FUJITSU

FUJITSU Software Enterprise Service Catalog Manager V18.0.0

Operator's Guide

April 2019

Contents

About this Manual

This manual describes the basic tasks involved in the operation and maintenance of) FUJITSU Software Enterprise Service Catalog Manager, hereafter referred to as ESCM.

The manual is structured as follows:

Chapter	Description
Introduction on page 10	Describes ESCM, its containers and their communication, its architecture, its organizations, users, and roles.
Getting Started on page 16	Describes how to access the ESCM administration portal, lists the initial steps to be performed after installing ESCM, and shows how to start and stop the ESCM containers.
Configuring ESCM and APP on page 18	Describes how to update the ESCM and APP configuration settings, how to add a currency and a language, how to use and configure timers for automatic task processing, as well as how to manage platform-wide LDAP settings.
Monitoring, Backup and Recovery on page 26	Describes how to retrieve ESCM logging information, how to configure the log level, as well as which data is subject to regular backups. In addition, it describes how to export the audit log data and how to check the load on the JMS queues.
Managing Organizations and Users on page 32	Describes how to create and manage organizations and user accounts.
Reporting on page 34	Describes the reports available for operators in ESCM.
Managing Billing and Payment on page 36	Describes how to define revenue shares and how to handle billing data: how to restart payment processes and billing runs, preview and export billing data.
Integrating Custom SSL Certificates and Key Files on page 40	Describes the usage of certificates for secure communication between ESCM and applications integrated with it.
Managing Marketplaces on page 41	Describes how to create, update, delete, and customize marketplaces.
Configuration Settings on page 43	Describes the ESCM and APP configuration settings.
LDAP Keys on page 56	Describes the keys to be defined for enabling access to an organization's LDAP system.
Audit Log on page 58	Describes the elements of the audit log.
Language Resource Bundles on page 82	Describes the resources that can be provided and customized in different languages.

Chapter	Description
User Data File for Multiple User Import on page 89	Describes the content and format of a user data file for importing multiple users in one operation.
Customer Billing Data on page 91	Describes the elements of an XML file created by exporting customer billing data.
Revenue Share Data on page 108	Describes the elements of XML files created by exporting revenue share data.
Menu Options and Required Roles on page 128	Gives an overview of the ESCM administration portal menu options and the roles required for using them.

Readers of this Manual

This manual is directed to operators who maintain and operate ESCM in their environment.

It assumes that you are familiar with the following:

- Container technology, particularly Docker and Docker Compose.
- 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.
- Web services concepts.
- Installation and administration of Web servers.
- Certificate-based authentication and communication.

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.

Abbreviations

This manual uses the following abbreviations:

API	Application Programming Interface
APP	Asynchronous Provisioning Platform

ESCM	Enterprise Service Catalog Manager
CA	Certification authority
JAX-WS	Java API for XML - Web Services
JMS	Java Message Service
JSP	Java Server Pages
LDAP	Lightweight Directory Access Protocol
PaaS	Platform as a Service
SaaS	Software as a Service
SAML	Security Assertion Markup Language
SOAP	Simple Object Access Protocol
STS	Security Token Service
WSDL	Web Services Description Language
WSIT	Web Services Interoperability Technologies
XSD	XML Schema Definition

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.
- *Microsoft Azure Integration:* A PDF manual for operators describing how to offer and use virtual systems controlled by Microsoft Azure 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.
- *OpenStack Integration:* A PDF manual for operators describing how to offer and use virtual systems controlled by OpenStack through services in ESCM.
- *VMware vSphere Integration:* A PDF manual for operators describing how to offer and use virtual machines provisioned on a VMware vSphere server through services in ESCM.

- *Shell Integration:* A PDF manual for operators describing how to use Shell scripts 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 Introduction

Enterprise Service Catalog Manager (ESCM) is a set of services which provide all business-related functions and features required for turning on-premise applications and tools into "as a Service" (aaS) offerings and using them in the Cloud. This includes ready-to-use account and subscription management, online service provisioning, billing and payment services, and reporting facilities.

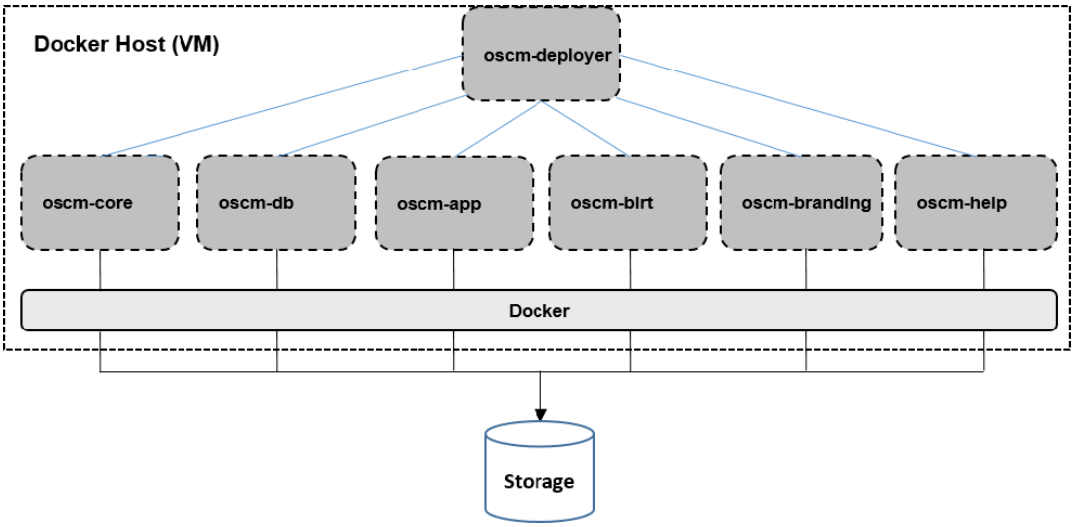
With its components, ESCM supports software vendors as well as their customers in leveraging the advantages of Cloud Computing.

1.1 ESCM Containers

ESCM is provided in Docker containers and deployed in a container environment such as OpenStack or a Kubernetes cluster providing for optimum performance, scalability, failover, and non-stop operation. The applications integrated with ESCM and their data may be hosted in the same Virtual Machine (VM) as ESCM or in different locations.

The `oscm-deployer` container is used for the configuration and deployment of the following ESCM containers:

- `oscm-core`: The ESCM core application.
- `oscm-birt`: The report engine that ESCM uses for generating reports.
- `oscm-branding`: A static Web server providing an empty directory structure for customizing the layout and branding of ESCM marketplaces.
- `oscm-help`: A static Web server providing the online help for the ESCM administration portal and marketplaces.
- `oscm-app`: The Asynchronous Provisioning Platform (APP) together with an OpenStack, an Amazon Web Services (AWS), a Microsoft Azure, a VMware, and a Shell service controller.
- `oscm-db`: Database SQL server providing the database schema required for running ESCM and APP.



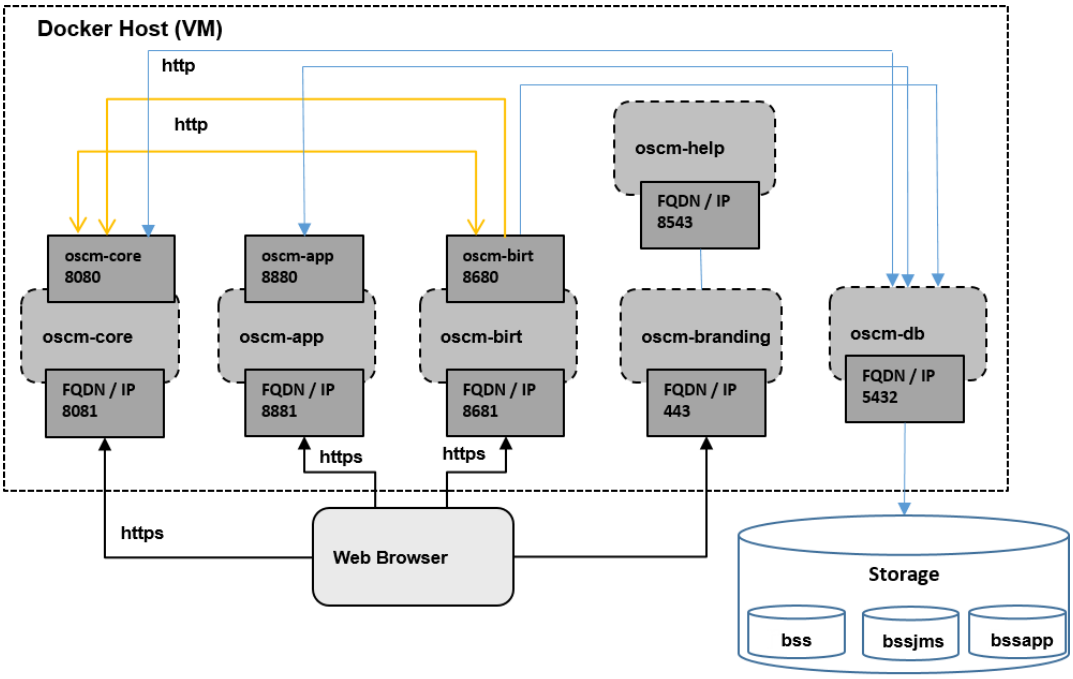
ESCM and APP store their data in PostgreSQL databases. For each database, a volume for persistent storage is mounted during the deployment process. The databases write to these volumes so that data is preserved in case of database updates. By default, data is persisted in the following directory on the machine hosting the VM where ESCM is deployed (the Docker host):

`<docker>/data/oscm-db/data:/var/lib/postgresql/data`

Hereafter, the `<docker>` directory is referred to as `/docker`.

Container Communication

The following figure provides an overview of the container communication on a Docker host (a VM):



The host-internal communication always relies on the HTTP protocol, whereas calls from outside of the host are always secured via HTTPS. The platform operator is responsible for opening the indicated ports, which can either be addressed using the FQDN or the floating IP address of the respective container.

To look inside a container:

1. List all containers (even stopped ones) to show the container names:

```
docker ps -a
```

2. Log in to a container using the following command:

```
docker exec -it <container name> /bin/bash
```

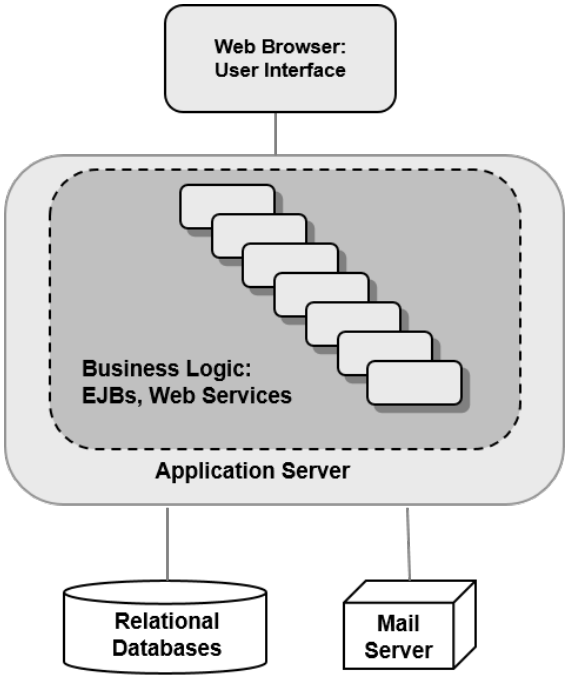
For example:

```
docker exec -it oscm-core /bin/bash
```

1.2 ESCM Architecture

ESCM is implemented in Java, using Java Platform, Enterprise Edition (Java EE) technology. It is deployed on an application server supporting this technology.

The following figure provides an overview of the architecture:



ESCM has a three-tier architecture:

- The **presentation layer** in the application server's Web container includes the **user interface** (administration portal and marketplaces), realized as JavaServer Faces. Users access the user interface in Web browsers.
- The **business logic** is implemented in Enterprise JavaBeans (EJB). Both the Enterprise JavaBeans and the **public Web services** are available in the application server's EJB container. The public Web services and their interfaces are mainly used for integrating applications and external systems with ESCM. However, they can also be employed for accessing ESCM functionality from a Web service client. HTTPS must be used for communication with the public Web services.
- ESCM **persists** its data through the Java Persistence API in **relational databases**.

For informing users about relevant issues (e.g. their registration or assignment to a subscription), ESCM must have access to a mail server.

1.3 Organizations and User Roles

Each user working in ESCM is a member of a specific organization. An organization typically represents a company, but it may also stand for a department of a company or a single person. Each organization in ESCM has a unique account and ID as well as one or more of the following roles: **operator**, **technology provider**, **supplier**, **reseller**, **broker**, **marketplace owner**, **customer**.

Customers can register themselves with ESCM or be registered by a supplier, reseller, broker, or operator. In any case, an organization with the customer role is created. Organizations with other roles can also act as customers, i.e. they are implicitly assigned the customer role. These organizations are created and assigned their roles as follows:

- When ESCM is installed, an organization with the operator role is created.
- Operators can assign the supplier, reseller, broker, and technology provider role to any existing organization or create new organizations with these roles. An organization can have both the supplier and the technology provider role. The reseller and broker roles, however, cannot be combined with each other or with the supplier or technology provider role.
- When operators create a marketplace, they specify an existing organization as its owner. In this way, the organization is assigned the marketplace owner role.

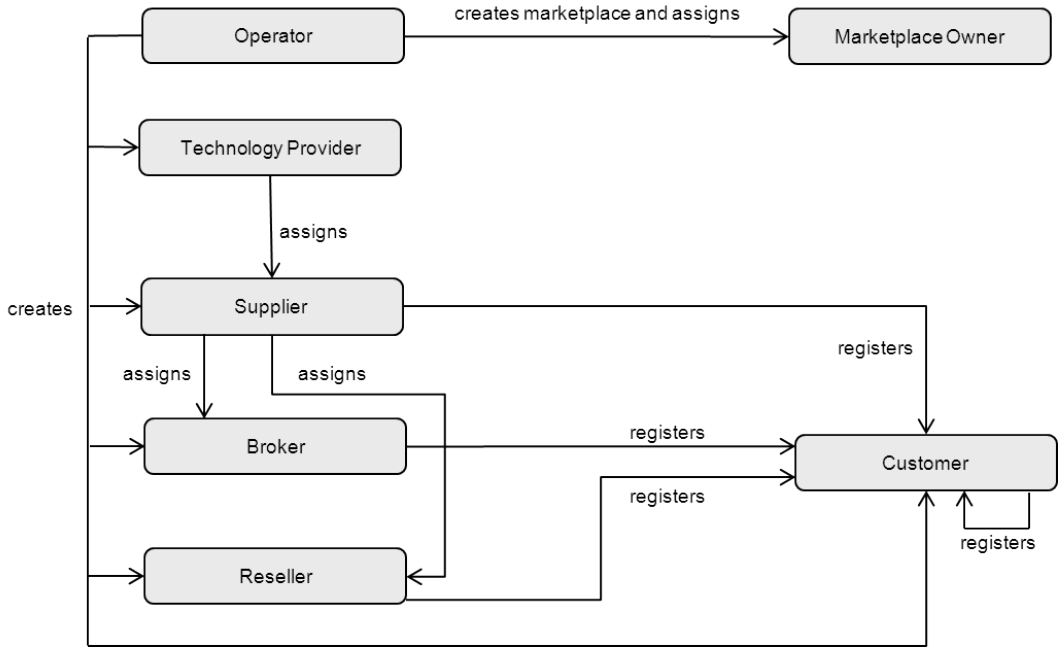
The roles of an organization determine which features are available to its users at the ESCM interfaces and which roles the users can be assigned. These user roles control the actions an individual user is allowed to carry out:

- **Standard user**: Users with this non-privileged role can work with services their organization has subscribed to. Every user registered in ESCM automatically is a standard user. Additional user roles must be assigned explicitly by an administrator.
- **Administrator**: Each organization must have at least one user with this role. An administrator can manage the organization's account and subscriptions as well as its users and their roles. The first administrator of an organization is defined when the organization is created.
- **OU administrator**: The users of an organization can be grouped in organizational units (OUs). The OU administrator role allows a user to manage the organizational units for which he has been appointed as an administrator, to create, modify, and terminate subscriptions for these units, as well as generate reports for cost-controlling purposes.
- **Subscription manager**: This role allows a user to subscribe to services and manage his own subscriptions. Unlike administrators, subscription managers are not permitted to work on subscriptions belonging to others or on subscription data related to billing and payment.
- **Technology manager**: This role allows a user to define technical services. It can be assigned to users of technology provider organizations.
- **Service manager**: This role allows a user to define marketable services and price models as well as publish marketable services. It can be assigned to users of supplier organizations.
- **Reseller**: This role allows a user to publish a supplier's marketable services applying different terms and conditions. It can be assigned to users of reseller organizations.
- **Broker**: This role allows a user to publish a supplier's marketable services without changing the terms and conditions defined by the supplier. It can be assigned to users of broker organizations.
- **Marketplace manager**: This role allows a user to define the organizations who are permitted to access a marketplace and publish services to it as well as update and customize a marketplace. This role can be assigned to users of marketplace owner organizations. It is

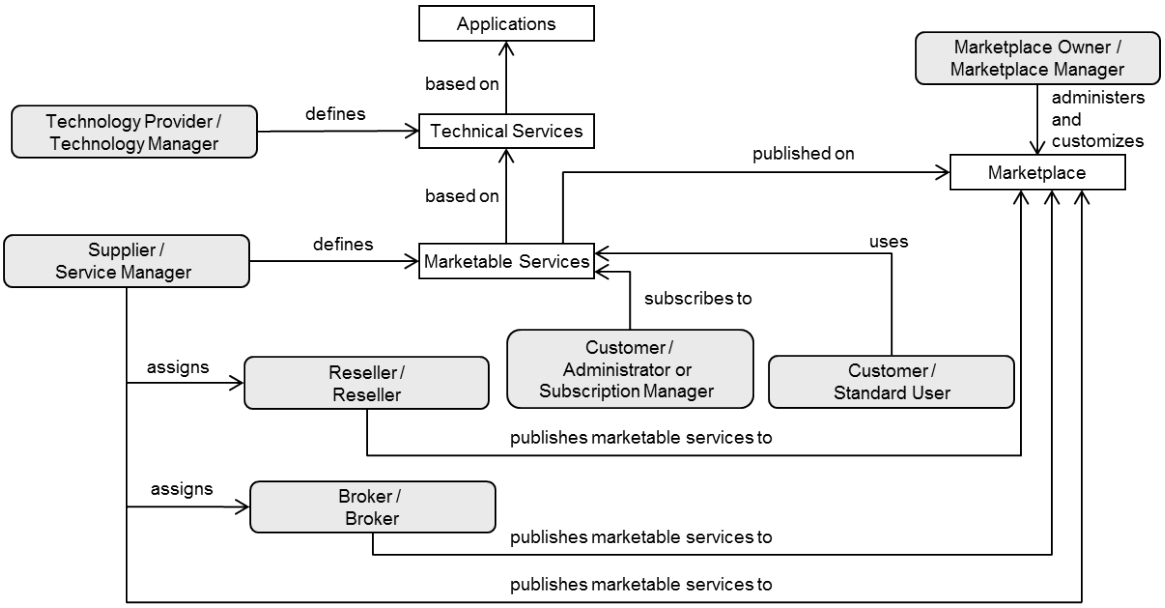
automatically assigned to all administrators of the marketplace owner organization when a marketplace is created.

- **Operator**: This role allows a user to carry out configuration and maintenance tasks, manage organizations, and create marketplaces. The first operator is created together with its operator organization when ESCM is installed.

The following illustration provides an overview of how organizations with the different roles are created and related with each other:



The following illustration provides an overview of the user roles and the main tasks of users with these roles:



2 Getting Started

Accessing ESCM

After the deployment, you have access to the ESCM administration portal where you can start with creating organizations, viewing and partially updating configuration settings, etc.

Access the ESCM administration portal in a Web browser using an URL in the following format:

`https://<hostname.fqdn>:<port>/oscm-portal`

`<hostname.fqdn>` is the name and the fully qualified domain name of the machine where ESCM has been deployed. `<port>` is the port to address the machine (default: `8081`), `oscm-portal` is the default context root of ESCM and cannot be changed.

You are prompted for the user ID and password. The initial credentials are as follows:

User ID: `administrator`

Password: `admin123`

It is recommended that you change the initial password in the ESCM administration portal (**Change Password** page in the **Account** menu).

After login, the operator functionality is available in the **Operation** menu.

Next Steps

In order for a supplier to be able to define price models and for the rating and billing engine of ESCM to be able to calculate usage costs, you need to add one or several currencies to the system. For details, refer to *Adding a Currency to ESCM* on page 21.

In a next step, you can create technology provider and seller organizations (suppliers, resellers, brokers), and set up marketplaces so that services can be offered to customers. For details, refer to *Managing Organizations and Users* on page 32 and *Managing Marketplaces* on page 41.

Startup and Shutdown

The ESCM containers can be started, stopped, removed, and restarted using the standard utilities of Docker.

To **stop** a container, use the following command on your Docker host:

```
docker stop <container-name>
```

For example:

```
docker stop oscm-core
```

To **delete** a container:

1. Stop the container using the above command.
2. Remove the container:

```
docker-compose -f docker-compose-oscm.yml rm <container-name>
```

To **create and start** a container:

```
docker-compose -f docker-compose-oscm.yml up <container-name>
```

To **restart** a container:

```
docker-compose -f docker-compose-oscm.yml restart <container-name>
```

When shutting down the containers, the `oscm-db` container must be the last one to be stopped. When starting or restarting the containers, make sure that the `oscm-db` container is first.

3 Configuring ESCM and APP

This chapter describes:

- How to update configuration settings for ESCM and APP.
- How to add a currency definition to ESCM.
- How to add an additional language to ESCM and how to customize existing texts.
- How to use and configure timers in ESCM.
- How to define and manage system-wide LDAP connection settings.
- How to enable access to the Asynchronous Provisioning Platform (APP) and service controllers.

3.1 Updating ESCM and APP Configuration Settings

Initial Settings

The ESCM software as well as the Asynchronous Provisioning Platform (APP) require a number of settings for configuring their container runtime environment. The mandatory settings have already been specified in environment variables in Docker files before deploying the containers. Usually, you needed to adapt the initial settings to your environment, in particular server names, ports, paths, and user IDs.

After the deployment, you can update the configuration settings by editing the following configuration files in the `/docker` directory, where `/docker` is the directory on the Docker host where all ESCM- and APP-specific data is located:

1. `.env`: Configuration settings for Docker, such as images and the base data directory (default: `/docker`)
2. `var.env`: Configuration settings for ESCM and APP, such as mail server, database and other settings.

The initial, mandatory settings are stored in the `bss` and `bssapp` databases. They are described in detail in *Initial Configuration Settings* on page 43.

Optional Settings for ESCM

There are additional configuration settings whose keys are also stored in the `bss` database. The settings have default values that can be viewed and partially be changed in the ESCM administration portal (**Update configuration settings** in the **Operation** menu).

Note:	Be aware that changing the settings in the administration portal should only be used for testing purposes. Your changes will be lost as soon as ESCM is restarted.

The ESCM configuration settings are described in detail in *ESCM Configuration Settings* on page 47.

Optional Settings for APP

For APP, the configuration settings are stored in the `bssapp` database. You can view and change the settings using the Web interface of APP. To access this interface, use an URL in the following format:

`https://<hostname.fqdn>:<port>/oscm-app`

<hostname.fqdn> is the name and the fully qualified domain name of the machine where APP has been deployed. <port> is the port to address the machine (default: 8881), oscm-app is the default context root of APP and cannot be changed.

The APP configuration settings are described in detail in *APP Configuration Settings* on page 54.

Persistently Updating ESCM Configuration Settings

If you want to persistently update configuration settings in the bss or bssapp database, proceed as follows:

1. On your Docker host, edit the .var and var.env files located in the /docker directory as required.
2. If you want to change the default value of a configuration setting that is not included in the var.env file yet:

   Add the key and the value the setting shall take on to the var.env file.

   You can find the keys of the settings on the **Update configuration settings** page or in the APP Web interface. They are described in detail in *ESCM Configuration Settings* on page 47 and *APP Configuration Settings* on page 54. For example:

   ```
   TIMER_INTERVAL_SUBSCRIPTION_EXPIRATION=345600000
   ```

3. Save the var.env file to its original location:

   ```
   /docker/var.env
   ```

4. In the docker-compose-initdb.yml file, set the OVERWRITE flag to true (under the environment settings) for changing a setting in the desired database.

   For example, for persistently updating a setting in the bss database, set the flag to true for oscm-initdb-core:

   ```
   oscm-initdb-core:
       image: ${IMAGE_INITDB}
       container_name: oscm-initdb-core
       env_file: var.env
       environment:
         - TARGET=CORE
         - SOURCE=INIT
         - OVERWRITE=true
         - LOG_LEVEL=INFO
       links:
   - oscm-db:oscm-db
   ```

5. Stop and remove all ESCM containers:

   ```
   docker-compose -f docker-compose-oscm.yml stop
   docker-compose -f docker-compose-oscm.yml rm -f
   ```

6. Start the container for initializing the database and delete the finished containers:

   ```
   docker-compose -f docker-compose-initdb.yml up -d oscm-db
   docker-compose -f docker-compose-initdb.yml up oscm-initdb-core
   docker-compose -f docker-compose-initdb.yml up oscm-initdb-jms
   docker-compose -f docker-compose-initdb.yml up oscm-initdb-app
   docker-compose -f docker-compose-initdb.yml
    up oscm-initdb-controller-openstack
   docker-compose -f docker-compose-initdb.yml
    up oscm-initdb-controller-aws
   ```

```
docker-compose -f docker-compose-initdb.yml
 up oscm-initdb-controller-vmware
docker-compose -f docker-compose-initdb.yml
 up oscm-initdb-controller-azure
docker-compose -f docker-compose-initdb.yml
 up oscm-initdb-controller-shell
docker-compose -f docker-compose-initdb.yml stop
docker-compose -f docker-compose-initdb.yml rm -f
```

7. Restart all application containers:

```
docker-compose -f docker-compose-oscm.yml up -d
```

Updating Controller Configuration Settings

If you want to persistently update the common controller settings individually for every controller, you use the following variables in the `var.env` file:

- `CONTROLLER_ORG_ID`. The ID of the organization in ESCM which is responsible for the service controller. The organization must have the technology provider role.
- `CONTROLLER_USER_KEY`: The user key for accessing ESCM.

  The user specified here must have the administrator and technology manager role, and belong to the organization specified by the organization ID.
- `CONTROLLER_USER_NAME`: The name of the user specified by the user key for accessing ESCM.
- `CONTROLLER_USER_PASS`: The password of the user specified by the user key for accessing ESCM.

You can change organization and user responsible for each individual service controller by setting environment variables as follows:

1. On your Docker host, edit the `docker-compose-initdb.yml` file and temporarily set the `OVERWRITE` flag to `true` (under the `environment` settings) for changing a setting in the desired database.

   For example, for changing the current organization and user responsible for Shell integration controller, edit the `docker-compose-initdb.yml` file and set the controller settings as follows:

```
CONTROLLER_ORG_ID=959c9bf7
CONTROLLER_USER_KEY=10000
CONTROLLER_USER_NAME=supplier@adfs.com
CONTROLLER_USER_PASS=supplier
```

2. Run the following command:

```
docker-compose -f docker-compose-initdb.yml up <container_name>
```

   where `<container-name>` is one of the following:

   - `oscm-initdb-controller-openstack`
   - `oscm-initdb-controller-aws`
   - `oscm-initdb-controller-azure`
   - `oscm-initdb-controller-vmware`
   - `oscm-initdb-controller-shell`

3. Edit the `var.env` file located in the `/docker` directory again, and set the `OVERWRITE` flag back to `false`.

3.2 Adding a Currency to ESCM

After installation, you need to add the currency or currencies to be supported by ESCM. These currencies will be available to suppliers when defining the price models for marketable services. The selected currencies are used by the integrated rating and billing engine of ESCM when calculating subscription usage charges.

To add a currency:

In the ESCM administration portal, choose **Manage currencies** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

3.3 Adding a Language to ESCM and Customizing Texts

ESCM supports multiple languages in which users can work. After installation, English, German, and Japanese language bundles are available. The operator can add additional languages and customize the texts provided in the English, German, and Japanese language bundles. Refer to *Supported Language Codes* on page 86 for the list of languages that can be added.

Adding a language or customizing existing texts comprises the following steps:

1. Translating all resources of a language bundle:
   - The user interface resources are translated or updated in a Microsoft Excel file.
     Refer to *User Interface Resources* on page 82 for details.
   - The online help and FAQ files are translated or updated directly in the HTML files.
     Refer to *Online Help and FAQs* on page 83 for details.
2. For a new language: Registering the language with ESCM.
3. Importing the translated or updated Microsoft Excel file.
4. Deploying the translated or updated online help and FAQ files.
5. For a new language: Activating the language in ESCM. All activated languages are available to users for selection when they edit their user profile.

To add a language:

In the ESCM administration portal, choose **Manage languages** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

3.4 Configuring Timers

Timers are used to handle background tasks, for example to check for expired subscriptions that are to be deleted. Each timer has a time interval specifying when it is executed periodically. The values are indicated in milliseconds.

Time intervals are defined and can be changed in the ESCM configuration settings.

Note:	Be aware that several functions in ESCM will not work if you do not enable the timers. For example, if a customer specifies that a subscription is to expire in 10 days, and you did not configure the timer `RESTRICTED_SUBSCRIPTION_USAGE_PERIOD`, the subscription will not expire after 10 days.

The initial expiration time of a timer for which a time interval is defined is calculated based on January 1st, 00:00:00.000, of the current year. For example, if you specify an interval of one week for a timer on January 5th, 14:30:00, the timer will expire for the first time on January 8th at 00:00:00.000, next on the 15th, the 22nd, etc. Or, if you specify an interval of one month for a timer on August 4th, 17:00:00, the timer will expire for the first time on September 1st at 00:00:00.000, next on October 1st, November 1st, etc. To avoid the expiration of several timers at the same time, which would result in heavy load on the system, there is an additional setting: An **offset** for each timer. The offset is added to the expiration time.

Example

For the timer used to remove customer accounts that have not been confirmed, the following configuration settings are defined:

```
PERMITTED_PERIOD_UNCONFIRMED_ORGANIZATIONS=604800000
TIMER_INTERVAL_ORGANIZATION=86400000
TIMER_INTERVAL_ORGANIZATION_OFFSET=300000
```

The first setting indicates how long an organization account is allowed to remain unconfirmed: 7 days. The second setting indicates the time interval at which the check for unconfirmed accounts is executed: every 24 hours. The third setting, the offset, is set to 5 minutes. The timer will expire at 0:05 a.m. every day. The offset is not accumulated, but stays the same every day.

Available Timers

There are the following timers:

- **`ORGANIZATION_UNCONFIRMED`**: Timer to check for organization accounts that have not been confirmed by a login of the initial administrator within a certain period of time. When this timer expires, the respective organization accounts are removed.

  This timer requires the following configuration settings:

  - `TIMER_INTERVAL_ORGANIZATION`
  - `TIMER_INTERVAL_ORGANIZATION_OFFSET`
  - `PERMITTED_PERIOD_UNCONFIRMED_ORGANIZATIONS`

- **`USER_NUM_CHECK`**: Timer to check for the current amount of users registered with the platform.

  This timer requires the following configuration setting:

  - `TIMER_INTERVAL_USER_COUNT`

- **`RESTRICTED_SUBSCRIPTION_USAGE_PERIOD`**: Timer used to ensure that subscriptions can only be used for the time specified in the underlying service's parameters. If this period is exceeded, the timer-related operations must be executed to make sure that the subscription cannot be used anymore unless the supplier upgrades or downgrades the underlying service.

  This timer requires the following configuration settings:

  - `TIMER_INTERVAL_SUBSCRIPTION_EXPIRATION`
  - `TIMER_INTERVAL_SUBSCRIPTION_EXPIRATION_OFFSET`

- **`TENANT_PROVISIONING_TIMEOUT`**: Timer used to check pending subscriptions. When the timeout time is reached, an email is sent to the administrators and the relevant OU administrators and subscription managers of the organizations who created the subscriptions, informing them about the timeout.

  This timer requires the following configuration settings:

  - `TIMER_INTERVAL_TENANT_PROVISIONING_TIMEOUT`

    - TIMER_INTERVAL_TENANT_PROVISIONING_TIMEOUT_OFFSET

- **BILLING_INVOCATION**: Timer for billing runs calculating subscription usage costs (customer billing data) or revenue share data. The interval for this timer is one day and cannot be changed.

  This timer requires the following configuration setting:

  - TIMER_INTERVAL_BILLING_OFFSET

- **DISCOUNT_END_CHECK**: Timer used to check whether the end date for discounts granted to customers has been reached. The timer interval is one day and cannot be changed.

  This timer requires the following configuration settings:

  - TIMER_INTERVAL_DISCOUNT_END_NOTIFICATION_OFFSET

- **INACTIVE_ON_BEHALF_USERS_REMOVAL**: Timer used to remove non-existing users from the database that were created because an organization acted on behalf of another organization.

  The timer for database cleanup requires the following configuration settings:

  - TIMER_INTERVAL_INACTIVE_ON_BEHALF_USERS
  - TIMER_INTERVAL_INACTIVE_ON_BEHALF_USERS_OFFSET

For a detailed description of the timers, refer to *ESCM Configuration Settings* on page 47.

To configure a timer:

To view the current settings for the available timers, choose **Update configuration settings** in the **Operation** menu in the ESCM administration portal. To persistently update the setting of a timer, proceed as described in *Updating ESCM and APP Configuration Settings* on page 18.

Retrieving Expiration Times

You can check when the currently registered timers expire.

To retrieve the expiration times:

In the ESCM administration portal, choose **Manage timers** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

Re-Initializing Timers

When you set a timer or update the settings for a timer, you need to re-initialize the timers in order to start them.

To re-initialize the timers:

In the ESCM administration portal, choose **Manage timers** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

3.5 Managing LDAP Settings

User IDs and passwords of an organization can be created and maintained in the platform or in an existing LDAP system of an organization.

When maintained in the platform, the user data is stored in the platform's database. An organization's administrator can register new users, and, if required, request passwords to be reset by the operator.

When using an LDAP system, an organization does not need to register its users manually with the platform. The organization's administrator can import the users from the LDAP system, thus automatically registering them with the platform. The users are managed in the LDAP system. The

platform continuously synchronizes its information on the users. Connection settings have to be defined in a configuration file so that the platform can connect to the LDAP system.

Whether an organization uses an LDAP system for user management is determined when the organization is created in the platform. An organization can be created in several ways:

- A customer registers himself. In this way, an organization with the customer role is created. Users are always managed in the platform, and an LDAP system cannot be used.
- A seller (supplier, reseller, or broker) registers a customer. In this way, an organization with the customer role is created. The seller can specify whether user management in an external LDAP system is to be used.
- You as the platform operator create an organization of any role. For any organization, you can specify whether user management in an external LDAP system is to be used. For details, refer to *Creating an Organization* on page 32.

A mixture of maintaining users in the platform and in the LDAP system is not supported. In addition, the type of user management can no longer be changed once an organization has been created.

The operator organization cannot use an external LDAP system. So you can always log in to the platform and change connection settings, for example if the LDAP system of an organization is not available and thus authentication against it is not possible.

As an operator, you can define default LDAP configuration settings for the entire platform. These settings apply as long as no organization-specific LDAP settings are specified. You can also check an organization's connection to its LDAP system.

Defining Default LDAP Settings

Defining default LDAP settings for the entire platform is useful if settings are to be reused for several organizations. If organization-specific LDAP settings exist, they overrule the default LDAP settings.

To define default LDAP settings:

In the ESCM administration portal, choose **Manage LDAP settings** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help. For a list of LDAP keys, refer to *LDAP Keys* on page 56.

Checking the Connection to an LDAP System

If for some reason an organization's LDAP system cannot be reached, you can check and restore the connection so that user authentication is possible again.

To check an LDAP connection:

In the ESCM administration portal, choose **Manage LDAP settings** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help. For a list of LDAP keys, refer to *LDAP Keys* on page 56.

3.6 Configuring and Enabling Access to APP and Service Controllers

When integrating applications with ESCM, the instance provisioning can be done in two provisioning modes: synchronous or asynchronous mode.

Asynchronous provisioning is required if provisioning operations take a long time because long-running processes or manual steps are involved. This is the case, for example, when

provisioning virtual machines on a virtual machine server. ESCM supports the integration of such applications with its asynchronous provisioning platform (APP). This is a framework which provides a provisioning service as well as functions, data persistence, and notification features which are always required for integrating applications in asynchronous mode.

Having deployed the `oscm-app` container, the following service controllers are registered with ESCM and initialized:

- AWS service controller: Can be used for integrating the Amazon Elastic Compute Cloud Web service with ESCM. Refer to the *AWS Integration* guide for details.
- OpenStack service controller: Can be used for integrating OpenStack services with ESCM. Refer to the *OpenStack Integration* guide for details.
- VMware service controller: Can be used for integrating VMware vSphere services and using virtual machines provisioned on a VMware vSphere server with ESCM. Refer to the *VMware vSphere Integration* guide for details.
- Azure service controller: Can be used for integrating Microsoft Azure services with ESCM. Refer to the *Microsoft Azure Integration* guide for details.
- Shell service controller: Can be used for executing Shell scripts through services in ESCM. Refer to the *Shell Integration* guide for details.

Before technology providers can use the service controllers, they must be entered as responsible organizations in APP as well as in the service controllers.

To register an organization that is to be responsible for a service controller:

1. Login to the ESCM administration portal, and choose **Create organization** in the **Operation** menu.

   Make sure to set the **Technology provider** role for the organization. Refer to the online help for details.

2. Access the Web interface (base URL) of APP. The URL has the following format:

   `https://<hostname.fqdn>:<port>/oscm-app`

   `<hostname.fqdn>` is the name and the fully qualified domain name of the machine where APP has been deployed. `<port>` is the port to address the machine (default: `8881`), `oscm-app` is the default context root of APP and cannot be changed.

3. Log in with your user ID and password (default: `administrator`, `admin123`).
4. Specify the ID of the technology provider organization that is to be responsible for the respective service controller (`ess.openstack`, `ess.aws`, `ess.azureARM`, or `ess.vmware`).
5. Click **Save Configuration** to save the settings.

4 Monitoring, Backup and Recovery

Regular system operation and maintenance includes the monitoring of the system and its processes as well as the backup of the databases, configuration settings, and log files.

This chapter describes:

- The monitoring of ESCM.
- How to configure the log level.
- How to export audit log data.
- The backup of the databases, configuration settings, and log files.
- The monitoring of the JMS queues.

4.1 Monitoring ESCM

In addition to entries in the standard application server log file, ESCM provides a log file of its own that helps you detect problems and identify, for example, illegal access to the system.

The logging of ESCM is based on the `log4j` tool.

4.1.1 Retrieving Logging Information

For each running container, ESCM writes its logging information into a dedicated log file on the storage volume for each container. New messages are continually appended to the log file at runtime.

Note: The ESCM deployer creates empty log files in a dedicated directory on the Docker host:
`/docker/logs/<container_name>/<container_name>.out.log`
Log information is, however, only written to these files if you configure `docker-compose` and `syslog` on your Docker host to write the container logs to these files. By default, Docker uses the `json-file` driver to record the logs of your containers. The log output in JSON format can be found in `/var/lib/docker/containers/<container-id>/<container-id>-json.log`. For more information on log handling, refer to the Docker documentation.

To show all running containers, execute the following command on the VM where ESCM has been deployed:

```
docker ps
```

To show the logs of a specific container:

```
docker logs -f <container name>
```

For example:

```
docker logs -f oscm-core
```

In addition to entries in the ESCM log files, you can find application server-specific log entries by logging in to the individual containers. Be aware that these are available only if you set the

TOMEE_DEBUG configuration setting in the var.env configuration file to true. To view the logs, log in to the respective container as follows:

```
docker exec -it <container name> /bin/bash
```

For example:

```
docker exec -it oscm-core /bin/bash
```

You find the logs in the following directory:

/opt/apache-tomee/logs

4.1.2 Configuring the Log Level

ESCM supports the following types of log information with their corresponding log level:

- **ERROR**: Problems that do not allow to continue working with ESCM in the current transaction or that indicate an issue which must be solved. With the ERROR log level set, the log file contains all exceptions that occurred at runtime with a complete stack trace.
- **WARN**: Problems that allow for the completion of an operation, irrespective of whether the operation is completed fully or only partially. For example, an email could not be sent. With the WARN log level set, the log file comprises all messages of type WARN and ERROR.
- **INFO**: Basic information such as the state of the server, whether it was started or stopped, whether a user logged in. With the INFO log level set, the log file comprises all messages of type INFO, WARN, and ERROR.
- **DEBUG**: Detailed information with references to the ESCM implementation, mainly start and exit of methods as well as proposals of how to solve the problem. With the DEBUG log level set, the log file comprises all messages.

To change the log level:

Proceed as described in *Updating ESCM and APP Configuration Settings* on page 18 and add the LOG_LEVEL configuration setting to the var.env file. You can set its value to INFO, DEBUG, WARNING, or ERROR respectively.

4.2 Exporting Audit Log Data

You can export information on all kinds of user operations related to subscriptions and marketable services including their price model. The data represents an audit log. Exporting this data may be useful, for example to check when and by whom a price model was changed or when subscriptions were created.

You can export information on subscription-related operations performed by administrators or standard users, as well as information on service and price model-related operations performed by service managers of supplier organizations. Administrator operations include actions performed by OU administrators and subscription managers. Service manager operations include actions performed by resellers and brokers.

Note: The system records user operations only if the AUDIT_LOG_ENABLED configuration setting is set to true.

You can specify the start and end date of the time period for which you want to export the log data. The dates must be specified in the format YYYY-MM-DD.

To export the audit log:

In the ESCM administration portal, choose **Export audit log** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

You can choose to view the exported data or save it to a file. For details on the content, refer to *Audit Log* on page 58.

4.3 Backup and Recovery

ESCM does not offer integrated backup and recovery mechanisms. Use the standard file system, application server, and database mechanisms instead.

It is recommended to create a regular backup of all persisted data on the Docker host from the mounted storage volume (default: `/docker` directory on the Docker host):

- `/config`: Certificates, customized brandings, etc.
- `/data`: ESCM databases (`bss`, `bssapp`, `bssjms`). The frequency of database backups depends on the amount of changes and on the availability of time slots with low load. PostgreSQL supports database backups without previous shutdown. For details, refer to the PostgreSQL documentation.
- `/logs`: ESCM log files.
- The `.env` configuration file.
- The Docker Compose `.yml` files.
- Customizations made to marketplaces, especially style sheets and localized texts presented at the user interface.

Note: When preparing for an update installation of your current ESCM release, always create a backup of all data mentioned above.

Below you find some **example procedures** for backing up and restoring the ESCM and APP databases.

Backup

1. Login to the Docker host where ESCM is deployed via SSH using the floating IP address of the VM.

```
ssh root@<floating_ip>
```

2. Access the ESCM configuration file and note down the database superuser password set with the `DB_SUPERPWD` variable:

```
less /docker/var.env
```

3. Create a directory where the database backups are to be stored. For example:

```
mkdir /docker/data/backup
```

4. Start a temporary Docker container with access to the databases for which you want to create backups. For example:

```
docker run -it --name dbbackup --rm --network docker_default -v
   /docker/data/backup:/backup servicecatalog/oscm-db:latest /bin/bash
```

If you use a custom Docker registry, you need to replace the `servicecatalog/oscm-db:latest` image name with your image name.

To list the local image names, execute the following command:

```
docker images
```

5. Export the database superuser password in your environment:

```
export PGPASSWORD="<your DB_SUPERPWD>"
```

6. Create the SQL dumps of the ESCM databases:

```
pg_dumpall -g -c --if-exists -f /backup/globals.sql -h
   oscm-db -U postgres

pg_dump -c --if-exists -C --quote-all-identifiers -f
   /backup/bss.sql -h oscm-db -U postgres bss

pg_dump -c --if-exists -C --quote-all-identifiers -f
   /backup/bssapp.sql -h oscm-db -U postgres bssapp

pg_dump -c --if-exists -C --quote-all-identifiers -f
   /backup/bssjms.sql -h oscm-db -U postgres bssjms
```

7. Exit the temporary Docker container:

```
exit
```

Restore

1. Login to the Docker host where ESCM is deployed via SSH using the floating IP address of the VM.

```
 ssh root@<floating_ip>
```

2. Access the ESCM configuration file and note down the database superuser password set with the `DB_SUPERPWD` variable:

```
less /docker/var.env
```

3. Go to your Docker base directory:

```
cd /docker
```

4. Stop and remove all running containers:

```
docker-compose -f docker-compose-oscm.yml stop
docker-compose -f docker-compose-oscm.yml rm -f
```

Optionally, completely delete the existing database:

```
rm -rf /docker/data/oscm-db/data ; mkdir /docker/data/oscm-db/data
```

5. Start the ESCM database container:

```
docker-compose -f docker-compose-oscm.yml up -d oscm-db
```

6. Start a temporary Docker container with access to the database from which you can restore the data. For example:

```
docker run -it --name dbrestore --rm --network docker_default -v
   /docker/data/backup:/backup servicecatalog/oscm-db:latest /bin/bash
```

   If you use a custom Docker registry, you need to replace the `servicecatalog/oscm-db:latest` image name with your image name.

   To list the local image names, execute the following command:

```
docker images
```

7. Export the database superuser password in your environment:

```
export PGPASSWORD="<your DB_SUPERPWD>"
```

8. Restore the SQL dumps:

```
psql -h oscm-db -U postgres < /backup/globals.sql
psql -h oscm-db -U postgres < /backup/bss.sql
psql -h oscm-db -U postgres < /backup/bssapp.sql
psql -h oscm-db -U postgres < /backup/bssjms.sql
```

9. Exit the temporary Docker container:

```
exit
```

10. Go to your Docker base directory:

```
cd /docker
```

11. Start all ESCM containers:

```
docker-compose -f docker-compose-oscm.yml up -d
```

4.4 Checking the Load on the JMS Queues

ESCM uses JMS queues for asynchronous processing of requests of different types:

- The **trigger queue** handles calls to the notification service of an external process control system.
- The **task queue** handles requests from the Java Mail session for sending notification emails to ESCM users.

The requests are stored in the relevant JMS queues before they are actually executed.

It is recommended to check the load on the JMS queues on a regular basis using the application server administration console. If the load is too high, you may want to set up more cluster nodes or take another appropriate action.

Note:	In case a restart of the database used for JMS data (`bssjms`) is required, make sure to also restart the `oscm-core` container in order to speed up the JMS recovery.

5 Managing Organizations and Users

This chapter describes how to:

- Create an organization.
- Maintain the address data of organizations.
- Add a role to an organization.
- Manage user accounts: lock and unlock user accounts, and reset the password for a user.
- Register users for organizations.

5.1 Creating an Organization

You can create an organization and specify its roles (technology provider, supplier, reseller, broker, or - implicitly - customer). For every new organization, you must define a user who is to become its first administrator. This user can later register new users and assign roles to them.

To create an organization:

In the ESCM administration portal, choose **Create organization** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

5.2 Changing the Address Data of an Organization

As an operator, you can at any time update the address data of an organization.

To update the address data of an organization:

In the ESCM administration portal, choose **Manage organization** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

5.3 Adding a Role to an Organization

As an operator, you can add a role to an organization (technology provider, supplier, reseller and/or broker).

To add a role:

In the ESCM administration portal, choose **Manage organization** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

5.4 Managing User Accounts

As an operator, you can lock the account of a specific user, for example when you assume unauthorized access. You can also unlock an account, for example in case it was locked because the user tried to log in using a wrong password for the configured number of times (default: 3). In addition, you can initiate the generation of a new password for a user. In all cases, the user is notified by email.

You can, at any time, view the number of users registered with ESCM as well as the configured maximum number of registered users (default: 1000). At regular intervals (default: 12 hours), the system checks the number of registered users. You are informed by email when this check returns that the allowed number of registered users is exceeded.

Note:	If an organization uses an external LDAP system for user management, only those fields that are not mapped to settings in the LDAP system can be edited. For example, passwords can only be changed in the LDAP system.

To manage user accounts:

In the ESCM administration portal, choose **Manage users** in the **Operation** menu and click the appropriate button. For detailed step-by-step instructions, refer to the online help.

5.5 Registering Multiple Users for Organizations

Users that are to work with an organization's subscriptions are usually registered by an organization's administrator.

As an operator, you can also register multiple users on behalf of organizations that do not use an external system for user authentication. The data of the users must be specified in a user data file which can then be imported into ESCM.

The user data file must be in `csv` (comma-separated values) format. For details, refer to *User Data File for Multiple User Import* on page 89.

To register multiple users for an organization:

In the ESCM administration portal, choose **Manage users** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

6 Reporting

ESCM offers comprehensive reports for different purposes and at different levels of detail. You can choose from various predefined reports.

The following reports are available for operators:

- **Supplier revenue report**: Shows the accumulated revenues of all suppliers and resellers registered with the platform.

  A time frame can be specified. Dates are to be entered in the format `YYYY-MM-DD`.

  The report shows the name, ID, and revenue of each supplier and reseller, and one accumulated value per currency. A supplier's revenue includes all revenues generated by his authorized brokers.

- **External services report**: Shows all sellers (suppliers, brokers, resellers) who have published services with the external access type to a marketplace on the platform.

  For each seller, the report shows the corresponding marketable services with their activation and deactivation time. If the services were activated and deactivated several times, all timestamps are listed.

- **Supplier revenue share report**: Shows the revenue share data for all suppliers registered with the platform for a specified month.

  For each marketplace, the revenue share data is calculated from the accumulated charges for subscriptions which were due during the past calendar month, irrespective of supplier-specific billing periods. For each supplier, the data is broken down to the organizations which generated the revenue as well as to the individual services. The suppliers' liabilities to the other participating parties are calculated. Discounts granted by a supplier to his customers are deducted from the revenue shares. The operator can specify the month for which to generate the report. The month is to be entered in the format `MM`, the year in the format `YYYY`.

- **Broker/reseller revenue share report**: Shows the revenue share data for all brokers and resellers registered with the platform.

  For each marketplace, the revenue share data is calculated from the accumulated charges for subscriptions which were due during the past calendar month. For each broker and reseller, the data is broken down to the suppliers who are providing the services to the brokers and resellers. The operator can specify the month for which to generate the report. The month is to be entered in the format `MM`, the year in the format `YYYY`.

- **Service report (of a supplier)**: Shows all marketable services of a supplier with their existing subscriptions. Subscriptions to services offered by brokers are also listed. The operator selects the supplier by specifying the supplier organization ID.
- **Customer report (of a supplier)**: Shows all customers of a supplier and his brokers with their organization ID and the services they have subscribed to. The report outputs whether a subscription is still active or has already ended. The operator selects the supplier by specifying the supplier organization ID.
- **Billing report (of a supplier)**: Shows a summary of all billing data for each customer of a supplier and his brokers.

  The operator selects the supplier by specifying the supplier organization ID. The billing data includes the billing data key of each subscription.

- **Detailed billing report for an existing invoice of a supplier's customer**: Shows the billing data of the current billing period for a selected subscription of a supplier's customer.

With pro rata cost calculation, the costs for the real service usage are calculated based on milliseconds. With per time unit calculation, the report contains the subscription costs for the time units that ended in the billing period. The operator selects the subscription by specifying a billing data key. The billing data key of each subscription is output by the **Billing report (of a supplier)**.

To create a report:

In the ESCM administration portal, choose **Create report** in the **Account** menu, and select the supplier revenue report. For detailed step-by-step instructions, refer to the online help.

The generated report is instantly displayed at the ESCM administration portal. You can choose to print the report or save it in several formats using the icons in the pane where the report is displayed.

7 Managing Billing and Payment

This chapter describes concepts and tasks of the operator related to billing and payment. It explains what billing runs are and how to:

- Define revenue shares.
- Handle billing data, including how to manually start a billing run or payment processing and retry failed payment processes, as well as how to preview or export billing data to an XML file.

7.1 Billing Runs

At daily intervals, the database content is checked for organizations which have produced billing-relevant data, the billing data is calculated, and the data is collected and stored in the database. These checks and calculations are called "billing runs for customer billing data".

Each supplier and reseller can define his preferred start day of the monthly billing periods. When a billing period ends, the costs for all customer subscriptions that were produced in the course of this period are calculated. This is done in the billing run that is executed on the start day of the next billing period plus the offset defined by the `TIMER_INTERVAL_BILLING_OFFSET` configuration setting. For example: A supplier defines that his billing period is to start on the 8th of a month. The `TIMER_INTERVAL_BILLING_OFFSET` is set to 5 days and 4 hours. The billing run calculating the costs for the supplier's customers is executed on the 13th of each month at 04:00:00.000.

You should check at regular intervals whether billing runs or payment processing fail, and, if yes, explicitly execute them.

7.2 Defining Revenue Shares

Suppliers may involve brokers and resellers in selling their services. The brokers and resellers as well as the platform operator and the owners of the marketplaces on which the services are published, usually receive a share of the revenue for the services. ESCM calculates these revenue shares based on the billing data for the customers who use the services. The operator revenue share applies irrespective of whether services are sold by the supplier or by his resellers and brokers. Discounts granted by a supplier to his customers are deducted from the operator revenue share.

Suppliers, brokers, resellers, and marketplace owners can generate reports for their revenue shares and export the revenue share data for a specific month. As an operator, you can export the data for all the suppliers, brokers, resellers, or marketplace owners known to your platform installation. The exported data can be forwarded, for example, to an accounting system which continues to process it. For details, refer to *Handling Billing Data* on page 37.

As an operator, you are responsible for defining the revenue shares. You can define the following:

- Operator revenue share to be paid by each supplier for using the platform.
- Marketplace owner revenue share to be paid by a supplier for publishing services on a specific marketplace.
- Revenue shares for broker and reseller organizations to be paid by a supplier for selling his services.

The revenue shares specify the percentage of the revenue the operator, marketplace owners, brokers, or resellers are entitled to.

The values you enter for revenue share percentages are based on agreed conditions between the operator, marketplace owners, suppliers, resellers, and brokers. You as the operator are

responsible for setting correct values. It is possible to set the percentages to a total of over 100%. This might be intentional. For example, a supplier who wants to strongly promote a service for a limited period of time may grant a broker or reseller a revenue share of 80%. In addition, the supplier may need to pay 30% of the revenue to the marketplace owner and 10% to the operator. This results in a total revenue share percentage of 120 and thus in a negative revenue for the supplier.

Revenue shares can be defined on the following levels:

1. **Operator revenue share**. It is defined for every supplier organization that wants to sell its services on your platform. The default operator revenue share is specified when you create a supplier organization. It is independent of the marketplace to which a service is published, and whether the supplier sells services himself or authorizes resellers and brokers to do so. Discounts granted by a supplier to his customers are deducted from the operator revenue share. The default operator revenue share applies to all services of a supplier as long as you do not define service-specific operator revenue shares. A service-specific operator revenue share applies to a specific marketable service and overrules the default operator revenue share defined for the respective supplier organization.
2. **Marketplace-specific revenue shares**. They are defined for one marketplace and comprise a percentage for the marketplace owner as well as a default for all broker and reseller organizations. Discounts granted by a supplier to his customers are deducted from the revenue share for the marketplace owner.
3. **Service-specific revenue shares**. For every marketable service a supplier offers to brokers or resellers for publishing, a service-specific revenue share can be defined. This revenue share applies irrespective of the marketplace the service is published to. It overrules any defined marketplace-specific revenue shares.
4. **Individual broker or reseller revenue shares**. An individual revenue share can be defined for each broker and reseller organization. This revenue share definition overrules the service-specific and marketplace-specific revenue share definitions.

To define operator revenue shares, proceed as follows:

- Default operator revenue share for a supplier organization:

  In the ESCM administration portal, choose **Create organization** in the **Operation** menu for a new supplier organization or **Manage organization** in the **Operation** menu for an existing supplier organization.

- To define a service-specific operator revenue share:

  In the ESCM administration portal, choose **Manage operator revenue share** in the **Operation** menu.

  For detailed step-by-step instructions, refer to the online help.

To define marketplace-specific revenue shares:

In the ESCM administration portal, choose **Update marketplace** in the **Marketplace** menu. For detailed step-by-step instructions, refer to the online help.

To define revenue shares for specific services, brokers, or resellers:

In the ESCM administration portal, choose **Manage broker revenue share** or **Manage reseller revenue share** in the **Marketplace** menu, respectively. For detailed step-by-step instructions, refer to the online help.

7.3 Handling Billing Data

This section describes how you can check whether billing runs have failed, re-invoke payment processes, or explicitly start a billing run. In addition, you can preview customer billing data for a specific period or export revenue share data to an XML file.

Use the ESCM reporting facilities to retrieve detailed information on all billing-relevant data of the suppliers, resellers, brokers, and marketplace owners managed on your platform. For details, refer to *Reporting* on page 34.

7.3.1 Start Billing Run

You can explicitly start a billing run for calculating the customer billing data for a billing period. The billing period for which the data is generated depends on the day when you start the billing run and on the `TIMER_INTERVAL_BILLING_OFFSET` configuration setting.

If you start a billing run on the first day of a month plus the day(s) defined in the `TIMER_INTERVAL_BILLING_OFFSET` setting, the revenue share data is also calculated and stored.

To start a billing run:

In the ESCM administration portal, choose **Execute billing tasks** in the **Operation** menu, and click **Execute** in the respective section of the Web page. For detailed step-by-step instructions, refer to the online help.

7.3.2 Retry Failed Payment Processes

When communication problems caused the automatic payment processing for an organization to fail, you can re-invoke these payment processes manually.

To retry failed payment processes:

In the ESCM administration portal, choose **Execute billing tasks** in the **Operation** menu, and click **Execute** in the respective section of the Web page. For detailed step-by-step instructions, refer to the online help.

7.3.3 Preview Billing Data

You can preview the billing data for a customer's subscriptions for a specified time frame. The billing data can be saved to an XML file or opened in an editor of your choice. You can edit the billing data and work with it as required, for example when forwarding the data to an accounting system.

The billing data preview collects the billing-relevant data for the specified customer and accumulates it for every day within the specified time frame. The data is not stored in the database; the result is just a cost projection for the customer organization.

To preview billing data:

In the ESCM administration portal, choose **Billing data preview** in the **Operation** menu. For detailed step-by-step instructions, refer to the online help.

The billing data is saved to an XML file (`<date>BillingData.xml`).

Refer to *Customer Billing Data* on page 91 for a detailed description of the XML file elements.

7.3.4 Export Revenue Share Data

You can export the revenue share data for all organizations with a specific role for a specific time frame. Based on the customer billing data calculated for the given time frame, the costs

are analyzed to determine the revenue shares for the operator, marketplace owners, brokers, and resellers and their effects on the suppliers' revenues. Discounts granted by a supplier to his customers are deducted from the revenue shares.

You can use the data to get an overview of who is to receive which revenue shares. The exported data can be forwarded, for example, to an accounting system which continues to process it. For example, you can invoice your revenue share to marketplace owners or suppliers. For details on defining revenue shares, refer to *Defining Revenue Shares* on page 36.

To export billing data:

In the ESCM administration portal, choose **Export billing data** in the **Account** menu. For detailed step-by-step instructions, refer to the online help.

The billing data can be saved to an XML file (`<date>BillingData.xml`) and opened in an editor of your choice. You can edit the data and work with it as required.

Refer to *Defining Revenue Shares* on page 36 for a detailed description of the XML file elements.

8 Integrating Custom SSL Certificates and Key Files

Certificates are required for ESCM to allow for trusted communication between ESCM and the Asynchronous Provisioning Platform (APP), or an application underlying a technical service . The ESCM deployer has already created a respective directory structure and a suitable Docker Compose configuration. In this way, default certificates have been inserted into the respective containers after deployment, thus communication between ESCM and APP is secured.

It is however possible to use custom SSL keypairs for the application listeners. They may be self-signed or official. Privacy Enhanced Mail (PEM) format is mandatory. This is a container format that may include just the public certificate, or may include an entire certificate chain including public key, private key, and root certificates. It is only necessary to place the respective certificate and/or key files in PEM format into the appropriate directories.

8.1 Importing SSL Key Pairs for the Application Listeners

If you want to use your own SSL key pairs that your application is to use, replace the default key pair by your PEM files in the following directories on your Docker host:

- Private key: `/docker/config/<container-name>/ssl/privkey`
- Public certificate: `/docker/config/<container-name>/ssl/cert`
- Intermediates / chain (optional): `/docker/config/<container-name>/ssl/chain`

Replace `/docker` with the base directory on the Docker host that you use for persisting data, and `<container-name>` with the name of the respective ESCM container, for example, `oscm-core` or `oscm-app`.

The custom certificates must also be placed into the following trusted directory so that a trusted relationship between the containers is established:

- `/docker/config/certs`

The `/docker/config/certs` directory is shared by all containers. By default, if you use your own SSL key pairs, you must also place all the public certificate files here.

For example, if you have a custom SSL keypair for the `oscm-core` container, you need to place the private key into the `/docker/config/oscm-core/ssl/privkey` directory, and the public certificate into the `/docker/config/oscm-core/ssl/cert` directory. Additionally, you need to place the public certificate into the `/docker/config/certs` directory on your Docker host. In this case, a restart of the `oscm-core` and `oscm-app` containers is required.

8.2 Importing Trusted SSL Certificates

If you want your application to trust certain, possibly self-signed SSL certificates, put them in PEM format in the following directory on your Docker host:

`/docker/config/certs`

Replace `/docker` with the base directory on the Docker host that you use for persisting data.

For example, if you want to use the VMware service controller, you need to export the vSphere certificate in PEM format, and copy it to the `/docker/config/certs` directory. Since the VMware service controller is running in the `oscm-app` container, a restart of this container is required.

9 Managing Marketplaces

This chapter describes

- How to create marketplaces.
- How to change the owner of a marketplace.
- How to delete a marketplace.
- How to configure marketplaces.
- How to apply customizations by a marketplace owner to a marketplace.

For details on administrating and customizing marketplaces, refer to the *Marketplace Owner's Guide*.

Use the ESCM reporting facilities, for example, to retrieve information on the services published to your marketplace. For details, refer to *Reporting* on page 34.

9.1 Creating a Marketplace

You are responsible for creating marketplaces for the organizations that want to authorize suppliers, brokers, and resellers to publish their services using the facilities of ESCM. Creating a marketplace includes defining the marketplace properties and assigning an organization as the owner of the marketplace. All administrators of the assigned owner organization automatically receive the marketplace manager role. The marketplace manager role enables them to administrate and customize the marketplace.

As a prerequisite for creating a marketplace, the organization to be assigned as the marketplace owner must already exist.

To create a marketplace:

In the ESCM administration portal, choose **Create marketplace** in the **Marketplace** menu. For detailed step-by-step instructions, refer to the online help.

9.2 Changing the Owner of a Marketplace

The owner of a marketplace is responsible for administrating and customizing the marketplace to which suppliers, brokers, and resellers can publish their services. Assigning a new owner to a marketplace may remove the marketplace owner role from the previous owner organization and the marketplace manager role from its users. This is the case if the marketplace for which you change the owner is the last one owned by the organization.

To assign another owner to a marketplace:

In the ESCM administration portal, choose **Update marketplace** in the **Marketplace** menu. For detailed step-by-step instructions, refer to the online help.

9.3 Deleting a Marketplace

When deleting a marketplace with activated services, these services are automatically deactivated. Customers can no longer subscribe to them. Existing subscriptions, however, are not affected.

To delete a marketplace:

In the ESCM administration portal, choose **Delete marketplace** in the **Marketplace** menu. For detailed step-by-step instructions, refer to the online help.

9.4 Configuring Marketplaces

ESCM provides several configuration settings that influence the behavior of marketplaces. For example:

- `CUSTOMER_SELF_REGISTRATION_ENABLED`: Specifies whether customer organizations can register on a marketplace.
- `MP_ERROR_REDIRECT_HTTP` or `MP_ERROR_REDIRECT_HTTPS` : URL specifying a Web page that is to be displayed in case a visitor tries to access a marketplace without a valid marketplace ID. Note that this URL is used platform-wide. It is not tenant-specific.
- `TAGGING_MAX_TAGS`: The maximum number of tags composing the tag cloud.
- `TAGGING_MIN_SCORE`: The minimum number of times a tag must be used in services to be shown in the tag cloud.

For details on the settings, refer to *ESCM Configuration Settings* on page 47.

For details on how to change configuration settings, refer to *Updating ESCM and APP Configuration Settings* on page 18.

9.5 Customizing the Layout of Marketplaces

After deployment, ESCM provides a neutral branding for marketplaces. Marketplace owners can download the source files of this branding as a `branding-package.zip` file and use this as a basis for customizing the layout and branding of their marketplaces. Once a marketplace owner is finished with any customizations, you are responsible for uploading the changes to the `oscm-branding` container.

To upload and deploy a customized branding, proceed as follows:

1. Log in to the Docker host as user root.
2. Extract the archive file provided by the marketplace owner (`<folder-name>.zip` file) to the following folder on your Docker host:

   `/docker/config/oscm-branding/brandings/`

3. Provide the URL pointing to the customized files to the respective marketplace owner.

   By default, the URL is as follows:

   `https://<hostname.fqdn>:8443/<new-folder-name>/css/mp.css`

   where `<hostname.fqdn>` is the name and the fully qualified domain name of the host used to access ESCM, and `<new-folder-name>` is the name of the folder containing the customized files.

Appendix A: Configuration Settings

The ESCM software as well as the Asynchronous Provisioning Platform (APP) require a number of settings for configuring their container runtime environment. The mandatory settings have already been specified in environment variables in Docker files before deploying the containers. Usually, you needed to adapt the initial settings to your environment, in particular server names, ports, paths, and user IDs.

After the deployment, you can update the configuration settings by editing the following configuration files in the `/docker` directory, where `/docker` is the directory on the Docker host where all ESCM- and APP-specific data is located:

1. `.env`: Configuration settings for Docker, such as images and the base data directory (default: `/docker`)
2. `var.env`: Configuration settings for ESCM and APP, such as mail server, database and other settings.

Refer to *Updating ESCM and APP Configuration Settings* on page 18 for details.

This appendix describes all settings that can be specified in detail.

A.1 Initial Configuration Settings

.env File

The configuration settings that must be set in the `.env` file are by default:

- Which Docker images to use and from which registry they are to be fetched:
  - `IMAGE_DB`: image for the `oscm-db` container
  - `IMAGE_CORE`: image for the `oscm-core` container
  - `IMAGE_APP`: image for the `oscm-app` container
  - `IMAGE_BIRT`: image for the `oscm-birt` container
  - `IMAGE_BRANDING`: image for the `oscm-branding` container
  - `IMAGE_INITDB`: image for the temporary, stateless `oscm-initdb` container
  - `IMAGE_PROXY`: image for the `oscm-proxy` container
  - `IMAGE_HELP`: image for the `oscm-help` container
- The base directory where data is persisted on your Docker host. The default directory is `/docker` as defined with the `DOCKER_PATH` configuration setting.
- HTTP timeout for Docker Compose. You need to adjust this value if you get timeout errors during high load or network congestion.

var.env File

The configuration settings that must be set in the `var.env` file described in detail below.

Settings for Connecting to the Databases

- `DB_PORT_CORE`

  The port of the PostgreSQL database (`bss`) where the configuration settings for the `oscm-core` container are stored.

  Must be set to `5432`.

- `DB_PORT_JMS`
  The port of the PostgreSQL database (`bssjms`) where the JMS messages are stored.
  Must be set to `5432`.
- `DB_PORT_APP`
  The port of the PostgreSQL database (`bssapp`) where the configuration settings for the `oscm-app` container are stored.
  Must be set to `5432`.
- `DB_PWD_CORE`
  Specify the password of the user to connect to the `bss` database.
  Default: `bssuser`
- `DB_PWD_APP`
  Specify the password of the user to connect to the `bssapp` database.
  Default: `bssappuser`
- `DB_SUPERPWD`
  Specify the password of the PostgreSQL database superuser.
  Default: `postgres`

Settings for Connecting to the Mail Server

- `SMTP_HOST`
  The host name or IP address of your mail server used for notifications by ESCM.
- `SMTP_PORT`
  The port used by your mail server.
- `SMTP_FROM`
  The email address to be used for emails sent by ESCM.
- `SMTP_USER`
  If your mail server requires authentication, the name of the user allowed to access the mail server. If no authentication is required, set to `none`.
- `SMTP_PWD`
  If your mail server requires authentication, the password of the user allowed to access the mail server. If no authentication is required, set to `none`.
- `SMTP_AUTH`
  Defines whether your mail server requires authentication. Can be set to `true` or `false`.
- `SMTP_TLS`
  Defines if TLS (Transport Layer Security) is to be used for mail server communication. Can be set to `true` or `false`.
- `APP_ADMIN_MAIL_ADDRESS`
  The email address used for emails sent by the Asynchronous Provisioning Platform (APP). For example, this is the email address of the administrator of the technology provider organization responsible for the OpenStack service controller.

Settings for Connecting to the Service Controllers

- docker-compose -f docker-compose-oscm
- `CONTROLLER_ORG_ID`

  The ID of the organization in ESCM responsible for the service controller. The organization must have the technology provider role.

  Before the first deployment, the setting's value must be set to `PLATFORM_OPERATOR`. The operator can then log in to the service controller and define the technology provider organization that is to be responsible.

- `CONTROLLER_USER_KEY`

  The identifier of the user for accessing the service controller.

  Before the first deployment, the setting's value must be set to `1000`.

- `CONTROLLER_USER_NAME`

  The name of the user for accessing the service controller. The user specified here must have the technology manager role in ESCM and belong to the organization specified in the `CONTROLLER_ORG_ID` setting. It is recommended that the user account is used only for carrying out actions on behalf of the service controller in ESCM.

  Before the first deployment, the setting's value must be set to `administrator`.

- `CONTROLLER_USER_PASS`

  The password of the user for accessing the service controller. The password is encrypted when it is stored in the database.

  Before the first deployment, the setting's value must be set to `admin123`.

Settings for Connecting the VMware Service Controller with the vSphere Server

The VMware service controller requires the following additional settings:

- `DB_USER_VMWARE`

  The name of the user to connect to the `vmware` database. This database stores the vSphere configuration.

- `DB_PWD_VMWARE`

  The password of the user to connect to the `vmware` database. This database stores the vSphere configuration.

- `VCENTER_NAME`

  The name of the vCenter running on the vSphere server.

- `DATACENTER_NAME`

  The name of data center managed by the vCenter specified above.

- `CLUSTER_NAME`

  The name of the cluster where VMs are to be provisioned.

- `LOADBALANCER_NAME`

  The name of the load balancer used by vSphere. Usually, you can use the name of the VM Network in vSphere: `VM Network`.

Note: Be aware that the configuration of an entire vSphere environment including, for example, various data centers and clusters, requires the uploading of `.csv` files using the instance status interface of the VMware service controller. Refer to the *VMware vSphere Integration* guide for details. The initial configuration settings specified in the `var.env` file setup a connection to one vSphere vCenter, with one data center and one cluster only where the VMs will be provisioned.

Additional Settings

- `KEY_SECRET`

  A string which will be used in the `bss` database as a seed for encryption and decryption of service parameters with data type `PWD` and custom attributes marked for encryption. Make sure not to forget this string so that your database is persisted.

Note: Be aware that you must use the same key when updating the database. Otherwise, encryption and decryption is no longer possible after an update.

- `HOST_FQDN`

  The fully qualified domain name or the IP address of the host to be used to access ESCM.

- `REPORT_ENGINEURL`

  The URL of the report engine used to generate the ESCM reports. If you do not specify a correct URL template, ESCM will not be able to generate any reports, since the Report Web service cannot be called correctly.

  The setting must be as follows:

```
https://<HOST_FQDN>:8681/birt/frameset?
   __report=\${reportname}.rptdesign&SessionId=\${sessionid}&
   __locale=\${locale}&WSDLURL=\${wsdlurl}&SOAPEndPoint=\
   ${soapendpoint}&wsname=Report&wsport=ReportPort
```

  Replace `<HOST_FQDN>` with the value entered for the `HOST_FQDN` setting. The other values must not be changed.

- `TOMEE_DEBUG`

  Defines if the application server is to generate log files for ESCM. Can be set to `true` or `false`.

Application Server Settings

- `CONTAINER_CALLBACK_THREADS`

  The number of threads for constructing and destroying beans. Required for the stateless `init-db` container.

  Default: `50`

- `CONTAINER_MAX_SIZE`

  The size of the instance pool for the stateless `init-db` container.

  Default: `50`

The following settings are used for the Docker engine environment which needs access to the Docker registry as well as for the ESCM containers which access the Internet. They are used for

the `oscm-app` container, meaning that each HTTP request made from APP is processed using these settings.

- `PROXY_ENABLED`
  Defines whether the proxy is enabled in the container. Can be set to `true` or `false`.
- `PROXY_HTTP_HOST`
  Host of the proxy used for HTTP connections
- `PROXY_HTTP_PORT`
  Port of the proxy used for HTTP connections
- `PROXY_HTTPS_HOST`
  Host of the proxy used for HTTPS connections
- `PROXY_HTTPS_PORT`
  Port of the proxy used for HTTPS connections
- `PROXY_NOPROXY`
  Pipe-separated list of hosts for which the proxy is to be bypassed. By default, this list already contains localhost, 127.0.0.1, and the floating IP address of the ESCM instance.

A.2 ESCM Configuration Settings

This section describes the ESCM configuration settings.

AUDIT_LOG_ENABLED

Optional. Specifies whether user operations related to subscriptions, marketable services, and price models are logged and stored in the database. If set to `true`, the operator can export audit log data to retrieve information on the user operations.

Allowed values: `true`, `false`

Default: `false`

AUDIT_LOG_MAX_ENTRIES_RETRIEVED

Optional. Specifies how many log entries are retrieved in one export of audit log data. If this number is exceeded, a warning is displayed asking the operator to change his filter criteria and start the export again. This setting is required to keep the number of SQL requests to the database low when audit log data is exported. Too many requests may lead to a decrease in system performance.

Allowed values: Any value between `1` and `1000`

Default: `100`

AUTH_MODE

Specifies that ESCM is used for user authentication. This configuration setting is evaluated at the first startup of ESCM and cannot be changed.

Default: `INTERNAL`

BASE_URL_HTTPS

The base URL is used to access the ESCM home page and to create the URL for accessing services via HTTPS.

Syntax: `https://<hostname.fqdn>:<port>/oscm-portal`

`<hostname.fqdn>` is the name and the fully qualified domain name of the machine where ESCM has been deployed. `<port>` is the port to address the machine (default: `8081`), `oscm-portal` is the default context root of ESCM and cannot be changed.

Note: If the SSL/HTTPS port was changed, then this setting must also be updated.

CUSTOMER_SELF_REGISTRATION_ENABLED

Optional. Specifies whether customer organizations can register on a marketplace. If set to `false`, the operator needs to create an organization for the customer who wants to register, or a seller (supplier, broker, reseller) needs to register the customer.

Allowed values: `true`, `false`

Default: `false`

DECIMAL_PLACES

Optional. Specifies the number of decimal places in which usage charges are calculated.

Allowed values: `2`, `3`, `4`, `5`, `6`

Default: `2`

HIDDEN_UI_ELEMENTS

Optional. Specifies user interface elements to be hidden from the ESCM administration portal and the marketplaces operated on your platform. You can use this setting to hide user interface elements both from the marketplaces and the administration portal.

Marketplaces

If you want to hide a menu option from the **Account** menu of the marketplaces operated on your platform, enter one of the following values:

- `marketplace.navigation.Profile`: **Profile** menu
- `marketplace.navigation.Payment`: **Payment** menu
- `marketplace.navigation.Subscriptions`: **Subscriptions** menu
- `marketplace.navigation.Users`: **Users & Units** menu
- `marketplace.navigation.Reports`: **Reports** menu
- `marketplace.navigation.Processes`: **Processes** menu
- `marketplace.navigation.Operations`: **Operations** menu

To hide several options from the **Account** menu, separate the options by a comma.

Administration Portal

If you want to hide a specific page from the ESCM administration portal, you can find out which value needs to be specified here as follows:

1. Open the respective page at the administration portal.
2. Display the online help for this page.
3. Have a look at the name of the online help HTML page.

4. Omit the file extension `.htm` and replace the underscore by a dot.

Example:

You want to hide the **Manage VAT rates** page. The online help HTML page name is `organization_manageVats.htm`. Thus, the respective administration portal page is `organization.manageVats`. You need to set the configuration setting as follows:

`HIDDEN_UI_ELEMENTS=organization.manageVats`

To hide several pages from the administration portal, separate the entries by a comma.

Below, you find some more examples of values that can be used to hide a specific page. The list is not complete.

- `organization.edit`: **Edit profile** page
- `shop.editSkin`: **Customize layout** page
- `techService.edit`: **Update service definition** page

To hide a complete menu from the administration portal, enter one of the following values:

- `navigation.myAccount`: **Account** menu
- `navigation.customer`: **Customer** menu
- `navigation.operator`: **Operation** menu
- `navigation.techService`: **Technical service** menu
- `navigation.service`: **Marketable service** menu
- `navigation.priceModel`: **Price model** menu
- `navigation.marketplace`: **Marketplace** menu

Note:	The **Update configuration settings** page in the **Operation** menu is the default page the operator is directed to when logging in. If you hide the page from the menu or hide the complete menu, you are still directed to the **Update configuration settings** page where you can make changes, if required.

KEY_FILE_PATH

The path to the file containing the key required for encryption and decryption of service parameters with data type `PWD` and custom attributes marked for encryption:
`/opt/apache-tomee/bin/key`

This setting cannot be changed.

LOG_LEVEL

The log level for ESCM.

Allowed values: `ERROR`, `WARN`, `INFO`, `DEBUG`

Default: `INFO`

MAIL_JA_CHARSET

Optional. Special character encoding for emails sent in Japanese.

Default: `UTF-8`

MAX_NUMBER_ALLOWED_USERS

The maximum number of users that can be registered within the ESCM installation.

Allowed values: Any value between `1` and `9223372036854775807`

Default: `1000`

MAX_NUMBER_LOGIN_ATTEMPTS

Optional. The maximum number of allowed login attempts to ESCM. If a user does not log in successfully with this number of attempts, his account is locked.

Allowed values: Any value between `1` and `9223372036854775807`

Default: `3`

MP_ERROR_REDIRECT_HTTPS

Optional. The URL of a Web page that is to be displayed in case a visitor tries to access a marketplace without a valid marketplace ID by HTTPS. This Web page will be shown instead of the default error message. Note that this URL is used platform-wide.

Syntax: `https://<your Web page>`

Make sure to specify a valid URL that does not exceed a maximum of 255 characters.

PERMITTED_PERIOD_INACTIVE_ON_BEHALF_USERS

The time in milliseconds after which a user who logged in on behalf of a customer and was inactive will be removed from the system: `604800000`, i.e. 7 days.

This setting cannot be changed.

PERMITTED_PERIOD_UNCONFIRMED_ORGANIZATIONS

Optional. The maximum time in milliseconds until an organization's initial administrative account must be confirmed. When this time has passed, the account is removed.

Allowed values: Any value between `1` and `9223372036854775807`

Default: `604800000`, i.e. 7 days

REPORT_ENGINEURL

The URL of the report engine used to generate the ESCM reports. If you do not specify a correct URL template, ESCM will not be able to generate any reports, since the Report Web service cannot be called correctly.

The setting must be as follows:

```
https://${HOST_FQDN}:8681/birt/frameset?
   __report=\${reportname}.rptdesign&SessionId=\${sessionid}&
   __locale=\${locale}&WSDLURL=\${wsdlurl}&SOAPEndPoint=\
   ${soapendpoint}&wsname=Report&wsport=ReportPort
```

Replace `HOST_FQDN` with the fully qualified name of the Docker host where ESCM is deployed.

REPORT_SOAP_ENDPOINT

The SOAP end point of the Report Web service. All report data is retrieved via a call to the Report Web service. If you do not specify a correct value, ESCM will not be able to generate any reports, since the Report Web service cannot be called correctly.

The required value is:

`https://oscm-core:8081/oscm-reporting/ReportingServiceBean`

This setting cannot be changed.

REPORT_WSDLURL

URL of the WSDL file of the Report Web service. All report data is retrieved via a call to the Report Web service. If you do not specify a correct value, ESCM will not be able to generate any reports, since the Report Web service cannot be called correctly.

The required value is:

`https://oscm-core:8081/oscm-reporting/ReportingServiceBean?wsdl`

This setting cannot be changed.

SSO_SIGNING_KEY_ALIAS

The alias of the private key of ESCM to be used for signing requests.

Requests are signed for the communication with Web pages or applications from custom tabs in subscriptions. The application server used for the Web page or Web application needs the corresponding certificate of ESCM in its truststore for verifying the signature.

Default: `s1as`

This setting cannot be changed.

SSO_SIGNING_KEYSTORE

The path and name of the application server's keystore where the private key of ESCM specified in the `SSO_SIGNING_KEY_ALIAS` setting is stored.

Default: `./keystore.jks`

This setting cannot be changed.

SSO_SIGNING_KEYSTORE_PASS

The password for accessing the keystore specified in the `SSO_SIGNING_KEYSTORE` setting.

Default: `changeit`

This setting cannot be changed.

SUPPLIER_SETS_INVOICE_AS_DEFAULT

Specifies whether invoice is to be used as the default payment type for all customers.

Default: `false`

This setting cannot be changed.

TAGGING_MAX_TAGS

The maximum number of tags composing the tag cloud.

The tag cloud is the area of a marketplace containing defined search terms (tags). The more often a tag is used in services, the bigger the characters of the tag are displayed. Customers can use the tags to search for services, provided that the tag cloud is enabled for the marketplace by the marketplace owner.

Allowed values: Any value between `0` and `2147483647`

Default: `20`

TAGGING_MIN_SCORE

The minimum number of times a tag must be used in services to be shown in the tag cloud.

The tag cloud is the area of a marketplace containing defined search terms (tags). The more often a tag is used in services, the bigger the characters of the tag are displayed. Customers can use the tags to search for services, provided that the tag cloud is enabled for the marketplace by the marketplace owner.

Allowed values: Any value between `1` and `2147483647`

Default: `1`, i.e. a tag must have been used at least once so that it is shown in the tag cloud.

TIME_ZONE_ID

Specifies that the `GMT` time zone is used for display.

This setting cannot be changed.

TIMER_INTERVAL_BILLING_OFFSET

Optional. The offset in milliseconds for the timer for billing runs calculating subscription usage costs (customer billing data) or revenue share data. The interval for this timer is one day and cannot be changed. If no offset is defined, the default offset of 4 days is applied.

Customer billing data is calculated for a period of one month (billing period). Suppliers and resellers can define individual start days for their billing periods. Revenue share data is always calculated for the past month on the first day of a month.

The offset for the billing run timer defines the following:

- Number of days after which the billing run calculating the customer billing data or the revenue share data is executed.
- Time the timer for the daily billing runs expires on the current day.

Example:

A supplier defines the 10th of a month as the billing period start date. The offset is set to 4 days and 4 hours. The billing run that calculates the customer billing data for the past billing period of this supplier is started on the 14th of the following month at 04:00:00.000. The revenue share data is calculated on the 5th of the following month at 04:00:00.000. The daily check whether a billing period of any supplier has ended is started at 04:00:00.000 every day.

Allowed values: Any value between `0` and `2419200000` (28 days)

Default: `345600000`, i.e. 4 days.

TIMER_INTERVAL_DISCOUNT_END_NOTIFICATION_OFFSET

Optional. The offset in milliseconds for the timer for terminating the discounts for all organizations. The timer interval is one day and cannot be changed.

Allowed values: Any value between `0` and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_INACTIVE_ON_BEHALF_USERS

Optional. The time interval in milliseconds at which a check for non-existing users acting on behalf of another organization is executed. A value of 0 indicates that this timer is disabled.

A technical service definition may contain a flag (`allowingOnBehalfActing`) to indicate that an organization can act in the name of another organization. The organization must be a customer of the other organization, which must have both the technology provider and supplier role.

Additionally, the customer organization must have allowed the other organization to log in on its behalf. This is achieved via a subscription whose underlying technical service has the `allowingOnBehalfActing` flag set to `true`.

When an organization acts in the name of another organization, an artificial user ID is generated.

Cleaning up the ESCM database from time to time to remove such users who no longer exist might be required since it cannot be ensured that a technical service always removes such users itself.

Allowed values: `0` and any value between `10000` (10 seconds) and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_INACTIVE_ON_BEHALF_USERS_OFFSET

Optional. The offset in milliseconds for the timer for removing inactive "on behalf" users.

Allowed values: Any value between `0` and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_ORGANIZATION

Optional. The time interval in milliseconds at which tasks related to organizations are executed. A value of 0 indicates that this timer is disabled.

Allowed values: `0` and any value between `10000` (10 seconds) and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_ORGANIZATION_OFFSET

Optional. The offset in milliseconds for the timer for organization-related tasks.

Allowed values: Any value between `0` and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_SUBSCRIPTION_EXPIRATION

Optional. The time interval in milliseconds at which a check for expired subscriptions is executed. This timer cannot be disabled, i.e. it cannot be set to 0.

Allowed values: Any value between `10000` (10 seconds) and `9223372036854775807`

Default: `86400000`, i.e. 1 day

TIMER_INTERVAL_SUBSCRIPTION_EXPIRATION_OFFSET

Optional. The offset in milliseconds for the timer for subscription expiration checks.

Allowed values: Any value between `0` and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_TENANT_PROVISIONING_TIMEOUT

Optional. The time interval in milliseconds at which a check for timed-out subscriptions is executed. A value of 0 indicates that this timer is disabled.

Allowed values: `0` and any value between `10000` (10 seconds) and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_TENANT_PROVISIONING_TIMEOUT_OFFSET

Optional. The offset in milliseconds for the timer for pending subscription checks.

Allowed values: Any value between `0` and `9223372036854775807`

Default: `0`

TIMER_INTERVAL_USER_COUNT

The time interval in milliseconds at which the amount of users registered with the platform is checked. This timer cannot be disabled, i.e. it cannot be set to 0.

Allowed values: Any value between `1` and `9223372036854775807`

Default: `43200000`, i.e. 12 hours

WS_TIMEOUT

The timeout for outgoing Web service calls in milliseconds. After this time has passed, a timeout exception is thrown by the JAX-WS framework.

An outgoing Web service call is a call initiated by ESCM. A typical example is the invocation of the `createUsers` method of the `ProvisioningService` interface, which is implemented by an application. If the timeout is reached before the Web service call returns, the operation is aborted and an exception is thrown.

Allowed values: Any value between `1` and `9223372036854775807`

Default: `30000`, i.e. 30 seconds

A.3 APP Configuration Settings

This section describes the configuration settings that are written to the `bssapp` database during deployment.

APP_BASE_URL

The URL for accessing the Web interface of APP

Syntax: `https://<hostname.fqdn>:<port>/oscm-app`

`<hostname.fqdn>` is the name and the fully qualified domain name of the machine where ESCM has been deployed. `<port>` is the port to address the machine (default: `8881`), `oscm-app` is the default context root of ESCM and cannot be changed.

APP_TIMER_INTERVAL

The interval (in milliseconds) at which APP polls the status of instances.

Default: `15000`.

APP_TIMER_REFRESH_SUBSCRIPTIONS

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, for example in OpenStack. The number is updated only in case the technical service definition specifies a `VMS_NUMBER` parameter.

Default: `86400000` (once a day).

APP_TIMER_REFRESH_USAGEDATA

The interval (in milliseconds) at which APP triggers the execution of a Shell script for collecting events that caused usage costs in the cloud. As prerequisites, a script defining which events are to be collected must exist, the technical service definition must contain the definition of these events, and specify the execution of the usage data script.

Default: `86400000` (once a day).

APP_ADMIN_MAIL_ADDRESS

The email address to which email notifications are sent.

BSS_USER_KEY

The user key for accessing ESCM.

The user specified here must have the administrator role for a technology provider organization in ESCM. The user account is used for carrying out actions on behalf of APP in ESCM.

Default: `1000`

Note:	To persistently change the key, you need to add the `APP_USER_KEY` setting to the `var.env` file.

BSS_USER_ID

The identifier of the user specified in `BSS_USER_KEY` for accessing ESCM.

Default: `administrator`

Note:	To persistently change the user name, you need to add the `APP_USER_NAME` setting to the `var.env` file.

BSS_USER_PWD

The password of the user specified in `BSS_USER_KEY` for accessing ESCM.

Default: `_crypt:admin123`

Note:	To persistently change the key, you need to add the `APP_USER_PWD` setting to the `var.env` file.

Appendix B: LDAP Keys

The following keys must be defined in a configuration file for enabling access to an organization's LDAP system:

Key	Description
LDAP_URL	Mandatory. Provider URL of the LDAP server. This LDAP server is used for user authentication. Example: `LDAP_URL=ldap://myldapserver.lan.est.company.de:389`
LDAP_BASE_DN	Mandatory. Position in the LDAP directory tree at which to start looking for users. Example: `LDAP_BASE_DN=ou=people,dc=est,dc=mycompany,dc=de`
LDAP_PRINCIPAL	Optional. Name of the user who is allowed to query the LDAP server. Example: `LDAP_PRINCIPAL=uid=admin,ou=system`
LDAP_CREDENTIALS	Optional. Password of the user who is allowed to query the LDAP server. Example: `LDAP_CREDENTIALS=secret`
LDAP_ATTR_UID	Mandatory. LDAP attribute from which a user ID is read. The default used when an organization is created is `uid` unless the operator has defined a different value in the platform LDAP settings. Example: `LDAP_ATTR_UID=uid`
LDAP_ATTR_EMAIL	Optional. LDAP attribute from which the email address of a user is read. Example: `LDAP_ATTR_EMAIL=scalixEmailAddress`
LDAP_ATTR_FIRST_NAME	Optional. LDAP attribute from which the first name of a user is read. Example: `LDAP_ATTR_FIRST_NAME=givenName`
LDAP_ATTR_LAST_NAME	Optional. LDAP attribute from which the last name of a user is read. Example: `LDAP_ATTR_LAST_NAME=sn`
LDAP_ATTR_ADDITIONAL_NAME	Currently not used.
LDAP_ATTR_LOCALE	Optional. LDAP attribute from which the default language to be stored for a user is read. Example: `LDAP_ATTR_LOCALE=locale`

LDAP_CONTEXT_FACTORY	Mandatory. Context factory which provides the API to query the LDAP server. The default used when an organization is created is `com.sun.jndi.ldap.LdapCtxFactory` unless the operator has defined a different value in the platform LDAP settings. Example: `LDAP_CONTEXT_FACTORY=com.sun.jndi.ldap.LdapCtxFactory`
LDAP_ATTR_REFERRAL	Optional. Property defining how LDAP referrals are to be processed. If an organization uses an Active Directory with sub-domains from which users are to be imported into ESCM, the sub-domains can be modeled as referrals. In this sense, a referral is a reference to another directory partition or sub-domain. By default, values from referrals are not retrieved. This property can take on the following values: `follow`: Referrals are followed, i.e. users are imported from all referenced directory partitions or sub-domains. `ignore`: Referrals are ignored (default), i.e. users are imported from the current domain directory only. Example: `LDAP_ATTR_REFERRAL=ignore`

Appendix C: Audit Log

The operator can view and export audit log data on all kinds of user operations related to subscriptions and marketable services including their price models. This may be useful, for example, to check when and by whom a price model was changed, when subscriptions were created, when and by whom a license description was changed.

Single entries of the audit log consist of a **header** and a **message**, separated by a comma.

The header consists of the following elements, separated by a blank:

- `MM/DD/YYYY_hh:mm:ss.SSS`: The local server date and time.
- `FSP_SW/CT-MG_CTMG-BSS`: The predefined log label for log entries resulting from user operations on your platform. This label cannot be changed.
- `INFO`: The default log level. This level cannot be changed.
- `3<nnnn>`: The ID of the operation that was logged. This is a number between 30000 and 39999. The ID is unique for each operation. For example, the ID of the `Define service` operation is 30090, the ID of the `Subscribe to service` operation is 30000.
- `<operation>`: The operation that was logged, for example, `Define service`.

Following the header, the detailed log message is appended. It consists of name-value pairs with additional information, separated by vertical bars (|).

Every message starts with the following name-value pairs:

- `userId=<user ID>`: The ID of the user who executed the operation.
- `orgId=<organization ID>`: The ID of the organization the user belongs to.
- `orgName=<organization name>`: The name of the organization the user belongs to.
- `<additional name-value pairs>`: Additional name-value pairs specific to the logged operation, separated by vertical bars (|).

The log entries are categorized by the role of the user who executed an operation:

- *User Operations* on page 58
- *Administrator Operations* on page 59
- *Service Manager Operations* on page 65

The log entries show the information as stored in the database. If there are updated values, the initial values are not contained in the audit log file.

C.1 User Operations

This section explains the messages that are output in the audit log for operations executed by standard users on a marketplace.

The following `<additional name-value pairs>` are appended to the header and the name-value pairs common to all log entries:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.

Additional name-value pairs may be appended depending on the logged operation.

Operation: **Execute service operation**

Description: A user selected an operation to be executed for the service he subscribed to. The operations and their parameters are defined in the technical service underlying the marketable service.

Additional name-value pairs:

- `serviceOperation=<operation>`: The ID of the service operation executed by the user.
- `<parameter ID>=<parameter value>`: List of operation parameters, separated by vertical bars (|). For every parameter, its ID and its corresponding value are indicated.

C.2 Administrator Operations

This section explains the messages that are output in the audit log for operations executed by an administrator of an organization with any role on a marketplace. Most of these operations can also be executed by OU administrators and subscription managers.

`<additional name-value pairs>` are appended depending on the logged operation.

Operation: **Assign service role**

Description: An administrator, OU administrator, or subscription manager set or changed a service role for a user assigned to a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `user=<user ID>`: The ID of the user who was assigned the service role.
- `userRole=<service role>`: The name of the service role as defined in the underlying technical service.

Operation: **Assign user to organizational unit**

Description: An administrator or OU administrator added one or more users to an organizational unit.

Additional name-value pairs:

- `user=<user ID>`: The IDs of the users who were assigned to the organizational unit, separated by commas.
- `organizationalUnit=<organizational unit name>`: The name of the organizational unit to which the users were added.

Operation: **Assign user to subscription**

Description: An administrator, OU administrator, or subscription manager assigned a user to a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `user=<user ID>`: The ID of the user who was assigned to the subscription.

Operation: **Deassign service role**

Description: An administrator, OU administrator, or subscription manager removed a service role from a user assigned to a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `user=<user ID>`: The ID of the user who was deassigned the service role.
- `userRole=<service role>`: The name of the service role as defined in the underlying technical service.

Operation: **Deassign user from subscription**

Description: An administrator, OU administrator, or subscription manager removed the assignment of a user to a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `user=<user ID>`: The ID of the user who was deassigned from the subscription.

Operation: **Disable access to services**

Description: An administrator or OU administrator specified the services that can no longer be accessed by the members of an organizational unit. For every service, a separate log entry is written.

Additional name-value pairs:

- `organizationalUnit=<organizational unit name>`: The name of the organizational unit whose members cannot access the service.
- `marketplaceId=<marketplace ID>`: The ID of the marketplace on which the members of the organizational unit were able to access the service.
- `marketplaceName=<marketplace name>`: The name of the marketplace on which the members of the organizational unit were able to access the service.
- `serviceID=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `sellerID=<seller ID>`: The organization ID of the supplier, broker, or reseller who offers the service.

Operation: **Edit customer attribute by customer**

Description: An administrator or subscription manager defined or changed an attribute value for a customer attribute. For every attribute value, a separate log entry is written.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `attributeName=<attribute name>`: The name of the customer attribute.
- `attributeValue=<attribute value>`: The value of the customer attribute.

Operation: **Edit subscription attribute by customer**

Description: An administrator, OU administrator, or subscription manager defined or changed an attribute value for a custom subscription attribute. For every attribute value, a separate log entry is written.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `attributeName=<attribute name>`: The name of the custom attribute.
- `attributeValue=<attribute value>`: The value of the custom attribute.

Operation: **Edit subscription owner**

Description: An administrator or OU administrator set or changed the owner of a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `subscriptionOwner=<user ID>`: The ID of the administrator, OU administrator, or subscription manager who was set as the new owner of the subscription.

Operation: **Edit subscription parameter configuration**

Description: An administrator, OU administrator, or subscription manager changed the parameter options for a subscription. For every parameter option, a separate log entry is written.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `parameterName=<option name>`: The name of the parameter option.
- `parameterValue=<option value>`: The new option value. For boolean parameter options, `ON` or `OFF` (`ON`: the option has been selected, `OFF`: the option has not been selected). For enumerations, the value set for the parameter option.

Operation: **Enable access to services**

Description: An administrator or OU administrator specified the services that can be accessed by the members of an organizational unit. For every service, a separate log entry is written.

Additional name-value pairs:

- `organizationalUnit=<organizational unit name>`: The name of the organizational unit whose members can access the service.
- `marketplaceId=<marketplace ID>`: The ID of the marketplace on which the members of the organizational unit can access the service.
- `marketplaceName=<marketplace name>`: The name of the marketplace on which the members of the organizational unit can access the service.
- `serviceID=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `sellerID=<seller ID>`: The organization ID of the supplier, broker, or reseller who offers the service.

Operation: **Remove user from organizational unit**

Description: An administrator or OU administrator removed one or more users from an organizational unit.

Additional name-value pairs:

- `user=<user IDs>`: The IDs of the users who were removed from the organizational unit, separated by commas.
- `organizationalUnit=<organizational unit name>`: The name of the organizational unit from which the users were removed.

Operation: **Report issue**

Description: An administrator, OU administrator, or subscription manager reported an issue on a subscription to the responsible supplier or reseller.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `issue=<text>`: The subject of the issue as entered by the administrator, OU administrator, or subscription manager.

Operation: **Set subscription billing address**

Description: An administrator, OU administrator, or subscription manager set or changed the billing address for a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `address=<display name>`: The name of the billing address as shown to the user.
- `addressDetails=<address details>`: The name of the organization, email, and postal address to which invoices are to be sent.

Operation: **Set subscription payment type**

Description: An administrator, OU administrator, or subscription manager set or changed the payment type for a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `paymentName=<display name>`: The name of the payment type as shown to the user.
- `paymentType=<payment type>`: The payment type name as offered by the supplier.

Operation: **Subscribe to service**

Description: An administrator, OU administrator, or subscription manager completed subscribing to a marketable service offered on a marketplace.

Additional name-value pairs.

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.

Operation: **Unsubscribe from service**

Description: An administrator, OU administrator, or subscription manager terminated a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.

Operation: **Up/downgrade subscription**

Description: An administrator, OU administrator, or subscription manager upgraded or downgraded a subscription.

Additional name-value pairs:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.
- `subscriptionName=<subscription name>`: The name of a subscription as entered when subscribing to a service.
- `newServiceId=<service ID>`: The ID of the marketable service to which the subscription was upgraded or downgraded.
- `newServiceName=<service name>`: The service name for customers of the service to which the subscription was upgraded or downgraded.

C.3 Service Manager Operations

This section explains the messages that are output in the audit log for operations that can be executed in the administration portal by a service manager of an organization with the supplier role. Some of these operations can also be executed by brokers or resellers.

The following `<additional name-value pairs>` are appended to the header and the name-value pairs common to all log entries:

- `serviceId=<service ID>`: The ID of a marketable service as entered during its creation.
- `serviceName=<service name>`: The service name for customers of a marketable service as entered during its creation.

Additional name-value pairs may be appended depending on the logged operation.

Operation: **Activate/deactivate service**

Description: A service manager, broker, or reseller activated or deactivated a marketable service.

Additional name-value pairs:

- `marketplaceId=<ID>`: The ID of the marketplace to which the service is published.
- `marketplaceName=<name>`: The name of the marketplace to which the service is published.
- `activation=<on or off>`: Specifies whether the service was activated or deactivated. Can be `ON` or `OFF`.
- `inCatalog=<on or off>`: Specifies whether the service is to be displayed in the service catalog. Can be `ON` or `OFF`.

Operation: **Assign brokers**

Description: In the publishing options, a service manager authorized a broker to sell the supplier's services.

Additional name-value pair:

`BrokerId=<ID>`: The organization ID of the assigned broker.

Operation: **Assign categories**

Description: In the publishing options, a service manager, broker, or reseller assigned or deassigned one or several categories to/from a marketable service.

Additional name-value pair:

`listOfCategories=<names>`: The categories that are assigned, separated by commas.

Operation: **Assign resellers**

Description: In the publishing options, a service manager authorized a reseller to sell the supplier's services.

Additional name-value pair:

`ResellerId=<ID>`: The organization ID of the assigned reseller.

Operation: **Assign service to marketplace**

Description: In the publishing options, a service manager, broker, or reseller specified or changed the marketplace to which a marketable service is to be published.

Additional name-value pairs:

- `marketplaceId=<ID>`: The ID of the marketplace.
- `marketplaceName=<name>`: The name of the marketplace.

Operation: **Copy service**

Description: A service manager created a copy of a marketable service.

Additional name-value pairs:

- `copyId=<service ID>`: The ID of the marketable service copy as stored in the database.
- `copyName=<service name>`: The name of the marketable service copy as entered when the copy was created.

Operation: **Deassign brokers**

Description: In the publishing options, a service manager removed a broker's right to sell the supplier's services.

Additional name-value pair:

`BrokerId=<ID>`: The organization ID of the deassigned broker.

Operation: **Deassign resellers**

Description: In the publishing options, a service manager removed a reseller's right to sell the supplier's services.

Additional name-value pair:

`ResellerId=<ID>`: The organization ID of the deassigned reseller.

Operation: **Define service**

Description: A service manager defined a marketable service. One additional **Update service parameter** log entry is created for each parameter and parameter option defined in the service.

Additional name-value pairs:

- `technServiceName=<name>`: The name of the technical service the marketable service is based on.
- `shortDescription=<yes or no>`: Specifies whether the short description of the marketable service was entered. Can be `YES` or `NO`.
- `description=<yes or no>`: Specifies whether the description of the marketable service was entered. Can be `YES` or `NO`.
- `locale=<language code>`: The code of the language in which the descriptions were saved.
- `autoAssignUser=<yes or no>`: Specifies whether the user subscribing to the service is automatically assigned to the subscription. Can be `YES` or `NO`.

Operation: **Define up/downgrade options**

Description: A service manager defined options to which a subscription can be upgraded or downgraded, or he removed a service from the list of up/downgrade options.

Additional name-value pairs:

- `targetId=<service ID>`: The ID of the marketable service to which subscriptions can be up/downgraded as stored in the database.
- `targetName=<service name>`: The name of the marketable service to which subscriptions can be up/downgraded.
- `upDownGrade=<on or off>`: Specifies whether the service was added or removed as an up/downgrade option. Can be `ON` or `OFF`.

Operation: **Delete customer price model**

Description: A service manager deleted a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.

Operation: **Delete service**

Description: A service manager deleted a marketable service.

No additional name-value pairs.

Operation: **Edit event price in customer price model**

Description: A service manager defined or changed the price for an event in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `eventName=<name>`: The name of the event as defined in the underlying technical service.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit event price in service price model**

Description: A service manager defined or changed the price for an event in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `eventName=<name>`: The name of the event as defined in the underlying technical service.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit event price in subscription price model**

Description: A service manager defined or changed the price for an event in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `eventName=<name>`: The name of the event as defined in the underlying technical service.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit one time fee in customer price model**

Description: A service manager defined or changed a one-time fee in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `oneTimeFee=<fee>`: The one-time fee for a subscription.

Operation: **Edit one time fee in service price model**

Description: A service manager defined or changed a one-time fee in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `oneTimeFee=<fee>`: The one-time fee for a subscription.

Operation: **Edit customer price model**

Description: A service manager defined or changed the currency, time unit, calculation mode, and/or a free trial period of a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit for recurring charges and per time unit calculation. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `calculationMode=<calculation option>`: The way charges for using a service are calculated. Can be `PRO_RATA` or `PER_UNIT`.
- `trialPeriod=<on or off>`: Specifies whether a free trial period has been defined in the price model. Can be `ON` or `OFF`.
- `daysOfTrial=<Number of days>`: The number of days defined for the free trial period.

Operation: **Edit service price model**

Description: A service manager defined or changed the currency, time unit, calculation mode and/or a free trial period of a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit for recurring charges and per time unit calculation. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `calculationMode=<calculation option>`: The way charges for using a service are calculated. Can be `PRO_RATA` or `PER_UNIT`.
- `trialPeriod=<on or off>`: Specifies whether a free trial period has been defined in the price model. Can be `ON` or `OFF`.
- `daysOfTrial=<Number of days>`: The number of days defined for the free trial period.

Operation: **Edit price model type to free of charge for customer**

Description: A service manager defined or changed a free-of-charge, customer-specific price model so that no costs are charged for using the service.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.

Operation: **Edit price model type to free of charge for service**

Description: A service manager defined or changed a free-of-charge service price model so that no costs are charged for using the service.

No additional name-value pairs.

Operation: **Edit price per subscription parameter in customer price model**

Description: A service manager defined or changed the price per subscription for a parameter or a parameter option in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `range=<value>`: For numeric parameters, stepped prices can be defined. If this is the case, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per subscription parameter in service price model**

Description: A service manager defined or changed the price per subscription for a parameter or a parameter option in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `range=<value>`: For numeric parameters, stepped prices can be defined. If this is the case, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per subscription parameter in subscription price model**

Description: A service manager defined or changed the price per subscription for a parameter or a parameter option in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `range=<value>`: For numeric parameters, stepped prices can be defined. If this is the case, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user parameter in customer price model**

Description: A service manager defined or changed the price per user for a parameter or a parameter option in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user parameter in service price model**

Description: A service manager defined or changed the price per user for a parameter or a parameter option in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user parameter in subscription price model**

Description: A service manager defined or changed the price per user for a parameter or a parameter option in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user role parameter in customer price model**

Description: A service manager defined or changed the price per user with a specific service role for a parameter or a parameter option in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user role parameter in service price model**

Description: A service manager defined or changed the price per user with a specific service role for a parameter or a parameter option in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit price per user role parameter in subscription price model**

Description: A service manager defined or changed the price per user with a specific service role for a parameter or a parameter option in a subscription-specific specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `parameterName=<name>`: The name of the parameter as defined in the underlying technical service.
- `optionName=<name>`: The name of the parameter option as defined in the underlying technical service.
- `price=<price>`: The price as defined in the price model.

Operation: **Edit recurring charge for subscription in customer price model**

Description: A service manager defined or changed the recurring charge per subscription in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `oneTimeFee=<fee>`: The one-time fee for a subscription.
- `recurringCharge=<charge>`: The recurring charge per subscription.

Operation: **Edit recurring charge for subscription in service price model**

Description: A service manager defined or changed the recurring charge per subscription in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `oneTimeFee=<fee>`: The one-time fee for a subscription.
- `recurringCharge=<charge>`: The recurring charge per subscription.

Operation: **Edit recurring charge for subscription in subscription price model**

Description: A service manager defined or changed the recurring charge per subscription in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `oneTimeFee=<fee>`: The one-time fee for a subscription.
- `recurringCharge=<charge>`: The recurring charge per subscription.

Operation: **Edit recurring charge for users in customer price model**

Description: A service manager defined or changed the recurring charge for users in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `recurringCharge=<charge>`: The recurring charge for users.

Operation: **Edit recurring charge for users in service price model**

Description: A service manager defined or changed the recurring charge for users in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `recurringCharge=<charge>`: The recurring charge for users.

Operation: **Edit recurring charge for users in subscription price model**

Description: A service manager defined or changed the recurring charge for users in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `range=<value>`: If stepped prices are defined, this is the step limit up to which the price applies. If no stepped prices are defined, the value is `1-ANY`.
- `action=<action>`: If stepped prices are defined, action that was executed. Can be `INSERT` (when a new stepped price was defined), `UPDATE` (when a stepped price was updated), or `DELETE` (when a stepped price was deleted. If no stepped prices are defined, the `action` parameter is not written to the audit log.
- `recurringCharge=<charge>`: The recurring charge for users.

Operation: **Edit service role prices for customer price model**

Description: A service manager defined or changed the recurring charge for users with a given service role in a customer-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `price=<price>`: The recurring charge for users having the selected role.

Operation: **Edit service role prices for service price model**

Description: A service manager defined or changed the recurring charge for users with a given service role in a service price model.

Additional name-value pairs:

- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `price=<price>`: The recurring charge for users having the selected role.

Operation: **Edit service role prices for subscription price model**

Description: A service manager defined or changed the recurring charge for users with a given service role in a subscription-specific price model.

Additional name-value pairs:

- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `currency=<currency code>`: The ISO code of the currency in which a customer is charged for using the service.
- `timeUnit=<time unit>`: The time unit defined for recurring charges. Can be `MONTH`, `WEEK`, `DAY`, or `HOUR`.
- `userRole=<role name>`: The name of the service role as defined in the underlying technical service.
- `price=<price>`: The recurring charge for users having the selected role.

Operation: **Edit subscription attribute by service manager**

Description: A service manager defined or changed an attribute value for a custom subscription attribute. For every attribute value, a separate log entry is written.

Additional name-value pairs:

- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `attributeName=<attribute name>`: The name of the custom attribute.
- `attributeValue=<attribute value>`: The value of the custom attribute.

Operation: **Localize price model for customer**

Description: A service manager translated the price model elements for a customer into a given language.

Additional name-value pairs:

- `customerKey=<ID>`: The ID of the customer organization.
- `customerName=<name>`: The name of the customer organization.
- `locale=<language code>`: The code of the language in which the texts were saved.
- `description=<yes or no>`: Specifies whether the description of the price model was changed. Can be `YES` or `NO`.
- `license=<yes or no>`: Specifies whether the license information was changed. Can be `YES` or `NO`.

Operation: **Localize price model for service**

Description: A service manager translated the price model elements for a service into a given language.

Additional name-value pairs:

- `locale=<language code>`: The code of the language in which the texts were saved.
- `description=<yes or no>`: Specifies whether the description of the price model was changed. Can be `YES` or `NO`.
- `license=<yes or no>`: Specifies whether the license information was changed. Can be `YES` or `NO`.

Operation: **Localize price model for subscription**

Description: A service manager translated the price model elements for a subscription into a given language.

Additional name-value pairs:

- `customerKey=<ID>`: The ID of the customer organization.
- `customerName=<name>`: The name of the customer organization.
- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `locale=<language code>`: The code of the language in which the texts were saved.
- `description=<yes or no>`: Specifies whether the description of the price model was changed. Can be `YES` or `NO`.
- `license=<yes or no>`: Specifies whether the license information was changed. Can be `YES` or `NO`.

Operation: **Localize service**

Description: A service manager added a service description or short description in another language.

Additional name-value pairs:

- `shortDescription=<yes or no>`: Specifies whether the short description of the marketable service was changed. Can be `YES` or `NO`.
- `description=<yes or no>`: Specifies whether the description of the marketable service was changed. Can be `YES` or `NO`.
- `locale=<language code>`: The code of the language in which the descriptions were saved.

Operation: **Set service as public**

Description: In the publishing options, a service manager, broker, or reseller specified that a marketable service is to be public or that a public marketable service is no longer to be public.

Additional name-value pair:

`public=<true or false>`: Specifies whether the service has been marked as public. Can be `true` (service is public) or `false` (service is not public).

Operation: **Terminate subscription**

Description: A service manager or reseller explicitly terminated a customer's subscription.

Additional name-value pairs:

- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `reason=<termination reason>`: Text explaining the reason for terminating the subscription.

Operation: **Update service**

Description: A service manager changed a marketable service.

Additional name-value pairs:

- `shortDescription=<yes or no>`: Specifies whether the short description of the marketable service was changed. Can be `YES` or `NO`.
- `description=<yes or no>`: Specifies whether the description of the marketable service was changed. Can be `YES` or `NO`.
- `locale=<language code>`: The code of the language in which the descriptions were saved.
- `autoAssignUser=<yes or no>`: Specifies whether the user subscribing to the service is automatically assigned to the subscription. Can be `YES` or `NO`.

Operation: **Update service parameter**

Description: A service manager defined or changed a parameter or parameter option.

Additional name-value pairs:

- `parameterName=<name>`: Name of the parameter as defined in the underlying technical service.
- `userOption=<on or off>`: Specifies whether the parameter is offered as an option to customers who subscribe to the marketable service. Can be `ON` or `OFF`.
- `parameterValue=<value>`: For a parameter, the actual string or number; for a parameter option, this can be `YES` or `NO`.

Operation: **View subscription**

Description: A service manager, broker, or reseller displayed the details of a subscription.

Additional name-value pairs:

- `subscriptionName=<subscription name>`: The name of the subscription as entered when subscribing to a service.
- `customerKey=<organization ID>`: The ID of the customer organization.
- `customerName=<organization name>`: The name of the customer organization.

Appendix D: Language Resource Bundles

A language bundle consists of the following resources:

- **User interface resources**: All texts that appear on the user interface or in generated emails.
- **Online help** and **FAQ** HTML files: The online help topics are available in the administration portal, the FAQs are available on the marketplaces.

This appendix provides details on how to translate the language resources and how to provide them to the system so that a new language is available to users to set it in their user profile. Refer to *Adding a Language to ESCM and Customizing Texts* on page 21 for details on how to add a language to ESCM.

D.1 User Interface Resources

The user interface resources are translated or updated in a Microsoft Excel file generated with the **Export** function on the **Manage languages** page.

The exported Microsoft Excel file contains three worksheets:

- **User interface**: All texts, labels, and messages that appear on the user interface of the platform's administration portal and the marketplaces.
- **Email**: The subject and body texts of all email messages which are generated automatically by the platform.
- **Platform objects**: The names of service parameters and events, report titles and names of payment types provided by the platform. The parameter and event names are visible to suppliers when defining a marketable service as well as to users subscribing to a service.

Each worksheet contains five columns:

- **Key** used by the platform to identify the label or string. The keys must not be changed.
- **de system**, **en system**, **ja system**: The system default names and labels provided with the language bundles after installation (German, English, Japanese). The system default strings are for your reference when providing translations.
- **Add your language code here** or ***<Language Code>***: In this column, the strings of the new language to be supported are to be entered.

  Add your language code here is shown if you did not select a language on the **Manage languages** page before exporting the data. In this case, you usually want to provide a new language. The column is empty.

  If you selected a language on the **Manage languages** page, the column is headed by the code of this language. In this case, you usually update existing translations of the selected language. You can also select the system default languages (de, en, or jp) if you want to customize the system default strings.

  The strings can be also be changed by marketplace managers using the **Marketplace** --> **Customize texts** menu function. Their changes apply to the marketplaces owned by the organization of the marketplace managers.

Proceed as follows to translate the file:

1. **On the three worksheets**, enter the ISO language code of the language you want to provide in the header of the fifth column.
2. For each key, enter your translation in the fifth column.

If a translation is missing, the English label or string is used. If the English label or string is not defined either, the language-independent technical key is displayed.

Be aware of the following:

- You can use HTML markups in the texts for all keys which represent a descriptive text (keys ending with `.description`). For example, you can use `<b>` for bold text, `<br>` for line breaks, and so on.
- You can use the complete Unicode character set.
- You can remove the text from the table cells which you do not want to change. This may be useful for managing and tracking your changes to the user interface.
- You must make sure that HTML fields contain valid data and do not break the page layout. For example, text which is too long will be broken into multiple lines on the user interface. Use a separate test environment for testing your translations.

3. Save the Microsoft Excel file in `.xls` format.
4. Import the file on the **Manage languages** page.

   By importing the file, existing labels and strings in the language with the code entered in the fifth column are overwritten. The language can be used instantly after its activation. Users who have set the language in their profile will instantly see your customizations.

Note: Make sure that all cells on all worksheets of the Excel file are formatted as text.

D.2 Online Help and FAQs

In the administration portal, online help is available in the language the user has set in his profile. The same applies to the FAQs that can be opened on a marketplace.

The online help and FAQ files are provided after installation in a separate container: `oscm-help`.

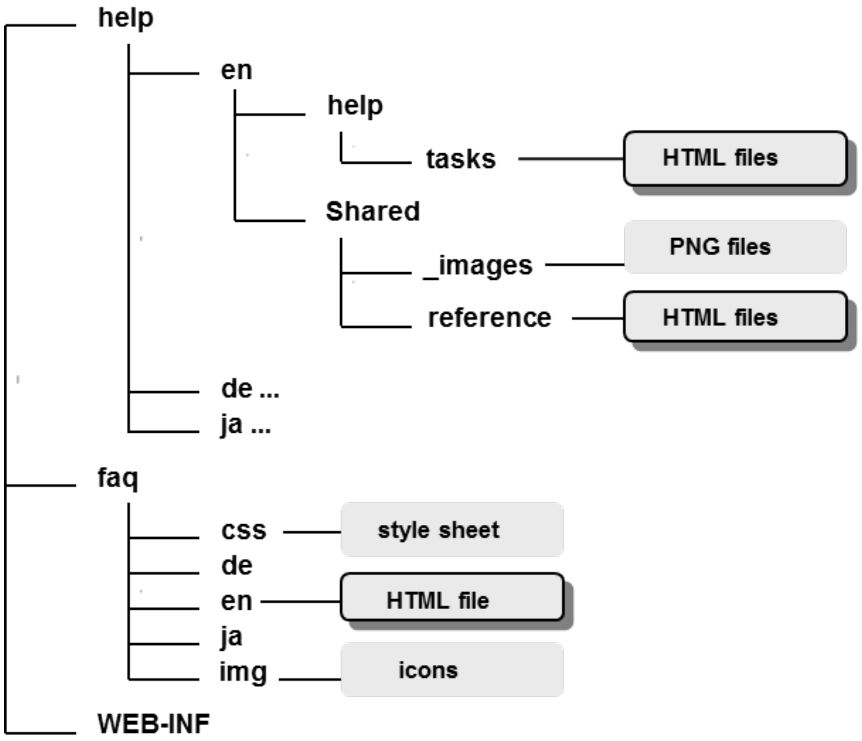
You can update the existing online help and FAQ files and/or provide the files in another language. Refer to *Adding a Language to ESCM and Customizing Texts* on page 21 for details on how to add a language to ESCM.

1. Log in to the Docker host where ESCM is deployed.
2. Log in to the `oscm-help` container:

```
docker exec –it oscm-help /bin/bash
```

3. Copy the `/opt/oscm-portal-help.war` archive file to a temporary directory on the Docker host.
4. Extract the `oscm-portal-help.war` file on the Docker host.

   You see the following directory structure:



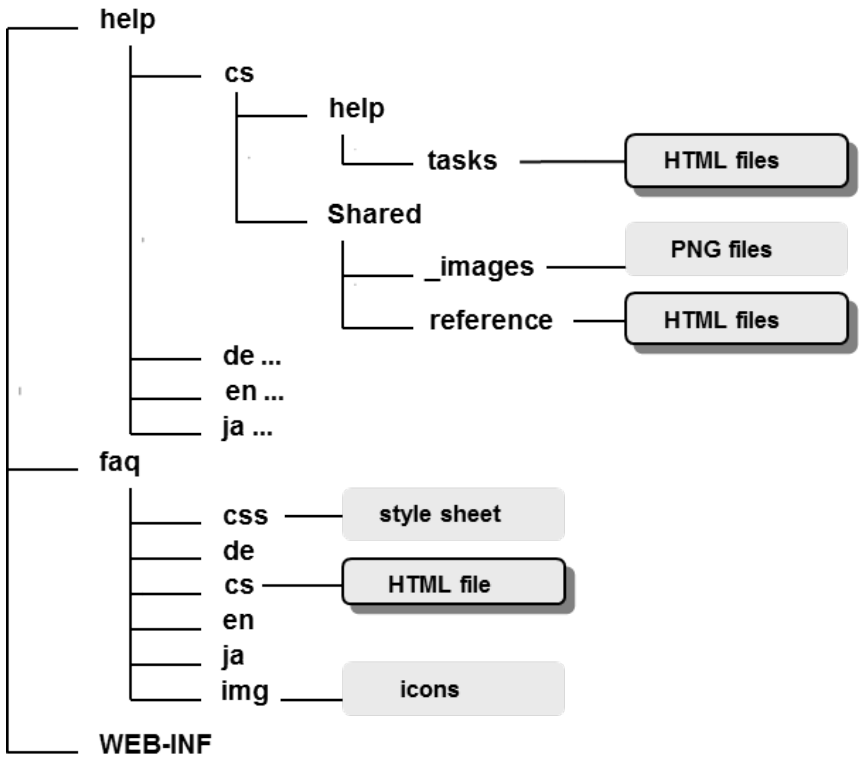
The languages provided after installation are English (en), German (de), and Japanese (ja). You can update the texts in their respective directories. The procedure is the same as when providing the texts in a different language (see below).

5. Copy the `help/en` directory and name it `help/<ISO code>`.

   Copy the `faq/en` directory and name it `faq/<ISO code>`.

   The ISO language code denotes the language for which you want to provide translations. For example, add a `cs` directory to the `help` as well as to the `faq` directory.

   The directory structure now looks as follows:



6. Translate the HTML files.

   Be aware of the following:

   - Make sure not to change any style sheet or icon.
   - Do not remove any files that you do not translate.
   - In the HTML files, make sure not to enter tags or invalid HTML code that may affect the page layout.
   - Use a separate test environment for testing your translations.
   - Images used in the online help topics can be translated as follows:
     1. Create the image with an image drawing tool.
     2. Save the image with the file name referenced in the related HTML file to the `Shared/_images` directory.
     3. Check the size of the image in the HTML topic and adapt it, if required.

7. Create a zipped archive file named `oscm-portal-help.tar.gz` containing the additional subdirectories for the new language, and copy this archive to the following directory on the Docker host:

   `/docker/config/oscm-help`

8. Stop the `oscm-help` container:

```
docker stop oscm- help
```

9. Remove the `oscm-help` container:

```
docker-compose -f docker-compose-oscm.yml rm oscm-help
```

10. Recreate and restart the `oscm-help` container:

```
docker-compose -f docker-compose--oscm.yml up -d oscm-help
```

By the deployment, existing HTML files are overwritten.

Note: You are responsible for keeping the directory structures and files consistent!

D.3 Supported Language Codes

Below is the list of languages and their codes that can be used for user interface and email texts, parameter and event names, as well as online help and FAQs:

Language Code	Language
az	Azerbaijani
be	Belarusian
bg	Bulgarian
bn	Bengali
br	Breton
bs	Bosnian
ca	Catalan
ch	Chamorro
cs	Czech
cy	Welsh
da	Danish
de	German
el	Greek
en	English
es	Spanish
et	Estonian
fi	Finnish
fr	French
gl	Gallegan
gu	Gujarati
hi	Hindi

Language Code	Language
hr	Croatian
hu	Hungarian
ia	Interlingua
in	Indonesian
ii	Sichuan Yi
is	Icelandic
it	Italian
ja	Japanese
ko	Korean
lt	Lithuanian
lv	Latvian
mk	Macedonian
ml	Malayalam
mn	Mongolian
ms	Malay
nb	Norwegian Bokmål
nl	Dutch
nn	Norwegian Nynorsk
no	Norwegian
pl	Polish
pt	Portuguese
ro	Romanian
ru	Russian
sc	Sardinian
se	Northern Sami
si	Sinhalese
sk	Slovak
sl	Slovenian
sq	Albanian
sr	Serbian
sv	Swedish

Language Code	Language
ta	Tamil
te	Telugu
th	Thai
tr	Turkish
tt	Tatar
tw	Twi
uk	Ukrainian
vi	Vietnamese
zh	Chinese

Appendix E: User Data File for Multiple User Import

An administrator can import multiple users of his organization and register them with the platform. A platform operator can import multiple users of his own organization as well as of any organization managed on his platform.

The user data must be provided in a file in `csv` (comma-separated values) format. This file can then be imported into ESCM.

The following rules apply:

- The data for one user is provided in one line. Empty lines are ignored.
- The user data file is saved in UTF-8 encoding.
- The user data is provided in the following sequence:
  1. User ID (mandatory)
  2. Email address (mandatory)
  3. Language (mandatory)
  4. Title (optional)
  5. First name (optional)
  6. Last name (optional)
  7. One or several user roles to be assigned to the imported user (mandatory if no marketplace is selected at the user interface)
- The user data fields are separated by a comma each. If an optional field does not contain any data, it must be empty and separated by a comma from the next field.

The fields can take on the following values:

Field	Value(s)
User ID	Mandatory. ID with which the user is to log in to the platform. User IDs are restricted to 100 characters and must not contain any of the following characters: `! " # $ % & ' * + , / : ; < = > ? \ ^ ` `
Email address	Mandatory. Email address of the user. It is used for notifying the user about the registration. The system checks whether the syntax of the given email address is valid, and whether the domain name corresponds to the standards as defined and maintained by the Internet Assigned Numbers Authority (IANA). Examples: `user.name@domain.arpa`, `user.name@domain.org`, `user@mycompany.lan.uk`
Language	Mandatory. ISO code of the language in which the user will work by default, for example, `en` (English).
Title	Optional. Salutation. Allowed values: `MR` or `MS` Observe that the value is case-sensitive and must be specified as indicated.
First name	Optional.

Field	Value(s)
Last name	Optional.
User role	Mandatory if no marketplace is selected at the user interface. If a marketplace is selected and no role is specified, a standard user without any privileges is registered. The user roles that can be specified depend on the role of the organization for which the users are to be imported. Valid user roles are: `ORGANIZATION_ADMIN` (administrator) for organizations with any role. `SUBSCRIPTION_MANAGER` (subscription manager) for organizations with any role. `PLATFORM_OPERATOR` (operator) for platform operator organizations. `MARKETPLACE_OWNER` (marketplace manager) for marketplace owner organizations. `SERVICE_MANAGER` (service manager) for supplier organizations. `TECHNOLOGY_MANAGER` (technology manager) for technology provider organizations. `BROKER_MANAGER` (broker manager) for broker organizations. `RESELLER_MANAGER` (reseller manager) for reseller organizations. Observe the following: • The role names are case-sensitive and must be specified as indicated above. • If several user roles are specified, they must be separated by a comma and enclosed in double quotes (Example: `"ORGANIZATION_ADMIN, RESELLER_MANAGER"`). Blanks between role names and fields are ignored. • If one user role is specified that is not available for the organization for which to import users, it is ignored and the affected users are not registered. • If several roles are specified for a user, and at least one of these roles is valid for the organization, the user is imported and registered successfully, invalid roles are ignored.

Sample user data file:

```
user1,user1@company.com,en,MS,Jane,Smith,ORGANIZATION_ADMIN
user2,user2@company.com,de,MR,John F.,Cool,SUBSCRIPTION_MANAGER
user3,user3@company.com,en,,,,ORGANIZATION_ADMIN
user4,user4@company.com,en,,,Admin,"SERVICE_MANAGER,ORGANIZATION_ADMIN"
user5,user5@company.com,en,MR,,Mueller-Siegel,SERVICE_MANAGER
user6,user6@company.com,en,MR,Joe,StandardUser,
```

Appendix F: Customer Billing Data

The charges for the usage of a service in ESCM are calculated based on the price model defined for the service, customer, or subscription.

A supplier or reseller can export the billing data for one or several of his customers for a specific time. Suppliers also have access to the billing data of customers of broker organizations that sell their services. The exported data can be forwarded, for example, to an accounting system for further processing.

The result of the export is stored in an XML file, the customer billing data file. The billing data file conforms to the XML schema `BillingResult.xsd`, which can be found in the ESCM integration package.

The billing data file is named `<date>BillingData.xml`, where `<date>` represents the creation date.

This appendix describes the meaning of the elements and attributes that may occur in a billing data file.

BillingDetails

Top-level container element of a billing data file. For each subscription, a `BillingDetails` element is added to the billing data file.

A `BillingDetails` element contains the following subelements:

- `Period` (see *Period* on page 91)
- `OrganizationDetails` (see *OrganizationDetails* on page 92)
- `Subscriptions` (see *Subscriptions* on page 92)
- `OverallCosts` (see *OverallCosts* on page 105)

A `BillingDetails` element has the following attributes:

`key` - (optional, data type `long`) Unique identifier allowing, for example, accounting systems to relate billing data to an invoice. The billing data key is printed on the invoice. Suppliers and customers may use the billing data key to create a detailed billing report for an existing invoice or subscription. A supplier can retrieve the key from a billing report, a customer gets the billing data key from the corresponding invoice.

`timezone` - (required, data type `string`) Time zone based on the UTC time standard. It reflects the standard server time without daylight saving time. For example, 18:00:00 o'clock on June 1st in Berlin (UTC+1) will be output as follows in the XML file:

```
<BillingDetails timezone="UTC+01:00" key="31122">
  <Period startDate="1370106000000"
  startDateIsoFormat="2013-06-01T16:00:00.000Z" ..."/>
```

The time zone is relevant for price models with costs (see *PriceModel* on page 94).

Period

Specifies the billing period for which the data is exported. The start and end time of the billing period are output according to the start day of the billing period which was defined by the supplier or reseller.

A `Period` element has the following attributes:

- **`startDate`** - (data type `long`) Start time of the period. The start time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.

- **startDateIsoFormat** - (optional, data type dateTime) Same as startDate, but specified in ISO 8601 format (YYYY-MM-DDThh:mm:ss.fffZ) .
- **endDate** - (data type long) End time of the period. The end time is specified in milliseconds, the starting point for the calculation is 1970-01-01, 00:00.
- **endDateIsoFormat** - (optional, data type dateTime) Same as endDate, but specified in ISO 8601 format (YYYY-MM-DDThh:mm:ss.fffZ).

Example:

```
<Period startDateIsoFormat="2012-08-31T22:00:00.000Z"
  startDate="1346450400000"
  endDateIsoFormat="2012-09-30T22:00:00.000Z"
  endDate="1349042400000"/>
```

OrganizationDetails

Provides details of the customer for which the billing data have been exported. The details may include a Udas element with custom attributes that store additional information on the customer organization.

An OrganizationDetails element contains the following subelements:

Email

Specifies the email address of the organization (data type string).

Name

Specifies the name of the organization (data type string).

Address

Specifies the address of the organization (data type string).

Paymenttype

Specifies the payment type used for subscriptions of the organization (data type string).

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
  <OrganizationDetails>
    <Email>info@company.com</Email>
    <Name>company</Name>
    <Address>Street</Address>
    <Paymenttype>INVOICE</Paymenttype>
  </OrganizationDetails>
...
</BillingDetails>
```

Subscriptions

Contains the billing data for the subscriptions of the customer which are relevant for the current billing period.

For every subscription of an organization, the Subscriptions element contains a Subscription element. In this element, the costs of the affected subscription are specified.

A Subscription element has the following attributes:

- **id** - (required, data type string) Unique subscription name.

- **purchaseOrderNumber** - (data type string) Optional reference number as specified by the customer when subscribing to a service.

A Subscription element contains a PriceModel element with the billing data for the price model of the subscription (see *PriceModel* on page 94). A Udas element with custom attributes that store additional information on the subscription may also be included (see *Udas* on page 93).

If a subscription is assigned to an organizational unit at the end of the billing period, the Subscription element also contains an OrganizationalUnit element with the following attributes:

- **name** - (required, data type string) Required name of the organizational unit to which the subscription is assigned.
- **referenceID** - (optional, data type string) Optional reference ID of the organizational unit to which the subscription is assigned.

Be aware that the organizational units to which the subscription may have been assigned before the generation of the billing data are not shown.

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
  <Subscriptions>
    <Subscription id="Mega Office Basic" purchaseOrderNumber="12345">
      <OrganizationalUnit name="ProjectTeam" referenceID="123abc"/>
      <PriceModels>
        <PriceModel calculationMode="PRO_RATA" id="14001">
...
        </PriceModel>
      </PriceModels>
    </Subscription>
  </Subscriptions>
...
</BillingDetails>
```

Udas

Contains custom attributes that store additional information on an organization or subscription. This could be, for example, the profit center to which a customer's revenue is to be accounted.

A Udas element may be included in an OrganizationDetails or a Subscription element.

For every custom attribute, a Uda element is included in the Udas element.

A Uda element has the following attributes:

- **id** - (required, data type string) ID of the custom attribute.
- **value** - (required, data type string) Value of the custom attribute.

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
  <Subscriptions>
    <Subscription id="Mega Office Basic" purchaseOrderNumber="12345">
...
      <Udas>
        <Uda id="Profit Center" value="My Company"/>
      </Udas>
    </Subscription>
  </Subscriptions>
```

```
...
</BillingDetails>
```

PriceModel

Contains the billing data for a price model used to calculate the utilization charges for a subscription.

A `PriceModel` element is included in every subscription element. It contains the following subelements:

- `UsagePeriod` (see *UsagePeriod* on page 94)
- `GatheredEvents` (see *GatheredEvents* on page 95)
- `PeriodFee` (see *PeriodFee* on page 96)
- `UserAssignmentCosts` (see *UserAssignmentCosts* on page 97)
- `OneTimeFee` (see *OneTimeFee* on page 98)
- `PriceModelCosts` (see *PriceModelCosts* on page 98)
- `Parameters` (see *Parameters* on page 99)

A `PriceModel` element has the following attributes:

`id` - (required, data type `string`) Unique name identifying the price model.

`calculationMode` - (required, data type `string`) Cost calculation option of the price model. Can be set to one of the following values: `FREE_OF_CHARGE` (the service is free of charge), `PRO_RATA` (the costs are calculated exactly for the time a service is used, based on milliseconds), `PER_UNIT` (the costs are calculated based on fixed time units).

UsagePeriod

Specifies the actual period in which a price model is used for calculating the charges of a subscription.

A usage period usually begins when a customer subscribes to a service and ends when the subscription is terminated. In case a free trial period is defined for the service, the usage period begins when the free trial period has ended. When the customer upgrades or downgrades the subscription, a new usage period is started in which the price model of the new service is applied. If the customer changes elements that determine how the charges for the service are calculated (e.g. the number of assigned users, the service roles of the assigned users, or the parameter values), a new usage period is started in which the updated elements are applied.

A `UsagePeriod` element is contained in a `PriceModel` element.

A `UsagePeriod` element has the following attributes:

- **`startDate`** - (data type `long`) Start time of the period. The start time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`startDateIsoFormat`** - (optional, data type `dateTime`) Same as `startDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`) .
- **`endDate`** - (data type `long`) End time of the period. The end time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`endDateIsoFormat`** - (optional, data type `dateTime`) Same as `endDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`).

Example:

```
<PriceModel calculationMode="PRO_RATA" id="14001">
```

```
  <UsagePeriod endDate="1306879200000"
   endDateIsoFormat="2011-05-31T22:00:00.000Z"
   startDate="1304755088065"
   startDateIsoFormat="2011-05-07T07:58:08.065Z"/>
...
</PriceModel>
```

GatheredEvents

Specifies the costs for all chargeable events that occurred in the current usage period of the subscription. These include, for example, login and logout by users to the underlying application, the completion of specific transactions, or the creation or deletion of specific data. It depends on the implementation and integration of the underlying application which events are available.

A `GatheredEvents` element is contained in a `PriceModel` element.

A `GatheredEvents` element contains the following subelements:

- `Event`
- `GatheredEventsCosts`

Event

For every event, an `Event` element is included in the `GatheredEvents` element.

An `Event` element has the following attribute:

`id` - (required, data type `string`) Event ID as specified in the technical service definition.

An `Event` element contains the following subelements:

- `Description`
- `SingleCost`
- `NumberOfOccurence`
- `CostForEventType`

Description

Contains the description of the event.

SingleCost

Specifies the price for the event as defined in the price model. If an event has stepped prices, this element is omitted. A `SteppedPrices` element is included instead (see *SteppedPrices* on page 104).

A `SingleCost` element has the following attribute:

`amount` - (required, data type `decimal`) Price for a single event.

NumberOfOccurence

Specifies how often the event occurred.

A `NumberOfOccurence` element has the following attribute:

`amount` - (required, data type `long`) Number of times the event occurred.

CostForEventType

Specifies the total costs for the event in the billing period.

A `CostForEventType` element has the following attribute:

`amount` - (required, data type `decimal`) Total costs for the event. The total costs for an event are calculated from the singular costs (`SingleCost`) multiplied with the number of occurrences (`NumberOfOccurence`). If role-based costs and/or stepped prices are specified for events, these

costs are added (see *RoleCosts* on page 103 and *SteppedPrices* on page 104). The value is rounded to two decimal places.

GatheredEventsCosts

Specifies the total costs for all events in the current `GatheredEvents` element.

A `GatheredEventsCosts` element has the following attribute:

`amount` - (required, data type `decimal`) Total costs for events. The value is rounded to two decimal places.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="14001">
...
  <GatheredEvents>
    <Event id="USER_LOGOUT_FROM_SERVICE">
      <Description xml:lang="en">Logout from the service.</Description>
      <SingleCost amount="100.00"/>
      <NumberOfOccurrence amount="3"/>
      <CostForEventType amount="300.00"/>
    </Event>
...
    <GatheredEventsCosts amount="1200.00"/>
  </GatheredEvents>
...
</PriceModel>
```

PeriodFee

Specifies the costs for using the subscription in the given usage period.

For each subscription, a charge can be defined that a customer has to pay on a recurring basis. Monthly, weekly, daily, or hourly periods are supported. The recurring charge for a subscription is independent of the amount of users, events, or other usage data.

The calculation of the charges depends on the cost calculation option which was chosen for the price model (see *PriceModel* on page 94 for details).

A `PeriodFee` element is contained in a `PriceModel` element.

A `PeriodFee` element has the following attributes:

- **`basePeriod`** - (required, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (required, data type `decimal`) Recurring charge per base period according to the price model.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the period fee for the subscription. The factor is calculated from the usage period of the subscription divided by the base period (`basePeriod`). The recurring charge is multiplied with this factor to calculate the total costs (`price`).
- **`price`** - (required, data type `decimal`) Total period fee for the subscription. This value is calculated from the recurring charge (`basePrice`) multiplied with the factor (`factor`). The value is rounded to two decimal places.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="14001">
...
  <PeriodFee basePeriod="MONTH" basePrice="10.00"
 factor="0.4020212567204301" price="4.02"/>
```

```
...
</PriceModel>
```

UserAssignmentCosts

Specifies the costs for the user assignments to the subscription.

For the users assigned to a subscription, a charge can be defined that a customer has to pay on a recurring basis. Monthly, weekly, daily, or hourly periods are supported. The charge depends on the amount of time units one or more users are assigned to the subscription. This type of charge can only be defined for services with the login or user access type.

The recurring charge for users is independent of the recurring charge per subscription or other usage data.

For this type of charge, stepped prices can be applied: Recurring charges can be defined that depend on the sum of the time units of all user assignments.

The calculation of the charges depends on the cost calculation option which was chosen for the price model (see *PriceModel* on page 94 for details).

With per time unit calculation, the costs for a time unit in which a user is assigned to a subscription are always fully charged. There is no difference in the costs between a user who is assigned from the start until the end of the time unit and a user who is assigned for a part of the time unit only. A time unit is charged only once if a user is deassigned from and re-assigned to a subscription within the same time unit. Yet, canceling an assignment, deleting the user, and then creating a new user with the same user ID is treated as if two different users are assigned to the subscription. The time unit is charged twice, accordingly.

A `UserAssignmentCosts` element is contained in a `PriceModel` element.

A `UserAssignmentCosts` element has the following attributes:

- **`basePeriod`** - (optional, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (optional, data type `decimal`) Recurring charge for users per base period according to the price model. If the charge for users has stepped prices, this attribute is omitted.
- **`factor`** - (optional, data type `decimal`) Factor used to calculate the costs for the user assignments. The factor is calculated by summing up the factors for each user specified in the `UserAssignmentCostsByUser` element. The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`).
- **`numberOfUsersTotal`** - (optional, data type `long`) Number of users assigned to the subscription in the usage period.
- **`total`** - (data type `decimal`) Total costs for the user assignments including role-based costs and stepped prices. The value is rounded to two decimal places. For details on role-based costs and stepped prices, refer to *RoleCosts* on page 103 and *SteppedPrices* on page 104.
- **`price`** - (optional, data type `decimal`) Costs for the user assignments. This value is calculated from the recurring charge (`basePrice`) multiplied with the factor (`factor`). The value is rounded to two decimal places.

A `UserAssignmentCosts` element contains the following subelement. If role-based costs and/or stepped prices are specified, a `RoleCosts` element and/or `SteppedPrices` element is also present (see *RoleCosts* on page 103 and *SteppedPrices* on page 104).

UserAssignmentCostsByUser

Specifies the fraction of the usage period a user was assigned to the subscription.

A `UserAssignmentCostsByUser` element has the following attributes:

- **`factor`** - (required, data type `decimal`) Fraction of the usage period the given user was assigned to the subscription. The factors of the single user assignments are summed up to calculate the total costs for the user assignments.
- **`userId`** - (required, data type `string`) User ID.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="18000">
...
  <UserAssignmentCosts basePeriod="MONTH" basePrice="19.00"
   factor="0.5337726052867383" numberOfUsersTotal="2"
   total ="50.00" price="10.14">
     <UserAssignmentCostsByUser factor="1.0499215949820788E-4"
     userId="admin"/>
     <UserAssignmentCostsByUser factor="0.5336676131272401"
     userId="miller"/>
   </UserAssignmentCosts>
...
</PriceModel>
```

OneTimeFee

Specifies the one-time fee for the subscription.

A one-time fee defines the amount a customer has to pay for a subscription in the first billing period of the subscription. It is added to the total charges for the first billing period. It is independent of the number of users, events, or other usage data.

If a one-time fee is defined for a service to which a customer upgrades or downgrades a subscription, it is added to the total charges for the customer, even if the service from which the customer migrates also defines a one-time fee.

A `OneTimeFee` element is contained in a `PriceModel` element.

A `OneTimeFee` element has the following attributes:

- **`amount`** - (required, data type `decimal`) Total costs for the one-time fee. The value is rounded to two decimal places.
- **`baseAmount`** - (required, data type `decimal`) One-time fee as defined in the price model.
- **`factor`** - (required, data type `long`) Factor used for calculating the one-time fee. Since this charge occurs only once, the factor is 1 for the first billing period, and 0 in case the one-time fee has already been charged in a previous billing period.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="14001">
...
  <OneTimeFee amount="10.00" baseAmount="10.00" factor="1"/>
...
</PriceModel>
```

PriceModelCosts

Specifies the total costs for the subscription in the current usage period.

A `PriceModelCosts` element is contained in a `PriceModel` element.

A `PriceModelCosts` element has the following attributes:

- **`currency`** - (required, data type `string`) ISO code of the currency in which the costs are calculated.
- **`amount`** - (required, data type `decimal`) Total amount of the costs for the subscription. The value is rounded to two decimal places.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="14001">
...
  <PriceModelCosts currency="EUR" amount="990.00"/>
</PriceModel>
```

Parameters

Specifies the costs for parameters defined for the service underlying the subscription.

A price model can define prices for service parameters and options. It depends on the implementation and integration of the underlying application whether and which parameters and options are available.

A price can be defined for every parameter and option, and the price can be charged per subscription or per user assigned to the subscription. Numeric parameters are a multiplier for the price. For boolean parameters, the multiplier is `1` if the value is `true`. In all other cases, the multiplier is `0`.

The calculation of charges for parameters and options depends on the cost calculation option which was chosen for the price model (see *PriceModel* on page 94 for details).

If the charges for a subscription are calculated per time unit and a customer changes a parameter value within a time unit, the affected time unit is charged pro rata. This means that the customer is charged exactly for the time each parameter value is set.

For numeric parameters, stepped prices can be applied per subscription: Different prices can be defined depending on the parameter values.

The prices for parameters and options are independent of other price model elements.

A `Parameters` element is contained in a `PriceModel` element.

A `Parameters` element contains the following subelements:

- `Parameter`
- `ParametersCosts`

Parameter

For every parameter, a `Parameter` element is included in the `Parameters` element.

A `Parameter` element has the following attribute:

`id` - (required, data type `string`) Parameter ID.

A `Parameter` element contains the following subelements:

- `ParameterUsagePeriod`
- `ParameterValue`
- `PeriodFee`
- `UserAssignmentCosts`
- `ParameterCosts`
- `Options`

ParameterUsagePeriod

Specifies the actual usage period for the parameter.

The usage period for a parameter begins when a customer subscribes to the service with the given parameter definition in the price model and ends when the subscription is terminated.

In case a free trial period is defined for the service, the usage period for the subscription begins when the free trial period has ended. If a parameter value is changed, a new usage period is started in which the updated value is applied for calculating the costs.

A `ParameterUsagePeriod` element has the following attributes:

- **`startDate`** - (data type `long`) Start time of the period. The start time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`startDateIsoFormat`** - (optional, data type `dateTime`) Same as `startDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`) .
- **`endDate`** - (data type `long`) End time of the period. The end time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`endDateIsoFormat`** - (optional, data type `dateTime`) Same as `endDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`).

ParameterValue

Specifies the costs and data type for the parameter.

A `ParameterValue` element has the following attributes:

- **`amount`** - (required, data type `string`) Costs for the parameter as defined in the price model.
- **`type`** - (required, data type `string`) Data type of the parameter. Can be set to one of the following values: `BOOLEAN`, `INTEGER`, `LONG`, `STRING`, `ENUMERATION`, `DURATION`.

PeriodFee

Specifies the costs for using the parameter in the given usage period. If a parameter has stepped prices, a `SteppedPrices` element is included in the `PeriodFee` element.

A `PeriodFee` element has the following attributes:

- **`basePeriod`** - (required, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (optional, data type `decimal`) Recurring charge per base period according to the price model. If a parameter has stepped prices, this attribute is omitted.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the costs for using the parameter. The factor is calculated from the usage period divided by the base period (`basePeriod`). The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`).
- **`price`** - (required, data type `decimal`) Costs for using the parameter. This value is calculated from the recurring charge (`basePrice`) multiplied with the factors (`factor` and `valueFactor`). The value is rounded to two decimal places.
- **`valueFactor`** - (required, data type `float`) Factor to calculate the total costs for using the parameter depending on the parameter value. The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`). This factor is set depending on the data type of the parameter. For numeric parameters it is set to the value of the parameter. For boolean parameters, the factor is set to `1` if the value is `true`. In all other cases, the factor is set to `0`.

UserAssignmentCosts

Specifies the costs for the parameter related to the user assignments of the subscription based on the price per user for the parameter as defined in the price model. If costs for service roles are

defined, a `RoleCosts` element is included in the `UserAssignmentCosts` element (see *RoleCosts* on page 103 for details).

A `UserAssignmentCosts` element has the following attributes:

- **`basePeriod`** - (required, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (required, data type `decimal`) Recurring charge for users per base period for the parameter according to the price model.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the costs for using the parameter. The factor is calculated from the parameter usage period divided by the base period (`basePeriod`) multiplied with the number of users. The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`).
- **`price`** - (required, data type `decimal`) Costs for using the parameter. This value is calculated from the recurring charge (`basePrice`) multiplied with the factors (`factor` and `valueFactor`). The value is rounded to two decimal places. If stepped prices are defined for user assignments, the costs are given in the `price` attribute.
- **`total`** - (data type `decimal`) Total costs for using the parameter including role-based costs. The value is rounded to two decimal places. For details on role-based costs, refer to *RoleCosts* on page 103.
- **`valueFactor`** - (required, data type `float`) Factor to calculate the total costs for using the parameter depending on the parameter value. The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`). This factor is set depending on the data type of the parameter. For numeric parameters it is set to the value of the parameter. For boolean parameters, the factor is set to `1` if the value is `true`. In all other cases, the factor is set to `0`.

ParameterCosts

Specifies the total costs for using the parameter.

A `ParameterCosts` element has the following attribute:

`amount` - (required, data type `decimal`) Total costs for the parameter calculated by summing up the costs specified in the `PeriodFee` and the `UserAssignmentCosts` element for the parameter and its options. If role-based costs and/or stepped prices are specified for the parameter, these are added (see *RoleCosts* on page 103 and *SteppedPrices* on page 104). The value is rounded to two decimal places.

ParametersCosts

Specifies the total costs for all parameters.

A `ParametersCosts` element has the following attribute:

`amount` - (required, data type `decimal`) Total costs for the parameters calculated by summing up the costs of the individual parameters as specified in the `ParameterCosts` elements. The value is rounded to two decimal places.

Example:

```
<Parameters>
...
  <Parameter id="MAX_FOLDER_NUMBER2">
    <ParameterUsagePeriod endDate="1306879200000"
     endDateIsoFormat="2011-05-31T22:00:00.000Z"
     startDate="1304755088065"
     startDateIsoFormat="2011-05-07T07:58:08.065Z"/>
    <ParameterValue amount="200" type="INTEGER"/>
    <PeriodFee basePeriod="MONTH" basePrice="0.00"
     factor="0.5337789669205496" price="0.00" valueFactor="200.0"/>
```

```
    <UserAssignmentCosts basePeriod="MONTH" basePrice="0.00"
     factor="0.5337726052867383" total ="0.00"
     price="0.00" valueFactor="200.0"/>
    <ParameterCosts amount="0.00"/>
  </Parameter>

...
  <ParametersCosts amount="600.00"/>
</Parameters>
```

Options

Specifies the costs for parameter options.

An `Options` element is contained in a `Parameter` element.

For every option, an `Option` element is included in the `Options` element.

An `Option` element has the following attribute:

`id` - (required, data type `string`) Option ID.

An `Option` element contains the following subelements:

- `PeriodFee`
- `UserAssignmentCosts`
- `OptionCosts`

PeriodFee

Specifies the costs for using the parameter option in the given usage period.

A `PeriodFee` element has the following attributes:

- **`basePeriod`** - (required, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (required, data type `decimal`) Recurring charge per base period according to the price model.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the costs for using the parameter option. The factor is calculated from the usage period divided by the base period (`basePeriod`). The recurring charge (`basePrice`) is multiplied with this factor to calculate the total costs (`price`).
- **`price`** - (required, data type `decimal`) Costs for the parameter option. This value is calculated from the recurring charge (`basePrice`) multiplied with the factor (`factor`), and is rounded to two decimal places.

UserAssignmentCosts

Specifies the costs for the parameter option related to the user assignments of the subscription based on the price per user for the option as defined in the price model. If costs for service roles are defined, a `RoleCosts` element is included in the `UserAssignmentCosts` element (see *RoleCosts* on page 103).

A `UserAssignmentCosts` element has the following attributes:

- **`basePeriod`** - (required, data type `string`) Period on which the charges are based. Can be set to one of the following values: `MONTH`, `WEEK`, `DAY`, `HOUR`.
- **`basePrice`** - (required, data type `decimal`) Recurring charge for users per base period for the parameter option according to the price model.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the costs for using the parameter option. The factor is calculated from the usage period divided by the base period

(basePeriod). The recurring charge (basePrice) is multiplied with this factor to calculate the costs (price).

- **total** - (data type decimal) Total costs for using the parameter option including role-based costs. The value is rounded to two decimal places. For details on role-based costs, refer to *RoleCosts* on page 103.
- **price** - (required, data type decimal) Costs for using the parameter option. This value is calculated from the recurring charge (basePrice) multiplied with the factor (factor). The value is rounded to two decimal places. If stepped prices are defined for user assignments, the costs are given in the price attribute.

OptionCosts

Specifies the total costs for using the parameter option.

An OptionCosts element has the following attribute:

amount - (required, data type decimal) Total costs for the parameter option calculated by summing up the costs specified in the PeriodFee and the UserAssignmentCosts element. The value is rounded to two decimal places.

Example:

```
<Parameter id="MEMORY_STORAGE">
...
  <Options>
    <Option id="2">
      <PeriodFee basePeriod="MONTH" basePrice="100.00"
       factor="1.0" price="100.00"/>
      <UserAssignmentCosts basePeriod="MONTH" basePrice="0.00"
       factor="1.0" total ="0.00" price="0.00"/>
      <OptionCosts amount="100.00"/>
    </Option>
  </Options>
...
</Parameter>
```

RoleCosts

Specifies the costs for service roles.

If defined for the underlying application, roles can be used to grant specific privileges to different users. The roles are specified in the technical service definition as service roles. Service roles can be mapped to corresponding permissions in the application.

For each role, a price can be defined. This price is added to the base price per user in the cost calculation for a billing period.

The calculation of the charges for service roles depends on the cost calculation option which was chosen for the price model (see *PriceModel* on page 94 for details).

If the charges are calculated per time unit and the role assignment of a user is changed within a time unit, the affected time unit is charged pro rata. This means that the customer is charged exactly for the time each user role is assigned.

If the charges are calculated per time unit and a user with a specific role is removed from the subscription and assigned to it again with a different role in the same time unit, the customer is also charged for the time during which the user is not assigned to the subscription. This means that he is charged with the price for the first service role until the user is assigned to the subscription with the second service role.

A `RoleCosts` element is contained in a `UserAssignmentCosts` element (as subelement of the `PriceModel`, `Parameters`, or `Option` element).

A `RoleCosts` element has the following attribute:

`total` - (required, data type `decimal`) Total amount of costs for the service roles. The value is rounded to two decimal places.

For every service role, a `RoleCost` element is included in the `RoleCosts` element.

A `RoleCost` element has the following attributes:

- **`id`** - (required, data type `string`) ID of the service role.
- **`basePrice`** - (required, data type `decimal`) Recurring charge for the service role according to the price model.
- **`factor`** - (required, data type `decimal`) Factor used to calculate the costs for the service role. The factor is calculated as a fraction of the actual usage period. The recurring charge (`basePrice`) is multiplied with this factor to calculate the costs (`price`).
- **`price`** - (required, data type `decimal`) Costs for the service role. This value is calculated from the recurring charge (`basePrice`) multiplied with the factor (`factor`). The value is rounded to two decimal places.

Example:

```
<Parameter id="MEMORY_STORAGE">
...
  <RoleCosts total="0.00">
    <RoleCost basePrice="0.00" factor="0.4020087753882915"
     id="USER" price="0.00"/>
    <RoleCost basePrice="0.00" factor="0.8040175186678614"
     id="ADMIN" price="0.00"/>
  </RoleCosts>
...
</Parameter>
```

SteppedPrices

Specifies the stepped prices for a user assignment, event, or parameter.

Stepped prices allow for the definition of ranges for which different price factors apply. Step limits, i.e. the upper limits of ranges, can be set for:

- The **sum of the time units** users are assigned and work with a subscription in a billing period. For example, up to 10 hours one user is assigned to a subscription cost 10.00 € per hour, every additional hour the user is assigned costs 8.00 €.
- The **number of events** occurring in the usage of a subscription. For example, up to 10 file downloads cost 1.00 € per download, any additional download costs 0.50 €.
- **Values of numeric parameters**. For example, uploading up to 100 files costs 1.00 € per file, any additional upload costs 0.50 € per file.

Stepped prices are independent of any other price model elements.

A `SteppedPrices` element is contained in `UserAssignmentCosts` (as subelement of the `PriceModel` element), `Event`, and `PeriodFee` (as subelement of the `Parameters` element) elements.

A `SteppedPrices` element has the following attribute:

`amount` - (required, data type `decimal`) Summed up costs for all steps including the last one.

For every price step, a `SteppedPrice` element is included in a `SteppedPrices` element.

A `SteppedPrice` element has the following attributes:

- **`additionalPrice`** - (required, data type `decimal`) Summed up costs for the previous steps. The costs are calculated from the `limit`, `freeAmount` and `basePrice` attributes of the previous step (`(limit - freeAmount) * price`). The `additionalPrice` attribute of the first step always has a value of `0`. The value is rounded to two decimal places.
- **`basePrice`** - (required, data type `decimal`) Costs for the current step according to the price model.
- **`freeAmount`** - (required, data type `long`) Amount of units for the current step that are considered as a fixed discount, for example, the number of users that are free of charge. The value corresponds to the value of the `limit` attribute in the previous step. The `freeAmount` attribute of the first step always has a value of `0`.
- **`limit`** - (required, data type `string`) Step limit as defined in the price model.
- **`stepAmount`** - (optional, data type `decimal`) Summed up costs for the current step. These costs are calculated from the `basePrice` and `stepEntityCount` attributes of the current step. The value is rounded to two decimal places.
- **`stepEntityCount`** - (optional, data type `decimal`) Factor used to calculate the costs for the current step.

Example:

```
<PriceModel calculationMode="PRO_RATA" id="350001">
...
  <UserAssignmentCosts basePeriod="MONTH" factor="2.707940780619112"
 numberOfUsersTotal="4" price="1283.18">
    <SteppedPrices amount="1283.18">
      <SteppedPrice additionalPrice="0.00" basePrice="500.00"
       freeAmount="0" limit="2" stepAmount="1000.00"
 stepEntityCount="2"/>
      <SteppedPrice additionalPrice="1000.00" basePrice="400.00"
       freeAmount="2" limit="3" stepAmount="283.18"
       stepEntityCount="0.707940780619112"/>
      <SteppedPrice additionalPrice="1400.00" basePrice="300.00"
       freeAmount="3" limit="null" stepAmount="0.00"
 stepEntityCount="0"/>
    </SteppedPrices>
  </UserAssignmentCosts>
...
</PriceModel>
```

OverallCosts

Contains the total amount of the charges to be paid by a customer for all subscriptions in the current billing period. The costs are given in the currency specified in the price model.

If a discount was specified, the net amount of the costs is given in the `Discount` element (see *Discount* on page 107). If a VAT rate was defined, it is given in the `VAT` element (see *VAT* on page 106).

An `OverallCosts` element has the following attributes:

- **`netAmount`** - (required, data type `decimal`) Net costs after the net discount has been deducted from the original net costs (see *Discount* on page 107). The value is rounded to two decimal places.
- **`currency`** - (required, data type `string`) ISO code of the currency in which the costs are calculated.

- **grossAmount** - (required, data type `decimal`) Gross amount of the costs, calculated from the net costs (`netAmount`) plus VAT (see *VAT* on page 106). The value is rounded to two decimal places.

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
<OverallCosts netAmount="900.00" currency="EUR" grossAmount="1053.00"/>

</OverallCosts>
</BillingDetails>
```

VAT

Specifies the VAT rate to be applied.

A supplier can define a basic VAT rate that applies by default to all prices for his customers. In addition to this basic VAT rate, country-specific or even customer-specific VAT rates can be defined. You can:

- Enable VAT rate support for your organization.
- Define a default VAT rate that applies to all prices for all customers.
- Define a country-specific VAT rate for every country where you want to sell your services.
- Define a customer-specific VAT rate, for example, in case a customer organization has a subsidiary located in another country than its parent organization.

The VAT rate settings have the following effects on the cost calculation for a customer:

- If VAT rate support is disabled, prices are calculated as net prices; no VAT is added to the overall costs.
- A customer-specific VAT rate takes priority over any default or country-specific VAT rate.
- The country-specific VAT rate for the country where the customer organization is located is applied to the cost calculation when no customer-specific VAT rate is defined.
- The default VAT rate is used in all other cases.

The VAT rate does not affect any price model elements. The calculated VAT amount is added to the overall costs and results in the gross price to be paid by a customer.

A `VAT` element is contained in the `OverallCosts` element.

A `VAT` element has the following attributes:

- **percent** - (required, data type `float`) VAT rate in percent, specified as a decimal number.
- **amount** - (required, data type `decimal`) Net amount of VAT to be added to the net costs (`netAmount` attribute of the `OverallCosts` element). The value is rounded to two decimal places.

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
  <OverallCosts netAmount="900.00" currency="EUR" grossAmount="1053.00">
    <VAT percent="17.0" amount="153.00"/>
  </OverallCosts>
</BillingDetails>
```

Discount

Specifies the discount granted to the customer.

A discount can be defined for a customer which applies to all subscriptions of the customer to services. A discount may be valid as of the current or a future month. It can be restricted to a certain period of time. Before the time expires, the customer is notified by email so that he can react and contact the supplier.

The discount is defined as a percentage that is subtracted from the regular total price for a subscription. It is granted for all costs of a customer that incur in a billing period in which the discount is valid. It does not matter whether the discount is valid for the whole billing period or only a part of it.

A discount is completely independent of what a customer might purchase. If a discount is changed, the new discount is valid the next time the billing data is generated. Usually, a discount is only changed in agreement with the relevant customer.

A `Discount` element is contained in the `OverallCosts` element.

A `Discount` element has the following attributes:

- **`percent`** - (required, data type `float`) Percentage of costs to be deducted from the net costs, specified as a decimal number.
- **`discountNetAmount`** - (required, data type `decimal`) Net discount to be deducted from the original net costs (`netAmountBeforeDiscount`). The value is rounded to two decimal places.
- **`netAmountAfterDiscount`** - (required, data type `decimal`) Net costs after the net discount (`discountNetAmount`) has been deducted from the original net costs (`netAmountBeforeDiscount`). The value is rounded to two decimal places.
- **`netAmountBeforeDiscount`** - (required, data type `decimal`) Net costs before the net discount (`discountNetAmount`) has been deducted. The value is rounded to two decimal places.

Example:

```
<BillingDetails key="10002" timezone="UTC+01:00">
...
  <OverallCosts netAmount="900.00" currency="EUR" grossAmount="1053.00">
    <Discount percent="10.00" discountNetAmount="100.00"
     netAmountAfterDiscount="900.00"
     netAmountBeforeDiscount="1000.00" />
    <VAT percent="17.0" amount="153.00"/>
  </OverallCosts>
</BillingDetails>
```

Appendix G: Revenue Share Data

In extended usage scenarios, suppliers may involve brokers and resellers in selling their services. The brokers and resellers as well as the platform operator and the owners of the marketplaces on which the services are published, usually receive a share of the revenue for the services. ESCM calculates these revenue shares based on the billing data for the customers who use the services.

Suppliers, brokers, resellers, and marketplace owners can generate reports for their revenue shares and export the revenue share data for a specific time. Operators can export the data for all the suppliers, brokers, resellers, or marketplace owners known to their platform installation. The exported data can be forwarded, for example, to an accounting system for further processing.

The result of the export is stored in an XML file, the revenue data file. The file conforms to one of the following schemas, depending on the type of the revenue share data:

- `BrokerRevenueShareResult.xsd` : revenue share data for brokers
- `ResellerRevenueShareResult.xsd` : revenue share data for resellers
- `MPOwnerRevenueShareResult.xsd` : revenue share data for marketplace owners
- `SupplierRevenueShareResult.xsd` : revenue share data for suppliers

Each of these schemas includes the `BillingBase.xsd` with common definitions. All the schemas can be found in the ESCM integration package.

The XML files containing the revenue share data are named `<date>BillingData.xml`, where `<date>` represents the creation date.

This appendix describes the meaning of the elements and attributes that may occur in the different types of revenue share data file. The first section explains elements and attributes that are common to all revenue share data files. The subsequent sections describe the individual files.

G.1 Common Elements

The sections below describe the following elements that are common to all revenue share data files:

- `Period`
- `OrganizationData`

Period

Specifies the billing period for which the data is exported. The start and end time of the billing period are output according to the start day of the billing period which was defined by the supplier or reseller.

A `Period` element has the following attributes:

- **`startDate`** - (data type `long`) Start time of the period. The start time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`startDateIsoFormat`** - (optional, data type `dateTime`) Same as `startDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`) .
- **`endDate`** - (data type `long`) End time of the period. The end time is specified in milliseconds, the starting point for the calculation is `1970-01-01, 00:00`.
- **`endDateIsoFormat`** - (optional, data type `dateTime`) Same as `endDate`, but specified in ISO 8601 format (`YYYY-MM-DDThh:mm:ss.fffZ`).

Example:

```
<Period startDateIsoFormat="2012-08-31T22:00:00.000Z"
  startDate="1346450400000"
  endDateIsoFormat="2012-09-30T22:00:00.000Z"
  endDate="1349042400000"/>
```

OrganizationData

Provides details of an organization.

An `OrganizationData` element has the following attributes:

- **`id`** - (required, data type `string`) ID of the organization.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the organization.

An `OrganizationData` element contains the following subelements:

- **`Email`** - (data type `string`) Email address of the organization.
- **`Name`** - (data type `string`) Name of the organization.
- **`Address`** - (data type `string`) Address of the organization.
- **`CountryIsoCode`** - (data type `string`) ISO code of the country where the organization is located.

Example:

```
<OrganizationData id="8e8f596c" key="37003">
  <Email>info@company.com</Email>
  <Name>Company</Name>
  <Address>Postal Address</Address>
  <CountryIsoCode>DE</CountryIsoCode>
</OrganizationData>
```

G.2 Broker Revenue Share Data

The following sections describe the XML elements and attributes that make up the revenue share data for brokers.

BrokerRevenueShareResult

Top-level container element for broker revenue share data. For each broker organization in consideration, a `BrokerRevenueShareResult` element is added to the billing data file.

A `BrokerRevenueShareResult` element has the following attributes:

- **`organizationId`** - (required, data type `string`) ID of the broker organization.
- **`organizationKey`** - (required, data type `positiveInteger`) Internal numeric key of the broker organization.

A `BrokerRevenueShareResult` element contains the following subelements:

- An `OrganizationData` element specifying the details of the broker organization (see *OrganizationData* on page 109).
- A `Period` element specifying the billing period (see *Period* on page 108).
- A `Currency` element for each currency for which broker revenue share data is available (see *Currency* on page 110).

Example:

```
<BrokerRevenueShareResult organizationId="cd9ffaac"
 organizationKey="19000" >
  <OrganizationData> ... </OrganizationData>
  <Period> ... </Period>
  <Currency> ... </Currency>
</BrokerRevenueShareResult>
```

Currency

Contains the broker revenue share data for a specific currency.

A `Currency` element has the following attribute:

`id` - (required, data type `string`) ISO code of the currency.

A `Currency` element contains the following subelements:

- A `Supplier` element for each supplier organization that provides a service for which the current broker organization receives a revenue share (see *Supplier* on page 110).
- A `BrokerRevenue` element specifying the overall revenue for the currency in its `totalAmount` attribute (optional, data type positive `decimal`, scale 2), and the overall broker revenue share for the currency in its `amount` attribute (required, data type positive `decimal`, scale 2).

Example:

```
<Currency id="EUR">
  <Supplier>...</Supplier>
  <BrokerRevenue totalAmount="1000.50" amount="100.05" />
</Currency>
```

Supplier

Contains the broker revenue share data for the services provided by a specific supplier.

A `Supplier` element contains the following subelements:

- An `OrganizationData` element specifying the details of the supplier organization (see *OrganizationData* on page 109).
- A `Service` element for each service provided by the supplier for which the current broker organization receives a revenue share (see *Service* on page 110).
- A `BrokerRevenuePerSupplier` element specifying the overall revenue for the supplier in its `totalAmount` attribute (optional, data type positive `decimal`, scale 2), and the overall broker revenue share in its `amount` attribute (required, data type positive `decimal`, scale 2).

Example:

```
<Supplier>
  <OrganizationData> ... </OrganizationData>
  <Service> ... </Service>
  <BrokerRevenuePerSupplier totalAmount="200.50" amount="20.05" />
</Supplier>
```

Service

Specifies the broker revenue share data for a specific service.

A `Service` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the service.

- **key** - (required, data type positiveInteger) Internal numeric key of the service offered by the broker. Technically, this is a copy of the original service defined by the supplier.
- **templateKey** - (required, data type positiveInteger) Internal numeric key of the original service defined by the supplier.

A Service element contains a ServiceRevenue element which specifies the total revenue for the service and the broker revenue share in its attributes:

- **totalAmount** - (required, data type positive decimal, scale 2) Total revenue for the service in the billing period.
- **brokerRevenueSharePercentage** - (required, data type positive decimal, scale 2) Percentage of the revenue the broker is entitled to.
- **brokerRevenue** - (required, data type positive decimal, scale 2) The broker's revenue share for the service in the billing period.

A ServiceRevenue element contains a ServiceCustomerRevenue subelement for each customer who used the service. It specifies the total revenue for the service generated by the customer and the broker revenue share in its attributes:

- **customerName** - (optional, data type string) Name of the customer organization.
- **customerId** - (optional, data type string) ID of the customer organization.
- **totalAmount** - (optional, data type positive decimal, scale 2) Total revenue for the service generated by the customer in the billing period.
- **brokerRevenueSharePercentage** - (optional, data type positive decimal, scale 2) Percentage of the revenue the broker is entitled to.
- **brokerRevenue** - (optional, data type positive decimal, scale 2) The broker's revenue share for the service generated by the customer in the billing period.

Example:

```
<Service id="Mega Office" key="17005" templateKey="10501">
  <ServiceRevenue totalAmount="200.00"
   brokerRevenueSharePercentage="10.00" brokerRevenue="20.00" />
    <ServiceCustomerRevenue customerName="MyCompany"
     customerId="862cfb94" totalAmount="50.00"
     brokerRevenueSharePercentage="10.00" brokerRevenue="5.00" />
</Service>
```

G.3 Reseller Revenue Share Data

The following sections describe the XML elements and attributes that make up the revenue share data for resellers.

ResellerRevenueShareResult

Top-level container element for reseller revenue share data. For each reseller organization in consideration, a ResellerRevenueShareResult element is added to the billing data file.

A ResellerRevenueShareResult element has the following attributes:

- **organizationId** - (required, data type string) ID of the reseller organization.
- **organizationKey** - (required, data type positiveInteger) Internal numeric key of the reseller organization.

A `ResellerRevenueShareResult` element contains the following subelements:

- An `OrganizationData` element specifying the details of the reseller organization (see *OrganizationData* on page 109).
- A `Period` element specifying the billing period (see *Period* on page 108).
- A `Currency` element for each currency for which reseller revenue share data is available (see *Currency* on page 112).

Example:

```
<ResellerRevenueShareResult organizationId="cd9ffaac"
 organizationKey="19000">
  <OrganizationData> ... </OrganizationData>
  <Period> ... </Period>
  <Currency> ... </Currency>
</ResellerRevenueShareResult>
```

Currency

Contains the reseller revenue share data for a specific currency.

A `Currency` element has the following attribute:

`id` - (required, data type `string`) ISO code of the currency.

A `Currency` element contains the following subelements:

- A `Supplier` element for each supplier organization that provides a service for which the current reseller organization receives a revenue share (see *Supplier* on page 112).
- A `ResellerRevenue` element with the following attributes:

  `totalAmount` - (optional, data type positive `decimal`, scale 2) Overall revenue for the currency.

  `amount` - (required, data type positive `decimal`, scale 2) Overall reseller revenue share.

  `purchasePrice` - (optional, data type positive `decimal`, scale 2) Difference between the `totalAmount` and the `amount` attribute.

Example:

```
<Currency id="EUR">
  <Supplier>...</Supplier>
  <ResellerRevenue totalAmount="1000.50" amount="200.10"
   purchasePrice="800.40"/>
</Currency>
```

Supplier

Contains the reseller revenue share data for the services provided by a specific supplier.

A `Supplier` element contains the following subelements:

- An `OrganizationData` element specifying the details of the supplier organization (see *OrganizationData* on page 109).
- A `Service` element for each service provided by the supplier for which the current reseller organization receives a revenue share (see *Service* on page 113).
- A `ResellerRevenuePerSupplier` element with the following attributes:

  `totalAmount` - (optional, data type positive `decimal`, scale 2) Overall revenue for the supplier.

  `amount` - (required, data type positive `decimal`, scale 2) Overall reseller revenue share for the supplier.

`purchasePrice` - (optional, data type positive `decimal`, scale 2) Difference between the `totalAmount` and the `amount` attribute.

Example:

```
<Supplier>
  <OrganizationData> ... </OrganizationData>
  <Service> ... </Service>
  <ResellerRevenuePerSupplier totalAmount="200.50" amount="40.10"
   purchasePrice="160.40"/>
</Supplier>
```

Service

Specifies the reseller revenue share data for a specific service.

A `Service` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the service.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the service offered by the reseller. Technically, this is a copy of the original service defined by the supplier.
- **`templateKey`** - (required, data type `positiveInteger`) Internal numeric key of the original service defined by the supplier.

A `Service` element contains the following subelements:

- A `Subscription` element for each subscription to the service offered by the reseller (see *Subscription* on page 113).
- A `ServiceRevenue` element specifying the overall reseller revenue share for the service (see *ServiceRevenue* on page 114).

Example:

```
<Service id="Mega Office" key="17005" templateKey="10501">
  <Subscription> ... </Subscription>
  <ServiceRevenue> ... </ServiceRevenue>
</Service>
```

Subscription

Specifies the revenue for a specific subscription to a service.

A `Subscription` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the subscription.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the subscription.
- **`billingKey`** - (required, data type `positiveInteger`) Unique identifier allowing, for example, accounting systems to relate billing data to an invoice. The billing data key is printed on the invoice.
- **`revenue`** - (required, data type positive `decimal`, scale 2) The total revenue for the subscription in the billing period. Note that there is a `Service` element for each phase of the subscription if the subscription is upgraded or downgraded.

A `Subscription` element contains a `Period` subelement specifying the applicable billing period (see *Period* on page 108).

Example:

```
<Subscription id="Mega Office Basic" key="17005" billingKey="19032"
  revenue="600.00">
  <Period>... </Period>
</Subscription>
```

ServiceRevenue

Specifies the total revenue for a service and the reseller revenue share.

A `ServiceRevenue` element has the following attributes:

- **`totalAmount`** - (required, data type positive `decimal`, scale 2) Total revenue for all subscriptions to the service in the billing period.
- **`resellerRevenueSharePercentage`** - (required, data type positive `decimal`, scale 2) Percentage of the revenue the reseller is entitled to.
- **`resellerRevenue`** - (required, data type positive `decimal`, scale 2) The reseller's revenue share for the service in the billing period.

A `ServiceRevenue` element contains a `ServiceCustomerRevenue` subelement for each customer who used the service. It specifies the total revenue for the service generated by the customer and the reseller revenue share in its attributes:

- **`customerName`** - (optional, data type `string`) Name of the customer organization.
- **`customerId`** - (optional, data type `string`) ID of the customer organization.
- **`totalAmount`** - (optional, data type positive `decimal`, scale 2) Total revenue for the service generated by the customer in the billing period.
- **`resellerRevenueSharePercentage`** - (optional, data type positive `decimal`, scale 2) Percentage of the revenue the reseller is entitled to.
- **`resellerRevenue`** - (optional, data type positive `decimal`, scale 2) The reseller's revenue share for the service generated by the customer in the billing period.
- **`purchasePrice`** - (optional, data type positive `decimal`, scale 2) Difference between the `totalAmount` and the `resellerRevenue` attribute.

Example:

```
<ServiceRevenue totalAmount="200.00"
  resellerRevenueSharePercentage="10.00" resellerRevenue="20.00">
    <ServiceCustomerRevenue customerName="MyCompany"
     customerId="862cfb94" totalAmount="50.00"
     resellerRevenueSharePercentage="10.00" resellerRevenue="5.00"
     purchasePrice="45"/>
</ServiceRevenue>
```

G.4 Marketplace Owner Revenue Share Data

The following sections describe the XML elements and attributes that make up the revenue share data for marketplace owners.

MarketplaceOwnerRevenueShareResult

Top-level container element for marketplace owner revenue share data. For each marketplace owner organization in consideration, a `MarketplaceOwnerRevenueShareResult` element is added to the billing data file.

A `MarketplaceOwnerRevenueShareResult` element has the following attributes:

- **`organizationId`** - (required, data type `string`) ID of the marketplace owner organization.
- **`organizationKey`** - (required, data type `positiveInteger`) Internal numeric key of the marketplace owner organization.

A `MarketplaceOwnerRevenueShareResult` element contains the following subelements:

- An `OrganizationData` element specifying the details of the marketplace owner organization (see *OrganizationData* on page 109).
- A `Period` element specifying the billing period (see *Period* on page 108).
- A `Currency` element for each currency for which marketplace owner revenue share data is available (see *Currency* on page 115).

Example:

```
<MarketplaceOwnerRevenueShareResult organizationId="cd9ffaac"
 organizationKey="19000">
  <OrganizationData> ... </OrganizationData>
  <Period> ... </Period>
  <Currency> ... </Currency>
</MarketplaceOwnerRevenueShareResult>
```

Currency

Contains the marketplace owner revenue share data for a specific currency.

A `Currency` element has the following attribute:

`id` - (required, data type `string`) ISO code of the currency.

A `Currency` element contains the following subelements:

- A `Marketplace` element for each marketplace for which revenue share data for the current marketplace owner organization is available (see *Marketplace* on page 115).
- A `RevenuesOverAllMarketplaces` element summarizing the revenue shares across the marketplaces (see *RevenuesOverAllMarketplaces* on page 119).

Example:

```
<Currency id="EUR">
  <Marketplace>...</Marketplace>
  <RevenuesOverAllMarketplaces> ... </RevenuesOverAllMarketplaces>
</Currency>
```

Marketplace

Contains the revenue share data for a specific marketplace.

A `Marketplace` element has the following attributes:

- **`id`** - (required, data type `string`) ID of the marketplace.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the marketplace.

A `Marketplace` element contains the following subelements:

- A `Service` element for each service published on the marketplace for which revenue share data is available (see *Service* on page 116).
- A `RevenuesPerMarketplace` element summarizing the revenue shares for all organizations involved (see *RevenuesPerMarketplace* on page 118).

Example:

```
<Marketplace id="e1828fba" key="17021">
  <Service>...</Service>
  <RevenuesPerMarketplace> ... </RevenuesPerMarketplace>
</Marketplace>
```

Service

Specifies the revenue share data for a specific service published on the current marketplace.

A `Service` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the service.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the published service. If the service is offered by a broker or reseller, this is the key of an internal technical copy of the original service defined by the supplier. If the service is offered directly by its supplier, it is the key of the original service.
- **`model`** - (required, data type `string`) String specifying by which type of organization the service is offered. Possible values are:
  - `DIRECT` : The service is offered by its supplier.
  - `BROKER` : The service is offered by a broker.
  - `RESELLER` : The service is offered by a reseller.
- **`templateKey`** - (optional, data type `positiveInteger`) Internal numeric key of the original service defined by the supplier, if the service is published by a broker or reseller.

A `Service` element contains the following subelements:

- A `Supplier` element specifying the supplier organization who defined the service in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- If the service is offered by a broker: A `Broker` element specifying the broker organization in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- If the service is offered by a reseller: A `Reseller` element specifying the reseller organization in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- A `RevenueShareDetails` element specifying the revenue shares for the service (see *RevenueShareDetails* on page 117).

Examples:

The service is offered directly by its supplier:

```
<Service id="Mega Office" key="17005" model="DIRECT">
  <Supplier> <OrganizationData> ... </OrganizationData> </Supplier>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

The service is offered by a broker:

```
<Service id="Mega Office" key="17005" model="BROKER"
 templateKey="10501">
  <Supplier> <OrganizationData> ... </OrganizationData> </Supplier>
  <Broker> <OrganizationData> ... </OrganizationData> </Broker>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

The service is offered by a reseller:

```
<Service id="Mega Office" key="17005" model="RESELLER"
 templateKey="10501">
  <Supplier> <OrganizationData> ... </OrganizationData> </Supplier>
  <Reseller> <OrganizationData> ... </OrganizationData> </Reseller>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

RevenueShareDetails

Specifies the revenue for a specific service and the revenue shares for all organizations involved in selling the service.

A `RevenueShareDetails` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Total revenue for the service in the billing period.
- **`marketplaceRevenueSharePercentage`** - (required, data type `decimal`, scale 2) Percentage of the revenue the marketplace owner is entitled to.
- **`operatorRevenueSharePercentage`** - (required, data type `decimal`, scale 2) Percentage of the revenue the platform operator is entitled to.
- **`brokerRevenueSharePercentage`** - (optional, data type `decimal`, scale 2) If the service is offered by a broker: Percentage of the revenue the broker is entitled to.
- **`resellerRevenueSharePercentage`** - (optional, data type `decimal`, scale 2) If the service is offered by a reseller: Percentage of the revenue the reseller is entitled to.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) The marketplace owner's revenue share for the service in the billing period.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) The platform operator's revenue share for the service in the billing period.
- **`brokerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a broker: The broker's revenue share for the service in the billing period.
- **`resellerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a reseller: The reseller's revenue share for the service in the billing period.
- **`amountForSupplier`** - (required, data type `decimal`, scale 2) The supplier's revenue share for the service in the billing period. This is the remaining value of the total service revenue after subtracting the revenue shares for the operator, marketplace owner, broker, and/or reseller.

Examples:

The service is offered directly by its supplier:

```
<RevenueShareDetails serviceRevenue="500.00"
  marketplaceRevenueSharePercentage="15.00" marketplaceRevenue="75.00"
  operatorRevenueSharePercentage="10.00" operatorRevenue="50.00"
  amountForSupplier="375.00">
</RevenueShareDetails>
```

The service is offered by a broker:

```
<RevenueShareDetails serviceRevenue="4000.00"
  marketplaceRevenueSharePercentage="21.00" marketplaceRevenue="840.00"
  brokerRevenueSharePercentage="9.00" brokerRevenue="360.00"
  operatorRevenueSharePercentage="5.00" operatorRevenue="200.00"
  amountForSupplier="2600.00">
```

```
</RevenueShareDetails>
```

The service is offered by a reseller:

```
<RevenueShareDetails serviceRevenue="3000.00"
  marketplaceRevenueSharePercentage="16.00" marketplaceRevenue="480.00"
  resellerRevenueSharePercentage="20.00" resellerRevenue="600.00"
  operatorRevenueSharePercentage="5.00" operatorRevenue="150.00"
  amountForSupplier="1770.00">
</RevenueShareDetails>
```

RevenuesPerMarketplace

Provides an overview of the revenue shares for the different organizations involved in selling services on a specific marketplace.

A `RevenuesPerMarketplace` element contains the following subelements:

- A `Brokers` element listing the relevant broker organizations with their revenue shares in `Organization` subelements.
- A `Resellers` element listing the relevant reseller organizations with their revenue shares in `Organization` subelements.
- A `Suppliers` element listing the relevant supplier organizations with their revenue shares in `Organization` subelements.
- A `MarketplaceOwner` element specifying the revenue share for the marketplace owner in its `amount` attribute (required, data type `decimal`, scale 2).

Each `Brokers`, `Resellers`, or `Suppliers` element included in a `RevenuesPerMarketplace` element has the following attributes:

- **`amount`** - (optional, data type `decimal`, scale 2) Overall revenue share of the listed broker, reseller, or supplier organizations.
- **`marketplaceRevenue`** - (optional, data type `decimal`, scale 2) The marketplace owner's share of the revenue of the listed broker, reseller, or supplier organizations.
- **`totalAmount`** - (optional, data type `decimal`, scale 2) Overall revenue for all services offered by the listed broker, reseller, or supplier organizations on the marketplace.

An `Organization` element included in a `Brokers`, `Resellers`, or `Suppliers` element has the following attributes:

- **`identifier`** - (required, data type `string`) ID of the organization.
- **`amount`** - (required, data type `decimal`, scale 2) Revenue share of the organization.
- **`name`** - (optional, data type `string`) Name of the organization.
- **`marketplaceRevenue`** - (optional, data type `decimal`, scale 2) The marketplace owner's share of the organization's revenue.
- **`totalAmount`** - (optional, data type `decimal`, scale 2) Overall revenue for all services offered by the organization on the marketplace.

Example:

```
<RevenuesPerMarketplace>
  <Brokers amount="250.00" totalAmount="1000.00"
   marketplaceRevenue="200.00">
    <Organization identifier="da3cd3a3" amount="100.00" name="broker"
     marketplaceRevenue="80.00" totalAmount="400.00" />
    <Organization identifier="ea4cd3a3" amount="150.00" name="broker2"
```

```
    marketplaceRevenue="120.00" totalAmount="600.00" />
  </Brokers>
  <Resellers amount="600.00" totalAmount="2000.00"
   marketplaceRevenue="400.00">
    <Organization identifier="bc4cd3a3" amount="240.00" name="reseller"
     marketplaceRevenue="160.00" totalAmount="800.00" />
    <Organization identifier="fg5cd3a3" amount="360.00" name="reseller2"
     marketplaceRevenue="240.00" totalAmount="1200.00" />
  </Resellers>
  <Suppliers />
  <MarketplaceOwner amount="600.00" />
</RevenuesPerMarketplace>
```

RevenuesOverAllMarketplaces

Provides an overview of the revenue shares for the different organizations involved in selling services on any of the marketplaces that belong to a specific marketplace owner.

A `RevenuesOverAllMarketplaces` element contains the following subelements:

- A `Brokers` element listing the relevant broker organizations with their revenue shares in `Organization` subelements.
- A `Resellers` element listing the relevant reseller organizations with their revenue shares in `Organization` subelements.
- A `Suppliers` element listing the relevant supplier organizations with their revenue shares in `Organization` subelements.
- A `MarketplaceOwner` element specifying the revenue share for the marketplace owner in its `amount` attribute (required, data type `decimal`, scale 2).

Each `Brokers`, `Resellers`, or `Suppliers` element included in a `RevenuesOverAllMarketplaces` element has the following attributes:

- **`amount`** - (optional, data type `decimal`, scale 2) Overall revenue share of the listed broker, reseller, or supplier organizations.
- **`marketplaceRevenue`** - (optional, data type `decimal`, scale 2) The marketplace owner's share of the revenue of the listed broker, reseller, or supplier organizations.
- **`totalAmount`** - (optional, data type `decimal`, scale 2) Overall revenue for all services offered by the listed broker, reseller, or supplier organizations on the marketplaces.

An `Organization` element included in a `Brokers`, `Resellers`, or `Suppliers` element has the following attributes:

- **`identifier`** - (required, data type `string`) ID of the organization.
- **`amount`** - (required, data type `decimal`, scale 2) Revenue share of the organization.
- **`name`** - (optional, data type `string`) Name of the organization.
- **`marketplaceRevenue`** - (optional, data type `decimal`, scale 2) The marketplace owner's share of the organization's revenue.
- **`totalAmount`** - (optional, data type `decimal`, scale 2) Overall revenue for all services offered by the organizations on the marketplaces.

Example:

```
<RevenuesOverAllMarketplaces>
  <Brokers amount="250.00" totalAmount="1000.00"
   marketplaceRevenue="200.00">
    <Organization identifier="da3cd3a3" amount="100.00" name="broker"
```

```
    marketplaceRevenue="80.00" totalAmount="400.00" />
   <Organization identifier="ea4cd3a3" amount="150.00" name="broker2"
    marketplaceRevenue="120.00" totalAmount="600.00" />
  </Brokers>
  <Resellers amount="600.00" totalAmount="2000.00"
   marketplaceRevenue="400.00">
   <Organization identifier="bc4cd3a3" amount="240.00" name="reseller"
    marketplaceRevenue="160.00" totalAmount="800.00" />
   <Organization identifier="fg5cd3a3" amount="360.00" name="reseller2"
    marketplaceRevenue="240.00" totalAmount="1200.00" />
  </Resellers>
  <Suppliers />
  <MarketplaceOwner amount="600.00" />
</RevenuesOverAllMarketplaces>
```

G.5 Supplier Revenue Share Data

The following sections describe the XML elements and attributes that make up the revenue share data for suppliers.

SupplierRevenueShareResult

Top-level container element for all revenue share data a supplier has to pay to all participating parties (platform operator, marketplace owners, brokers, resellers) based on his contractual agreements. For each supplier organization in consideration, a `SupplierRevenueShareResult` element is added to the billing data file.

A `SupplierRevenueShareResult` element has the following attributes:

- **`organizationId`** - (required, data type `string`) ID of the supplier organization.
- **`organizationKey`** - (required, data type `positiveInteger`) Internal numeric key of the supplier organization.

A `SupplierRevenueShareResult` element contains the following subelements:

- An `OrganizationData` element specifying the details of the supplier organization (see *OrganizationData* on page 109).
- A `Period` element specifying the billing period (see *Period* on page 108).
- A `Currency` element for each currency for which supplier revenue share data is available (see *Currency* on page 120).

Example:

```
<SupplierRevenueShareResult organizationId="cd9ffaac"
 organizationKey="19000">
  <OrganizationData> ... </OrganizationData>
  <Period> ... </Period>
  <Currency> ... </Currency>
</SupplierRevenueShareResult>
```

Currency

Contains the supplier revenue share data for a specific currency.

A `Currency` element has the following attribute:

`id` - (required, data type `string`) ISO code of the currency.

A `Currency` element contains the following subelements:

- A `Marketplace` element for each marketplace for which revenue share data for the supplier organization is available (see *Marketplace* on page 121).
- A `SupplierRevenue` element specifying the detailed revenue share data for the current supplier organization (see *SupplierRevenue* on page 121).

Example:

```
<Currency id="EUR">
  <Marketplace>...</Marketplace>
  <SupplierRevenue>...</SupplierRevenue>
</Currency>
```

Marketplace

Contains the revenue share data for a specific marketplace.

A `Marketplace` element has the following attributes:

- **`id`** - (required, data type `string`) ID of the marketplace.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the marketplace.

A `Marketplace` element contains the following subelements:

- A `MarketplaceOwner` element specifying the marketplace owner in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- A `Service` element for each service published on the marketplace for which revenue share data for the supplier is available (see *Service* on page 123).
- A `RevenuePerMarketplace` element summarizing the revenue shares for all organizations involved (see *RevenuePerMarketplace* on page 127).

Example:

```
<Marketplace id="e1828fba" key="17021">
  <MarketplaceOwner>
    <OrganizationData> ... </OrganizationData>
  </MarketplaceOwner>
  <Service>...</Service>
  <RevenuePerMarketplace> ... </RevenuePerMarketplace>
</Marketplace>
```

SupplierRevenue

Contains the detailed revenue data for a supplier organization.

A `SupplierRevenue` element has the following attributes:

- **`amount`** - (required, data type `decimal`, scale 2) Overall revenue for the supplier.

A `SupplierRevenue` element may contain the following subelements, depending on which organizations offer the supplier's services:

- A `DirectRevenue` element specifying the revenue for the services offered directly by the supplier.
- A `BrokerRevenue` element specifying the revenue for the supplier's services offered by brokers.
- A `ResellerRevenue` element specifying the revenue for the supplier's services offered by resellers.

Example:

```
<SupplierRevenue amount="1500.00">
  <DirectRevenue> ... </DirectRevenue>
  <BrokerRevenue> ... </BrokerRevenue>
  <ResellerRevenue> ... </ResellerRevenue>
</SupplierRevenue>
```

DirectRevenue

Contains the revenue data for services offered directly by their supplier.

A `DirectRevenue` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the services offered by the supplier.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the owners of the marketplaces where the services are offered by the supplier.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the platform operator who provides the infrastructure for the marketplaces and services.

Example:

```
<SupplierRevenue amount="1500.00">
  <DirectRevenue serviceRevenue="600.00" marketplaceRevenue="75.00"
   operatorRevenue="25.00"/>
  <BrokerRevenue> ... </BrokerRevenue>
  <ResellerRevenue> ...</ResellerRevenue>
</SupplierRevenue>
```

BrokerRevenue

Contains the revenue data for a supplier's services offered by brokers.

A `BrokerRevenue` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the services offered by the broker.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the owners of the marketplaces where the services are offered by brokers.
- **`brokerRevenue`** - (required, data type `decimal`, scale 2) Overall revenue share for the broker offering the services.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the platform operator who provides the infrastructure for the marketplaces and services.

Example:

```
<SupplierRevenue amount="1500.00">
  <DirectRevenue> ... </DirectRevenue>
  <BrokerRevenue serviceRevenue="700.00" marketplaceRevenue="75.00"
   brokerRevenue="25.00" operatorRevenue="100.00"/>
  <ResellerRevenue> ...</ResellerRevenue>
</SupplierRevenue>
```

ResellerRevenue

Contains the revenue data for the supplier's services offered by resellers.

A `ResellerRevenue` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the services offered by resellers.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the owners of the marketplaces where the services are offered by the resellers.
- **`resellerRevenue`** - (required, data type `decimal`, scale 2) Overall revenue share for the resellers offering the services.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the platform operator who provides the infrastructure for the marketplaces and services.
- **`overallRevenue`** - (required, data type `decimal`, scale 2) Overall revenue for the services offered by the resellers minus the marketplace owner revenue (`marketplaceRevenue`), the reseller revenue (`resellerRevenue`), and the operator revenue (`operatorRevenue`).

Example:

```
<SupplierRevenue amount="1500.00">
  <DirectRevenue> ... </DirectRevenue>
  <BrokerRevenue> ... </BrokerRevenue>
  <ResellerRevenue serviceRevenue="650.00"
  marketplaceRevenue="75.00"
  resellerRevenue="25.00"
  operatorRevenue="50.00"
  overallRevenue="500.00" />
</SupplierRevenue>
```

Service

Specifies the revenue share data for a specific service.

A `Service` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the service.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the published service. If the service is offered by a broker or reseller, this is the key of an internal technical copy of the original service defined by the supplier. If the service is offered directly by its supplier, it is the key of the original service.
- **`model`** - (required, data type `string`) String specifying by which type of organization the service is offered. Possible values are:
  - `DIRECT` : The service is offered by its supplier.
  - `BROKER` : The service is offered by a broker.
  - `RESELLER` : The service is offered by a reseller.
- **`templateKey`** - (optional, data type `positiveInteger`) Internal numeric key of the original service defined by the supplier, if the service is published by a broker or reseller.

A `Service` element contains the following subelements:

- If the service is offered directly by the supplier: A `Subscription` element for each subscription to the service for which revenue share data is available (see *Subscription* on page 124).
- If the service is offered by a broker:
  - A `Subscription` element for each subscription to the service for which revenue share data is available (see *Subscription* on page 124).

    - A `Broker` element specifying the broker organization in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- If the service is offered by a reseller:
    - A `SubscriptionsRevenue` element summarizing the total revenue for all subscriptions to the service in its `amount` attribute (required, data type positive `decimal`, scale 2).
    - A `Reseller` element specifying the reseller organization in an `OrganizationData` subelement (see *OrganizationData* on page 109).
- A `RevenueShareDetails` element specifying the revenue shares for the service (see *RevenueShareDetails* on page 125).

Examples:

The service is offered directly by its supplier:

```
<Service id="Mega Office" key="17005" model="DIRECT">
  <Subscription> ... </Subscription>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

The service is offered by a broker:

```
<Service id="Mega Office" key="17005" model="BROKER"
 templateKey="10501">
  <Subscription> ... </Subscription>
  <Broker> <OrganizationData> ... </OrganizationData> </Broker>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

The service is offered by a reseller:

```
<Service id="Mega Office" key="17005" model="RESELLER"
 templateKey="10501">
  <SubscriptionsRevenue amount="6000.00" />
  <Reseller> <OrganizationData> ... </OrganizationData> </Reseller>
  <RevenueShareDetails> ... </RevenueShareDetails>
</Service>
```

Subscription

Specifies the revenue for a specific subscription to a service.

A `Subscription` element has the following attributes:

- **`id`** - (required, data type `string`) Name of the subscription.
- **`key`** - (required, data type `positiveInteger`) Internal numeric key of the subscription.
- **`billingKey`** - (required, data type `positiveInteger`) Unique identifier allowing, for example, accounting systems to relate billing data to an invoice. The billing data key is printed on the invoice.
- **`revenue`** - (required, data type positive `decimal`, scale 2) The total revenue for the subscription in the billing period. Note that there is a `Service` element for each phase of the subscription if the subscription is upgraded or downgraded.

A `Subscription` element contains a `Period` subelement specifying the applicable billing period (see *Period* on page 108).

Example:

```
<Subscription id="Mega Office Basic" key="17005" billingKey="19032"
  revenue="600.00">
  <Period>... </Period>
</Subscription>
```

RevenueShareDetails

Specifies the revenue for a specific service and the revenue shares for all organizations involved in selling the service.

A `RevenueShareDetails` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Total revenue for the service in the billing period.
- **`marketplaceRevenueSharePercentage`** - (required, data type `decimal`, scale 2) Percentage of the revenue the marketplace owner is entitled to.
- **`brokerRevenueSharePercentage`** - (optional, data type `decimal`, scale 2) If the service is offered by a broker: Percentage of the revenue the broker is entitled to.
- **`resellerRevenueSharePercentage`** - (optional, data type `decimal`, scale 2) If the service is offered by a reseller: Percentage of the revenue the reseller is entitled to.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) The marketplace owner's revenue share for the service in the billing period.
- **`brokerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a broker: The broker's revenue share for the service in the billing period.
- **`resellerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a reseller: The reseller's revenue share for the service in the billing period.
- **`operatorRevenueSharePercentage`** - (required, data type `decimal`, scale 2) Percentage of the revenue the operator is entitled to.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) The operator's revenue share for the service in the billing period.
- **`amountForSupplier`** - (required, data type `decimal`, scale 2) The supplier's revenue share for the service in the billing period. This is the remaining value of the total service revenue after subtracting the revenue shares for the operator, marketplace owner, broker, and/or reseller.

A `RevenueShareDetails` element contains a `CustomerRevenueShareDetails` subelement for each customer who used the service (see *CustomerRevenueShareDetails* on page 126).

Examples:

The service is offered directly by its supplier:

```
<RevenueShareDetails serviceRevenue="500.00"
  marketplaceRevenueSharePercentage="15.00" marketplaceRevenue="75.00"
  operatorRevenueSharePercentage="10.00" operatorRevenue="50.00"
  amountForSupplier="375.00">
  <CustomerRevenueShareDetails> ... </CustomerRevenueShareDetails>
</RevenueShareDetails>
```

The service is offered by a broker:

```
<RevenueShareDetails serviceRevenue="4000.00"
  marketplaceRevenueSharePercentage="21.00" marketplaceRevenue="840.00"
  brokerRevenueSharePercentage="9.00" brokerRevenue="360.00"
```

```
  operatorRevenueSharePercentage="10.00" operatorRevenue="400.00"
  amountForSupplier="2400.00">
  <CustomerRevenueShareDetails> ... </CustomerRevenueShareDetails>
</RevenueShareDetails>
```

The service is offered by a reseller:

```
<RevenueShareDetails serviceRevenue="3000.00"
  marketplaceRevenueSharePercentage="16.00" marketplaceRevenue="480.00"
  resellerRevenueSharePercentage="20.00" resellerRevenue="600.00"
  operatorRevenueSharePercentage="10.00" operatorRevenue="300.00"
  amountForSupplier="1620.00">
  <CustomerRevenueShareDetails> ... </CustomerRevenueShareDetails>
</RevenueShareDetails>
```

CustomerRevenueShareDetails

Specifies the revenue for a specific service generated by a given customer and the revenue shares for all organizations involved in selling the service.

A `CustomerRevenueShareDetails` element has the following attributes:

- **`customerName`** - (required, data type `string`) Name of the customer organization.
- **`customerId`** - (required, data type `string`) ID of the customer organization.
- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Total revenue for the service in the billing period.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) The marketplace owner's revenue share for the service in the billing period.
- **`resellerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a reseller: The reseller's revenue share for the service in the billing period.
- **`brokerRevenue`** - (optional, data type `decimal`, scale 2) If the service is offered by a broker: The broker's revenue share for the service in the billing period.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) The operator's revenue share for the service in the billing period.
- **`amountForSupplier`** - (required, data type `decimal`, scale 2) The supplier's revenue share for the service generated by the customer in the billing period. This is the remaining value of the total service revenue generated by the customer after subtracting the revenue shares for the operator, marketplace owner, broker, and/or reseller.

Examples:

The service is offered directly by its supplier:

```
<CustomerRevenueShareDetails customerName="MyCompany"
  customerId="8dac00a0" serviceRevenue="316.92"
  marketplaceRevenue="31.69" operatorRevenue="31.69"
  amountForSupplier="223.54"/>
```

The service is offered by a broker:

```
<CustomerRevenueShareDetails customerName="ITCompany"
  customerId="ff48ae0b" serviceRevenue="300.00"
  marketplaceRevenue="63.00" brokerRevenue="27.00"
  operatorRevenue="30.00" amountForSupplier="180.00" />
```

The service is offered by a reseller:

```
<CustomerRevenueShareDetails customerName="YourCompany"
  customerId="859069d7" serviceRevenue="500.00"
  marketplaceRevenue="80.00" resellerRevenue="250.00"
  operatorRevenue="50.00" amountForSupplier="120.00"/>
```

RevenuePerMarketplace

Provides an overview of the revenue shares for the different organizations involved in selling the services of a supplier on a specific marketplace.

A `RevenuePerMarketplace` element has the following attributes:

- **`serviceRevenue`** - (required, data type `decimal`, scale 2) Total revenue for all relevant services in the billing period.
- **`marketplaceRevenue`** - (required, data type `decimal`, scale 2) Total revenue share for the marketplace owner.
- **`brokerRevenue`** - (optional, data type `decimal`, scale 2) Total revenue share for all brokers.
- **`resellerRevenue`** - (optional, data type `decimal`, scale 2) Total revenue share for all resellers.
- **`operatorRevenue`** - (required, data type `decimal`, scale 2) Total revenue share for the operator.
- **`overallRevenue`** - (required, data type `decimal`, scale 2) Total revenue for the supplier. This is the remaining value of the total revenue for all services after subtracting the revenue shares for the operator, marketplace owner, brokers, and/or resellers.

Example:

```
<RevenuePerMarketplace serviceRevenue="80905.00"
  marketplaceRevenue="8090.50"
  resellerRevenue="0.00"
  brokerRevenue="20226.25"
  operatorRevenue="8899.55"
  overallRevenue="43688.70"/>
```

Appendix H: Menu Options and Required Roles

The roles of an organization determine which functions are available to its users at the ESCM user interface and which roles the users can be assigned. The user roles control the actions an individual user is allowed to execute.

This appendix provides a list of the available user interface functions and shows which user of which organization is allowed to execute a function.

Account Menu

This menu is available to organizations of all roles.

Function	Organization Role	User Role
Edit profile	Any	Any
Import users (if the organization uses LDAP-based user management)	Any	Administrator
Change password (if the organization does not use an external system for user management)	Any	Any
Register new users (if the organization does not use LDAP-based user management)	Any	Administrator
Manage users	Any	Administrator
LDAP settings (if the organization uses LDAP-based user management)	Any	Administrator
Create report	Any	Administrator
Manage suppliers	Technology provider	Technology manager
Manage custom attributes	Supplier	Service manager
Process triggers	Any	Administrator
Manage processes (if the organization is connected to an external process control system)	Any	Any
Export billing data	Supplier, reseller, broker, marketplace owner, operator	Service manager, reseller, broker, marketplace manager, operator

Function	Organization Role	User Role
Define billing period	Supplier, reseller	Service manager, reseller

Customer Menu

This menu is available to supplier, reseller, and broker organizations.

Function	Organization Role	User Role
Register customer	Supplier, reseller, broker	Service manager, reseller, broker
View customer	Reseller, broker	Reseller, broker
Manage customer	Supplier	Service manager
Manage payment types	Supplier, reseller	Service manager, reseller
Manage VAT rates	Supplier	Service manager
View subscription	Supplier, reseller, broker	Service manager, reseller, broker
Manage subscription attributes	Supplier	Service manager
Terminate subscription	Supplier, reseller	Service manager, reseller

Marketable Service Menu

This menu is available to supplier, reseller, and broker organizations.

Function	Organization Role	User Role
Define service	Supplier	Service manager
Update service	Supplier	Service manager
View service	Broker	Broker
Manage service	Reseller	Reseller
Copy service	Supplier	Service manager
Delete service	Supplier	Service manager
Define up/downgrade options	Supplier	Service manager
Define publishing options	Supplier, reseller, broker	Service manager, reseller, broker
Activate or deactivate services	Supplier, reseller, broker	Service manager, reseller, broker

Marketplace Menu

This menu is available to marketplace owner and operator organizations.

Function	Organization Role	User Role
Manage categories	Marketplace owner	Marketplace manager
Manage access	Marketplace owner	Marketplace manager
Manage sellers	Marketplace owner	Marketplace manager
Create marketplace	Operator	Operator
Update marketplace	Marketplace owner, operator	Marketplace manager or operator. The operator can assign a different marketplace owner and define revenue shares; the marketplace manager can change the name and settings of the marketplace.
Delete marketplace	Operator	Operator
Manage broker revenue share	Operator	Operator
Manage reseller revenue share	Operator	Operator
Add tracking code	Marketplace owner	Marketplace manager
Define featured services	Marketplace owner	Marketplace manager
Customize stage	Marketplace owner	Marketplace manager
Customize texts	Marketplace owner	Marketplace manager
Customize layout	Marketplace owner	Marketplace manager

Operation Menu

This menu is available to operator organizations only.

Function	Organization Role	User Role
Manage users	Operator	Operator
Create organization	Operator	Operator
Create payment service provider	Operator	Operator
Manage organization	Operator	Operator
Manage operator revenue share	Operator	Operator
Manage timers	Operator	Operator
Manage payment service provider	Operator	Operator

Function	Organization Role	User Role
Manage currencies	Operator	Operator
Manage LDAP settings	Operator	Operator
Update configuration settings	Operator	Operator
Billing data preview	Operator	Operator
Execute billing tasks	Operator	Operator
Export audit log	Operator	Operator
Manage languages	Operator	Operator
Manage billing systems	Operator	Operator
Manage tenants	Operator	Operator

Price Model Menu

This menu is available to supplier organizations only.

Function	Organization Role	User Role
Define for service	Supplier	Service manager
Define for customer	Supplier	Service manager
Delete for customer	Supplier	Service manager
Define for subscription	Supplier	Service manager

Technical Service Menu

This menu is available to technology provider organizations only.

Function	Organization Role	User Role
Register service definition	Technology provider	Technology manager
Import service definition	Technology provider	Technology manager
Update service definition	Technology provider	Technology manager
Export service definition	Technology provider	Technology manager
Delete service definition	Technology provider	Technology manager
View billing systems	Technology provider	Technology manager

Glossary

Administrator

A privileged user role within an organization with the permission to manage the organization's account and subscriptions as well as its users and their roles. Each organization has at least one administrator.

Application

A software, including procedures and documentation, which performs productive tasks for users.

Billing System

A system responsible for calculating the charges for using a service.

Broker

An organization which supports suppliers in establishing relationships to customers by offering the suppliers' services on a marketplace, as well as a privileged user role within such an organization.

Cloud

A metaphor for the Internet and an abstraction of the underlying infrastructure it conceals.

Cloud Computing

The provisioning of dynamically scalable and often virtualized resources as a service over the Internet on a utility basis.

Customer

An organization which subscribes to one or more marketable services in ESCM in order to use the underlying applications in the Cloud.

Infrastructure as a Service (IaaS)

The delivery of computer infrastructure (typically a platform virtualization environment) as a service.

Marketable Service

A service offering to customers in ESCM, based on a technical service. A marketable service defines prices, conditions, and restrictions for using the underlying application.

Marketplace

A virtual platform for suppliers, brokers, and resellers in ESCM to provide their services to customers.

Marketplace Owner

An organization which holds a marketplace in ESCM, where one or more suppliers, brokers, or resellers can offer their marketable services.

Marketplace Manager

A privileged user role within a marketplace owner organization.

Operator

An organization or person responsible for maintaining and operating ESCM.

Organization

An organization typically represents a company, but it may also stand for a department of a company or a single person. An organization has a unique account and ID, and is assigned one or more of the following roles: technology provider, supplier, customer, broker, reseller, marketplace owner, operator.

Organizational Unit

A set of one or more users within an organization representing, for example, a department in a company, an individual project, a cost center, or a single person. A user may be assigned to one or more organizational units.

OU Administrator

A privileged user role within an organization allowing a user to manage the organizational units for which he has been appointed as an administrator, and to create, modify, and terminate subscriptions for these units.

Payment Type

A specification of how a customer may pay for the usage of his subscriptions. The operator defines the payment types available in ESCM; the supplier or reseller determines which payment types are offered to his customers, for example payment on receipt of invoice, direct debit, or credit card.

Platform as a Service (PaaS)

The delivery of a computing platform and solution stack as a service.

Price Model

A specification for a marketable service defining whether and how much customers subscribing to the service will be charged for the subscription as such, each user assigned to the subscription, specific events, or parameters and their options.

Reseller

An organization which offers services defined by suppliers to customers applying its own terms and conditions, as well as a privileged user role within such an organization.

Role

A collection of authorities that control which actions can be carried out by an organization or user to whom the role is assigned.

Seller

Collective term for supplier, broker, and reseller organizations.

Service

Generally, a discretely defined set of contiguous or autonomous business or technical functionality, for example an infrastructure or Web service. ESCM distinguishes between technical services and marketable services, and uses the term "service" as a synonym for "marketable service".

Service Manager

A privileged user role within a supplier organization.

Standard User

A non-privileged user role within an organization.

Software as a Service (SaaS)

A model of software deployment where a provider licenses an application to customers for use as a service on demand.

Subscription

An agreement registered by a customer for a marketable service in ESCM. By subscribing to a service, the customer is given access to the underlying application under the conditions defined in the marketable service.

Subscription Manager

A privileged user role within an organization with the permission to create and manage his own subscriptions.

Supplier

An organization which defines marketable services in ESCM for offering applications provisioned by technology providers to customers.

Technical Service

The representation of an application in ESCM. A technical service describes parameters and interfaces of the underlying application and is the basis for one or more marketable services.

Technology Manager

A privileged user role within a technology provider organization.

Technology Provider

An organization which provisions applications as technical services in ESCM.