Medexter Healthcare

HOW TO CALL ARDEN SYNTAX MLMS ON AN ARDENSUITE SERVER USING REST AND SOAP

SIRS Notification as an Example

Datum: 11/26/2018

Version: 1.3

Medexter Healthcare GmbH, Borschkegasse 7/5, A-1090 Vienna, Austria, Telephone +43-1-968 03 24, Facsimile +43-1-968 09 22 www.medexter.com, Handelsgericht Wien: FN 225564m, UID: ATU54901101, IBAN: AT602011128014988900, BIC: GIBAATWW

Summary

The aim of this *how-to* instruction manual is to show how to call Arden Syntax Medical Logic Modules (MLMs) deployed on an ARDENSUITE Server and how to handle their return values using Representational State Transfer (REST) or Simple Object Access Protocol (SOAP) web-service communication protocols.

Document Information

Target Audience

This instruction manual was created for ARDENSUITE Server users and developers interested in developing, deploying, and using MLMs as well as using already deployed MLMs on an ARDENSUITE Server.

Impressum

Media owners, editors, publishers:
Medexter Healthcare GmbH, Borschkegasse 7/5, A-1090 Vienna, Austria
Telephone: +43-1-968 03 24, Fax: +43-1-968 09 22, Internet: www.medexter.com
Email: office@medexter.com
CEO: Klaus-Peter Adlassnig, PhD, MSc

Editorial, project management, coordination:
Klaus-Peter Adlassnig, PhD, MSc

Vienna, November 2018

Version: 1.3

Download at www.medexter.com

Introduction

In this *how-to* instruction manual, we will guide you step-by-step in communicating with Arden Syntax MLMs deployed on an ARDENSUITE Server via web service protocols (REST and SOAP). Throughout this *how-to*, we provide MLM use case examples containing notification rules for the systemic inflammatory response syndrome (SIRS). Clinically, the rules for SIRS notifications are as follows:

SIRS Notification

ALERT if ≥ 2 Criteria

Temperature > 38°C (100.4°F) or < 36°C (96.8°F)
and/or

Heart rate > 90 beats per minute
and/or

Respiratory rate > 20 breaths per minute or arterial carbon dioxide tension (PaCO2) < 32 mm Hg
and/or

White blood cell count (>12,000/µL or < 4,000/µL or >10% immature [band] forms)

Following these provisions, MLMs were constructed that implement these notification rules and generate alerts when patient data match these criteria.

In this *how-to*, we discuss three use case scenarios to show how to call these MLMs using web-service communication protocols and how to handle the MLMs' feedback:

- Calling an MLM without supplying any input parameters. All data are obtained from a database.
- Calling an MLM with one input parameter. This parameter is used to select which data set is read from a database.
- Calling an MLM and supplying all the data through parameters, thereby removing the need to access a database.

For all three scenarios, the return value is either a string with an alert or `null` in case no alert was generated. For each scenario, we discuss how MLMs can be called using REST or SOAP as well as what these calls' return values look like.

Requirements

For optimal use of this instruction manual, please install the following software on your computer:

- The ARDENSUITE IDE and ARDENSUITE Server with Database Connector
- A relational database management system (DBMS) (e.g., MySQL)
- REST & SOAP clients

The ARDENSUITE IDE and ARDENSUITE Server with Database Connector

In case you do not have access to the ARDENSUITE IDE or the ARDENSUITE Server with Database Connector yet, please contact us at support@medexter.com. A 30-day trial version of the ARDENSUITE IDE can also be downloaded here.

Relational Database Management System

The methods discussed in this *how to* document are suited for any relational database that supports JDBC. For the instruction in this document, we used MySQL.

REST & SOAP Clients

For instructional and testing purposes, SOAP and REST calls to MLMs and their return values are illustrated using a web browser. In this document, we recommend using Postman for all REST communication and the desktop application SOAP UI for all SOAP communication.

Files

This *how-to* is accompanied by six files: an SQL script that can be used to prepare the database tables used in the use case scenarios, two text files used to illustrate remote web service calls (`rsc_REST.txt` and `rsc_SOAP.txt`) and three MLM files (extension `.mlm`). These MLMs illustrate in three different implementation variants for SIRS notifications how web service connectivity with MLMs can be achieved using the ARDENSUITE Server.

- `RS_SIRS-Notification1`: This MLM evaluates the four SIRS notification criteria and returns an alert if two or more criteria are met. The MLM uses a database table and queries the data using a predefined patient ID.

- `RS_SIRS-Notification2`: This MLM is a modification of the SIRS Notification #1 MLM. Instead of using a predefined patient ID for the SQL query, the MLM receives an ID as parameter/argument included in the REST/SOAP call.
- `RS_SIRS-Notification3`: This MLM uses no database. All necessary values are included in the REST/SOAP call.

NOTE: *The MLM files can be opened using any standard text editor or viewer, but in order to compile and upload the MLMs, the ARDENSUITE IDE and ARDENSUITE Server are required.*

Preliminaries

Before you can start with the actual *how-to* part of this manual, the database has to be created and connected to the ARDENSUITE Server and MLMs have to be compiled and deployed on the ARDENSUITE Server before calls can be made to them.

Database Setup

The first step is creating a database (e.g., a MySQL database using XAMPP and phpMyAdmin). For the purpose of this *how-to*, a database called `SIRSDB` should be created. With this *how-to*, you also received an SQL script (`sirsdb.sql`) that you can load and execute in order to automatically create and fill database tables.

After creation of the database, a connection between the ARDENSUITE Server and the DBMS has to be established and configured in order for MLMs on the ARDENSUITE Server to retrieve data from the created database. To do this, we refer to the corresponding *how-to* document (*Achieving Database Connectivity in Arden Syntax Using the ARDENSUITE Database Connector*), which can be found here.

NOTE: *Naturally, you can choose any name for the database and tables. Should you choose a different database name, keep in mind that the ARDENSUITE Server Database Connection needs to be adapted in order for the examples to function properly. Similarly, if you choose a different table name, make sure to adapt the supplied MLMs accordingly.*

Compiling and Uploading MLMs

Before an MLM can be called, it has to be compiled and uploaded onto the ARDENSUITE Server. To do this, we refer to the corresponding *how-to* document (*How to Compile, Test, and Deploy MLMs with the ARDENSUITE*), which can be found here. In order to use the *how-to* at hand, please compile and upload all three MLMs that accompany this document.

REST Communication

To set up an MLM call using the REST protocol, please open Postman. As a first step, a URL has to be constructed. Generally, if the ARDENSUITE Server is installed at a location `$location$` with a port `$port$`, the URL for the ARDENSUITE Server REST interface is:

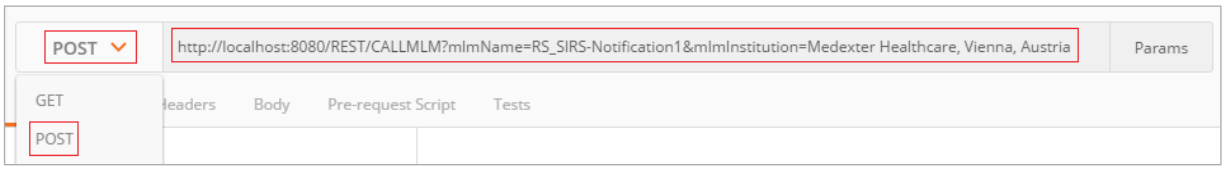
```
http://$location$:$port$/REST/CALLMLM?
```

In this *how-to*, we assume that the ARDENSUITE Server is installed locally with port `8080`, thus the URL for the ARDENSUITE Server REST interface would be

```
http://localhost:8080/REST/CALLMLM?
```

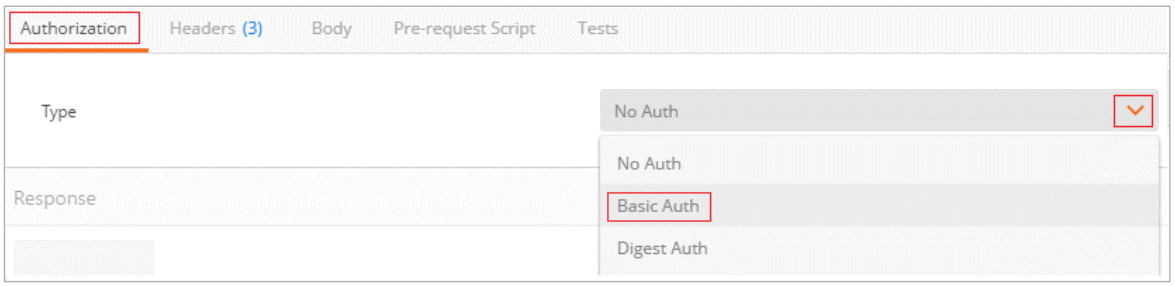
Additionally, it is necessary that each URL contains the name of the MLM and the MLM institution in order to identify the MLM to call. We have to combine these data in a `HTTP POST` command, as shown below.

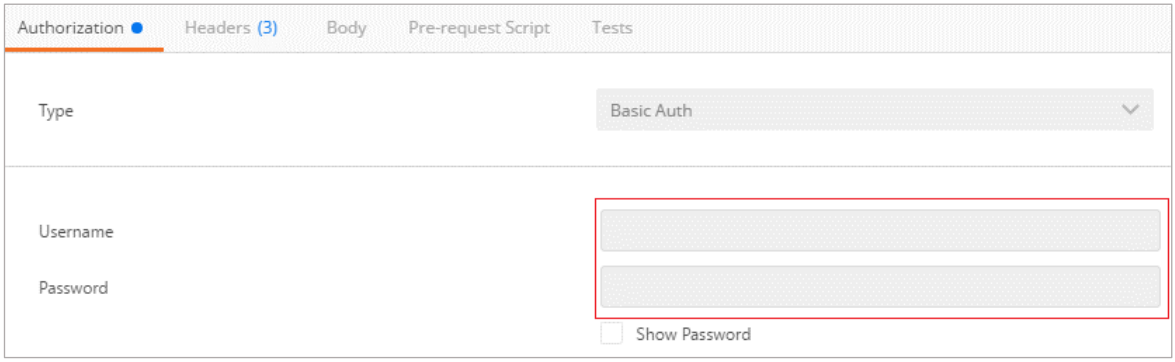
NOTE: *A URL will be supplied for each use case in their respective sections*.



After the URL has been constructed, the proper authorization credentials have to be entered. Click on the `Authorization` tab, select `Basic Auth` as type, and enter your ARDENSUITE Server user credentials (see figures below).

NOTE: *Be aware that your ARDENSUITE Server user needs the permission to use REST, which can be granted in the User Management section by an admin user.*

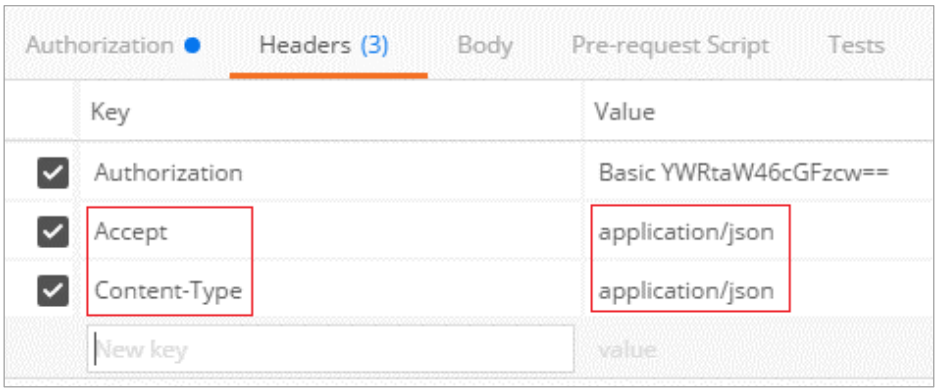




Finally, to save the modifications, click the `Update Request` button at the top right of the screen.

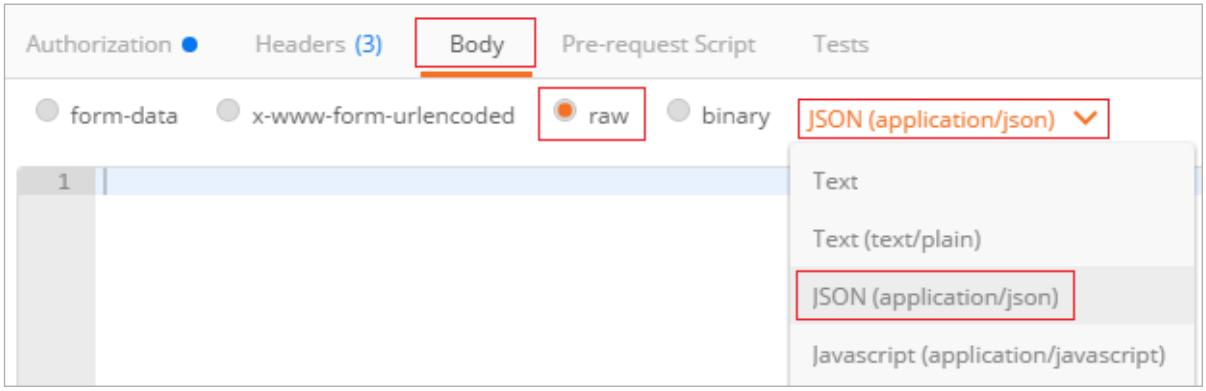
NOTE: *Make sure that your ARDENSUITE Server has a valid license; otherwise, the MLM call will not produce any results.*

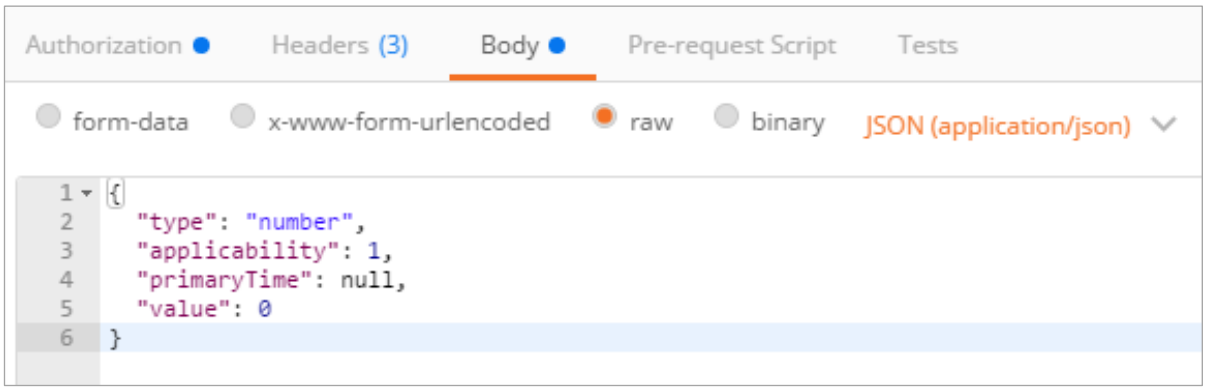
Next, you have to construct the message header. Click on the `Headers` tab. Here, you can specify the format type that should be returned by the server as well as the content type (the data format that will be sent to the ARDENSUITE Server). For the format type, enter `Accept` as key, and `application/json` as value. Similarly, for the content type, enter `Content-Type` as key, and again `application/json` as value. As an alternative, `application/xml` may be defined for the `Accept` header if XML is preferred as response data.



As a next step we will create the message body where the MLM parameters are specified. In order to do that, click on the `Body` tab and select `raw` and then `JSON (application/json)`. Parameters sent to an MLM can then be entered as JSON data in the text area below (see figures below).

NOTE: *The body content will be supplied for each use case in their respective sections*.





Finally, to send the request to the ARDENSUITE Server, press the blue `Send` button at the top right of the screen. The results will then be displayed in the `Result` area at the bottom of the screen.

SIRS Notification #1

In this MLM (`RS_SIRS-Notification1.mlm`), an alert is generated and returned if a patient record with an identifier predefined within the MLM matches the defined criteria for SIRS. In the data section of the MLM, using patient identifier `123`, the patient data is loaded into the MLM using an SQL statement on table `SIRSvalues`. After data collection, the MLM moves on to the logic part. In this part, the SIRS rules are implemented in conditional `if-then-else` statements.

For this MLM, the URL for the ARDENSUITE Server REST interface is:

`http://localhost:8080/REST/CALLMLM?mlmName=RS_SIRS-Notification1&mlmInstitution=Medexter Healthcare, Vienna, Austria`

As this MLM does not use any input parameters, the necessary JSON data to be supplied in the `Body` segment of the REST call is:

```
{
  "type": "number",
  "value": 0,
  "applicability": 1,
  "primaryTime": null
}
```

If everything worked out as it should and no errors occurred, the server returns an alert for SIRS. The alert looks like this:

```
{
  "type": "string",
  "primaryTime": null,
  "applicability": 1,
  "value": "Alert for SIRS"
}
```

SIRS Notification #2

In this MLM (`RS_SIRS-Notification2.mlm`), an alert is generated and returned if a patient record with an identifier specified in the REST body matches the defined criteria for SIRS.

For this MLM, the URL for the ARDENSUITE Server REST interface is:

```
http://localhost:8080/REST/CALLMLM?mlmName=RS_SIRS-
Notification2&mlmInstitution=Medexter Healthcare, Vienna, Austria
```

This MLM needs a single input parameter, the patent identifier, which is of the type `number`. If we select patient identifier `123`, the necessary JSON data to be supplied in the `Body` segment of the REST call is:

```
{
  "type": "number",
  "value": 123,
  "applicability": 1,
  "primaryTime": null
}
```

Again, if everything worked out as it should and no errors occurred, the server returns an alert for SIRS. The alert looks like this:

```
{
  "type": "string",
  "primaryTime": null,
  "applicability": 1,
  "value": "Alert for SIRS"
}
```

Note that if we were to select patient identifier `125`, which is a patient that has no SIRS, the server would return null, and the output would look like this:

```
{
  "type": "null",
  "primaryTime": null,
  "applicability": 1
}
```

SIRS Notification #3

In this MLM (`RS_SIRS-Notification3.mlm`), an alert is generated and returned if a patient record with an identifier specified in the REST body matches the defined criteria for SIRS. However, instead of connecting to the `SIRSDB` database, data for the MLM are provided using input parameters. For this MLM, data comprise values for a single patient at different points in time. The MLM selects the latest value set and proceeds to analyze it in the same fashion as the previous two MLMs did.

For this MLM, the URL for the ARDENSUITE Server REST interface is:

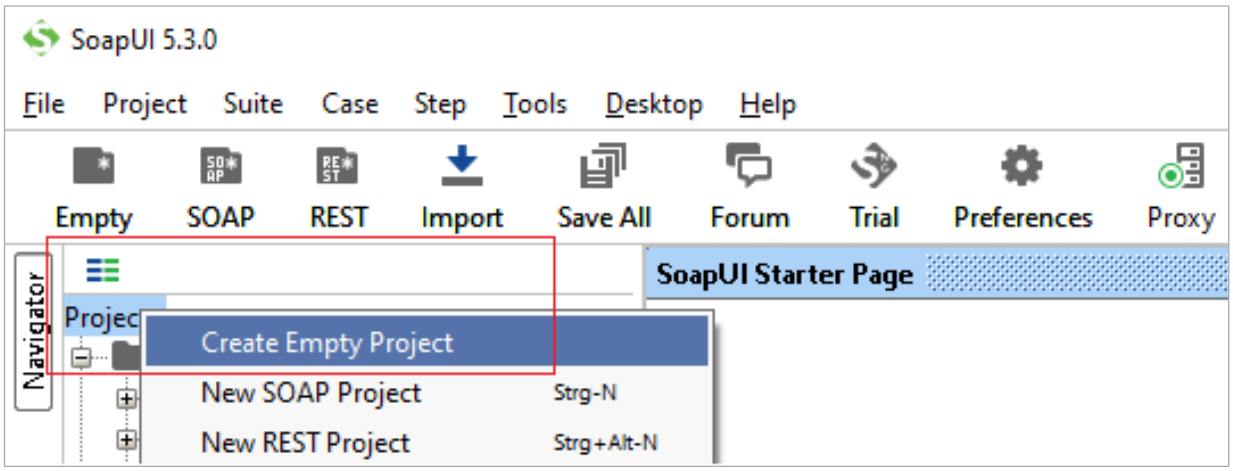
```
http://localhost:8080/REST/CALLMLM?mlmName=RS_SIRS-
Notification3&mlmInstitution=Medexter Healthcare, Vienna, Austria
```

As all data are supplied as input parameters, the REST body becomes quite large. As such, the complete JSON code is provided in the `rsc_REST.txt` file. Once more, the resulting server return message looks like this:

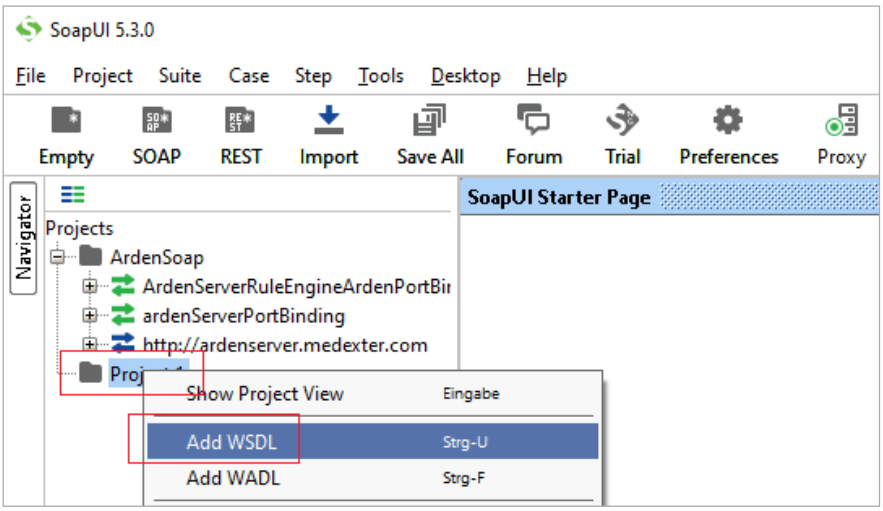
```
{
  "type": "string",
  "primaryTime": null,
  "applicability": 1,
  "value": "Alert for SIRS"
}
```

SOAP Communication

To set up an MLM call using SOAP, first open the SOAP UI desktop application and create an empty project by right-clicking on `Projects` and selecting `Create Empty Project` (see figure below).



Next, a Web Services Description Language (WSDL) resource location has to be added. Right-click on your newly created project and press `Add WSDL`.



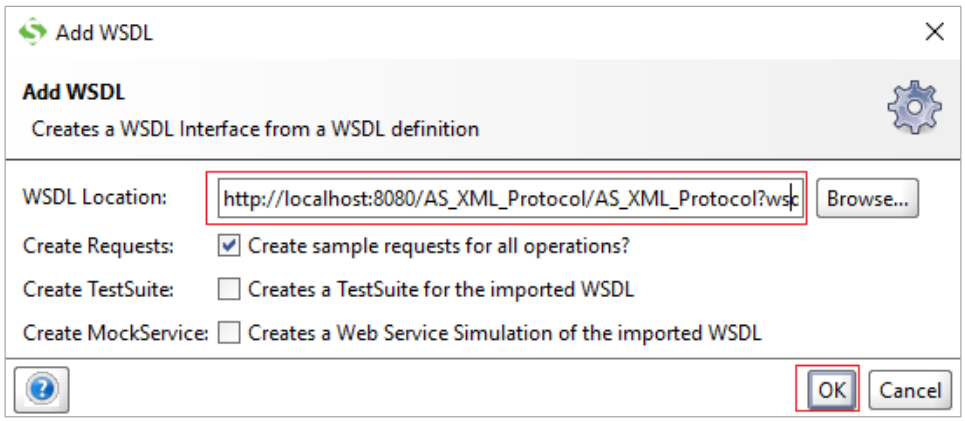
In general, if the ARDENSUITE Server is installed at a location `$location$` with a port `$port$`, the URL for the ARDENSUITE Server WSDL resource is:

`http://$location$:$port$/AS_XML_Protocol/AS_XML_Protocol?wsdl`

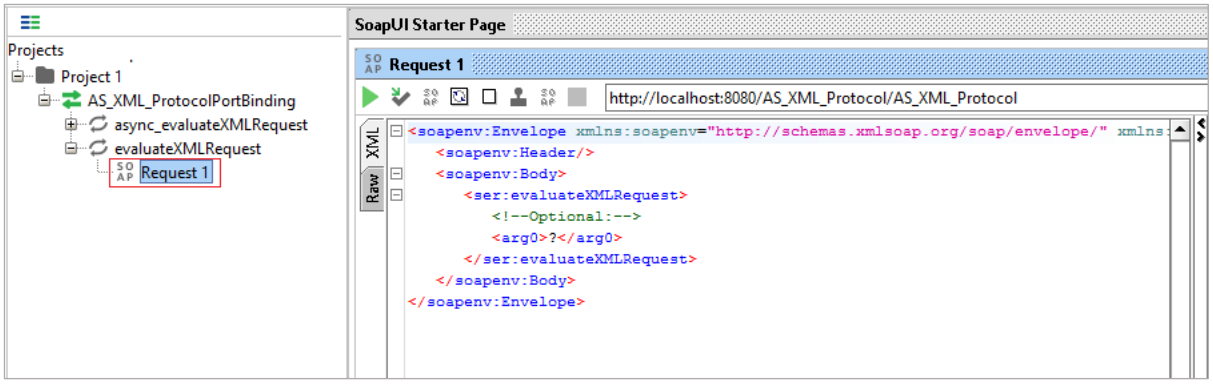
In this tutorial, we assume that the ARDENSUITE Server is installed locally with port `8080`, thus the URL to the ARDENSUITE Server WSDL would be

`http://localhost:8080/AS_XML_Protocol/AS_XML_Protocol?wsdl`

Copy this URL to the `WSDL Location` text field and press the `OK` button (see figure below).

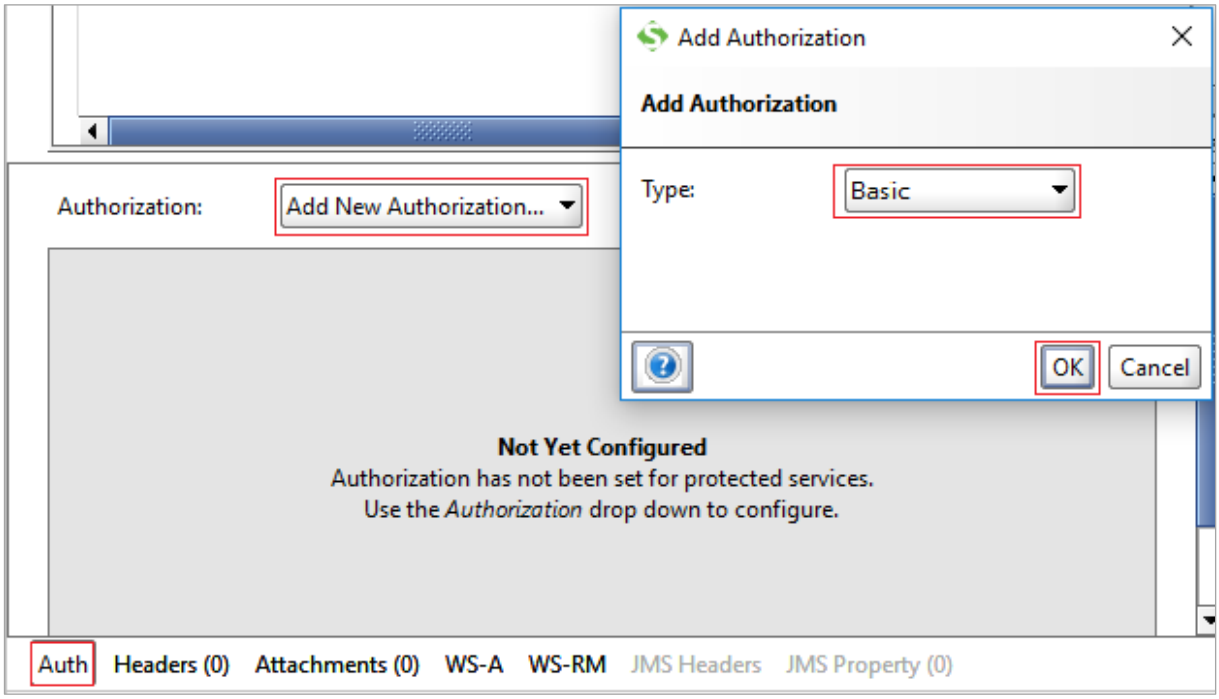


After adding the WSDL location, it is processed by the ARDENSUITE Server and two requests are made available within SOAP UI: `async_evaluateXMLRequest` and `evaluateXMLRequest`. Next, expand the `evaluateXMLRequest` element and double-click on `Request 1`. A new `Request 1` window opens on the right side (see figure below).

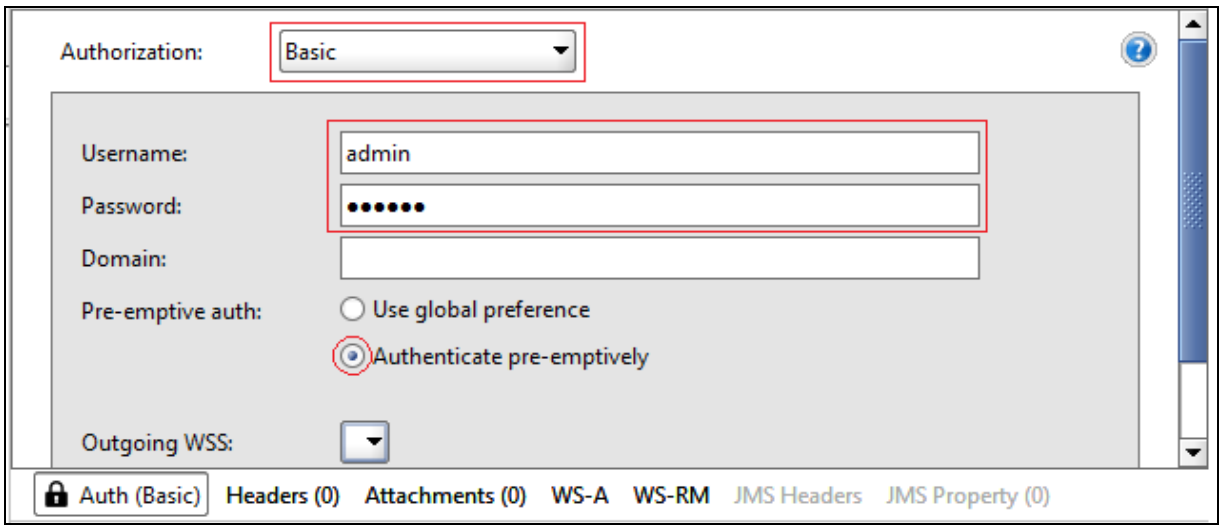


NOTE: *As only an ARDENSUITE Server user with permission for SOAP web services may call MLMs via SOAP, please make sure your user account has the respective permission (Permission for SOAP can be granted in the ARDENSUITE Server's* User Management *section by an admin user). Also, ensure that your ARDENSUITE Server has a valid license, otherwise the MLM call will not produce any results.*

The next step is to add an authorization header to the request. To construct an authorization header, click on `Auth` at the bottom left of the `Request 1` window. Next, open the `Authorization` dropdown menu, select `Add New Authorization`, and in the following dialog window, select the option `Basic`, and press `OK` (see figure below).



After a basic authentication has been configured, you have to enter your ARDENSUITE Server username and password in their respective input fields (see figure below). Also, please select the option `Authenticate pre-emptively` in order for SOAP UI to send the Authentication Header with each request.



The final step is adding MLM parameters to the request. In order to uniquely identify the MLM-to-be-

called, the MLM name and institution have to be supplied as parameters in XML (see code below):

```
<usecase>
  <callMlm>
    <key>
      <mlmName>RS_SIRS-Notification1</mlmName>
      <institution>Medexter Healthcare, Vienna, Austria</institution>
    </key>
    <arguments>
    </arguments>
  </callMlm>
</usecase>
```

However, as the data has to be interpreted as text, all '<' and '>' characters have to be transformed to HTML character entities:

```
&lt;usecase&gt;
  &lt;callMlm&gt;
    &lt;key&gt;
      &lt;mlmName&gt;RS_SIRS-Notification1&lt;/mlmName&gt;
      &lt;institution&gt;Medexter Healthcare, Vienna,
Austria&lt;/institution&gt;
    &lt;/key&gt;
    &lt;arguments&gt;
    &lt;/arguments&gt;
  &lt;/callMlm&gt;
&lt;/usecase&gt;
```

Copy this definition between the two `<arg0></arg0>` tags and hit the `Send` button (green arrow on the top left side) to receive a result from the MLM call. The server response will be displayed in the second window on the right, between the `<results>` tags:

NOTE: *XML code will be supplied for each use case in their respective sections*.

SIRS Notification #1

In this MLM (`RS_SIRS-Notification1.mlm`), an alert is generated and returned if a patient record with an identifier predefined within the MLM matches the defined criteria for SIRS. In the data section of the MLM, using patient identifier `123`, the patient data is loaded into the MLM using an SQL

statement on table SIRSvalues. After data collection, the MLM moves on to the logic part. In this part, the SIRS rules are implemented in conditional if-then-else statements.

As this MLM does not use any input parameters, the XML code to be inserted between the <arg0></arg0> tags comprises only the name of the MLM and the institution:

```
&lt;usecase&gt;
  &lt;callMlm&gt;
    &lt;key&gt;
      &lt;mlmName&gt;RS_SIRS-Notification1&lt;/mlmName&gt;
      &lt;institution&gt;Medexter Healthcare, Vienna,
Austria&lt;/institution&gt;
    &lt;/key&gt;
    &lt;arguments&gt;
    &lt;/arguments&gt;
  &lt;/callMlm&gt;
&lt;/usecase&gt;
```

If everything worked out as it should and no errors occurred, the server returns an alert for SIRS. The alert looks like this:

```
<results>
  <string applicability="1.0">Alert for SIRS</string>
</results>
```

SIRS Notification #2

In this MLM (RS_SIRS-Notification2.mlm), an alert is generated and returned if a patient record with an identifier specified in the REST body matches the defined criteria for SIRS.

This MLM needs a single input parameter, the patent identifier, which is of the type Number. If we select patient identifier 123, the XML code to be inserted between the <arg0></arg0> tags comprises this number, as well as the name of the MLM and the institution:

```
&lt;usecase&gt;
  &lt;callMlm&gt;
    &lt;key&gt;
      &lt;mlmName&gt;RS_SIRS-Notification2&lt;/mlmName&gt;
      &lt;institution&gt;Medexter Healthcare, Vienna,
Austria&lt;/institution&gt;
    &lt;/key&gt;
    &lt;arguments&gt;
      &lt;number&gt;123&lt;/number&gt;
    &lt;/arguments&gt;
  &lt;/callMlm&gt;
&lt;/usecase&gt;
```

Again, if everything worked out as it should and no errors occurred, the server returns an alert for SIRS. The alert looks like this:

```
<results>
  <string applicability="1.0">Alert for SIRS</string>
</results>
```

Note that if we were to select patient identifier `125`, which is a patient that has no SIRS, the server would return null, and the output would look like this:

```
<results>
  <null applicability="1.0"/>
</results>
```

SIRS Notification #3

In this MLM (`RS_SIRS-Notification3.mlm`), an alert is generated and returned if a patient record with an identifier specified in the REST body matches the defined criteria for SIRS. However, instead of connecting to the `SIRSDB` database, data for the MLM are provided using input parameters. For this MLM, data comprise values for a single patient at different points in time. The MLM selects the latest value set and proceeds to analyze it in the same fashion as the previous two MLMs did.

As all data are supplied as input parameters, the resulting XML code becomes quite large. As such, the complete XML code to be inserted between the `<arg0></arg0>` tags is provided in the

`rsc_SOAP.txt` file. Once more, the resulting server return message looks like this:

```
<results>
  <string applicability="1.0">Alert for SIRS</string>
</results>
```