Automatisierung von APIs mit Groovy Script

Inhalt

Einführung

In diesem Dokument wird beschrieben, wie eine SoapUI Application Programmers Interface (API)-Anforderung erstellt wird und wie ein SoapUI-Testfall erstellt wird, der über Testschritte schleift, die die API-Anforderungen an Quantum Policy Suite (QPS) automatisieren.

Im Beispiel SoapUI-Testfall in diesem Artikel werden Testschritte implementiert, die eine Datei mit Teilnehmer-IDs lesen und dann eine querySubscriberRequest erstellen und an QPS senden.

Erstellen eines SoapUI-Projekts

Bevor Sie dieses Verfahren beginnen, installieren Sie die SOAPUI-Anwendung auf Ihrem Desktop. Sie können die ausführbare Software für die SoapUI-Installation von www.soapui.org herunterladen.

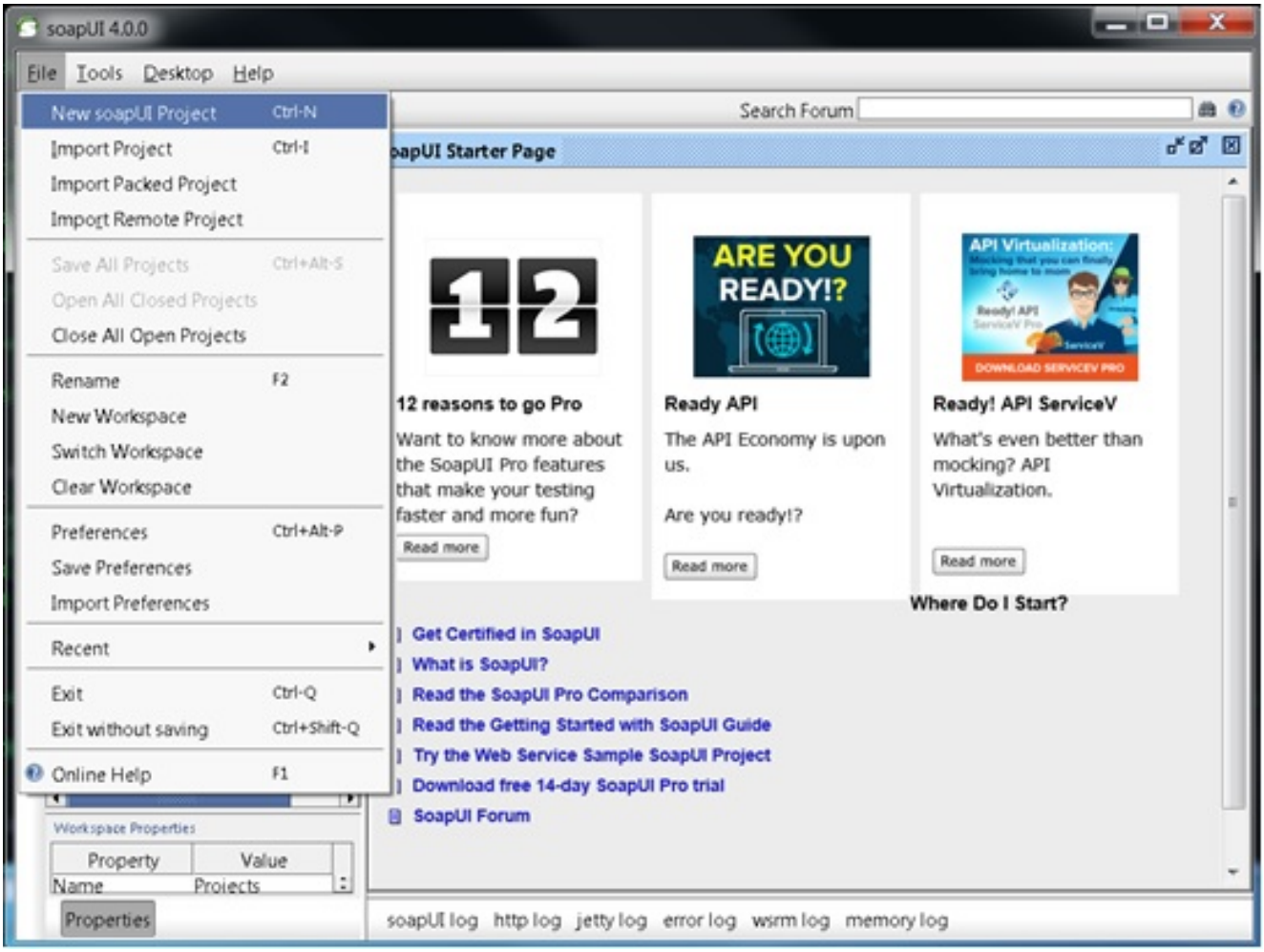
Bevor Sie eine API-Anforderung oder einen Testfall erstellen, müssen Sie zunächst ein soapUI-Projekt erstellen. Zum Erstellen des Projekts benötigen Sie die WSDL-Datei (Web Services Description Language) und die XSD-Datei (XML Schema Description). WSDL gibt die unterstützten APIs an. Normalerweise erhalten Sie WSDL und XSD von QPS, wenn Sie diese Befehle über den Load Balancer (LB) ausführen:

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

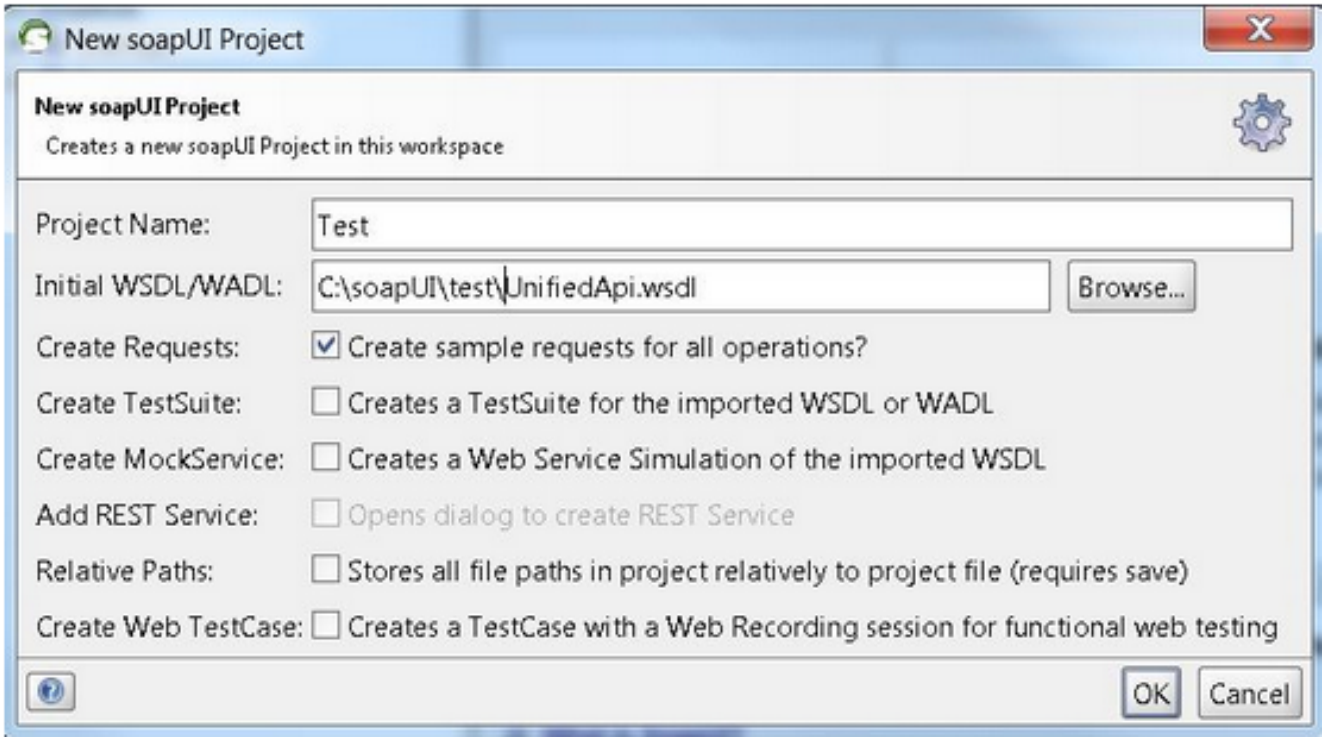
Speichern Sie WSDL und XSD im gleichen Verzeichnis auf dem Desktop, auf dem Sie die SoapUI-Anwendung ausführen möchten.

Gehen Sie wie folgt vor, um das soapUI-Projekt zu erstellen:

1. Wählen Sie **File > New soapUI Project** aus dem soapUI-Fenster aus:

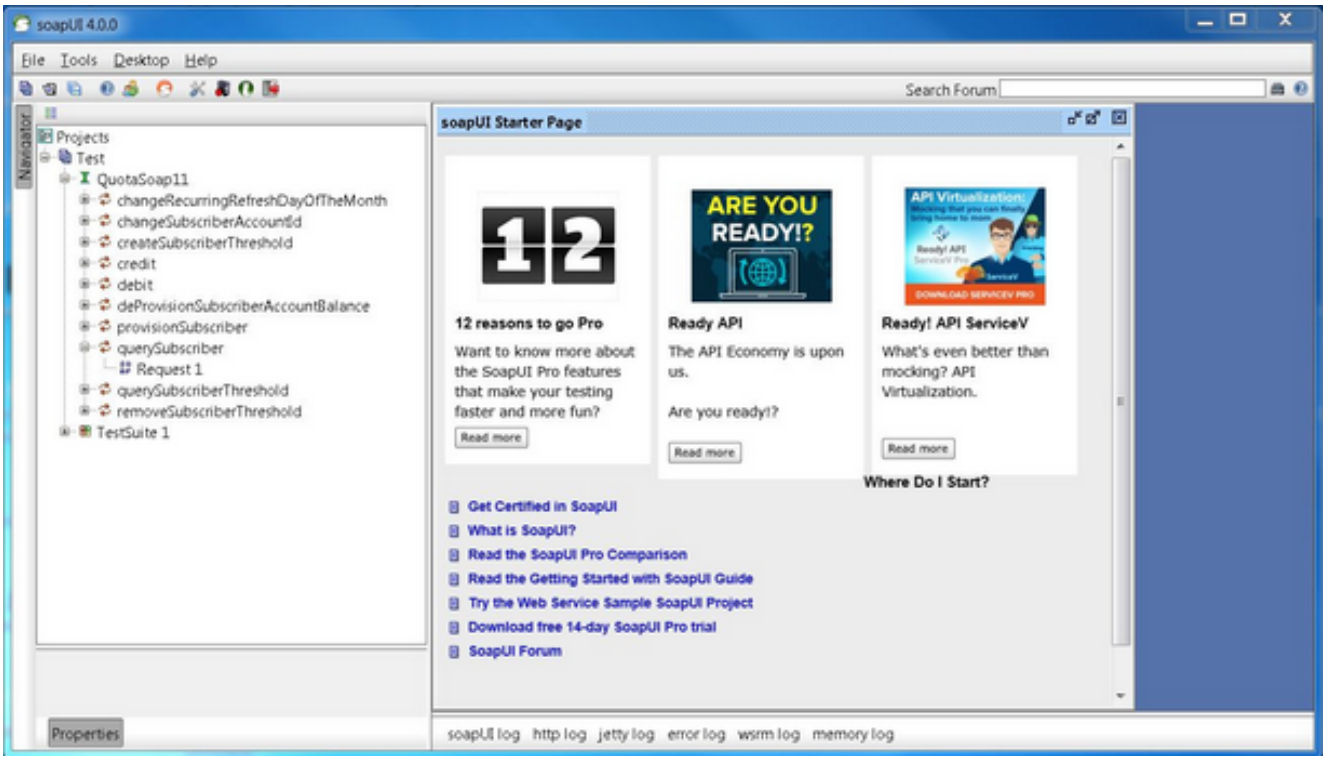


2. Geben Sie im Fenster Neues soapUI-Projekt im Feld Projektname einen Namen für das Projekt ein, und geben Sie den Speicherort der WSDL-Datei im Feld Initial WSDL/WADL ein. Klicken Sie abschließend auf
OK.

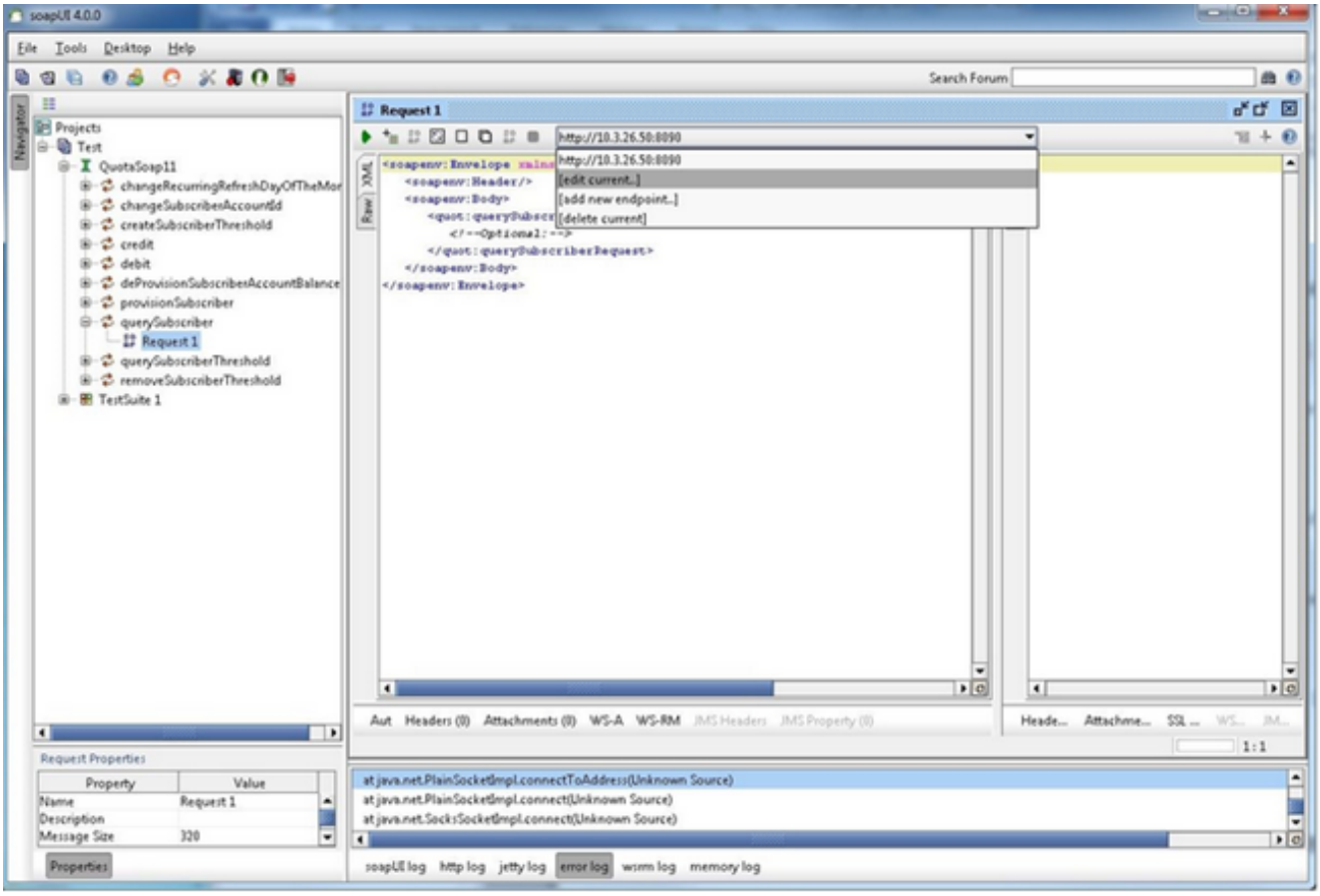


Erstellen einer SoapUI-API-Anforderung

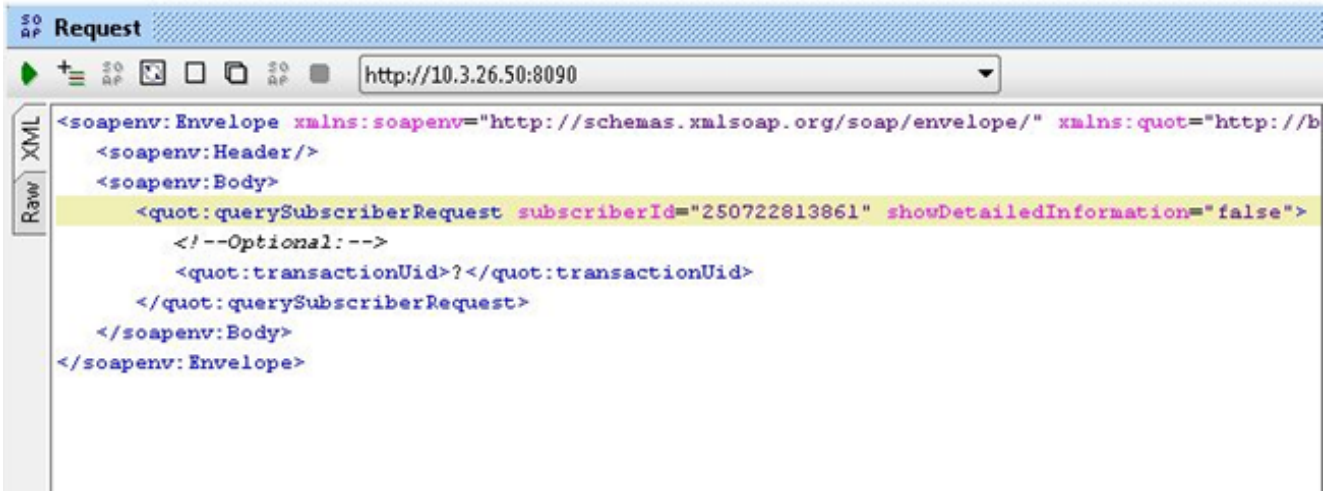
Gehen Sie wie folgt vor, um eine SoapUI-API-Anforderung zu erstellen:

1. Erweitern Sie das von Ihnen erstellte soapUI-Projekt, um die APIs anzuzeigen. Sie können auch eine der APIs erweitern, um die Anforderung anzuzeigen.In diesem Beispiel wird **querySubscriberRequest** erweitert:


2. Öffnen Sie die Anforderung, um das Anforderungsfenster mit dem XML anzuzeigen, das die Abfrage bildet. Bearbeiten Sie im Anforderungsfenster die IP-Adresse http:// in die IP-Adresse und den Port. Dies ist normalerweise die IP-Adresse und der Port von lbvip01, an die die Anforderung gesendet werden soll, wie im folgenden Beispiel gezeigt:



3. Ändern Sie die Felder im XML mit den Daten, die Sie in Ihrer Anforderung senden möchten. In diesem Beispiel ist die Anforderung eine querySubscriberRequest. Ändern Sie die Teilnehmer-ID für den Abonnenten, den Sie abfragen möchten, und setzen Sie showDetailedInformation auf **false**:



4. Klicken Sie auf die grüne Schaltfläche **Ausführen** am oberen Rand des Anfragefensters, um die Abfrage auszuführen.

Erstellen eines SoapUI-Testfalls

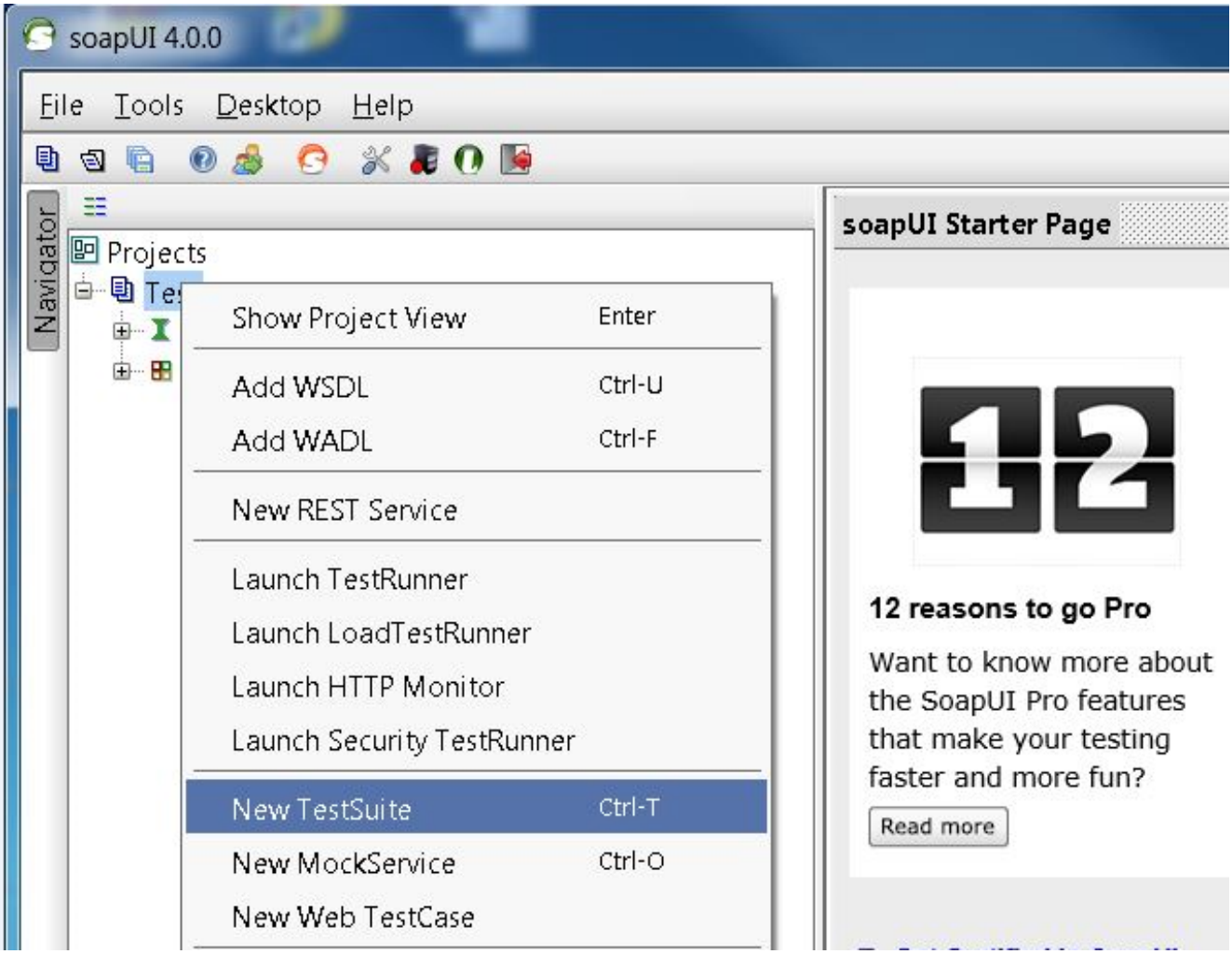
In diesem Verfahren wird erläutert, wie eine Testsuite erstellt wird, die automatisiert werden kann, wenn APIs an QPS gesendet werden.

In diesem Beispielverfahren wird die Testsuite über eine Liste von Subscriber-IDs hochgeladen und verwendet diese Teilnehmer-IDs in der querySubscriberRequest, die sie an QPS sendet. Die

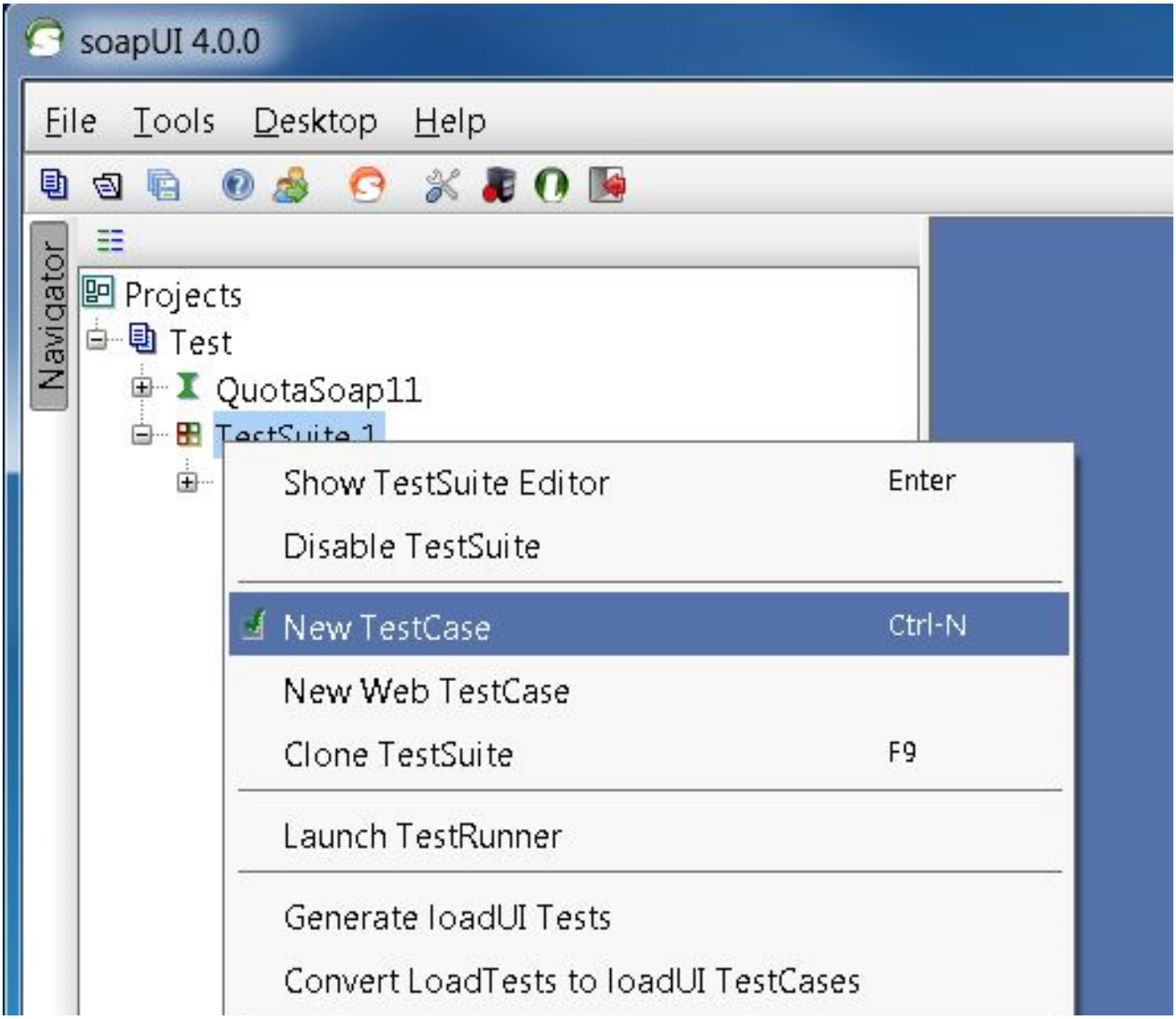
Liste der Teilnehmer-IDs befindet sich jeweils in einer einzelnen Zeile in einer Textdatei mit dem Namen **subid.txt**.

Gehen Sie wie folgt vor, um die Test Suite zu erstellen:

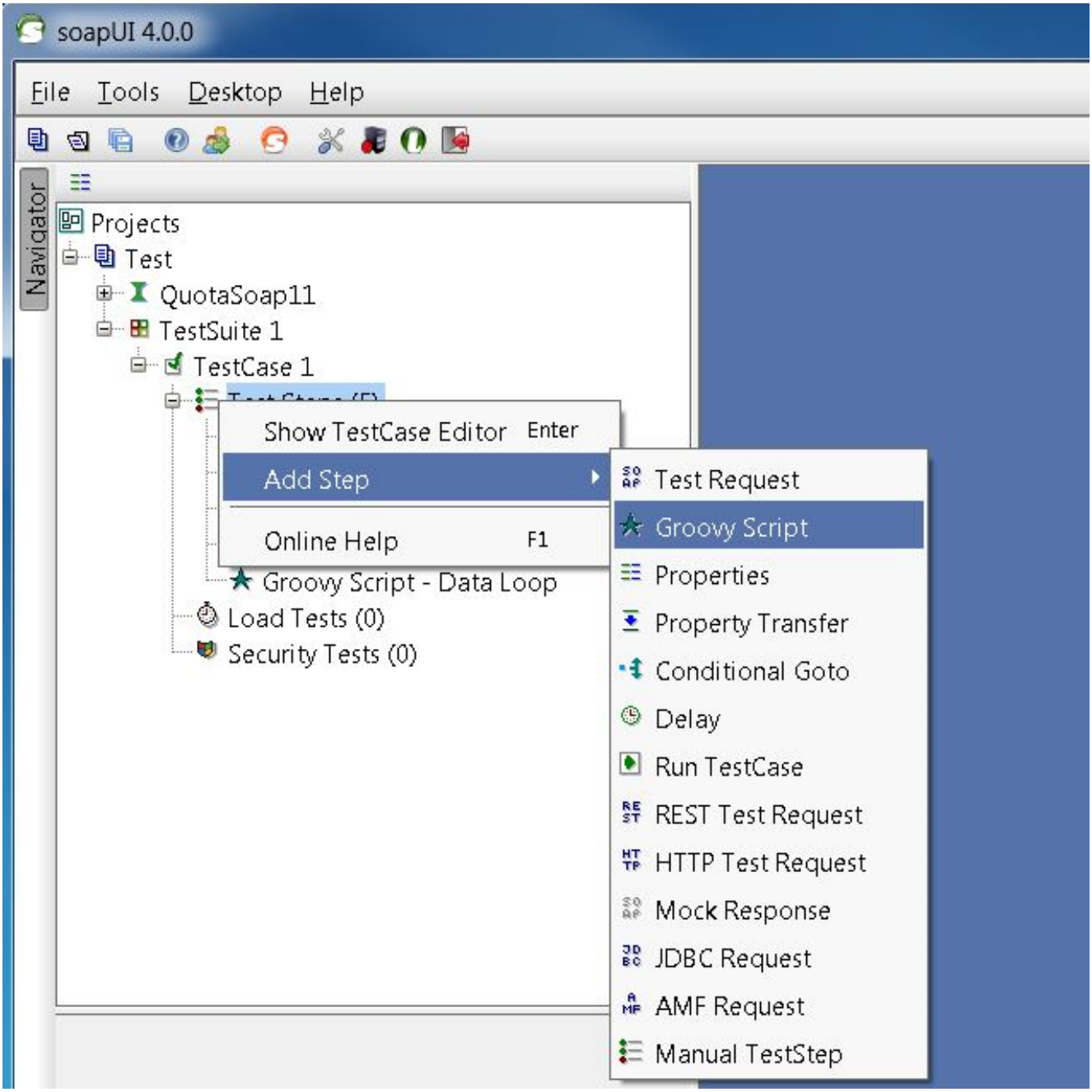
1. Erstellen Sie im von Ihnen erstellten soapUI-Projekt eine neue Test Suite. Klicken Sie mit der rechten Maustaste auf die soapUI, und wählen Sie **New TestSuite aus**.



2. Klicken Sie mit der rechten Maustaste auf die Testsuite, und wählen Sie **Neuer TestCase aus**.



3. Klicken Sie mit der rechten Maustaste auf den Testfall, und wählen Sie **Add Step > Groovy Script** aus, um einen Groovy Script-Testschritt hinzuzufügen. Nennen Sie es **Datenquelle**:

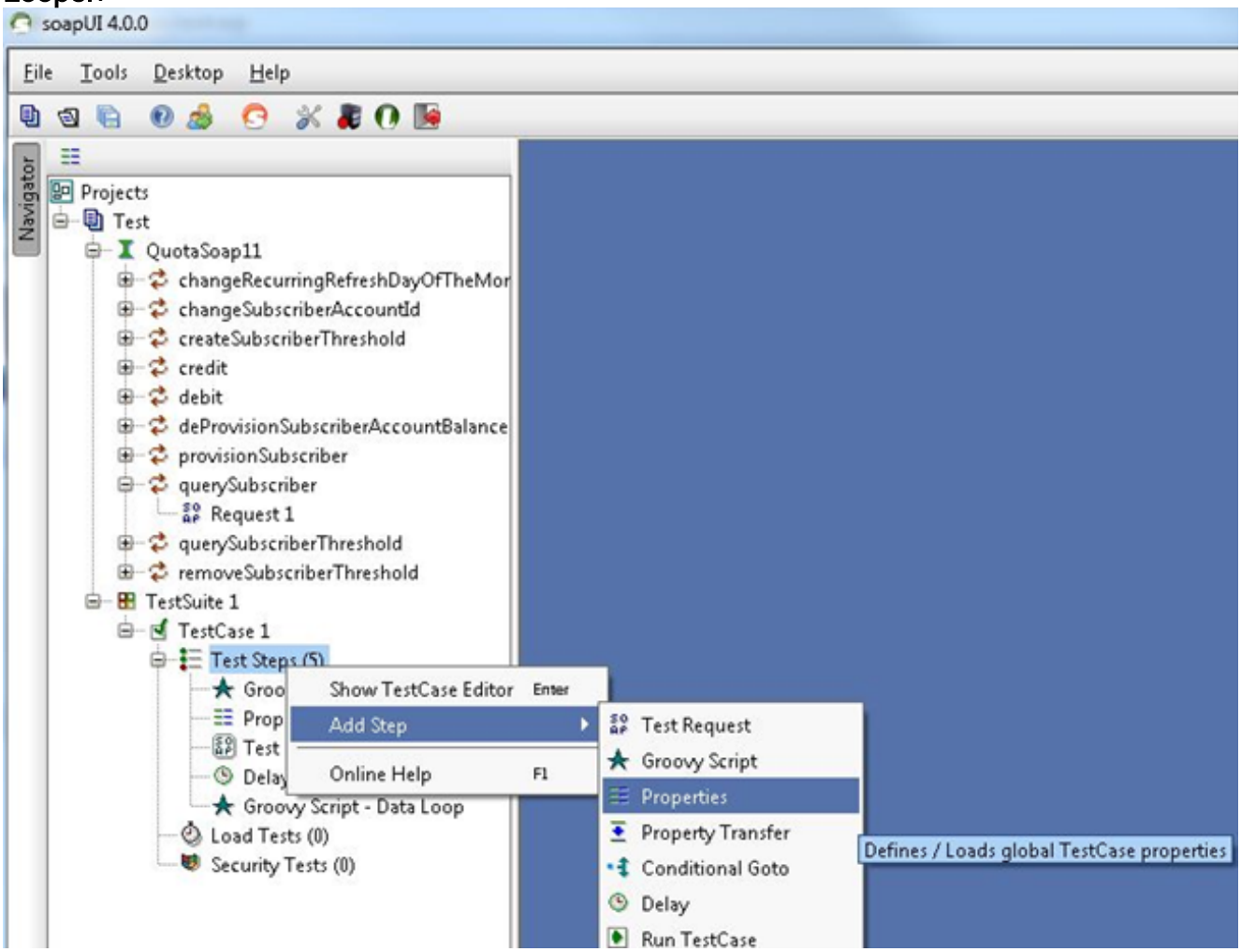


4. Fügen Sie diesen Code in der Datenquellendatei ein. Dieser Code liest Datei **C:/subid.txt**, die eine Teilnehmer-ID für jede Leitung enthält:

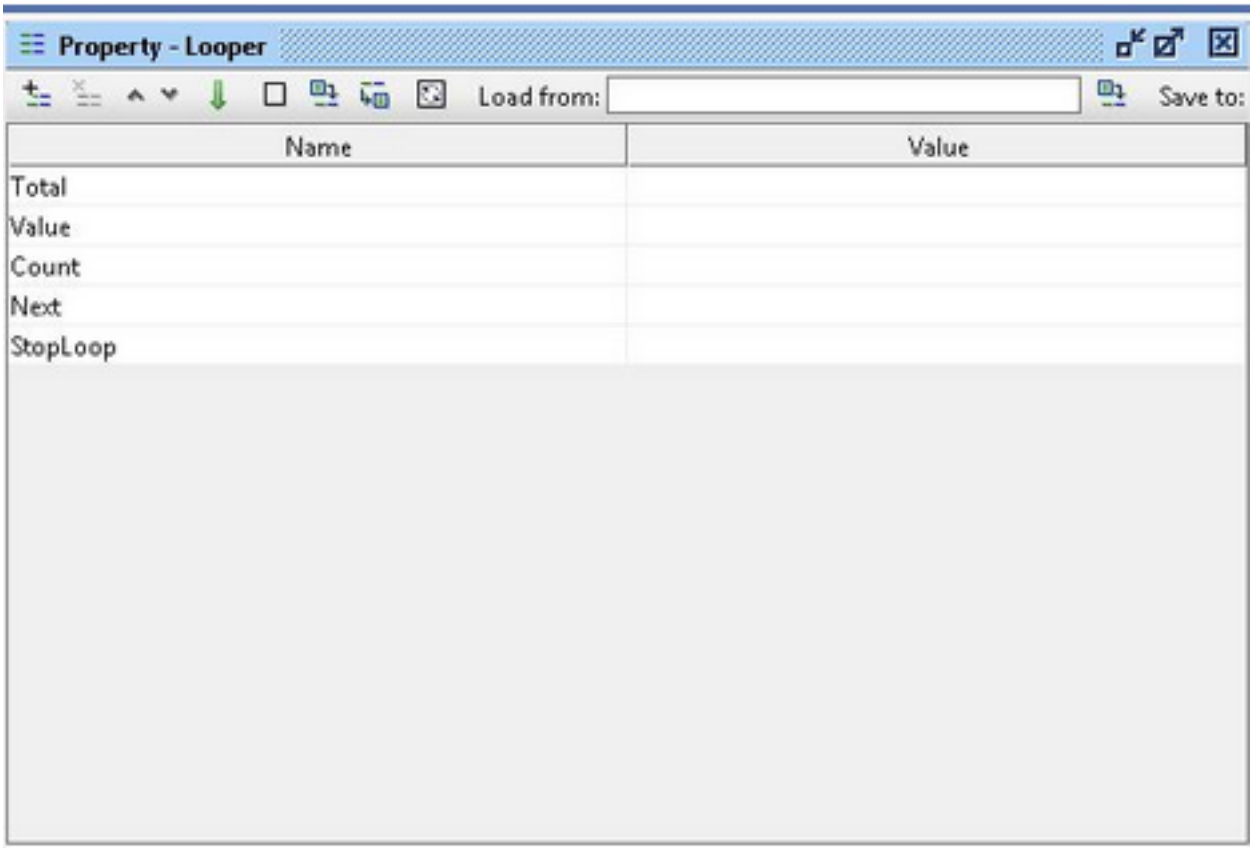
```
import com.eviware.soapui.support.XmlHolder def myTestCase = context.testCase
def counter,next,previous,sizeFile tickerEnumFile = new File("C:/subid.txt") //subscriber
IDs sepearted by new line (CR). List lines = tickerEnumFile.readLines() size =
lines.size.toInteger() propTestStep = myTestCase.getTestStepByName("Property - 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. Klicken Sie mit der rechten Maustaste auf den Testschritt, und wählen Sie **Add Step > Properties (Schritt hinzufügen > Eigenschaften)** aus, um einen Eigenschaftentestschritt

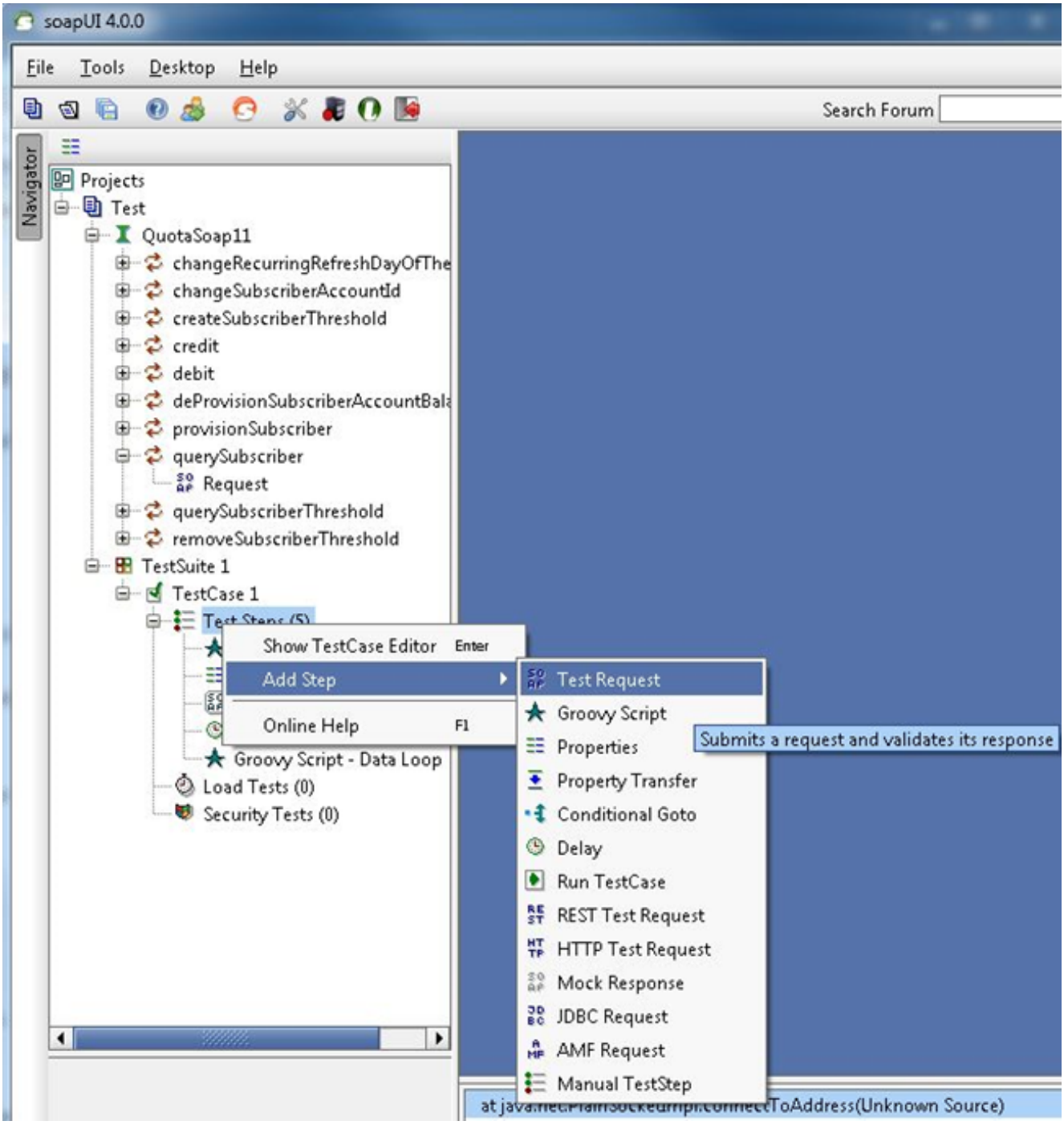
hinzuzufügen, und nennen Sie ihn **Property - Looper**.



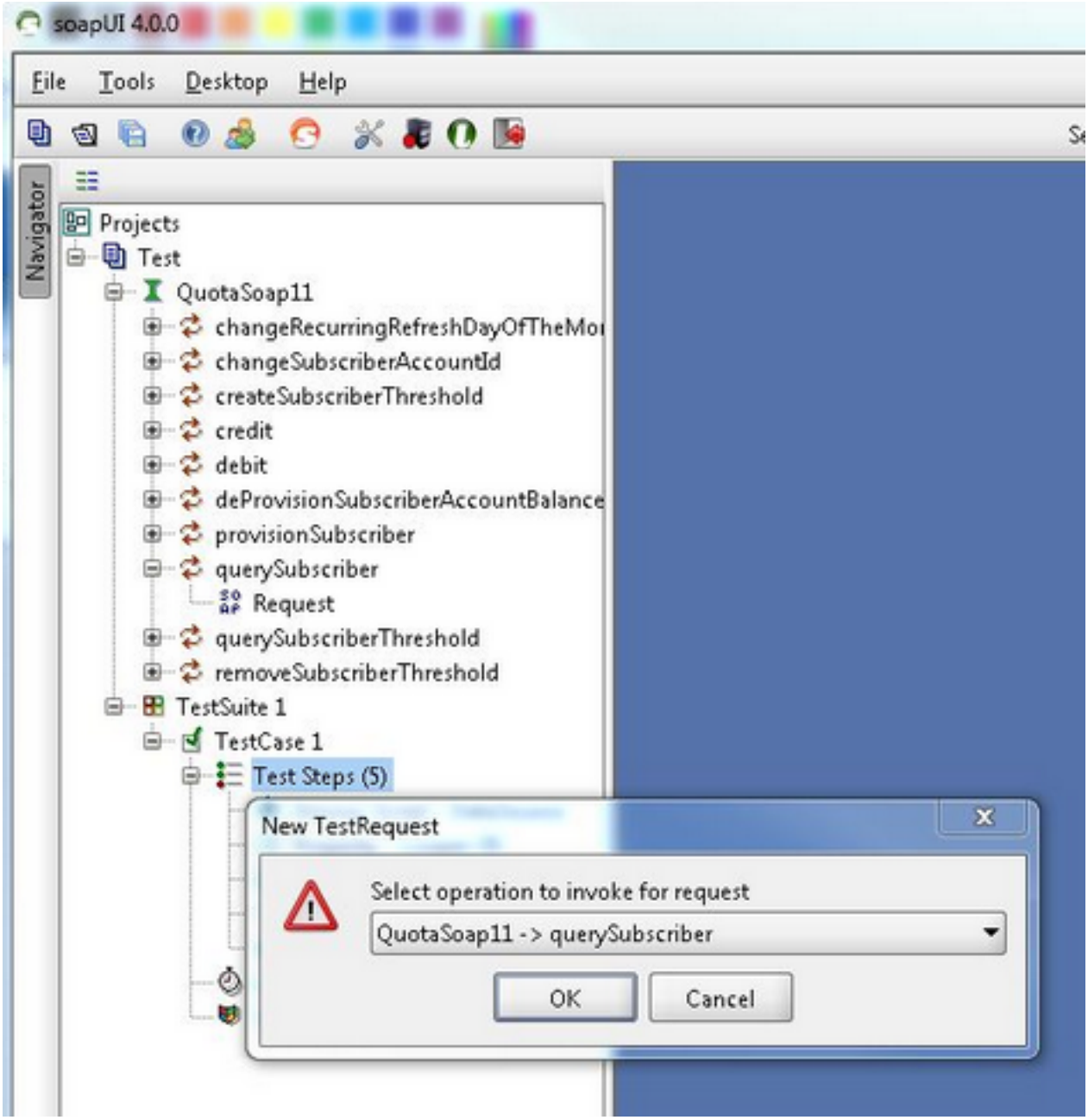
6. Fügen Sie diese benutzerdefinierten Eigenschaften des Looper-Testschritts hinzu: InsgesamtWert: In unserem Beispiel wird die Abonnent-ID aus der Datei Subscriber-IDs gelesen.AnzahlWeiterStoppLoop



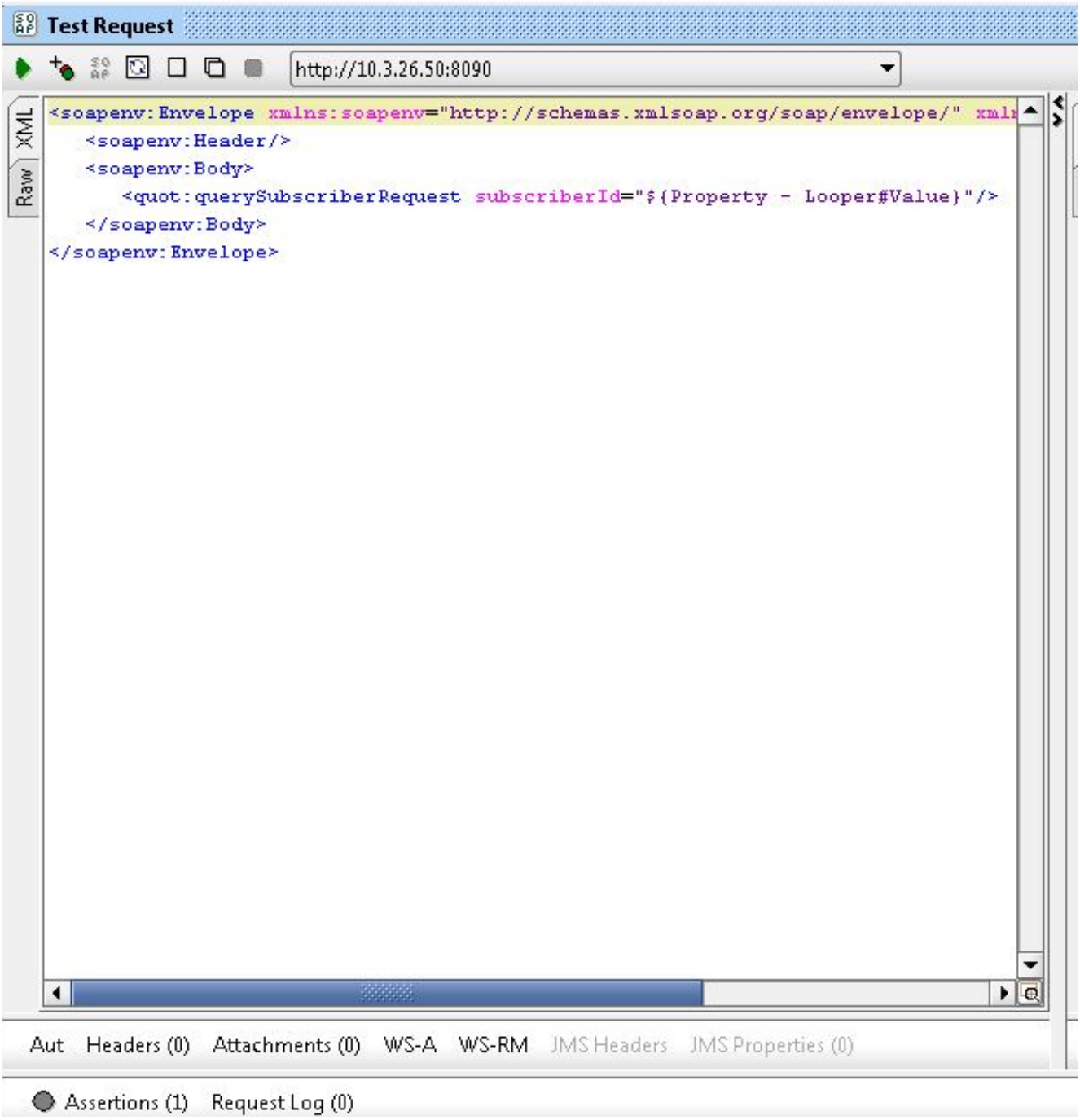
7. Klicken Sie mit der rechten Maustaste auf den Testschritt, und wählen Sie **Add Step > TestRequest**, um einen Test-Anforderungstest-Schritt hinzuzufügen, und wählen Sie die Anforderung aus, die Sie aufrufen
möchten:



In diesem Beispiel wird querySubscriberRequest verwendet.

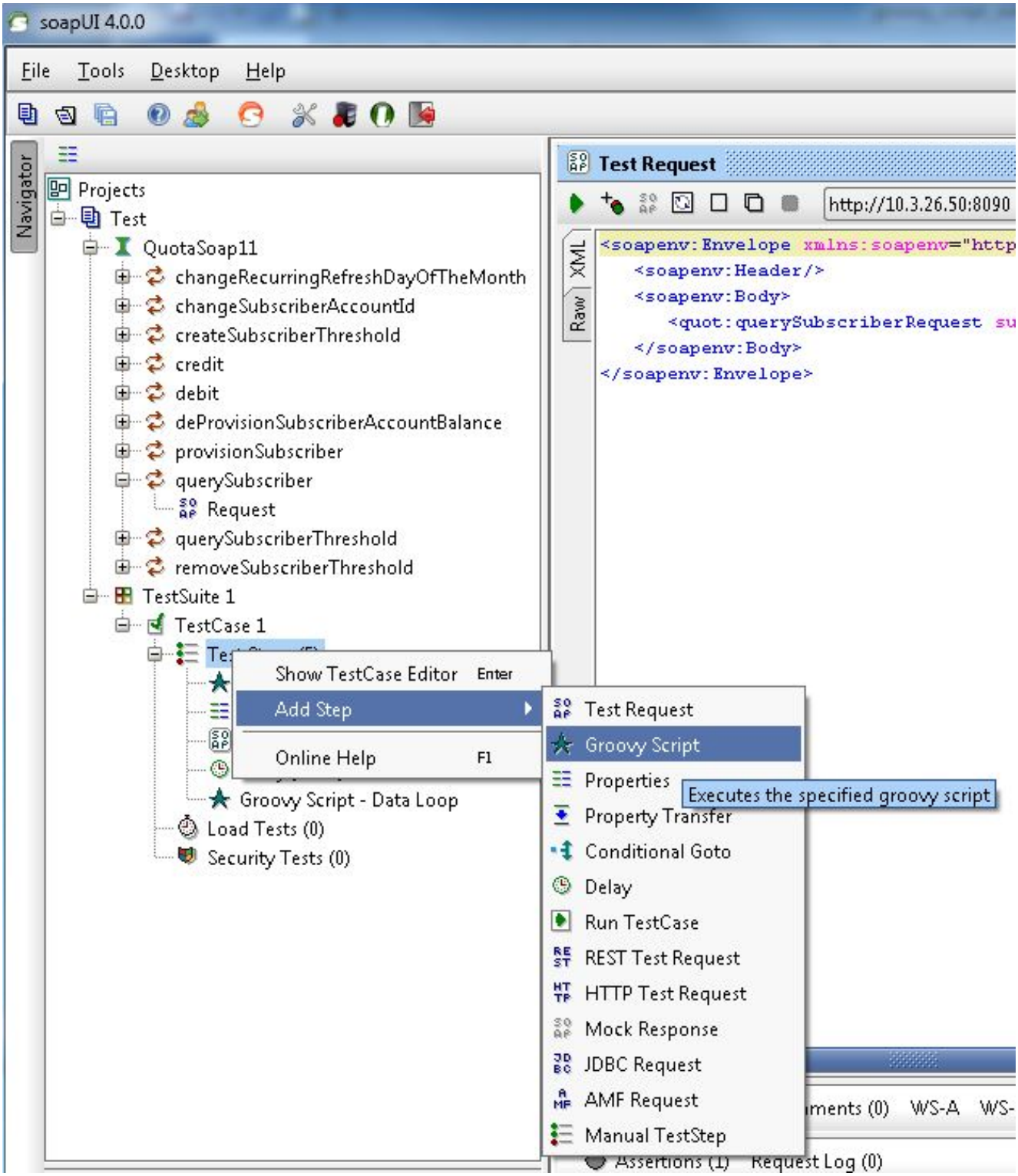


8. In der Anforderung ersetzt der Erweiterungscode die Feldwerte der Abfrage. In diesem Beispiel wird das **?** der **SubscriberId=?** in der querySubscriberRequest wird durch den Erweiterungscode **${Property - Looper#Value}** (soap_test_req_expansion_code) ersetzt:



Eigenschaft - Looper ist der Name des zuvor erstellten Property TestStep, und **Value** enthält die aktuelle Teilnehmer-ID, die aus der Datei der Teilnehmer-IDs gelesen wird.

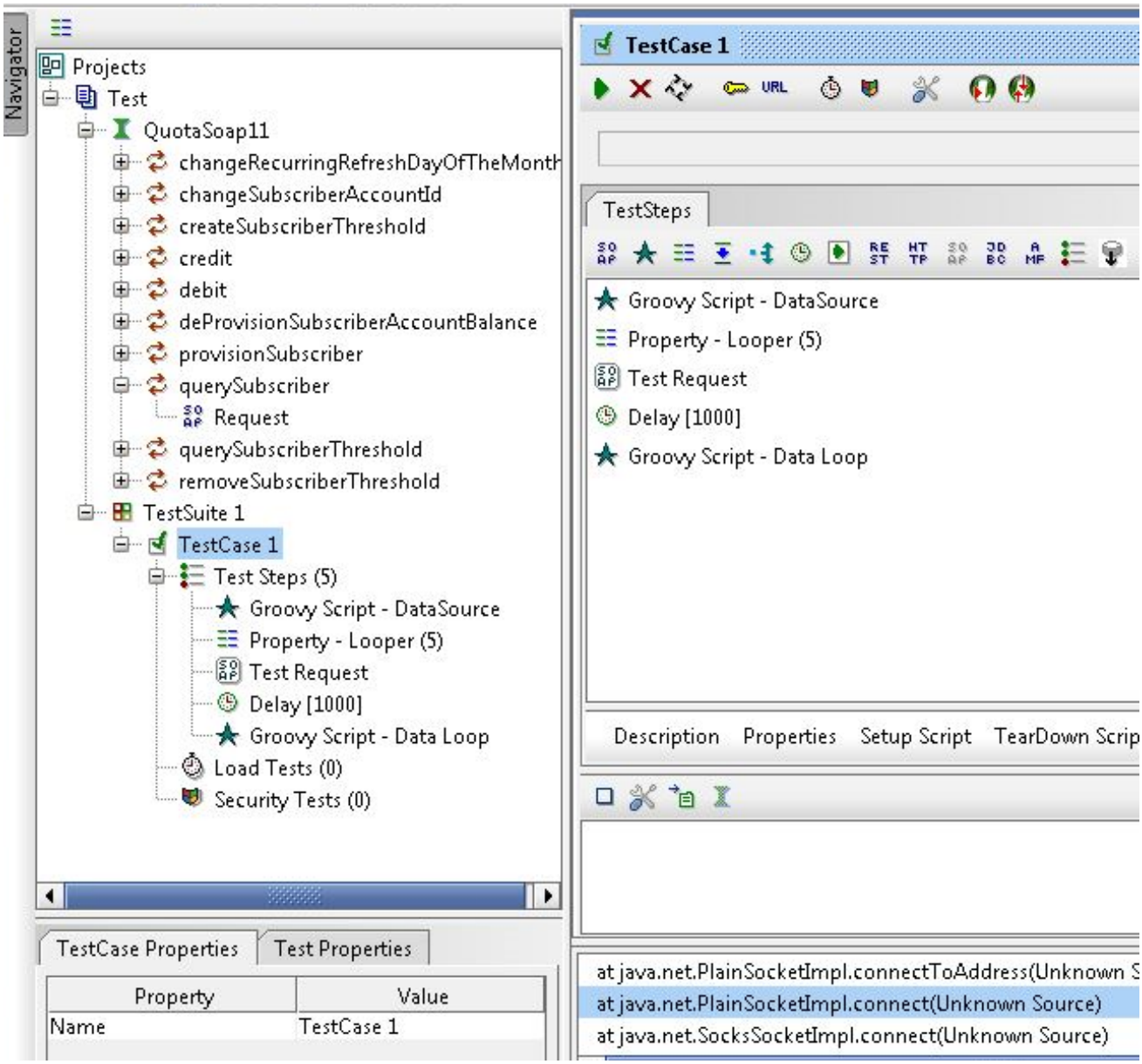
9. Klicken Sie mit der rechten Maustaste auf den Testschritt, und wählen Sie **Add Step > Groovy Script** und nennen Sie es **Datenschleife**:



10. Fügen Sie diesen Code in der Groovy Script Data Loop ein:

```
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. In diesem Beispiel wird eine 1000-ms-Verzögerung zwischen den einzelnen Schleifen hinzugefügt. Dieser Schritt ist optional.Mit der Verzögerung gibt es nun fünf Testschritte:



12. Klicken Sie auf die grüne Schaltfläche **Ausführen**, um die fünf Testschritte im Fenster TestCase auszuführen.