FUJITSU

FUJITSU Software Enterprise Service Catalog Manager V17.4

Release Notes

September 2017

Contents

About this Manual 4

1 What's New? 6

1.1 OpenStack Integration 6

1.2 Obtaining Subscription Usage Data 6

1.3 APP Configuration in Web Interface 7

1.4 Configuration of Ports for the ESCM Domain 8

1.5 Configuration of Signing Algorithm for SAML Transactions 8

1.6 Deprecated Compatibility Layer for ESCM V15.4 8

1.7 Extension of Parameter Configuration Tool Sample 9

1.8 Specification of a Marketplace ID 9

1.9 Documentation 9

2 Compatibility 10

2.1 SOAP-based Web Services and APIs 10

2.1.1 Migrating Existing Applications for Multi-Tenancy Support 11

2.1.2 Migrating Existing Applications Prior to V16.0 12

2.2 Update Installation 15

2.3 Asynchronous Provisioning Platform 15

3 Restrictions 16

About this Manual

This manual describes new features and changes to existing features of FUJITSU Software Enterprise Service Catalog Manager (ESCM) V17.4 as compared to V17.0. In addition, this manual provides compatibility information and information on known restrictions.

ESCM can be operated on the platforms specified in the *Installation Guide*.

This manual is structured as follows:

Chapter	Description
What's New? on page 6	Describes new features, changes, and enhancements.
Compatibility on page 10	Describes compatibility issues when upgrading an installation from a previous release of ESCM.
Restrictions on page 16	Describes the known restrictions of this release.

Readers of this Manual

This manual is intended for operators who are responsible for installing ESCM.

It assumes that you are familiar with the following:

- Administration of the operating systems in use, including the adaption and execution of batch files or shell scripts.
- Java EE technology, particularly as to the deployment on application servers.
- Relational databases and their administration, in particular the PostgreSQL database.
- ESCM concepts as explained in the *Overview* manual.
- Installation and administration of Web servers.
- Installation and administration of the GlassFish application server.

Notational Conventions

This manual uses the following notational conventions:

Add	Names of graphical user interface elements.
`init`	System names, for example command names and text that is entered from the keyboard.
`<variable>`	Variables for which values must be entered.
`[option]`	Optional items, for example optional command parameters.
`one | two`	Alternative entries.
`{one | two}`	Mandatory entries with alternatives.

Available Documentation

The following documentation on ESCM is available:

- *Overview:* A PDF manual introducing ESCM. It is written for everybody interested in ESCM and does not require any special knowledge.
- *Operator's Guide:* A PDF manual for operators describing how to administrate and maintain ESCM.
- *Technology Provider's Guide:* A PDF manual for technology providers describing how to prepare applications for usage in a SaaS model and how to integrate them with ESCM.
- *Supplier's Guide:* A PDF manual for suppliers describing how to define and manage service offerings for applications that have been integrated with ESCM.
- *Reseller's Guide:* A PDF manual for resellers describing how to prepare, offer, and sell services defined by suppliers.
- *Broker's Guide:* A PDF manual for brokers describing how to support suppliers in establishing relationships to customers by offering their services on a marketplace.
- *Marketplace Owner's Guide:* A PDF manual for marketplace owners describing how to administrate and customize marketplaces in ESCM.
- *OpenStack Integration:* A PDF manual for operators describing how to offer and use virtual systems controlled by OpenStack through services in ESCM.
- *Amazon Web Services Integration:* A PDF manual for operators describing how to offer and use virtual servers controlled by the Amazon Elastic Compute Cloud Web service through services in ESCM.
- *Online Help:* Online help pages describing how to work with the administration portal of ESCM. The online help is intended for and available to everybody working with the administration portal.

1 What's New?

This chapter describes changes and enhancements made in ESCM since V17.0.

1.1 OpenStack Integration

The OpenStack service controller now supports HEAT templates allowing to deploy multiple VMs.

With previous versions of ESCM, there were two options to provide templates for the provisioning of resources such as virtual machines in an OpenStack environment:

1. A technology provider organization responsible for the OpenStack service controller provides them on an external host in a location whose URL can be reached from ESCM via HTTP or HTTPS.
2. The ESCM operator provides them on the ESCM host in a location whose URL can be reached from ESCM via HTTP or HTTPS.

The technology manager now has the option to import templates that can be referenced in a technical service definition into the `bssapp` database.

The graphical user interface for configuring the OpenStack service controller has been extended so that templates can be imported, exported, and removed in the `bssapp` database. If a template with the same name is imported twice, the original (first) one is replaced by the second one.

The OpenStack integration software looks for the template with the name specified in the technical service definition as follows:

1. The template is searched for in the `bssapp` database.
2. If it is not found, the template is searched for in the location specified in the `TEMPLATE_BASE_URL` controller configuration setting.

1.2 Obtaining Subscription Usage Data

A platform operator can now create a report listing all currently active subscriptions of all customers, including the number of virtual machines (VMs) booked with subscriptions to IaaS services, for example, in OpenStack. The report also shows the number of users assigned to the subscriptions. This data can then be passed to suppliers so that they can charge customers for their usage of VMs.

Adapting Technical Service Definitions

In order to be able to retrieve the information on the number of provisioned VMs, the technical service definitions for the provisioning of VMs need to define a parameter with the `VMS_NUMBER` identifier. Note that technical services that are the basis of marketable services for which there are existing subscriptions cannot be changed.

To adapt the technical service definitions:

1. Export the definitions in the ESCM administration portal.
2. Add the following parameter definition, for example:

```
<ParameterDefinition configurable="false"
       default="0" id="VMS_NUMBER"
       mandatory="true" valueType="LONG">
       <LocalizedDescription locale="en">
           Number of VMs of IaaS subscriptions</LocalizedDescription>
```

```
</ParameterDefinition>
```

3. Import the service definitions again.

The VMS_NUMBER parameter will be added to existing subscriptions to marketable services based on the technical service definitions as soon as the new APP_TIMER_REFRESH_SUBSCRIPTIONS timer has run for the first time.

New Timer in Configuration Settings of APP

A new configuration setting has been introduced in the configsettings.properties file of APP: APP_TIMER_REFRESH_SUBSCRIPTIONS. It defines the interval (in milliseconds) at which APP polls the status of instances and updates the number of virtual machines (VMs) provisioned for subscriptions to IaaS services. The default is 86400000 milliseconds (once a day).

Retrieving the Number of VMs and Users

The platform operator can retrieve subscription usage data, including the number of VMs and the number of users assigned to the subscriptions to IaaS services using the new getsubscriptionusagedata command of the command line tool.

The syntax of the command is as follows:

```
<JAVA_HOME>/bin/java -jar lib/oscm-operatorsvc-client.jar
 <userkey> <password>
 getsubscriptionusage filename=<filename>
```

where

JAVA_HOME is the installation directory of your JDK

userkey is the numeric key of the operator (default: 1000)

password is the password of the operator (default: admin123)

getsubscriptionusage is the command to be executed

filename is the name of the file to which the report is to be written, for example, output.txt.

The command writes the following information in the given sequence to the specified output file:

- Customer organization ID
- Customer organization name
- Subscription name
- Marketable service name
- Technical service name
- Supplier organization name
- Supplier organization ID
- Number of users
- Number of VMs

1.3 APP Configuration in Web Interface

The APP administrator can now change some configuration settings for APP using the APP Web interface where service controllers are registered (for example, http://127.0.0.1:8880/oscm-app). The new settings overwrite the ones stored in the bssapp database as imported via the configsettings.properties file when APP was deployed.

The following settings can be edited:

- APP_BASE_URL
- APP_ADMIN_MAIL_ADDRESS
- BSS_USER_KEY
- BSS_USER_ID
- BSS_USER_PWD

1.4 Configuration of Ports for the ESCM Domain

The glassfish.domain.WS_PORT configuration setting has been renamed to glassfish.domain.WS_PORT_SECURE. This setting defines the port used for a secure HTTP listener for Web service connections of the application server. The listener now uses SSL with the default application server certificate (s1as).

This has been introduced because in some operational environments it is required to use separate certificates for the SOAP communication between ESCM and APP and for Web browser access, i.e. separate certificates for internal and external communication must be provided.

This setting is evaluated for new installations only. Existing installations are not affected.

Default: 8082

1.5 Configuration of Signing Algorithm for SAML Transactions

The platform operator can now configure which signing algorithm is used for SAML transactions.

The following optional configuration setting has been introduced and can be edited in the configsettings.properties file for ESCM:

SSO_SIGNING_ALGORITHM

The following values are supported:

- SHA1 (default)
- SHA256

1.6 Deprecated Compatibility Layer for ESCM V15.4

Up to version 17.0 of ESCM, the SOAP-based public Web services and APIs of ESCM were backward compatible to version 15.4. This compatibility layer has been deprecated.

The following Web services and APIs are affected:

- Platform services: v1.7
- Notification service API: v1.9
- Provisioning service API: v1.6
- Operation service API: v1.4
- PSP integration service: v1.7

1.7 Extension of Parameter Configuration Tool Sample

The parameter configuration tool sample included in the ESCM integration package has been extended. It is now possible for customers to view the price per subscription for a parameter when subscribing to a service.

The parameter tool API has also been extended by methods for retrieving information on parameter prices for a subscription and per user.

Note that the sample UI does not support prices per user, stepped prices, and one-time fees. The API also does not support stepped prices. In addition, currencies are not displayed/displayable.

1.8 Specification of a Marketplace ID

When a marketplace is created, it is now possible to optionally specify an ID. If nothing is specified, the system still generates the ID automatically.

This option is also available using the API.

1.9 Documentation

The manuals and online help pages have been revised, where applicable, to reflect the newly introduced and changed features.

2 Compatibility

This chapter describes compatibility issues when upgrading to ESCM V17.4.

2.1 SOAP-based Web Services and APIs

The SOAP-based public Web services and APIs of ESCM V17.4 come with a **compatibility layer** so that applications (clients) implemented with ESCM V16.0, or V16.1 can still be used without having to be rewritten. Versions older than V16.0 are not supported.

Web service clients must address the current version of the public Web services. The versioning pattern is as follows:

Current Web service versions	Shipped with ESCM version
Platform services: v1.9	V16.0; V16.1; V17.0 - V17.4
Notification service API: v1.9	V16.0; V16.1; V17.0 - V17.4
Provisioning service API: v1.8	V16.1 Fix 3-7; V17.0 - V17.4
Operation service API: v1.5	V16.0; V16.1; V17.0 - V17.4
PSP integration service: v1.8	V16.0; V16.1; V17.0 - V17.4

Supported Web service versions	Shipped with ESCM version
Provisioning service API: v1.7	V16.0; V16.1; V16.1 Fix 1-2

If ESCM is installed in INTERNAL authentication mode, Web services with the `BASIC` or `CLIENTCERT` suffix can be used. If ESCM is installed in SAML_SP mode, Web services with the `STS` suffix can be used.

A WSDL URL is used to address a specific Web service. The WSDL URL of a specific service in a current version can be found out as follows:

1. In the GlassFish administration console, go to **Common Tasks** -> **Applications** -> **oscm**.
2. On the **Descriptor** tab, open the `META_INF/sun-ejb-jar.xml` descriptor file of the `oscm-webservices.jar` subcomponent.

   For every platform service, the endpoint address URI shows the Web service name and whether it is to be addressed through basic authentication (`BASIC`), certificate-based authentication (`CLIENTCERT`), or a security token service (`STS`).

   The URL pointing to the WSDL definition of a platform service is constructed as follows:

   `<BASE_URL_HTTPS>/<endpoint-address-uri>?wsdl`

   where

   `<BASE_URL_HTTPS>` points to the local server and port where the platform services have been deployed.

   `<endpoint-address-uri>` is the address as defined in the `sun-ejb-jar.xml` descriptor file.

   `?wsdl` is the suffix to be used for identifying a WSDL file.

   Example: `https://myserver:8081/AccountService/BASIC?wsdl`

Note:	For the `STS` endpoint, it is possible to specify the ID of the tenant associated with the Identity Provider (IdP) system to be used for authentication, for example: `https://myServer:8181/oscm/v1.9/SessionService/STS?wsdl&tenantID=xxxxxxxx` If no tenant is specified, the default tenant as configured by the platform operator when installing ESCM is used.

ESCM uses an internal servlet for providing the content of the shipped WSDL files. This means that the WSDL files are delivered as static content. This static content enables runtime migration of Web service clients to a future release of ESCM.

If applications integrated with ESCM rely on pure HTTP calls, you need to change their setup to use HTTPS.

2.1.1 Migrating Existing Applications for Multi-Tenancy Support

With ESCM V16.1, multi-tenancy functionality has been introduced that requires adapting of existing Web service client applications that make use of STS (Security Token Service). The following basic steps are required:

1. Adapting the integration helper implementation.
2. Receiving correct SAML assertions from the Identity Provider (IdP) system.

Note:	Existing applications only need to be changed if you want to use the new functionality. They can still be run with this release.

Adapting the Integration Helper Implementation

If your Web service client application makes use of the ESCM integration helpers, it must allow for the specification of a tenant ID. A tenant defines the Identity Provider (IdP) system against which the Web service calls are authenticated. Proceed as follows:

1. Retrieve the `Integrationhelper.war` package from the ESCM integration package, and add the new `Integrationhelper.jar` and `oscm-webservices-proxy.jar` packages to the class path of your application.
2. In the `webserviceclient.properties` file, for the URL of the WSDL file of the platform's Session Service, add the `tenantID` parameter, for example:

   `https://myServer:8181/oscm/v1.9/SessionService/STS?wsdl&tenantID=xxxxxxxx`

   The platform operator is responsible for configuring tenants and associating an IdP (Identity Provider) system with them. When the system is installed, the operator configures the default tenant. If no tenant is specified when calling the ESCM session service, the settings for the default tenant are used. The tenant ID must refer to an existing tenant configured and set up by the platform operator.
3. In the `webserviceclient.properties` file, add the tenant ID associated with the technology provider organization to the user credentials for accessing the ESCM Web services (`cm.service.user.tenantID`).
4. Copy all newly delivered `.jar` files to the location where you implement your application.
5. Rebuild and redeploy your application.

Receiving Correct SAML Assertions from the IdP

The IdP must be configured such that its assertions contain two `<AttributeStatement>` elements. The first `<Attribute>` subelement must contain a `Name="userid"` property, and the

<AttributeValue> subelement must specify the user ID of the calling user in ESCM . The second <Attribute> subelement must contain a Name="tenantID" property, and the <AttributeValue> subelement must specify the ID of the tenant associated with the organization the user belongs to.

Example:

```
<saml:Assertion ...>
    ...
    <saml:AttributeStatement>
      <saml:Attribute Name="userid">
        <saml:AttributeValue>administrator</saml:AttributeValue>
      </saml:Attribute>
    <saml:Attribute Name="tenantID">
        <saml:AttributeValue>34ffd098</saml:AttributeValue>
      </saml:Attribute>
    </saml:AttributeStatement>
  </saml:Assertion>
```

Note: Be aware that the SAML assertions returned from the IdP always need at least contain the default tenant ID in case you do not want to make use of the multi-tenancy functionality.

2.1.2 Migrating Existing Applications Prior to V16.0

With ESCM V16.0, new functionality has been introduced that requires adapting of existing Web service client applications. In addition, the SOAP-based public Web services and APIs of ESCM V15.4 are deprecated.

- In Web service client applications of ESCM V15.4, the namespace needs to be changed.
- In Web service client applications of a ESCM release prior to V15.4, the namespace and version information needs to be adapted.

In both cases, the following basic steps are required:

1. Adapt your interface implementation of the provisioning, operation, notification, and/or PSP integration service.
2. Adapt the XML description file of the underlying technical service.
3. Adapt the source code of your application.

Note: Existing applications only need to be changed if you want to use the new functionality. They can still be run with this release.

Adapting the Provisioning Service Interface Implementation

The description below shows how to adapt the provisioning service for your application so that the multi-tenancy functionality is used.

1. From the oscm-integration-pack.zip package, copy the ProvisioningService.wsdl and its related schema file to the location where you implement your application (for example, to META-INF/wsdl). The ProvisioningService.wsdl and its schema file can be found in the oscm-integration-pack/SOAPapis/provisioning/schema folder.
2. If your provisioning service is implemented as an EJB (annotated bean) for an application that is to be deployed in a Java EE-compliant application server (GlassFish), add the WSDL file as follows in the webservices.xml descriptor file:

```
<webservices xmlns="http://java.sun.com/xml/ns/javaee"
```

```
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.2"
   xsi:schemaLocation=
      "http://java.sun.com/xml/ns/javaee
       http://www.ibm.com/webservices/xsd/
            javaee_web_services_1_2.xsd">
      <webservice-description>
         <display-name>ProvisioningService</display-name>
         <webservice-description-name>ProvisioningService
            </webservice-description-name>
         <wsdl-file>ProvisioningService.wsdl</wsdl-file>
         <port-component> ... </port-component>
         ...
      </webservice-description>
```

If your provisioning service is deployed as a standard `.war` archive (non-EJB implementation), add the provisioning service WSDL file to the `sun-jaxws.xml` descriptor file:

```
<endpoints version="2.0"
       xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime">
       <endpoint name="ProvisioningService"
 implementation="org.oscm.jaxws.ProvisioningServicImpl"
             url-pattern="/ProvisioningService"
             wsdl="ProvisioningService.wsdl" />
</endpoints>
```

3. Make sure that your application references the `oscm-extsvc-provisioning.jar` library provided with the integration package of this release (`oscm-integration-pack/SOAPapis/provisioning/lib`).
4. Rebuild your provisioning service.

Adapting the Interface Implementation

The description below shows how to adapt a provisioning service for your application. For implementations of the other interfaces, you can proceed analogously.

1. Change the target namespace of the ESCM Web services and remove the version value. For example:

   Replace `http://bss.fujitsu.com/xsd/v1.4` with `http://oscm.org/xsd`.

2. From the `oscm-integration-pack.zip` package, copy the `ProvisioningService.wsdl` and its related schema file to the location where you implement your application (for example, to `META-INF/wsdl`). The `ProvisioningService.wsdl` and its schema file can be found in the `oscm-integration-pack/SOAPapis/provisioning/schema` folder.

3. If your provisioning service is implemented as an EJB (annotated bean) for an application that is to be deployed in a Java EE-compliant application server (GlassFish), add the WSDL file as follows in the `webservices.xml` descriptor file:

```
<webservices xmlns="http://java.sun.com/xml/ns/javaee"
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="1.2"
   xsi:schemaLocation=
      "http://java.sun.com/xml/ns/javaee
       http://www.ibm.com/webservices/xsd/
            javaee_web_services_1_2.xsd">
      <webservice-description>
         <display-name>ProvisioningService</display-name>
         <webservice-description-name>ProvisioningService
```

```
            </webservice-description-name>
          <wsdl-file>ProvisioningService.wsdl</wsdl-file>
          <port-component> ... </port-component>
          ...
      </webservice-description>
```

If your provisioning service is deployed as a standard `.war` archive (non-EJB implementation), add the provisioning service WSDL file to the `sun-jaxws.xml` descriptor file:

```
<endpoints version="2.0"
       xmlns="http://java.sun.com/xml/ns/jax-ws/ri/runtime">
       <endpoint name="ProvisioningService"
 implementation="org.oscm.jaxws.ProvisioningServicImpl"
             url-pattern="/ProvisioningService"
             wsdl="ProvisioningService.wsdl" />
</endpoints>
```

4. Make sure that your application references the `oscm-extsvc-provisioning.jar` library provided with the integration package of this release (`oscm-integration-pack/SOAPapis/provisioning/lib`).
5. Rebuild your provisioning service.

Adapting the Technical Service Definition

In the XML description of the technical service underlying your application, adapt the namespace and the version value of the provisioning service and check the URL referencing your provisioning service.

For example:

```
<tns:TechnicalServices
   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
   xsi:schemaLocation="oscm.serviceprovisioning/1.9/
      TechnicalService.xsd
      ../../oscm-serviceprovisioning/javares/
      TechnicalServices.xsd"
   xmlns:tns="oscm.serviceprovisioning/1.9/
      TechnicalService.xsd">
```

If your application provides technical service operations, you need to adapt the URL referencing the operation service as follows:

```
<Operation actionURL="https://<host>:<port>/
   OperationService/AsynchronousOperationProxy?wsdl"
   id="STOP_VIRTUAL_SYSTEM">
      <LocalizedName locale="de">Stop</LocalizedName>
</Operation>
```

Adapting the Application Source Code

Adapt the source code of your application so that it can use and address the new ESCM Web services:

1. Take a look at the new `IntegrationHelper.zip` file contained in the `oscm-integration-pack.zip` archive. You can implement your Web service client in analogy to the integration helpers. They provide a sample ESCM Web service client implementing runtime API versioning.

2. Compare the sources of your previous ESCM installation with the ones contained in the new integration package. Adapt, for example, the code in your `WsProxyInfo.java` class.
3. Copy all newly delivered `.jar` files to the location where you implement your application.
4. Rebuild and redeploy your application.

2.2 Update Installation

ESCM

An update installation is possible from a ESCM V15.4, V16.0, V16.1, V17.0, V17.1, V17.2, or V17.3 installation to this release. If you want to upgrade from a previous release, you need to uninstall ESCM and restart with a new installation. Refer to the instructions in the *Installation Guide* for details.

APP and Controllers

The platform operator and technology managers must make sure that the following rules are observed when updating ESCM, APP, and/or service controllers: The ESCM version must be higher or equal to the APP version. The APP version must be equal to the controller version.

Example: If you want to use the VMware controller included in the V17.4 release, you must upgrade ESCM and APP to V17.4 first.

Note:	The extension interface for adding a custom tab to the **MySubscriptions** page on a marketplace cannot be used with versions of APP or service controllers prior to V16.1 Fix 7. For the extension interface to function correctly, you need to carry out an update installation for APP as well as the service controllers.

2.3 Asynchronous Provisioning Platform

Manual Migration of Technical Services

As of V15.3.2, APP implements HTTPS client authentication for Web services in APP (both, the Provisioning Service and the Operation Service). The Web service endpoints in APP have been configured for CLIENTCERT authentication.

In case you have implemented and provided technical service definitions using the HTTP protocol for provisioning service calls in APP, you must manually migrate the technical services so that the HTTPS protocol is used:

1. Export the relevant service definition.
2. Adapt the `provisioningUrl` attribute of the `TechnicalService` element to use the HTTPS protocol.
3. Adapt the `actionURL` attribute of the `Operation` element.
4. Import the service definition into ESCM.

The Web applications (APP and controller user interface) work as before.

No Backward Compatibility in SAML_SP Mode

If you are running ESCM and APP in SAML_SP mode, you need to upgrade APP to the same version as ESCM, at least to V16.0. In SAML_SP mode, versions of APP older than V16.0 cannot communicate with the current version of the ESCM server.

3 Restrictions

This chapter describes known restrictions of this ESCM release.

Certificate-Based Authentication

Certificate-based authentication cannot be used when calling ESCM functions whose execution requires that a user role be specified. The user role determines whether the calling user is allowed to execute the function. Instead, you must use basic authentication for such Web service calls.

If you need to use certificate-based authentication in such scenarios, contact your ESCM support organization.

Web Browsers

Concurrent Sessions

ESCM does not support multiple sessions in Web browsers. This means that you cannot run and log in to ESCM using several tabs or instances of the same Web browser. If you want to use multiple sessions, run ESCM in different Web browsers, for example, in Microsoft Internet Explorer and Mozilla Firefox.

In specific cases, ESCM itself opens a new tab in the Web browser, for example, when displaying a price model obtained from an external billing system. If you leave the new tab open and continue working in the initial, original Web browser tab, unexpected effects may occur. In this case, you need to refresh the content of the initial, original Web browser tab.

Browser Navigation and Refresh

ESCM does not support the usage of the standard navigation buttons in Web browsers, such as **Back**, **Forward** or **F5**. This means that you must use the ESCM buttons for refreshing the ESCM pages and moving forward and backward between them.

Multi-Tenancy and Authentication

ESCM does not support the usage of one and the same Web browser when working with multiple Identiy Providers (tenants).

HTTP or HTTPS Configuration

ESCM can be configured to use the HTTP or the HTTPS protocol. If you use both protocols at the same time, PSP (payment service provider) integration will not work.

We recommend to configure the HTTPS protocol for all URLs used by ESCM.

Report Error Messages

The following reports require input parameters, for example a billing data key or dates:

- Supplier revenue report (can be generated by platform operators)
- Customer billing report (can be generated by suppliers and operators)
- Detailed billing report for an existing invoice (can be generated by customers)

When the input parameter is entered in a wrong or invalid format, e.g. `My<>Key` as a billing data key or `2012.5.12` as a date, the error message generated by the report engine does not contain text explaining the reason for the error. If you receive an error message when trying to generate a report, check the input parameter(s) and ensure that they are in the correct format. For example, you can find the date format in the dialog for entering the start and end date for a report; the billing data key is printed on the invoice.

Asynchronous Provisioning - Termination of Subscription

When a supplier or reseller terminates a customer's subscription and APP is used for asynchronous provisioning, an email is sent to the technology provider of the underlying service as well as to the customer's administrator or subscription manager. The email contains the information that the subscription has been terminated as well as the reason why. In case, the technology manager uses a default language in his profile different from the one of the administrator or subscription manager, the email to the customer's administrator or subscription manager will show the text of the reason in the technology manager's language.

Logout Error with Active Directory Federation Services in Internet Explorer 11

An error occurs when a user working with ESCM tries to log out by clicking **Logout** in the administration portal or on a marketplace. This is true for a specific system configuration:

- ESCM is installed in SAML_SP authentication mode so that Web browser single sign-on can be used.
- Windows Active Directory Federation Services is installed and used as Identity Provider (IdP).
- In ESCM, the `SSO_IDP_AUTHENTICATION_REQUEST_METHOD` configuration setting is defined as `POST`.
- ESCM is accessed using Internet Explorer 11.
- Internet Explorer is set up for integrated Windows authentication (IWA) so that no authentication is required when logging in to ESCM.

The correct behavior would be that the Web browser is refreshed and the user is automatically logged in again. To achieve this, either use Mozilla Firefox or Google Chrome as the Web browser, or set the `SSO_IDP_AUTHENTICATION_REQUEST_METHOD` setting to `GET`.

Error Messages When Logging in to APP

Error messages displayed when logging in to the Web interface of APP or a service controller are always in English, not in German or any other language.

Specification of Security Groups for the AWS Service Controller

If you specify security groups for the AWS service controller using the `SECURITY_GROUP_NAMES` service parameter, you also need to specify the corresponding subnet using the `SUBNET` parameter. If no subnet parameter is specified, the AWS service controller ignores any specified security groups, and the service instance is created in a default subnet and a default security group is assigned.

Issues With Exporting Revenue Reports

The following issues may occur when a supplier is exporting a revenue report in the following formats:

- `.docx`: due to the page width, the pages are cut off at arbitrary locations. The issue does not occur if the report is generated as a `.doc` file.
- `.pptx`: the content of the report is very small, an increase of approx. 300% is needed to read it.

It is therefore recommended to export a revenue report as an Excel file.

Issues with Exporting the Payment Preview

If the payment preview report is exported in the `.xlsx` format, then layout issues may occur. This can be avoided by exporting the report in `.xls` format.