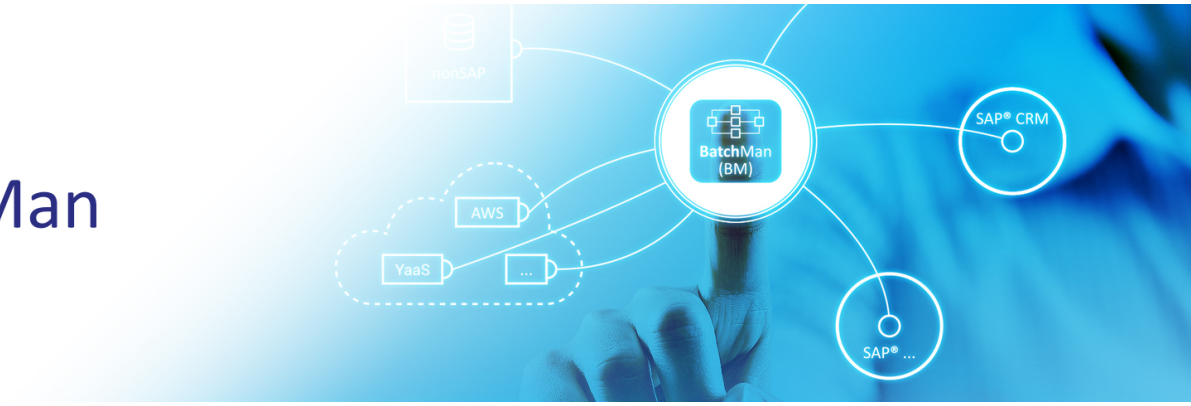
BatchMan



Web Services Manual

Product info & Support

Phone: + 49 (0) 40/32 80 86 - 0
E-Mail: batchman@honico.com
Web: www.honico.com
Release: BatchMan 5.0
Version: February 2018

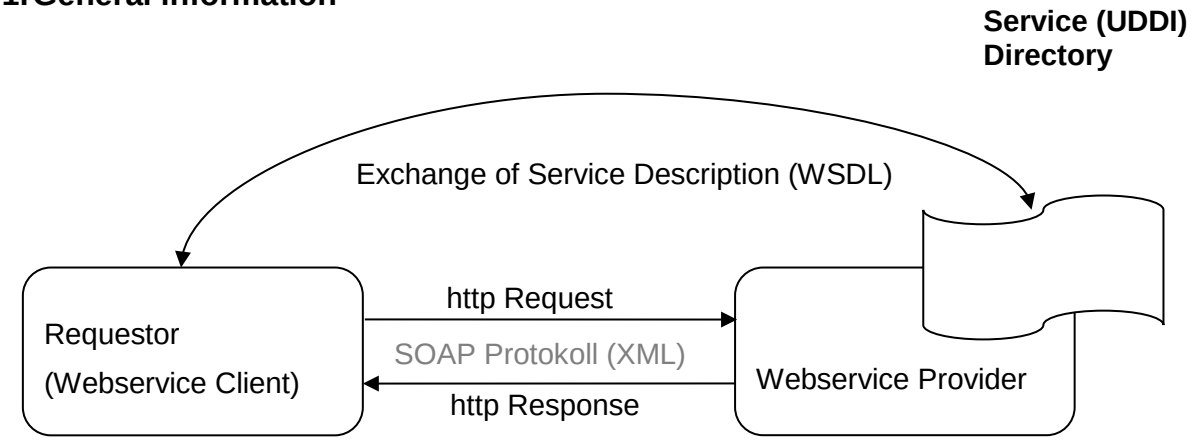
SAP® Certified
Powered by SAP NetWeaver®

SAP Partner

Contents

1. Web services

1.1. General information



A web service is a service provided by a network and that supports interaction with other network participants. The processes interact using SOAP (Simple Object Access Protocol), here data is usually swapped as XML via http(s).
Each web service defines the form of the data exchange in its service description, the WSDL (Web Service Definition Language).

1.2. Communication

To communicate with a web service using SOAP, you need the following information, which you can extract from the WSDL (service description):

- Web service endpoint
  - URL which can be used to address the web service
- SOAP action
  - Selection criterion in case a web service provides several services
- SOAP message
  - Input parameter in the XML form

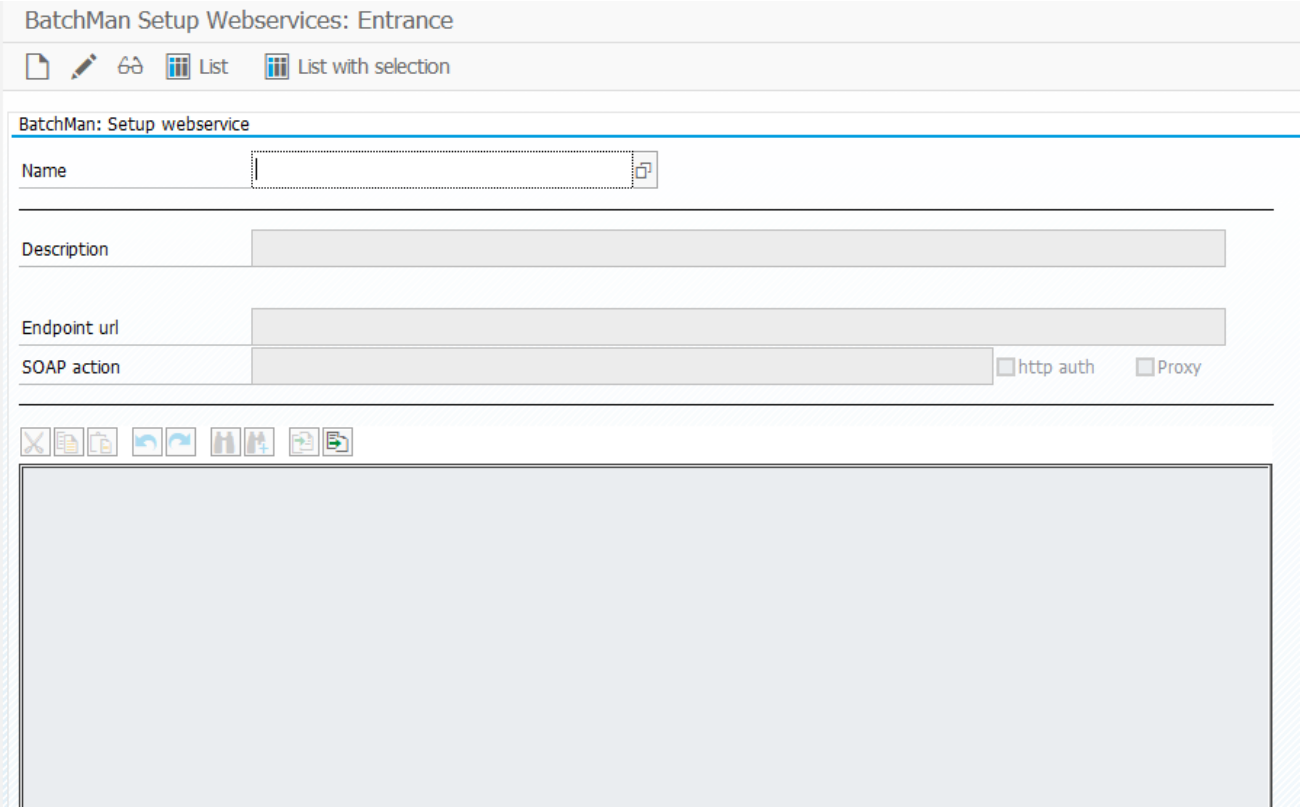
Since setting up the SOAP message manually from a WSDL file is usually very complicated depending on the complexity of the related web service, it is recommended you use utility programs (WSDL parser). These programs extract the information described above from the WSDL file.
An example for a relatively popular WSDL parser is SoapUI (soapui.org), which is used in part the section below. The basic version is subject to the GNU Lesser General Public License and is free. It allows functional tests of the web service in addition to extracting the WSDL file.

2. BatchMan web service client

2.1. Web service set up

To access the setup of the BatchMan web service client, choose:

BatchMan®-*Function Tree → Customizing → Technical Customizing → Set Up Web Service*



Enter a name for the web service. If you choose “Create” and enter a description, the three fields described above remain “Endpoint URL” “SOAP Action” and “SOAP Message” (cf. 1.2 Communication).
As of SAP Rel. 7.01, the system queries the WSDL when creating the URL and then parses – thus manual parsing is not required (cf. 1.2 Communication). There are also the fields “http Auth” with “User” and “Password”. You only need these fields if the http server of the web service requires an authentication (*Basic Authentication* per RFC 2617).

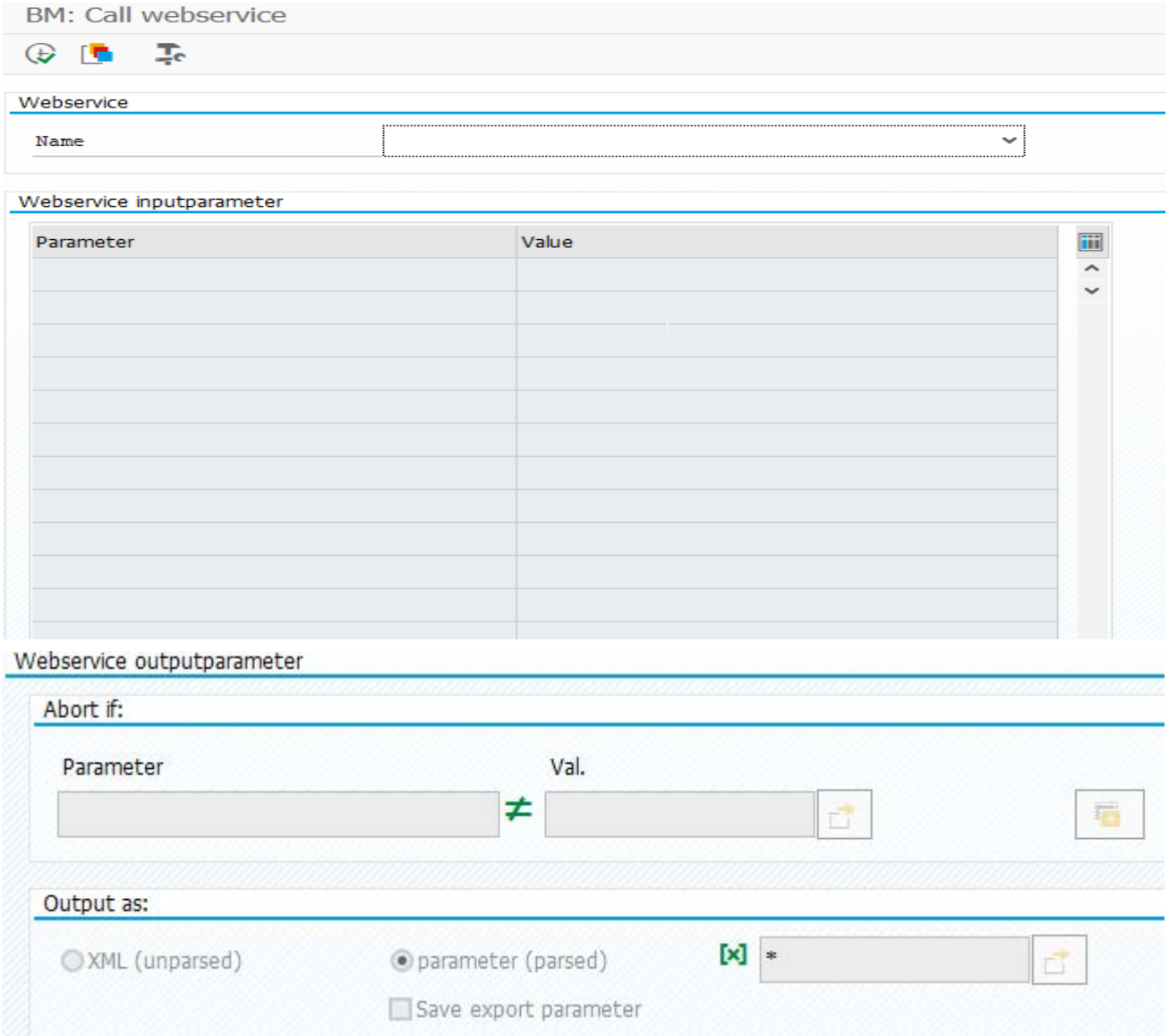
You will need the fields “Proxy” with “Proxy computer” “Proxy user” and “Proxy password” if a proxy server is required for the communication with the http server.

2.2. Web service client

After you set up a web service in BatchMan, you can call it using the BatchMan web service client.

To reach the web service client, choose:

BatchMan® *Function Tree → Customizing → Technical Customizing → Variants for Web Service*



You can use this report both in the dialog mode and in the batch mode, and for alarms as an alert tool.
You must create a variant to use the report in the batch mode or as an alert tool.
Under the “Name” field, you have a list box with all web services defined in BatchMan (cf. 2.1 Web service set up).

Once you choose a web service, the system displays the related input parameters in the parameter table. At this point, you can assign fixed or dynamic values to the input parameters.
To assign a fixed value to a parameter, enter it into the related table row. To assign a dynamic value to a parameter, enter the replacement parameter into the related table row. The following replacement parameters are possible (cf. BM4 UserHB, line 227):

<OPSYS>	Operating system
<INSTANCE>	Instance of R/3 application
<SYSID>	Name of the R/3 application as per system field SY- SYSID.
<DBSYS>	Database system as per system field SY- DBSYS
<SAPRL>	R/3 Release as per system field SY- SAPRL
<HOST>	Computer name as per system field SY- HOST
<CLIENT>	Client as per system field SY- MANDT
<LANGUAGE>	Logon language as per system field SY- LANGU
<DATE>	Date as per system field SY-DATUM
<YEAR>	Year as per system field SY-DATUM, with four places
<SYEAR>	Year as per system field SY-DATUM, with two places
<MONTH>	Month as per system field SY-DATUM
<DAY>	Day as per system field SY-DATUM
<WEEKDAY>	Weekday as per system field SY- FDAYW
<TIME>	Time of day as per system field SY- UZEIT
<STIME>	Hour and minute as per system field SY- UZEIT
<HOUR>	Hour as per system field SY- UZEIT
<MINUTE>	Minute as per system field SY- UZEIT
<SECOND>	Second as per system field SY- UZEIT
<P=name>	Value of a profile parameter of the running system
<V=name>	Value of a variable as per variable table (TCODE FILE)
<F=name>	Return value of a function module
<T=name>	TVARV variables
<L=logical file>	Defines a logical file name (TCODE FILE)
<A=logical path>	Defines a logical path (TCODE FILE)

To use as an alert tool, these replacement parameters are available:

&HOST	Host name (field SY-HOST from system table SYST)
&SID	SAP system identifier (field SY-SYSID from system table SYST)
&MDT	Client (field SY-MANDT from system table SYST)
&RFC	RFC connection (target system) of a job
&JOB	Job name
&COUNT	Job count
&DATE	Date
&TIME	Time
&CHECKP	Name of checkpoint that triggered the alarm
&OPA	Number of the order
&USER	User name from job step (only master system) as first assignment, otherwise user from message recipient table
&TELEFON	Telephone number from message recipient table
&FAX	Fax number from message recipient table
&PAGER	Mobile or pager number from message recipient table
&MAIL	E-mail address from message recipient table
&ACLASS	Alarm class from message recipient table
&MESS	Contains the compound text to use as the subject line
&TID	Assignment of a transaction ID (according to SAP: “a globally unique ID of the current unit of work”)
&MAILFILE	Placeholder for email text of alert tool

&LOGFILE	Placeholder for job log
&LOGFILE[]	Placeholder for job log, BASE64 coded

The reply from the web service can be handled using the table with the input parameters. Here you can define which output parameters will cause an abort and when.
You can also find the names of the output parameters in the WSDL file.
You can simply use the web service as a test to get to the output parameters.
Output options include a structured (only parameter) and an unstructured output (completes XML). With the structured output, you can also select the output parameters that are to be output. If you use the web service for alarms, you have the special feature that the system saves in the alarm history the output parameter defined at the first place here.

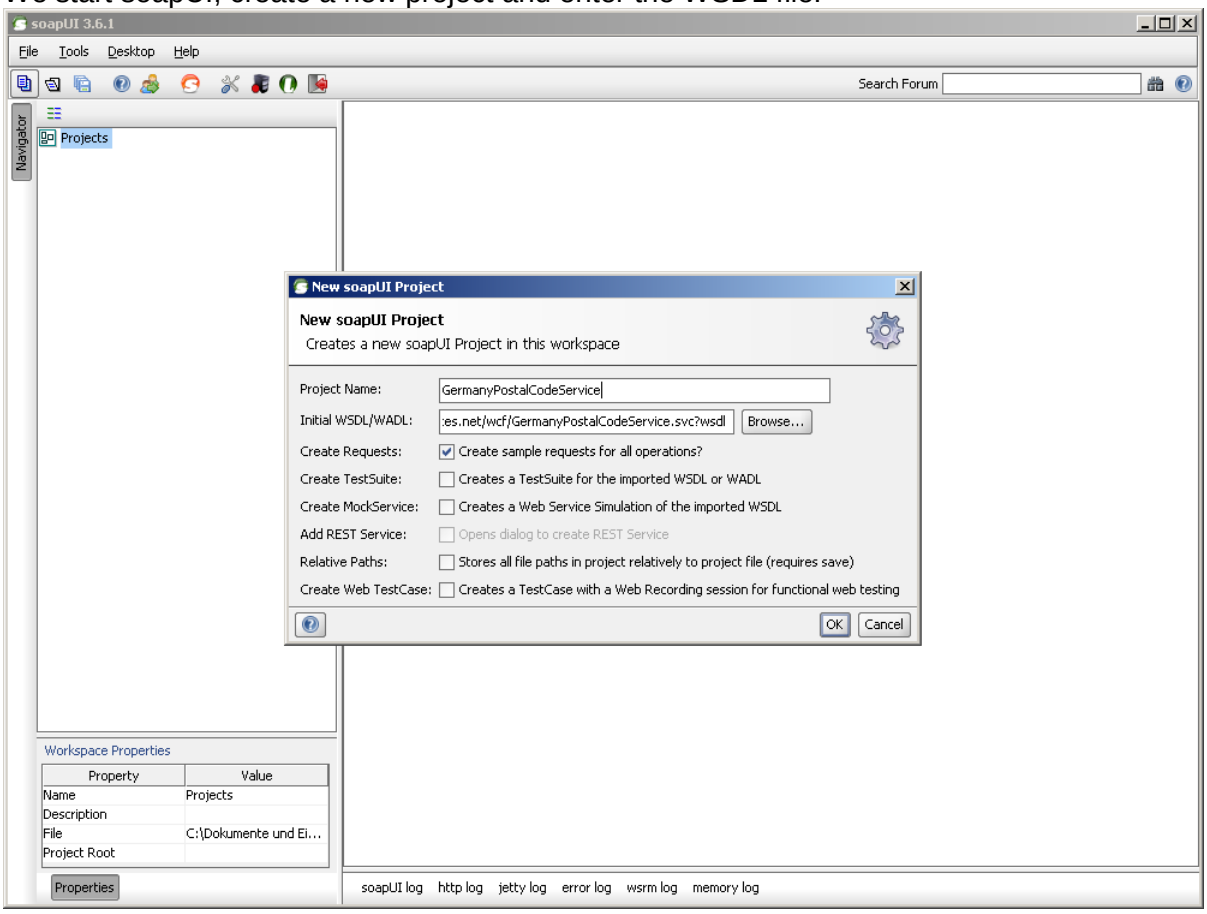
3. Web service example

3.1. WSDL extraction

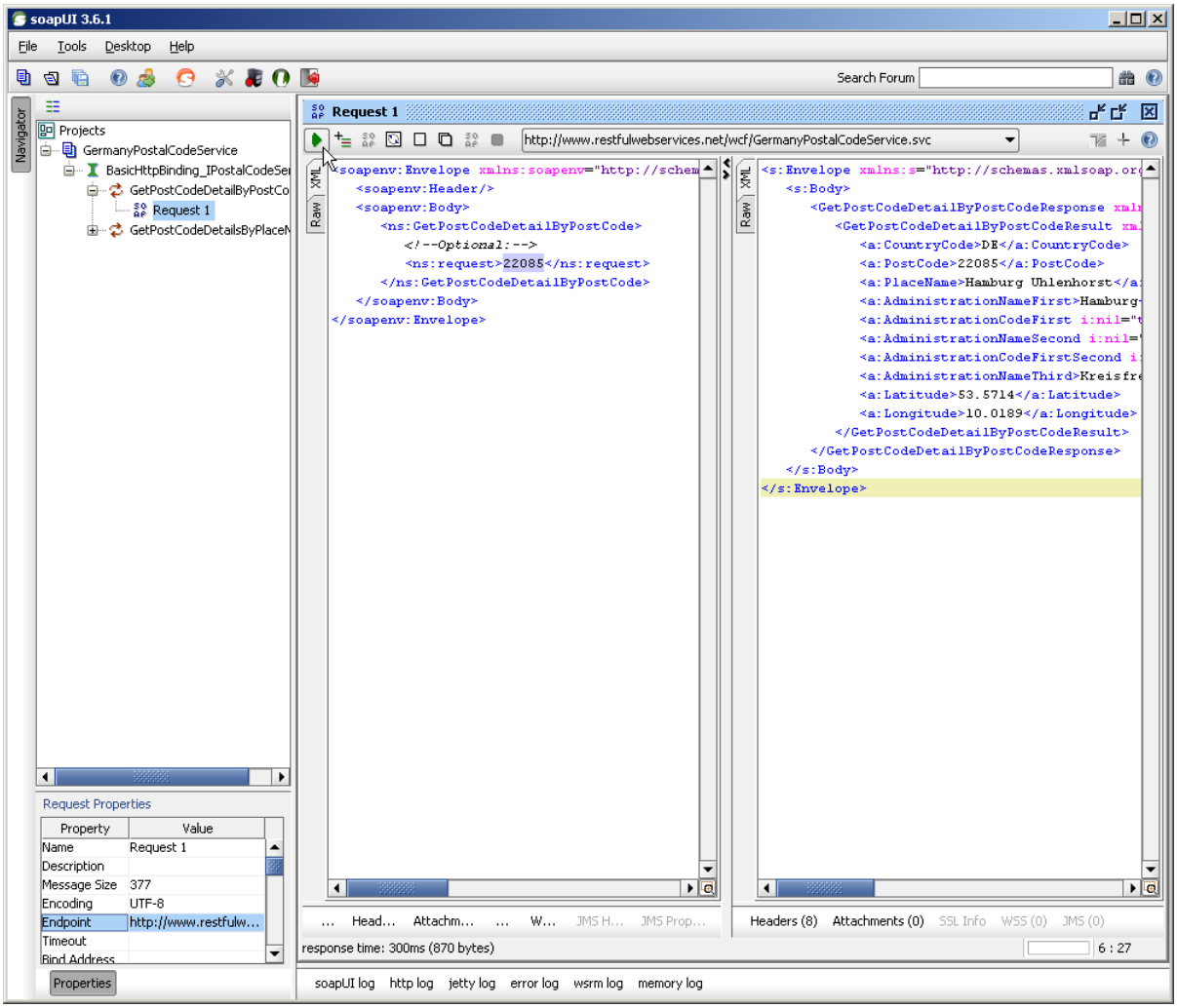
We now want to call a test web service as an example using BatchMan.
Our test web service is called “GermanyPostalCodeService” and has this WSDL:
http://www.restfulwebservices.net/wcf/GermanyPostalCodeService.svc?wsdl

We first need the three parameters “Web service endpoint", “SOAP action” and “SOAP message” (cf. 1.2 Communication).

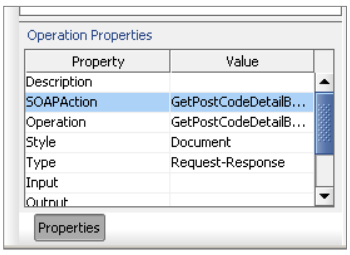
We start soapUI, create a new project and enter the WSDL file.



We can see in the left on the tree that our web service offers two methods and we want to look at the “GetPostCodeDetailByPostCode” method in more detail.
If you double-click on Request 1, the system generates the SOAP input message. On the right double window, you can see the input message (request) to the left; the output message (response) is to the right.
You only get the output message once you send the input message. To test the web service, you can enter an input parameter for <ns:request> but this is not required because we only want to get to the input message.



At the bottom right, you can see the entry for “endpoint” and so you have two of your three parameters.

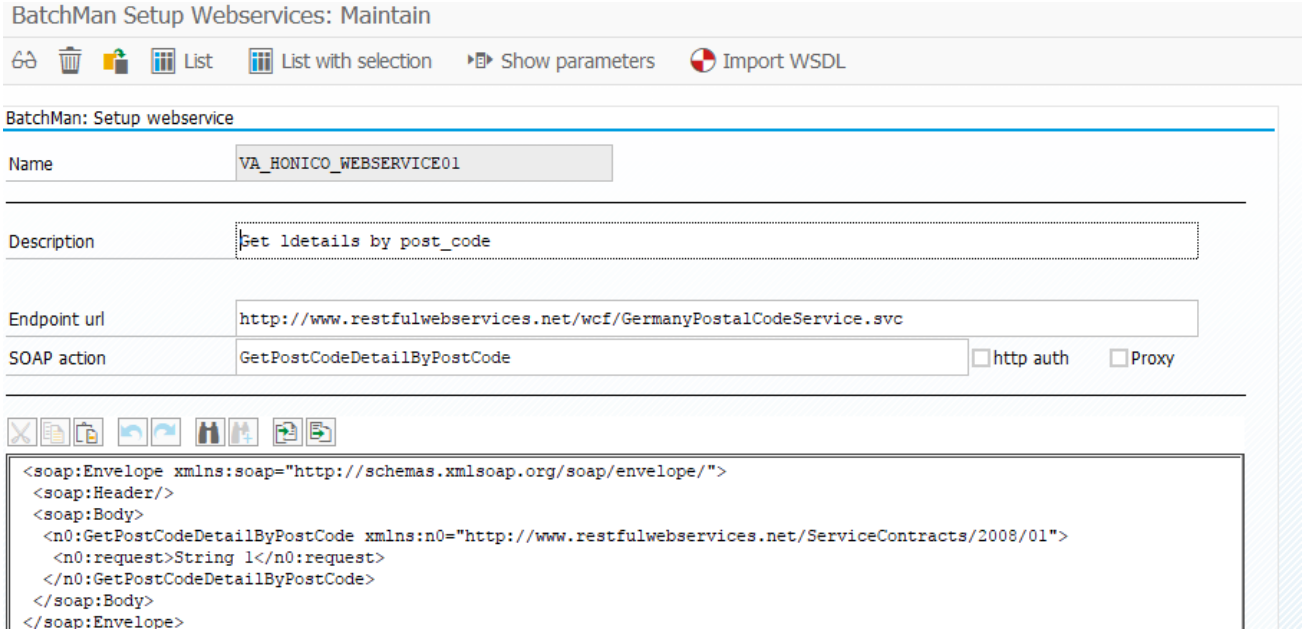


To get to the third parameter, again choose GetPostCodeDetailByPostCode on the left in the tree instead of on Request 1 and the system displays the last “SOAP action” parameter.

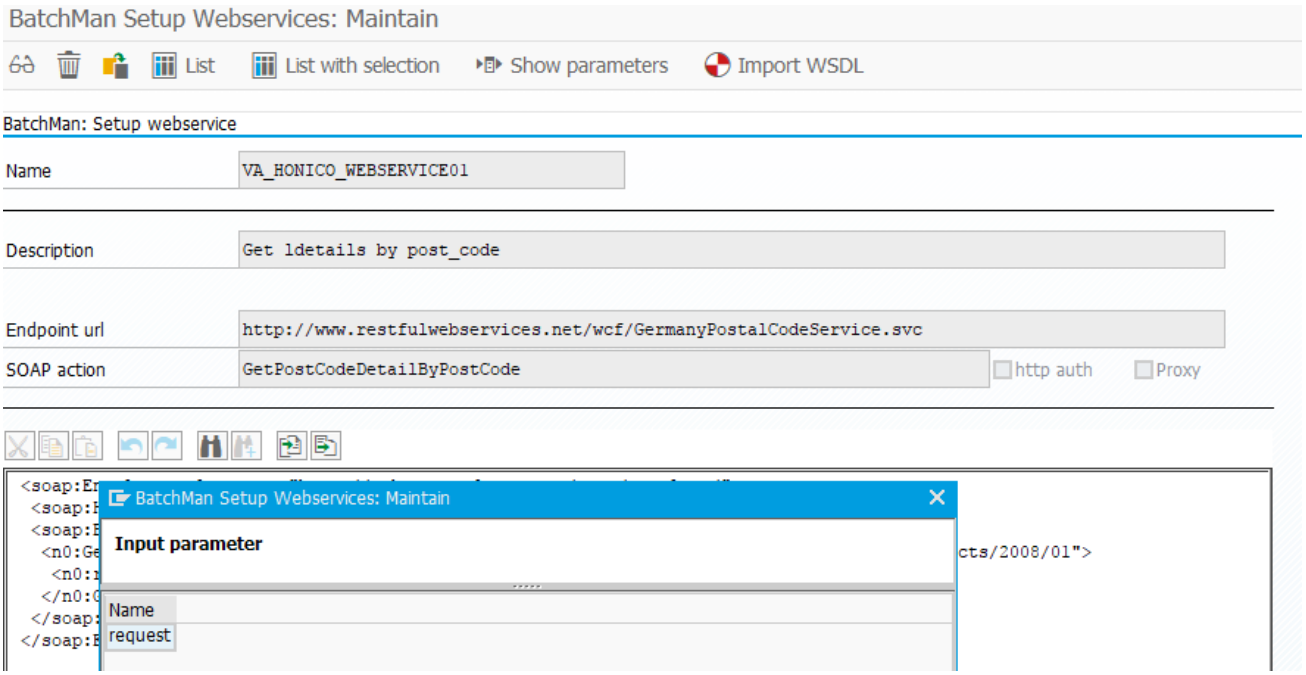
With our three parameters, we leave the program soapUI and go to BatchMan.

3.2. BatchMan web service set up

Start the BatchMan web service setup (cf. 2.1 Web service setup) and enter any name. Then choose “Create” and enter a description.
Now transfer into the other fields the copied values “Web service endpoint", “SOAP action” and “SOAP message” (cf. 3.1 WSDL Extraction).



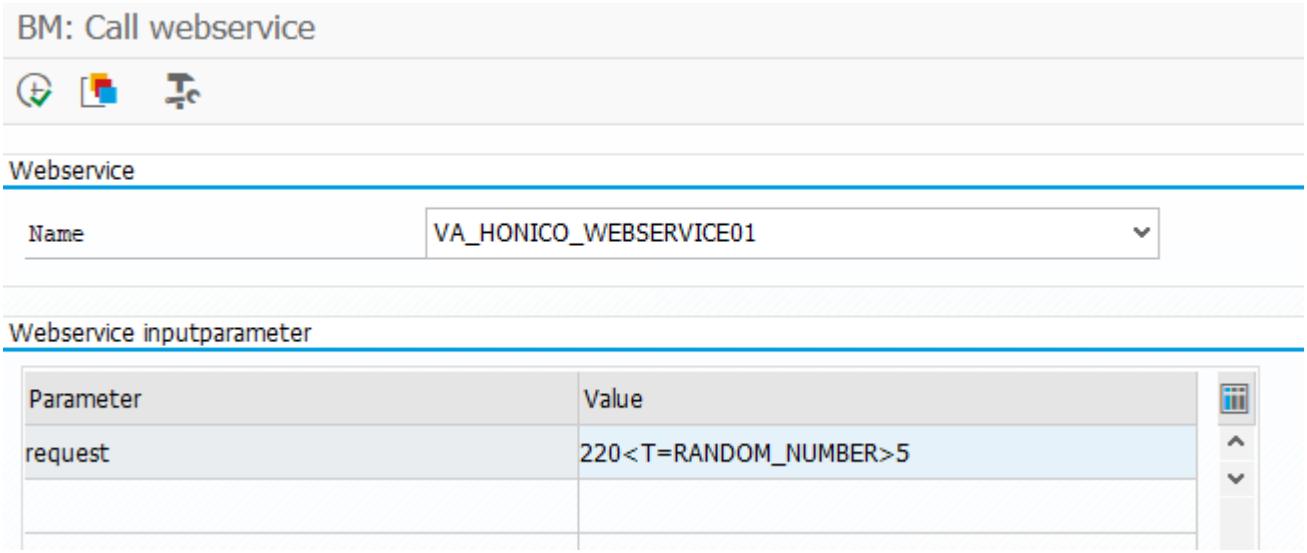
Use “Display parameters” to view the parameters from the SOAP message. Our test web service now has only one input parameter with the “request" name. You can give this parameter values or variables in the next section.



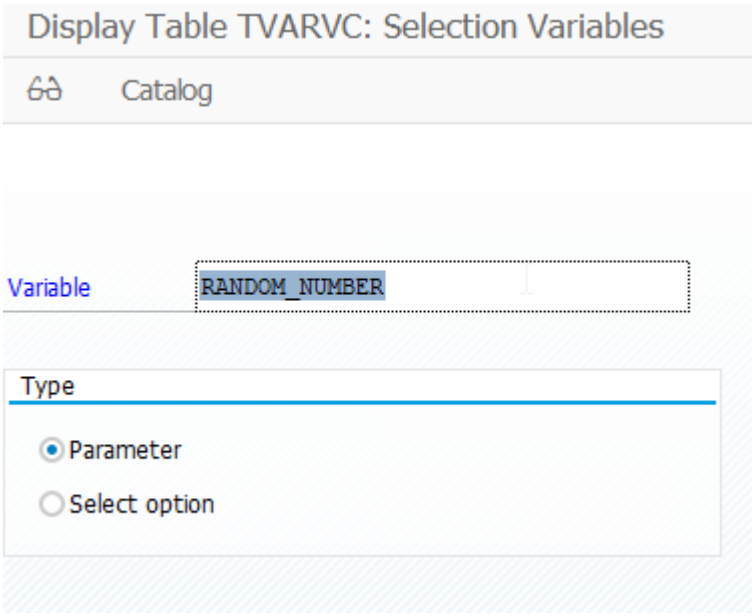
3.3. BatchMan web service client

Start the BatchMan web service client (cf. 2.2 Web service client).
From the list box, select the web service configuration previously defined (cf. 3.2 BatchMan web service set up).
In the parameter table, the system again displays your parameter “request” and you can assign it a value. This can be a fixed value like 22085 or a dynamic value like from TVARV.

We use a mix of both: 220<T=RANDOM_NUMBER>5 where the value from <T=RANDOM_NUMBER> is replaced at runtime by the corresponding value from the TVARV(C) table (in our case the value 8).
So, at runtime you have the value 22085.



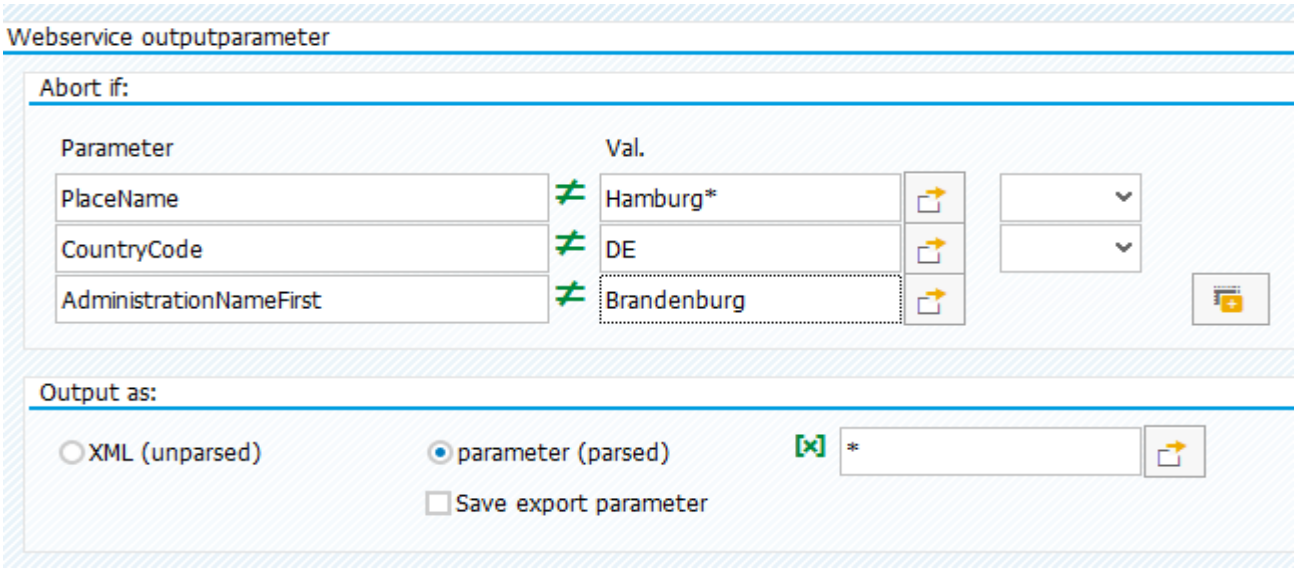
Save the selection in a variant and execute the report. You get the answer from the web service as an output list.



BM: Call webservice

Parameter	Value
GetPostCodeDetailByPostCodeResult:a	http://www.restfulwebservices.net/DataContracts/2008/01
GetPostCodeDetailByPostCodeResult:i	http://www.w3.org/2001/XMLSchema-instance
CountryCode	DE
PostCode	22085
PlaceName	Hamburg Uhlenhorst
AdministrationNameFirst	Hamburg
AdministrationCodeFirst:nil	true
AdministrationCodeFirst	
AdministrationNameSecond:nil	true
AdministrationNameSecond	
AdministrationCodeFirstSecond:nil	true
AdministrationCodeFirstSecond	
AdministrationNameThird	Kreisfreie Stadt Hamburg
Latitude	53.5714
Longitude	10.0189

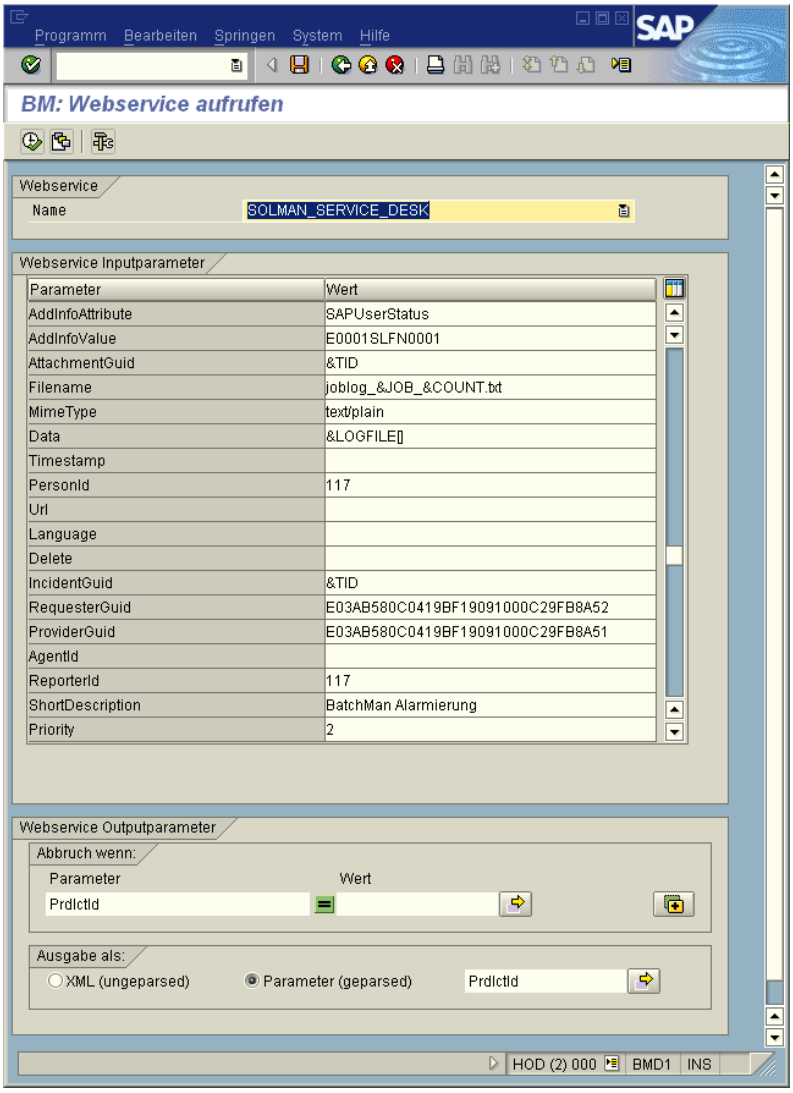
You can select parameters here that are to trigger an abort. Usually the web service issues return codes as parameter, but this is not so in your case.
Nevertheless, you should trigger for certain parameters because otherwise the system will only abort with a faulty http status, but the web service by default usually only issues an error for the http status with syntax issues.
For our example, trigger for the parameters “PlaceName”, “CountryCode” and “AdministrationNameFirst”.
So, the system will abort only if the zip code 22085 is not somewhere in Hamburg, Germany is not the country or Brandenburg is not returned as the state.


Webservice outputparameter
Abort if:
Parameter
Val.
PlaceName
Hamburg*
CountryCode
DE
AdministrationNameFirst
Brandenburg
Output as:
XML (unparsed)
parameter (parsed)
*
Save export parameter

4. Alert tool web service

4.1. Web service variant

To set up a web service variant for alarms, follow the steps in item 2.2 Web service client (prerequisite is web service set up, cf. 2.1).



Assign the values you want to the parameters. You can also use the replacement parameters for the alarms.
Define meaningful abort parameters, like (return ID is empty (image), RC not equal to 0 etc.).
At the bottom right, define the output parameter the system is to send later to the alarm history.

You can track a web service abort for the alarm at three places.

- SLG1 application log
- Job log of the job concerned
- Output parameter in the alarm history

4.2. Alert tool set up

You can set up the alert tool as described in the alert manual. Use the “Web service” system type. As a web service variant, you can use all variants of the web service client (cf. 4.1 Web service variant).