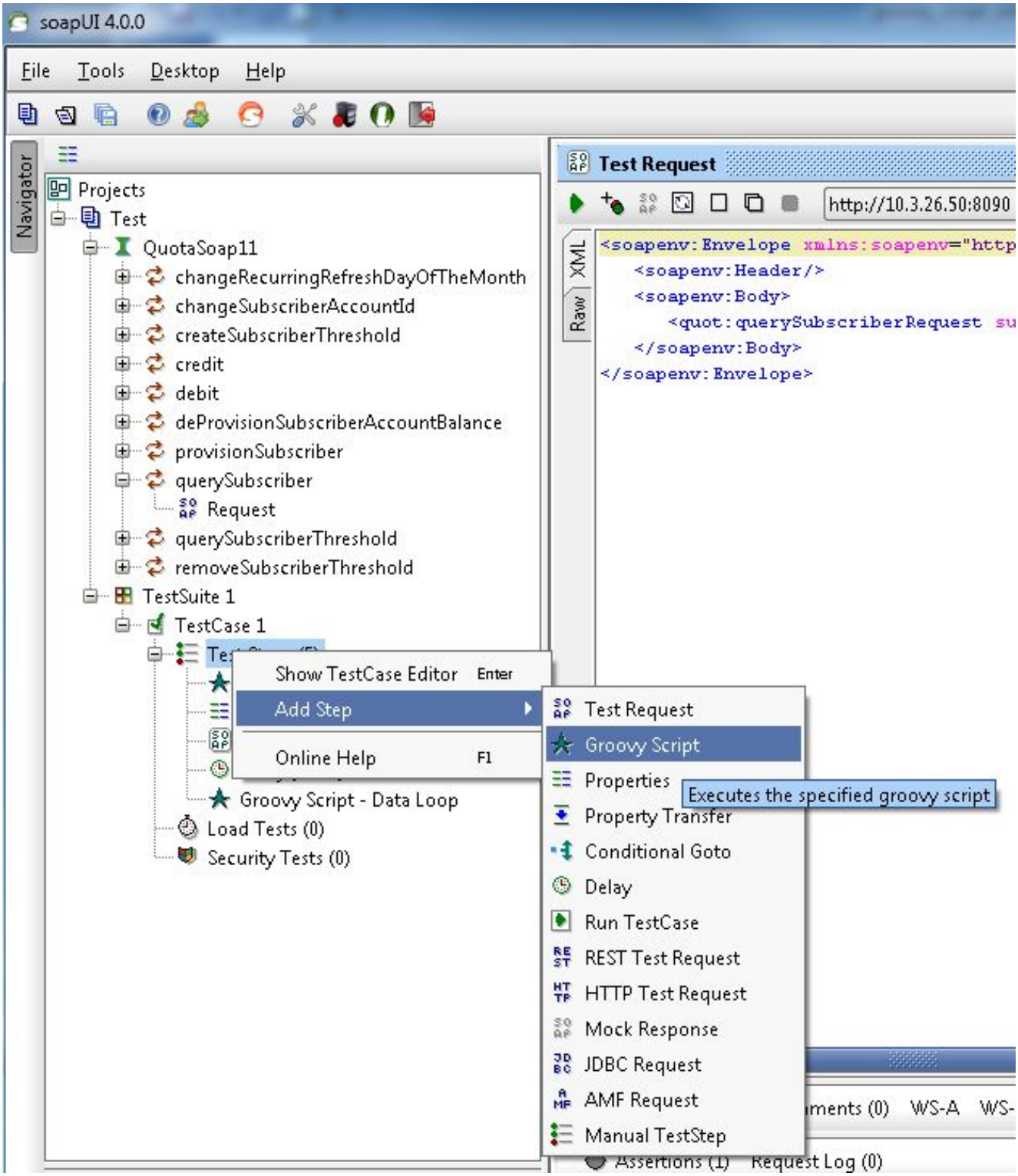


8. 要求では、拡張コードはフィールドの値をクエリーの内容で置き換えます。この例では、querySubscriberRequest 内の **SubscriberId=? の? が**拡張コード **${Property - Looper#Value}（soap_test_req_expansion_code）に置き換えられます。**



「Property - Looper」は、前の手順で作成したプロパティ TestStep の名前で、「Value」にはサブスクライバ ID のファイルから読み取った現在のサブスクライバ ID が保持されます。

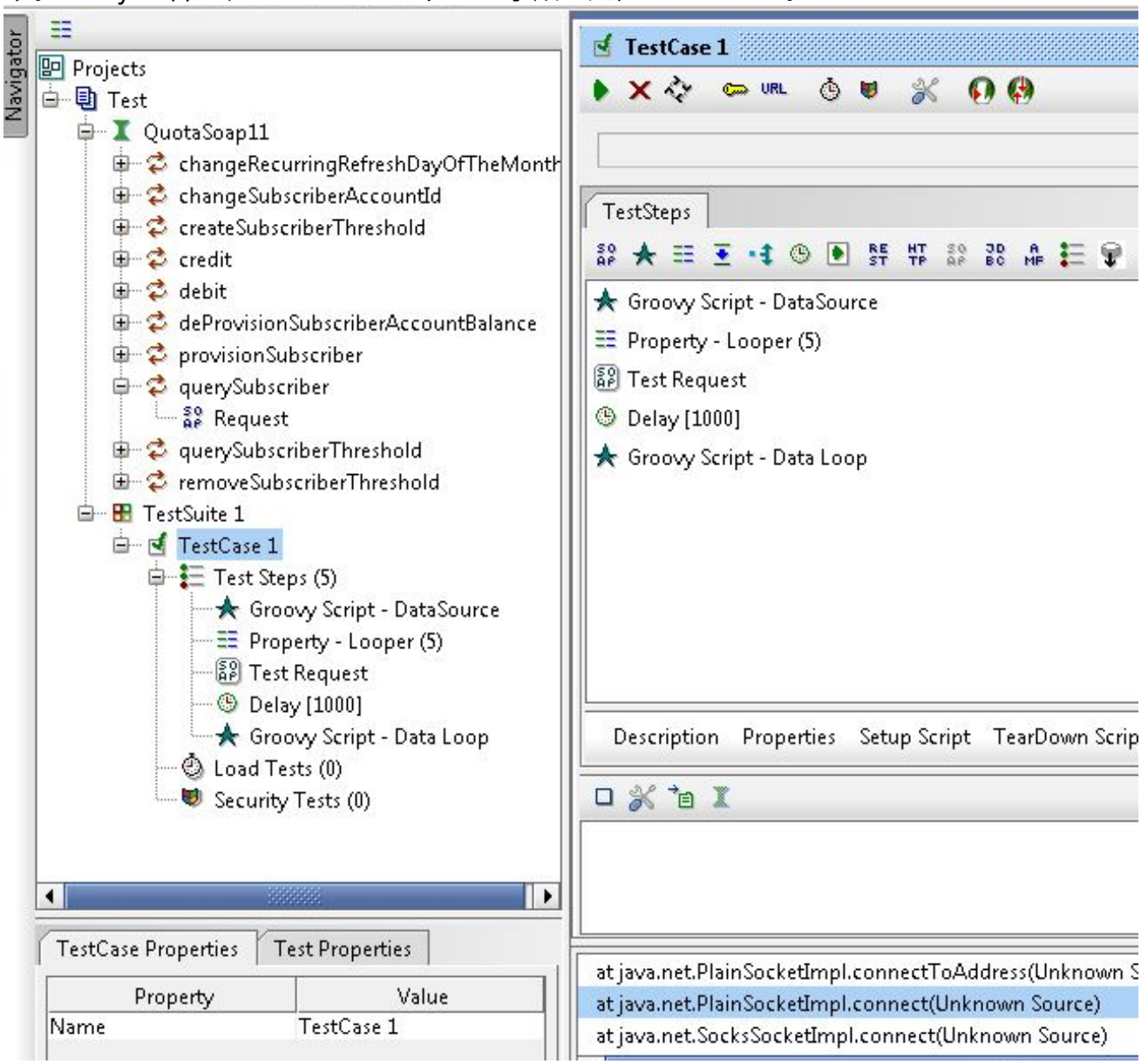
9. [Test Step] を右クリックし、[Add Step] > [Groovy Script] の順に選択して、これに「Data Loop」という名前を付けます。



10. Groovy Script の Data Loop に以下のコードを貼り付けます。

```
def myTestCase = context.testCase
def runner
propTestStep = myTestCase.getTestStepByName("Property - Looper")
endLoop = propTestStep.getPropertyValue("StopLoop").toString()
if (endLoop.toString() == "T" || endLoop.toString()=="True"
|| endLoop.toString()=="true")
{
log.info ("Exit Groovy Data Source Looper")
assert true
}
else
{
testRunner.gotoStepByName("Groovy Script - DataSource") //go to the DataSource
}
```

11. この例の手順では、各ループ間で 1000 ms の Delay を追加しています。この手順は任意です。Delay を含め、これで 5 つのテスト手順が追加されました。



12. 緑色の実行ボタンをクリックすると、[TestCase] ウィンドウで 5 つのテスト手順が実行されます。