Table of Contents

Adding/Editing Services

The following section describes how to work with the editor of services when adding a new service or editing the existing one. Services are described in the Communication with Other Systems section.

The editor of services can be started as described in the Working with Configuration Elements section.

You can define the following settings of the service in the working area of the editor:

Process

Select a process that will implement the service from a list of all processes defined in the business space version. When the service is requested the specified process will be executed. If the specified process requires input, this input must be provided by a service requestor.

Third parties will be able to call SOAP-based web services

Choose this option if you want AwareIM to expose the service as a “web service” implementing the SOAP protocol. When the business space containing such service is published AwareIM will automatically create a WSDL file for the service. See below for the names of the endpoints that the WSDL file and service calls will have. These URL's need to be shared with the callers of the services.

> **warning**
>
> If at least one SOAP service is exposed by your application, make sure to set SOAP Settings in the properties of a business space version

Third parties will be able to call REST-ful web services

Tick this checkbox if the service should be exposed as a REST-ful service

> **warning**
>
> Both REST-ful and SOAP services only work in production mode and are available as soon as the business space version is published

Service Will be Protected by OAuth2.0 Protocol

Tick this checkbox if you want to protect a service. If at least one service should be protected, you will

need to deploy a special web application. This web application will manage scopes, clients, will issue access tokens and refresh tokens. To deploy this web application:

1. Go to the Edit menu of the Configuration Tool
2. Select the entry with the name “SERVICE PROTECTION”

Note that thew deployment of the dynamic feature should only be done once

Your service documentation needs to convey to the callers of the service that before calling the service, they need to do the following:

1. Obtain an access token by cand
2. include the access token in the Authorization Bearer HTTP header, for example (

   ```
   Authorization: Bearer <access token>
   ```

).

3. The name of the business space must be in the HTTP header called AW-DOMAIN, for example

   ```
   AW-DOMAIN:CRM
   ```

4. The name of the service to call must be in the header called AW-SERVICE for a REST-ful service (for example,

   ```
   AW-SERVICE: AddCustomer
   ```

) and AW-SERVICE-SOAP for a SOAP service - for example

   ```
   AW-SERVICE-SOAP: AddCustomer
   ```

   .

Service Reply

Service reply defines a business object or an array of business objects sent by a service provider as a reply to the service request. If you select the “Service will return object” radio button you will need to select the object that will be returned from the list of all objects. Your implementing process must populate the context with the instance(s) of the business object being returned. Tick the “Return multiple instances” checkbox to return an array of instances. Again your implementing process must populate the context with instances of the object to be returned.

Exposed Objects and Attributes

An object returned by the service, as well as the object specified as input of the implementing process, may have attributes that the callers of the service do not need(want) to see. You can specify this by ticking only those objects and attributes that are relevant to the caller. All other attributes and objects will not be encoded in the service request and service reply.

Service Endpoints

Service endpoint is a URL that the caller should call for a particular service. These endpoints depend on whether this is a REST-ful or SOAP service and whether the service is protected. The following URL's should be used:

For unprotected services:

1. for any REST-ful unprotected service:

```
http(s)://YourServerName:yourPort/WebApplication/REST/BusinessSpace/ServiceName (where WebApplication is the name of your web application or a default value (AwareIM).
```

2. for the SOAP WSDL file:

```
http(s)://YourServer:YourPort/WebApplication/SOAP/BusinessSoace?WSDL
```

3. for SOAP service call:

```
http(s)://YourServer:YourPort/WebApplication/SOAP/BusinessSpace/ServiceName
```

For protected services:

1. for any Rest-ful service:

```
http(s)://YourServer:YourPort/CXF/services/serviceCall/api
```

2. for the SOAP WSDL file:

```
http(s)://YourServer:YourPort/WebApplication/SOAPP/BusinessSpace?WSDL
```

3. for SOAP Service call:

```
http(s)://YourServer:YourPort/CXF/services/serviceCall/api
```

> **warning**
> Note that for protected services the name of the business space and the name of the service to call are specified in the special HTTP header, as explained above

Reply Format

If a REST-ful service returns a reply, it can be either in the XML format (also Default) or in JSON format. The format can also vary depending on the URL used – the extension of the expected format can be specified in the URL itself. Choose the appropriate option here. A reply can also be just a string

calculated by the process implementing the service. If that is the case, choose the “Calculated String” radio button.

For SOAP services reply format is always XML.

Handing of Undefined Values

If values of the attributes of the returned object are undefined, you can choose how ***AwareIM*** will encode such values. By default ***AwareIM*** will skip undefined values (“Do not encode” option), but you can also encode an empty value or a particular string. You can also specify different encoding options depending on the name of the attribute.

Editing service properties in the Property Window

The following properties can be specified in the Element Properties window:

Name

Specify the name of the service uniquely identifying it within the business space version. The following restrictions apply:

1. The name of the service must be unique among the names of other services defined in the business space version.
2. The name must start with a character or underscore symbol; all other symbols must be characters (including underscore symbol) or digits. Space symbols are not allowed.

Description

Specify any text that describes what the services does etc. Providing a description is not mandatory but is highly recommended. Any description if defined is included into the generated documentation for the business space version – see Generating Documentation.

Service Log Settings

If you click on this property, the Service Log dialog will be displayed. If you tick “Save Communication In” checkbox the system will record the log of the communication. It will capture all communication in a log. It will record the name of the service called, time of the communication, URL parameters, HTTP method, HTTP body etc. It will also record HTTP headers of the request and response. The log will be captured in the instance of the object that you specify (you also need to define the attributes in this object that will store the log, as well as the attribute storing the name of the service and timestamp (the latter two attributes are required, so that you can sort by time and service.

Handling of HTTP headers

Values of HTTP headers of each service call can be stored in the object that is specified as the input of the process implementing the service. Enter the mapping between the name of the header and the attribute where the header value will be stored, in the dialog displayed when you edit this property.

Name mapping

If your REST-ful service is implemented by a process that takes some object as input (should only be one object) then values of attributes of this object have to be provided as parameters in the URL (see example above). The name of the parameters must match the names of the attributes of the object. If this, however, is impossible for whatever reason, then you need to provide a mapping between the names of the parameters (“External Name”) and the name of the corresponding attribute. You should only do it for those parameters that have different names to the names of the attributes.