

Living Systems® Process Suite

Management

Living Systems Process Suite Documentation

3.2
Tue Jan 12 2021

Copyright © 2007-2021 Whitestein Technologies AG.

This document is part of the Living Systems® Process Suite product, and its use is governed by the corresponding license agreement. All rights reserved.

Whitestein Technologies, Living Systems, and the corresponding logos are registered trademarks of Whitestein Technologies AG. Java and all Java-based trademarks are trademarks of Oracle and/or its affiliates. Other company, product, or service names may be trademarks or service marks of their respective holders.

Contents

1	Management	1
1.1	Module Management	2
1.2	Model Instance Management	2
1.3	Person Management	2
1.3.1	Substitution	3
1.3.2	Role Management	3
1.3.2.1	Parametric Roles	4
1.3.3	Security-Role Management	4
1.4	To-Do Management	5
1.4.1	Delegation	5
1.4.2	Escalation	6
1.4.3	Rejection	6
1.4.4	Resetting	6
1.4.5	Reassignment	6
1.5	Logs and Exceptions	6
1.5.1	Logs	6
1.5.2	Exceptions	6

2 Management Perspective	7
2.1 Model and Module Management	8
2.1.1 Uploading a Module	9
2.1.2 Generating Schema Update Scripts	9
2.1.3 Unloading a Module	10
2.1.4 Downloading a Model or Module	11
2.2 Model Instance Management	11
2.2.1 Creating Model Instances	13
2.2.2 Following Process Execution in Diagrams	14
2.2.3 Evaluating Expressions in Model Contexts	16
2.2.4 Changing Values of Variables	17
2.2.5 Deactivating and Activating Goals	18
2.2.6 Suspending and Resuming Model Instances	19
2.2.7 Invoking Model Instances	19
2.2.8 Finishing Model Instances	20
2.2.9 Exporting Model Instances to XML	20
2.2.10 Exporting and Importing a Model Instance	20
2.3 User Management	21
2.3.1 Creating a Person	21
2.3.2 Disabling and Enabling Persons	22
2.3.3 Exporting Persons List	23
2.3.4 Role Management	23
2.3.4.1 Managing Runtime Roles	23
2.3.4.2 Assigning Parametric Roles	24
2.3.5 Security Management	24
2.3.5.1 Creating or Editing Security Roles	25
2.3.5.2 Managing Security Roles of a Person	26
2.3.6 User Management	26
2.3.6.1 Defining Substitutes	26
2.3.6.2 Activating and Deactivating Substitution	27

2.4	Document Management	27
2.4.1	Exporting a Saved Document	27
2.4.2	Importing a Saved Document	27
2.5	To-Do Management	27
2.5.1	To-Do Detail View	29
2.5.2	Searching for Orphaned To-Dos	30
2.5.3	Delegating To-Dos	30
2.5.4	Escalating To-Dos	31
2.5.5	Cancelling Rejection	31
2.5.6	Resetting a Locked To-Do	31
2.5.7	Reassign a To-Do	32
2.5.8	Exporting and Importing To-Dos	32
2.6	Web Service Management	32
2.7	Exceptions and Logs Management	33
2.7.1	Logs	33
2.7.2	Exceptions	33
3	Management Console	35
3.1	Accessing Management Console	35
3.2	User Interface	36
3.2.1	To-dos View	37
3.2.2	To-Do Detail	38
3.2.3	Modules View	39
3.2.4	Model Instances View	39
3.2.5	Model Instance Detail	40
3.2.6	Saved Documents View	45
3.2.7	Logs View	46
3.2.8	Web Services View	46
3.2.9	Persons View	47
3.2.10	Person Detail View	48
3.2.11	Roles View	48

3.2.12	Security Roles View	49
3.2.13	Exceptions View	49
3.2.14	Model Update Logs View	50
3.3	To-Do Management	50
3.3.1	Searching for Orphaned To-Dos	51
3.3.2	Reassigning To-Dos	51
3.3.3	Escalating To-Dos	52
3.3.4	Delegating To-Dos	53
3.3.4.1	Undoing Delegation	54
3.3.5	Resetting To-Dos	55
3.3.6	Exporting and Importing a Saved To-Do for Editing	55
3.4	Module Management	55
3.4.1	Uploading Modules	56
3.4.2	Displaying Module Details	56
3.4.3	Downloading a Model or Module	57
3.4.4	Unloading Modules	57
3.5	Model Instance Management	57
3.5.1	Running a Model Instance	58
3.5.2	Filtering Model Instances	58
3.5.3	Display Model Instance Detail	58
3.5.3.1	Display Live Diagrams	59
3.5.4	Changing Model Instance Data	59
3.5.5	Exporting and Importing a Model Instance for Editing	60
3.5.6	Evaluating Expressions	60
3.5.7	Changing Persisted Data	61
3.5.8	Removing Waiting Signals	62
3.5.9	Filtering Wait Points	62
3.5.10	Suspending and Resuming a Model Instance	63
3.5.11	Finishing a Model Instance	63
3.5.12	Restarting a Model Instance	64

3.5.13	Updating a Model Instance	65
3.6	Saved Documents	65
3.6.1	Exporting a Saved Document	65
3.6.2	Importing a Saved Document	65
3.7	Person Management	66
3.7.1	Filtering Persons	66
3.7.2	Creating Persons	66
3.7.3	Editing Persons	66
3.7.4	Disabling and Enabling Persons	67
3.7.5	Activating Substitution	67
3.7.5.1	Defining Substitutes	68
3.8	Role Management	68
3.8.1	Assigning a Role to multiple Persons	68
3.8.2	Assigning multiple Roles to a Person	69
3.9	Security Role Management	70
3.9.1	Creating Security Roles	70
3.9.2	Assigning Security Roles	71
3.9.3	Deleting Security Roles	71
3.10	Log Management	71
3.10.1	Filtering Logs	71
3.11	Exception Management	72
3.11.1	Filtering Exceptions	72
3.11.2	Displaying Exception Stacktrace	72
3.11.3	Removing Exceptions	72
3.11.4	Resending Inputs on Model Instances with an Exception	72
3.12	Application Settings	73
3.12.1	Switching Languages	73
3.12.2	Changing User Details	73
3.12.3	Dynamic Themes	73
3.12.3.1	Creating Themes	73
3.12.3.2	Modifying Themes	74
3.12.3.3	Deploying Themes	74
3.12.3.4	Undeploying Themes	75
3.12.3.5	Downloading the Sampler Theme	75

4	Command-Line Console	77
4.1	Model Management	77
4.2	Model Instance Management	79
4.3	Model Update	82
4.4	Document Management	85
4.5	To-Do Management	85
4.6	Engine Management	86
4.7	Person Management	87
5	Reference	89
5.1	Security Rights	89

Chapter 1

Management

Management features serve to manage the LSPS Server and its resources:

- [modules](#)
- [model instances](#)
- User-related:
 - [persons](#)
 - [roles](#)
 - [security roles](#)
- [To-dos-related](#)
- Execution-related:
 - [logs](#)
 - [exceptions](#)

You can perform most of the related management task with any of the following tools:

- [Tools in the Management perspective](#): perspective in the PDS development environment
- [Management Console](#): web-based front-end application intended for Management of model instances, users, and the LSPS front-end applications, that is, the Process Management Console and the Application User Interface.
- [Command-Line Console](#): command-line management tool
The tool is primarily intended for application and LSPS administrators. However, also process designers and application developers might benefit from its features.

Often it is possible to perform a task in any of the tools, plus with functions and methods provided by the [Standard Library](#); the choice will depend on your preference and specific requirements, such as, automation of the task or integration in a script; ability to apply security restriction on the feature, etc.

Important: Feature accessibility might be restricted by [security rights](#) for security roles.

1.1 Module Management

Modules are created in PDS and once ready you can upload them to an LSPS Server and execute them. On the server, they live in the Module Repository. On upload, a module with all its module imports is uploaded into the Module Repository. If you need a copy of an uploaded module, you can download it from the repository. Once a module is no longer needed, you can unload it to keep the repository clean.

Whenever you are uploading a module, you need to define how to deal with any database changes. This is set as the database update strategy:

- Do nothing with database schema: Existing schema remains unchanged.
- Update the schema by model: New tables, columns and relationships are added; however, no data is deleted.
- Validate schema first: Existing schema is validated against the new model schema (if inconsistencies are detected, the model launching fails).
- Drop/create schema: Existing schema is dropped and a new schema following the new model definitions is created.

If the database already contains data and you want to upgrade the schema, [generate the schema update scripts](#) and use the scripts to migrate the schema with your business data.

To require confirmation on model upload, set *confirmModelUpload* to true. The flag is exposed as the MXBean property `com.whitestein.lsp.Settings -> Attributes -> ConfirmModelUpload`. You can change the setting also with a database insert into the LSPS_SETTINGS table:

```
INSERT INTO LSPS_SETTINGS(ID, VALUE) VALUES ('CONFIRM_MODEL_UPLOAD', 'true');
```

1.2 Model Instance Management

Model instances are runtime versions of models (a model designates an executable module with all its module imports): when you create a model instance, you "run a model". You can run multiple instances of a model possibly with different parameters.

For detailed information on behavior of Model instances, refer to the [GO-BPMN Model Language Guide](#).

1.3 Person Management

Users are represented by *persons* with personal details and access rights to the application.

A person defines the following:

- User information: user details, such as login, password
- Application data: person data, such as, name, email, phone
- [Security roles](#): enable or restrict access to general system features, for example, to to-do rejection, document submit, person management, etc.
- [Modeled roles](#): roles that define the person role in a model instance created based on organization models of the model. Typically they will restrict the access to to-dos and documents.
- [Substitution](#): list of substitutes

You can manage all of the data from the Management perspective and the Management Console.

1.3.1 Substitution

Important: The built-in substitute mechanism is provided as a base for custom substitution mechanism.

The substitution mechanism serves to define stand-ins for persons: when activated any unlocked to-dos assigned to the user are assigned also to all their substitutes. The to-dos are displayed in the to-do lists of both the users and their substitutes.

Note: If a substitute lacks a modeled role required by the to-do, they cannot see the content of a to-do.

At the moment the substitution is deactivated, all to-dos listed in the substitute's to-do list which were originally allocated to the substituted person and are not locked by the substitute are removed from the to-do list of the substitute and only displayed in the to-do list of the person.

Substitution is transitive: If a substitute has also activated their substitution, the substitution applies also the to-dos assigned due to a previous substitution.

When substitution is deactivated, any unlocked to-dos listed in the substitute's to-do list are removed; locked to-dos remain in the to-do list.

Individual users can activate and deactivate substitution from their Application User Interface, while administrators can manage substitution of all users for the *Management* perspective or web console.

1.3.2 Role Management

Modeled roles represent groups of persons that have the same role in a process. They are defined as part of a GO-BPMN Module. Once these elements are uploaded they can be assigned to persons as their runtime roles.

For parametric modeled roles, you can define the parameter values on their runtime versions.

A role is defined by the module name and its own name, *not* the module version. As a result, an organization element can define multiple parameters, which can originate from different module versions with the modeled roles.

- If another version of the same module with an updated organization definition is uploaded, the role is identified as the same role as the one created by the previous module version (assuming the role name remained unchanged).
- If another version of the same module contains the same role with a new parameter name, the parameter name is added to the already-existing parameter names of the modeled role.

You can assign a person to the role and units, and remove them as necessary either programatically using the `addPersonToRole()` function or from the management perspective and web console on runtime manually.

A typical usage of Roles is represented by To-Do performers: For example, you create an organization definition with a role. Then you create a process with a User Task. The User Task will define as its Performer parameter persons with the given role, for example, `Admins()`. When executed, the generated to-do will be allocated only to the persons with the runtime role `Admins`. Note that if you add a person to the role, the to-do is allocated to this person as well.

More information on roles and organization units is available in the [GO-BPMN Modeling Language Specification](#).

1.3.2.1 Parametric Roles

Parametric roles are modeled roles with defined parameter names. A runtime role created based on a parametric role may have a parameter value attached to the parameter name.

Runtime roles, based on parametric modeled roles, contain parameter definitions (parameter names and values).

If no parameter value is defined, the runtime role is considered superior to any roles with a parameter value and if a parameter value is defined, the runtime role is inferior to the runtime role without a defined parameter value.

Example: Consider a parametric modeled role, Developer, with a parameter, Language.

There are three persons in the system: Javan, Cee, and Deve. Each has the Developer runtime role but with a different Language value: Javan has the Developer role with the parameter Language set to Java, Cee with the parameter Language set to C, and Deve with no parameter value.

When a process dispatches a to-do with the initial performers set to the role Developer with parameter Java, the to-do will appear in to-do lists of Javan and Deve – the person with the role with no parameter value receives to-dos assigned to the role regardless of the required parameter value. However, since Cee has the parameter set to C, he will not receive the to-do.

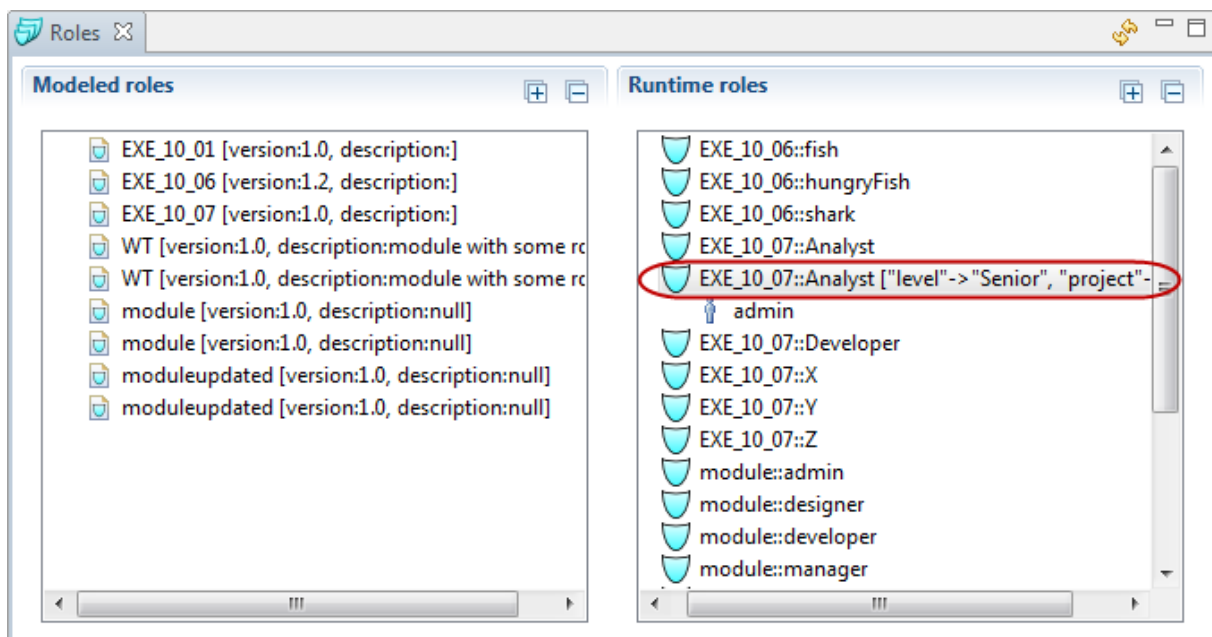


Figure 1.1 Roles view with a parameter runtime role outlined

1.3.3 Security-Role Management

Security management is implemented as a role-based security manager: A person's security rights are defined by the security roles the person has. Each security role defines the [security rights](#) it actually grants: The user can perform an action in front-end application (over runtime) only if they have a security role with the security right required for the action; otherwise, the respective feature is inaccessible or the action is interrupted.

The following predefined security roles are available:

- Admin represents a security role with rights of an administrator. Admin has all security rights and hence an unrestricted access to management features. The security role is read-only and cannot be deleted.
- ApplicationRoleManager can manage system users and application roles (both the modeled roles and the security roles).
- ProcessExecutor has access to end-user features.
- ProcessManager can manage model instances.

1.4 To-Do Management

A *to-do* represents a job that an end user has to accomplish: it is generated by the **User Tasks**, which is a type of task provided by the Standard Library.

When the execution flow enters the User Task, the task becomes *alive*, just like any other task, and the system generates a to-do. The to-do is allocated to its initial users: it is displayed in the To-Do lists of the Application User Interface to the users defined by the Performers property of the User Task.

Note: If the Performers property returns a set of Roles or Organization Units and the administrator or the system assigns or unassigns the Role or Organization Unit to a Person, the system allocates the to-do to this person accordingly.

The moment one of the users opens the to-do, it becomes *locked*: the to-do is removed from the To-Do lists of other users. Only the user who locked it, can access and accomplish the to-do. They can also decide to reset it so as to release the to-do.

As a result of such actions, assignees of the to-do change and to-do might end up with no assignees. Such to-dos are referred to as *orphaned* and require the attention of an administrator: in the Management Console, the administrator can [search specifically for such to-dos](#).

When the user accomplishes the to-do task, that is, they submit the form of the to-do: the User Task becomes *accomplished* and the underlying process workflow proceeds to the next element.

The accessibility of the actions can be restricted by [user security roles](#).

1.4.1 Delegation

>***Important:** The built-in delegation mechanism is provided as a showcase for a custom mechanism

and is under no circumstances to be considered production-ready.

Delegation enables users to assign an *alive* to-do to other users, for example, to ask for assistance or simply because they know better. When a user delegates their to-do, the to-do disappears from the to-do lists of all current assignees and appears in the to-do list of the selected users—delegates. A delegate can delegate the to-do further—delegation can take place on several levels.

1.4.2 Escalation

To-do escalation provides the user a mechanism that sends a special type of Signal, an escalation Signal, to the server while keeping the to-do locked. The underlying Model is expected to catch the signal and process it as appropriate: hence, the underlying process flow must define how the signal is caught and handled.

Important: Note that the to-do escalation mechanism is not related to the [Escalation of the GO-BPMN Modeling Language](#).

1.4.3 Rejection

Rejection allows a front-end user who is an assignee of a to-do to reject the to-do: when rejected, the to-do is removed from their to-do list and the fact is recorded by the server.

You can check the assignees who have rejected a to-do in the *Assignees* box of a to-do detail view: They are indicated by a red assignee icon. From the view, you can cancel the rejection and reallocate the to-do to the assignee again.

1.4.4 Resetting

To-do reset () erases the data in a saved to-do. This feature is useful if the data used by the saved to-do have changed; typically on model update. Note that the reset to-do remains locked by the same user. You can reset a to-do from the [management console](#) or [management perspective](#).

1.4.5 Reassignment

Important: The built-in reassignment mechanism is provided as a base for custom mechanism.

Reassignment enables an administrator to assign an Alive to-do to another set of performers, who become the initial performers. That means you can reassign a to-do to roles, organization units, or persons.

When you reassign a to-do, you modify the current performers as well as its initial performers: The to-do disappears from the to-do list of the original assignees and appears in the to-do lists of the new assignees, which become the current and initial assignees of the to-do. The information on original assignees are lost.

Important: Reassignment cannot be undone.

1.5 Logs and Exceptions

1.5.1 Logs

Logs are dedicated LSPS Application logs: they are created by the `Log` tasks and `log()` functions, which are part of the Standard Library and stored in system database table `LSPS_LOGS`.

1.5.2 Exceptions

When a model instance fails with an Exception, the Execution Engine rolls back the transaction that caused the exception. Such error on model instances is recorded in the Exceptions log. For further information on transactions in model instance, refer to [Transactions in Model Instances](#).

As part of the management features, you can resend inputs to such model instances.

Chapter 2

Management Perspective

Management perspective contains views and tools that allow you to communicate with the LSPS Server, that is, to upload and instantiate models, monitor and update model instances, manage roles and users, etc.)

Server data is presented in dedicated management views, which you can display by clicking the Management Views button () in the main toolbar. The data in the views is loaded, when the view is opened. Make sure to refresh the views so you work with the most recent data (click the Refresh () button in the toolbar of the view).

From the *Management Views* menu you can open the following views:

- [Module Management](#)
- [Model Instances](#)
- [To-do List](#)
- [Saved Documents](#)
- [Logs](#)
- [Model Update Logs](#)
- [Role Management](#)
- [Persons](#)
- [Profiling](#)
- [Security Management](#)
- [Web Services](#)
- [Exceptions and Logs](#)
- [Profiler View](#)

Note: You can display any views, not only management views, in the perspective: Go to **Window -> Show View -> Other**.

The screenshot displays the Management Perspective interface with two main panels:

- Model Instances:** A table listing model instances with columns for ID, Initiated, Status, and Model.

ID	Initiated	Status	Model
8000	2018-03-29 14:18	Finished	soapserver - 1.0 (2018-03-29 14:33:18)
8001	2018-03-29 14:14	Created	soapClient - 1.0 (2018-03-29 14:34:13)
8002	2018-03-29 14:14	Created	soapClient - 1.0 (2018-03-29 14:34:43)
8003	2018-03-29 14:15	Finished	soapClient - 1.0 (2018-03-29 14:34:14)
20000	2018-04-24 15:15	Finished	soapserver - 1.0 (2018-04-24 15:59:39)
20001	2018-04-24 16:16	Finished	soapClient - 1.0 (2018-04-24 16:00:01)
20002	2018-04-24 16:16	Running	soapserver - 1.0 (2018-04-24 15:59:39)
20003	2018-04-24 16:16	Running	7063 - 1.0 (2018-04-24 16:04:02)
- Module Management:** A table listing modules with columns for Name, Executable, Upload Date, Unfinished Instances, and Hash.

Name	Executable	Upload Date	Unfinished Instances	Hash
core - 3.2		2018-03-29 14:43:14		2549aa4fcc27415084d0
core - 3.2		2018-04-24 16:00:01		30cdeb832444431c9a8b2
human - 3.2		2018-03-29 14:43:14		aeb71965b4fd9eb0972c
human - 3.2		2018-04-24 16:00:01		c58eddf188ad98813ca
soapClient - 1.0		2018-03-29 14:34:13	1	3e96d3a9038d0a6b1861
soapClient - 1.0		2018-03-29 14:34:43	1	19678cbb018f62c4f6bb
soapClient - 1.0		2018-03-29 14:43:14	0	18ae503ae6989a81c0a0
soapClient - 1.0		2018-04-24 16:00:01	0	dedc342208f1f0941e23
soapserver - 1.0		2018-03-29 14:33:18	0	04e40dddec915bdc5a
soapserver - 1.0		2018-04-24 15:59:39	0	bb6ad7621b9f136c926e
ui - 3.2		2018-03-29 14:43:14		087895d074b0861ffdc
ui - 3.2		2018-04-24 16:00:01		5adbca2f201ea6ca6fc71
ulcommon - 3.2		2018-03-29 14:43:14		3ad92019ae664d68d2et
ulcommon - 3.2		2018-04-24 16:00:01		497c51322cfd9290c44c3

Below the Module Management table is a description field and a pagination bar showing "Page 1 of 1, total items: 14".

The bottom panel shows the **Error Log** with a table of workspace logs:

Message	Plug-in	Date
Keybinding conflicts occurred. They may interfere with normal accelerator operation.	org.eclipse.jface	4/2
A conflict occurred for CTRL+SHIFT+V:	org.eclipse.jface	4/2
Binding(CTRL+SHIFT+V, ParameterizedCommand(Command(org.eclipse.jdt.debug.ui.command.OpenFr	org.eclipse.jface	4/2
A conflict occurred for ALT+SHIFT+L:	org.eclipse.jface	4/2
Binding(ALT+SHIFT+L, ParameterizedCommand(Command(org.eclipse.jdt.ui.edit.text.java.extract.loc	org.eclipse.jface	4/2
A conflict occurred for CTRL+F:	org.eclipse.jface	4/2
Binding(CTRL+F, ParameterizedCommand(Command(org.eclipse.ui.edit.findReplace,Find and Re	org.eclipse.jface	4/2
Find and replace text, ... (Open log entry details for full message)		

Figure 2.1 Management perspective

Related topics:

- [Model and Module Management](#)
- [Model Instance Management](#)
- [User Management](#)
- [Document Management](#)
- [To-Do Management](#)
- [Web Services Management](#)
- [Logs and Exceptions](#)

2.1 Model and Module Management

You can manage Model instances and Modules on the LSPS Server from the Module Management view. The view contains the list of uploaded Modules with their details. Using the buttons in the view toolbar, you can perform Module-related actions:

- **Upload** (): upload a Model exported in GO-BPMN Export to the LSPS Server;
- **Download** (): download the uploaded Model or Module to a zip archive (you can import it to your workspace in the Process Design Suite using the GO-BPMN Import);

- **Unload** (): delete a model from the LSPS Server.

The Filters () button allows you to filter the list of uploaded Models based on their name or version.

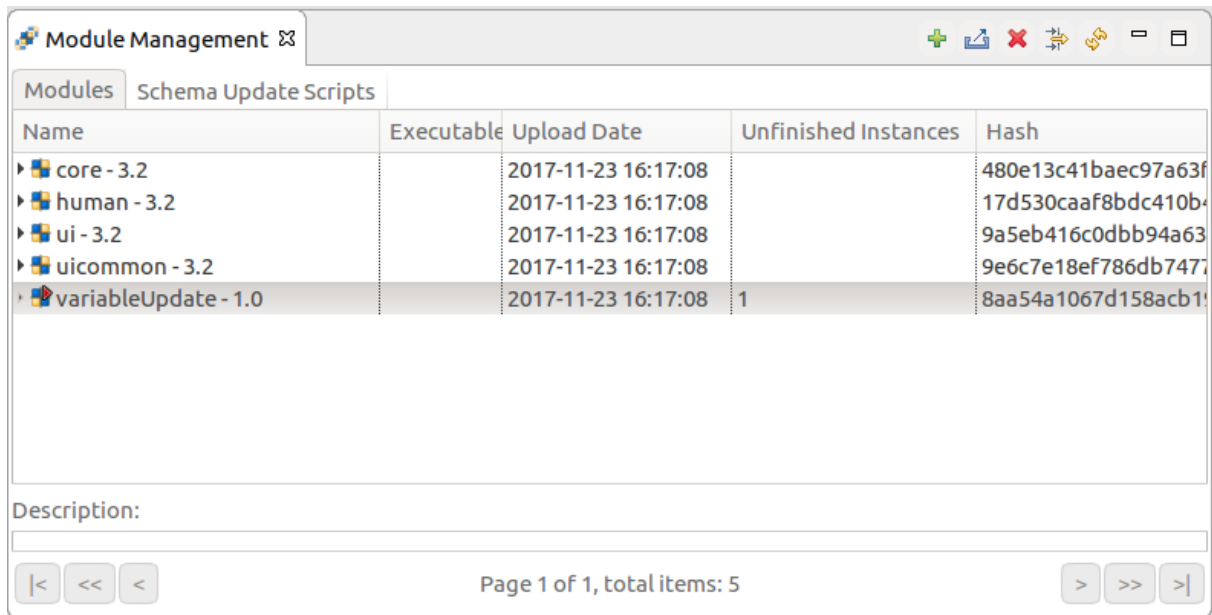


Figure 2.2 Module Management view

2.1.1 Uploading a Module

To upload a module from the Management perspective of your PDS, do the following:

1. Make sure your PDS is connected to the correct LSPS server.
2. In the Module Management view, click the **Upload**  button.
3. In the Module Upload dialog, define the upload details:
 - (a) Select either a zip file with an exported module on your filesystem or a module from your workspace.
 - (b) Define the [database update strategy](#).
 - (c) Unselect the *Do not execute schema update* scripts flag.
4. Click **OK**.

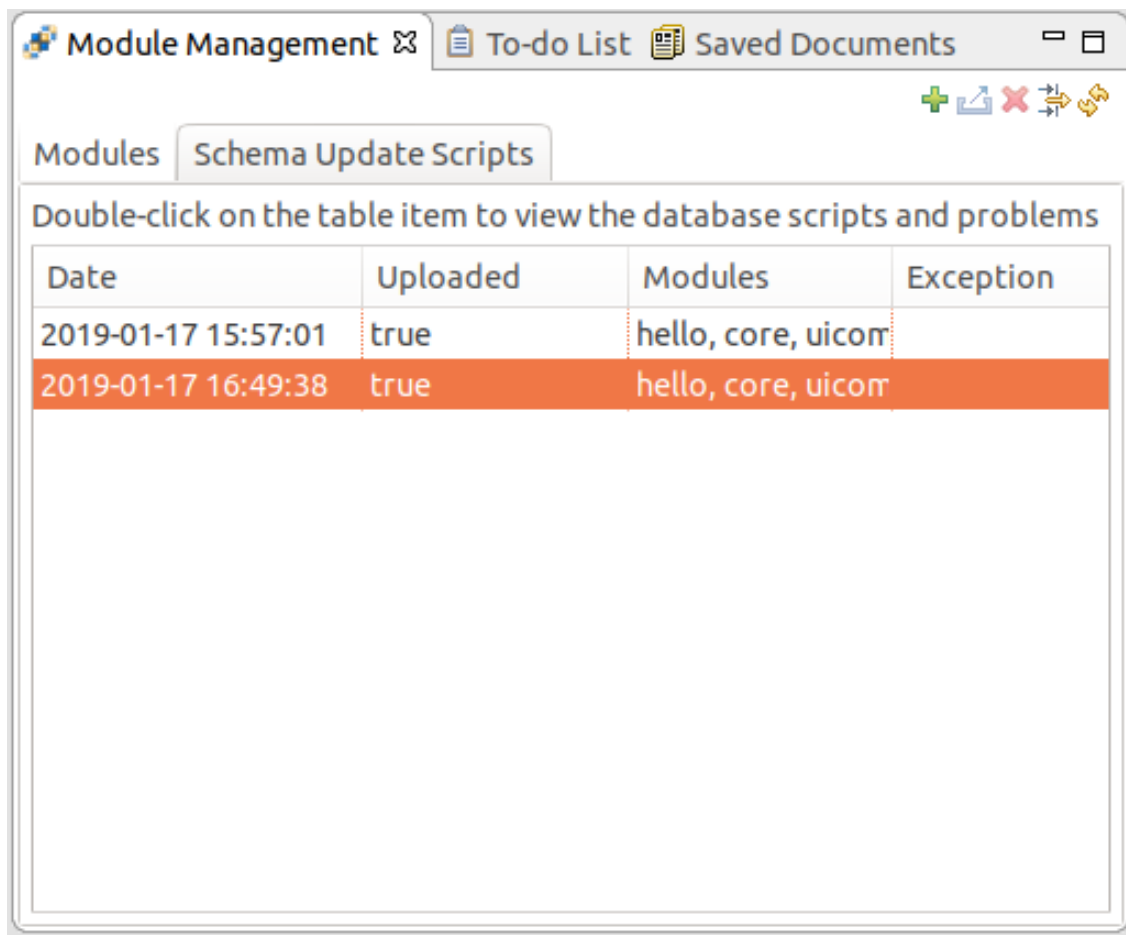
2.1.2 Generating Schema Update Scripts

When upgrading the data model of your model to a version which is not backward compatible and you want to migrate the existing data or migrate the underlying database schema manually, generate the schema update scripts for your models, modify them as required and apply them on the database.

To generate the schema update scripts for your modules do the following:

1. Open PDS and connect to the LSPS Server with the current modules.

2. Switch to the Management perspective.
3. In the Module Management view, click the **Upload**  button.
4. In the Module Upload dialog, define the details:
 - (a) Select either a zip files with the new modules or modules from your workspace.
 - (b) Set *Database Schema Strategy*: Set *Update the schema by model* to generate scripts that will update the current schema or *Drop/Create* to generate scripts that will create the new schema from scratch.
 - (c) Select the *Do not execute schema update scripts* flag: This prevent the schema update scripts execution and makes the scripts available in the tab *Schema Update Scripts*. You can find details such as lists of problems and conflicts discovered in the current schema in the script details.
5. Adapt the scripts as necessary:
 - Include the data stored in the tables of the old schemaUpdateScripts
 - *Do not remove* any tables or columns used by either the new or old models if their instances will be updated as part of migration (the new and old schema must be backward-compatible).



2.1.3 Unloading a Module

On unload, the module is deleted from the Module Repository, and if a module is also a model, its running Model instances are terminated. Roles and role assignments are preserved (for further information, refer to [Role Management](#)).

Note that some exploring features may become unavailable.

To unload a Module from the LSPS Server, do the following:

1. In the Module Management view select the model you wish to unload.
2. Click the Unload () button in the view toolbar.
3. In the Warning dialog box, click Yes.

The Module is unloaded and no longer available. Any Model instances based on the Model are terminated.

2.1.4 Downloading a Model or Module

You can download an uploaded Model or Module from the Model Repository as an archive file and then import the Module into the Process Design Suite with the GO-BPMN Import and edit it.

To download a Module from an LSPS Server, do the following:

1. Make sure PDS is connected to the correct server.
2. In the Module Management view, select the model.
3. Click the Download button.
4. In the Save As dialog box, define the target location and edit the file name if necessary.

2.2 Model Instance Management

To work with Model instances on the LSPS Server, use the Model Instances view with the list of Model instances on the server. Using the buttons in the toolbar, you can do the following:

- **Create**  a new model instance
- **Suspend**  a model instance
- **Resume**  a suspended model instance
- **Finish**  a running or suspended model instance
- **Perform and manage model update:**
 - **Update**  update a model instance based on a muc definition
 - **Continue Model Update**  trigger the next model update phase
 - **Abort Model Update**  of a model instance
- **Export runtime data of the selected instances to XML** .
- **Export the model instance list**  to a CSV file
- **Filter**  the model instances list
- **Refresh**  the view content

The action commands as well as additional actions are available also in the context menus of individual model instance entries:

- Show To-dos  generated by the model instance (filter for the respective model instance ID is applied to the To-do List view and the view is focused).
- Show Logs  generated by the model instance (filter for the respective model instance ID is applied in the Logs view and the view is focused).
- **Suspend**  and **Resume** () a model instance.
- **Finish**  a running or suspended model instance.
- Perform actions of **Model update**:
 - Update  a model instance based on a muc definition.
 - Continue Model Update  trigger the next model update phase.
 - Abort Model Update  of a model instance.
- **Export**  a model instance into an XML.
- **Refresh**  the view content.

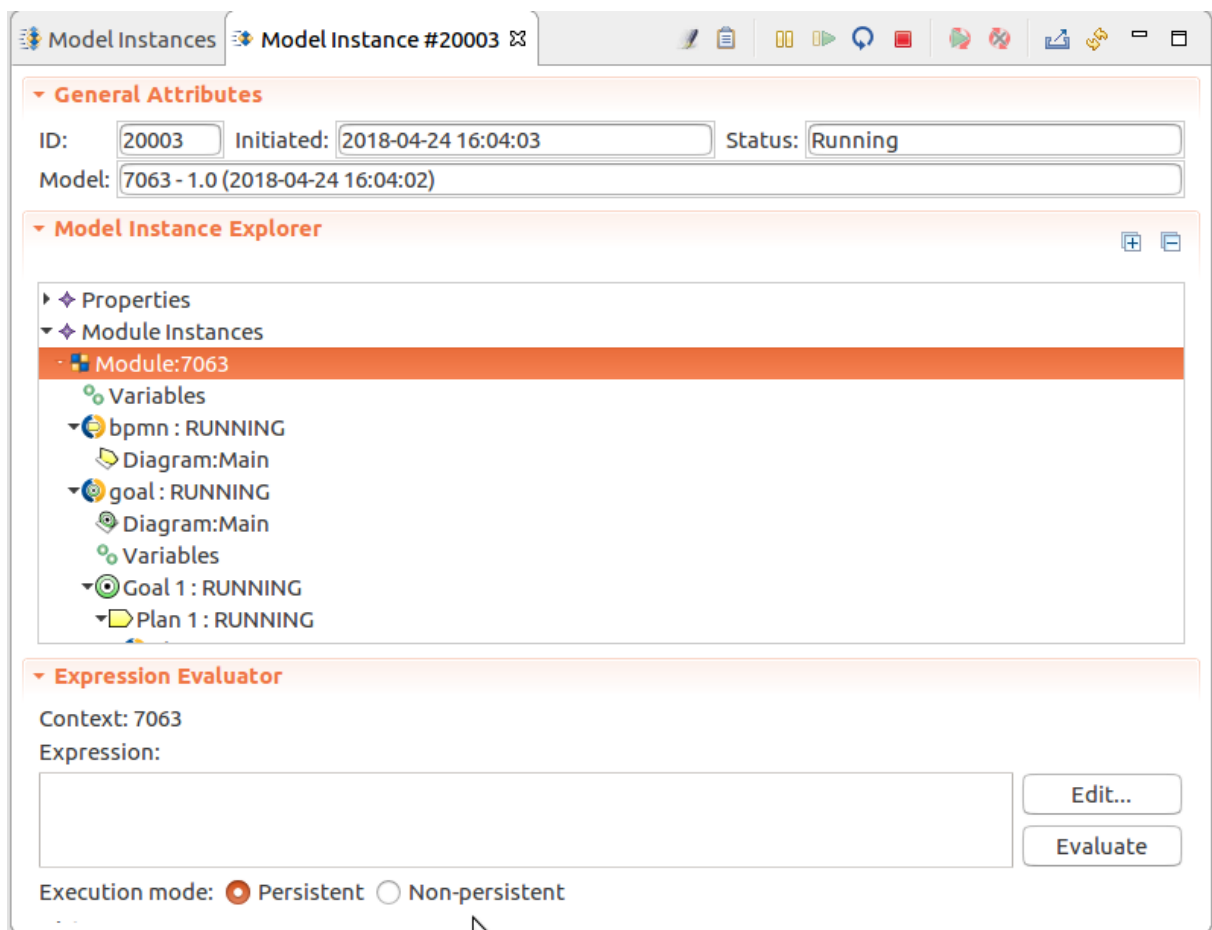


Figure 2.4 Model instance detail view

2.2.1 Creating Model Instances

To create a Model instance, do the following:

1. Open the Model Instances view.
2. Click the **Create**  button in the view toolbar.
3. Select one of the uploaded models and click **Finish**.

The new model instance appears in the list of model instances.

2.2.2 Following Process Execution in Diagrams

You can visualize the current status of Model instance resources in their live diagrams: Live diagrams are diagrams of Process resources as defined in their Model (process, goal, plan, or sub-process diagrams). Individual execution states are indicated by colors. Diagram legend can be displayed by clicking the legend button  in the view toolbar.

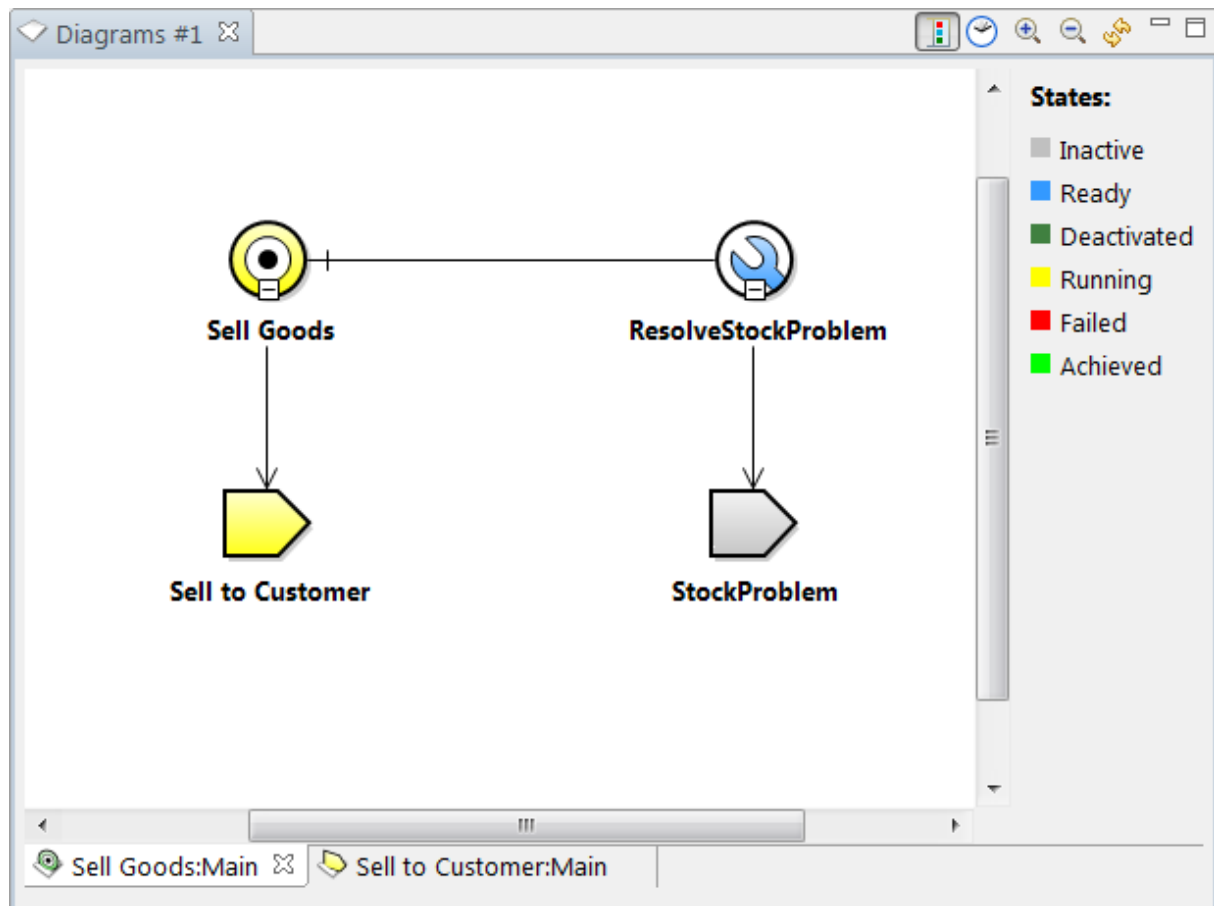


Figure 2.5 Live Goal diagram with legend

Goals with an implicit hyperlink (indicated by a small arrow in the top-left corner of the icon on the canvas), plans, and sub-processes are clickable: Click the element to open the respective linked live diagram.

If several live diagrams of the same model instance are activated, they are opened in the same view stacked on each other.

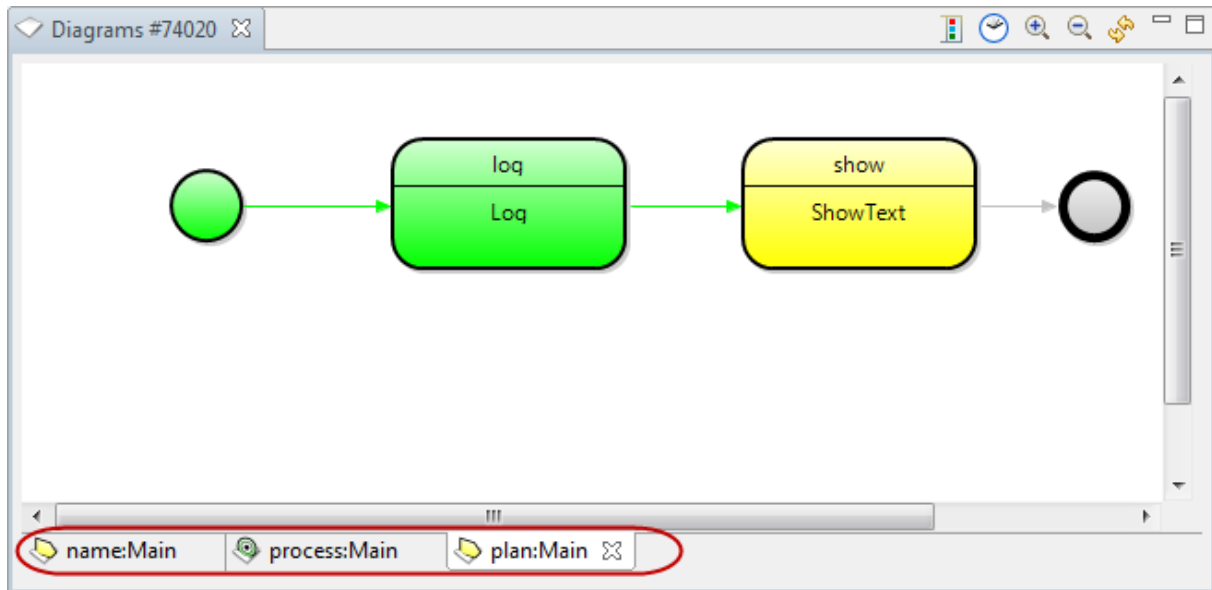


Figure 2.6 Live diagram with multiple pages

To display a live diagram, open the respective model instance detail view and double-click the diagram icon under Module Instances or Model Update History in the tree (also available for old module instances from before model update if applicable).

To visualize the history of the execution flow on the live diagram, click the History  button in the view toolbar: You can then use the slider at the bottom of the view to view the execution workflow.

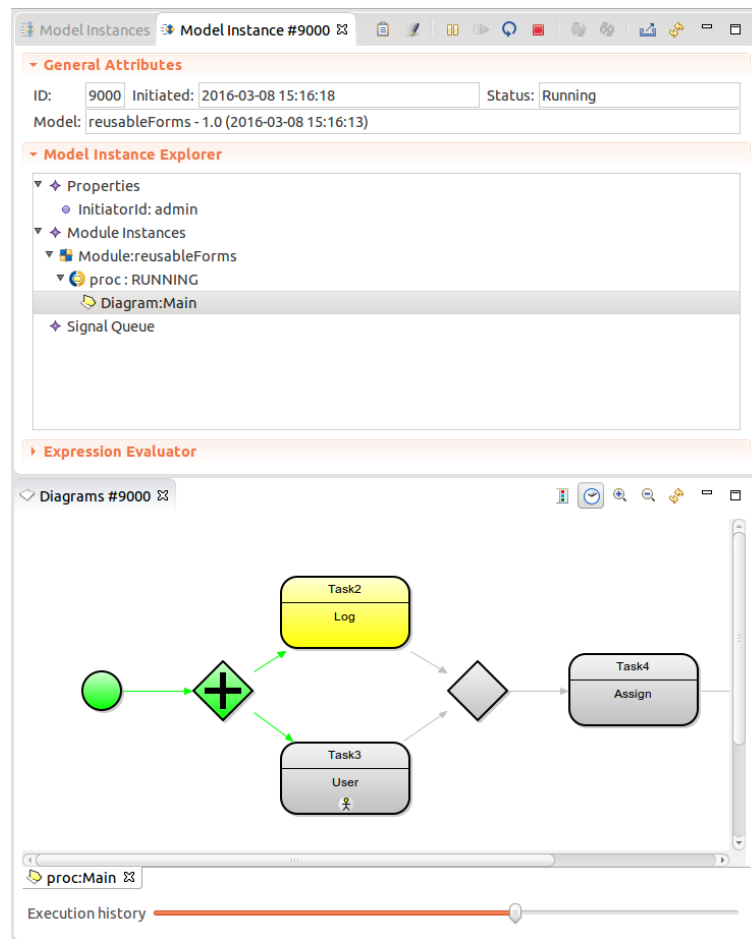


Figure 2.7 Live diagram with execution history slider

You can change the status colors used to indicate the element execution status in live diagrams under **Window > Preferences > Process Design Suite > Management > Appearance**.

2.2.3 Evaluating Expressions in Model Contexts

To evaluate expressions using the current context values of a Model instance, use the Expression Evaluator in the Model instance detail:

1. Open the Model instance detail view.
2. Expand the Expression Evaluator section.
3. In the tree, select the target context.

The context of the Expression Evaluator is set to the selected context: you can check the current context in the Expression Evaluator view.

The expressions can be executed in persistent or non-persistent mode:

- if in non-persistent mode, the expression is evaluated in the model instance context;

- if in persistent mode, the expression is evaluated and applied on the model instance context and on the database data if Shared Records are involved.

Information on the selected context is displayed in the Expression Evaluator: this is by default the Model instance context. To change the context to a Module, Process, Sub-Process context, select the respective element in the Model instance tree.

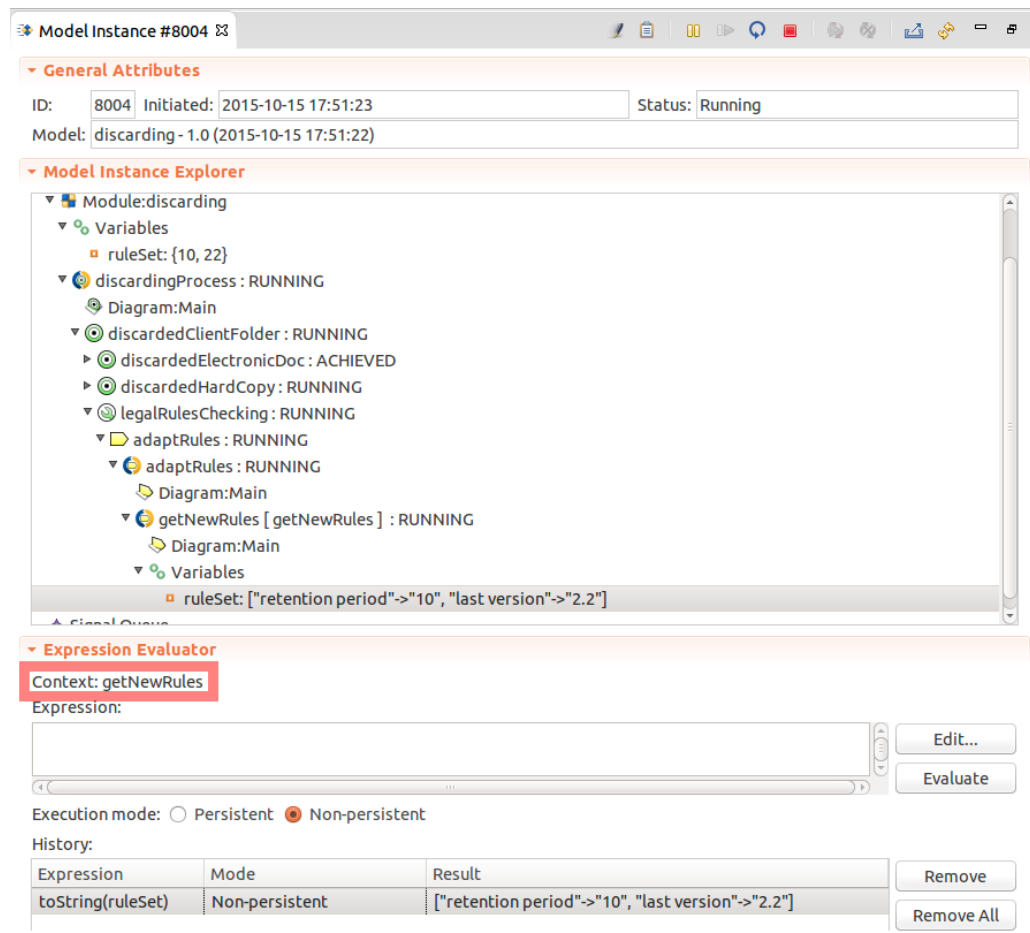


Figure 2.8 Expression Evaluator

2.2.4 Changing Values of Variables

You can change values of variables during execution either from the Expression Evaluator using assignments or from the Model instance detail.

To change the value of a variable in a running model instance, do the following:

1. Open the Model instance detail view.
2. Under Model Instance Explorer, expand Module Instances and the desired Module node (parent namespace of the context variable).
3. Expand the Variables node and double-click the required variable. Variables of a standard library type cannot be edited.
4. In the Update Variable dialog box, select:

- Value to assign the variable a particular value;

Note: Depending on the variable type (basic, record, collection, etc.), the Update Variable dialog box shows relevant buttons. For basic values, provide the desired value directly; make sure you follow the Expression Language rules. Collection and record values are edited recursively, that is, if you click Edit, there are gradually "split" in less complex values, until basic values are displayed.

- Expression and type an expression to be evaluated and used as the variable value. Note the information on the declared type of the variable and the current actual type.

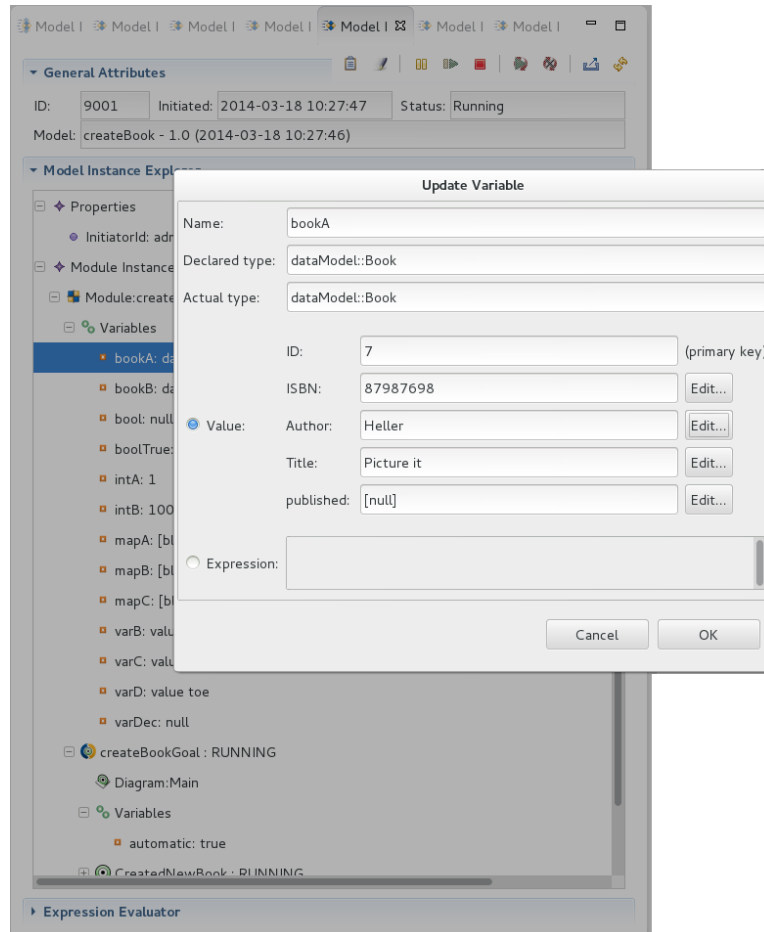


Figure 2.9 Updating variable

2.2.5 Deactivating and Activating Goals

Changing execution states of Goals during execution can help you to direct the flow of the execution. Note that you can activate or reactivate only achieved, failed, deactivated or inactive goals, and **deactivate only ready or running goals**.

To deactivate or activate a Goal, do the following:

1. Open a Model instance detail view.
2. Under Model Instance Explorer, expand Module Instances and the desired module node.
3. Expand the respective process, then right-click a goal and select Deactivate or Activate.

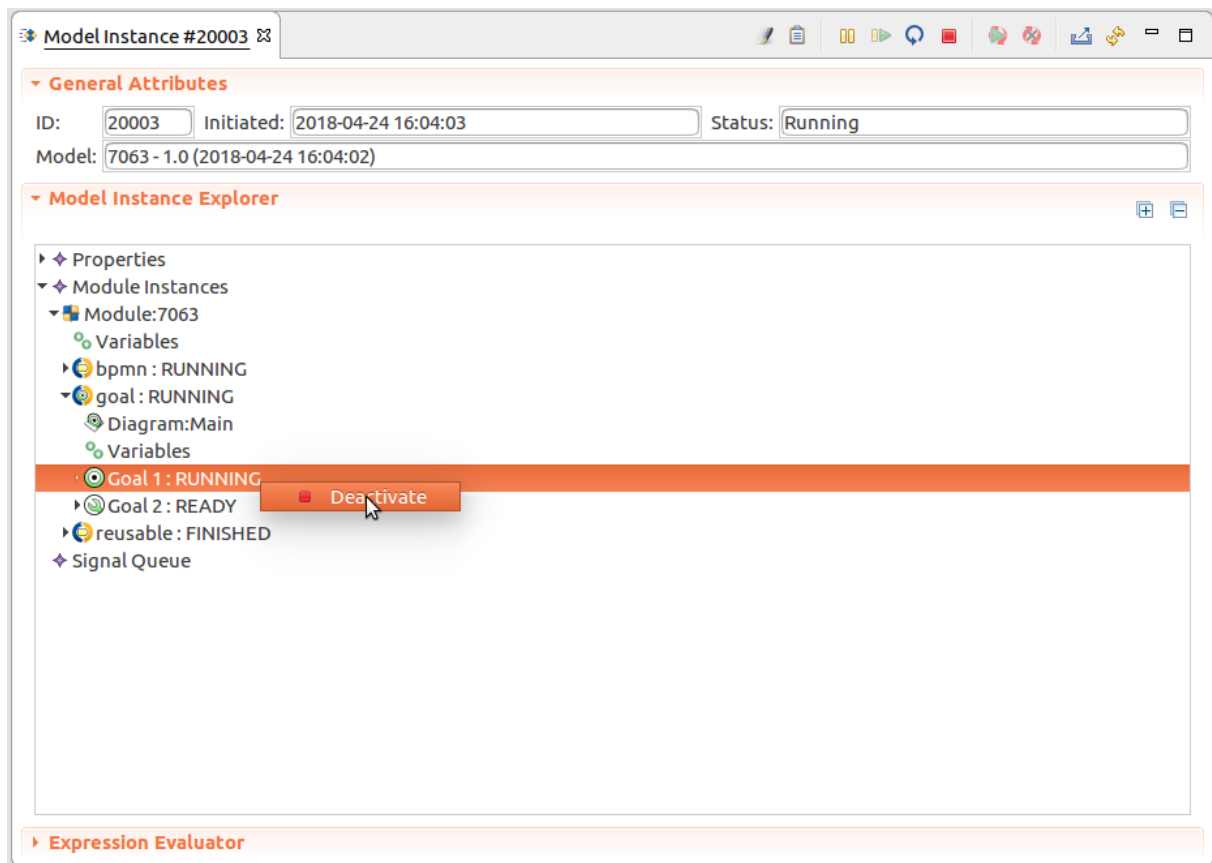


Figure 2.10 Deactivating a Goal

You can activate or deactivate a Goal also from its context menu in a live diagram; mind that disabled goals cannot be deactivated or activated.

2.2.6 Suspending and Resuming Model Instances

To suspend a running Model instance or resume a suspended Model instance, do the following:

1. Activate the Model Instances view.
2. Select the instances you wish to suspend/resume.
3. In the view toolbar, click Suspend  or Resume .

On detailed behavior of Model instances, refer to [GO-BPMN Model Language User Guide](#).

The command is available also in the context menu of individual model instances.

2.2.7 Invoking Model Instances

On invoke, a Model instance attempts to continue its execution: this feature is helpful if a Model instance remains in a state where it is waiting for an event which cannot occur, typically due to manual adjustments (for example, change on timer events or variable values), you can invoke the model instance to make the instance check if it can continue.

To invoke a model instance, go to Model Instances view, select the model instance and click the Invoke  button.

2.2.8 Finishing Model Instances

The Finish feature allows you to immediately terminate a Model instance.

To finish a running Model instance, do the following:

1. Open the Model Instances view.
2. Select the instances, you wish to finish.
3. In the view toolbar, click Finish . The command is available also in the context menu of individual model instances.

2.2.9 Exporting Model Instances to XML

You can export Model instances as formatted XML files. Note that the XML file contains only information on the runtime status of the Model instance, not the underlying Model, and serves as an information source, for example, during Model update.

To export a model instance to an XML file, do the following:

1. Double-click the model instance in the Model Instances to display the model instance detail.
2. In the view toolbar, click Export . The command is available also in the context menus of individual model instances in the Model Instances view.
3. In the Model Instance Export dialog box, define export details:

2.2.10 Exporting and Importing a Model Instance

If you want to modify runtime data of a model instance, you can do so directly in its raw XML. Such data includes values of variables, signals, etc. Note that if you modify data that is persisted in the database, the data remains unchanged.

To export a model-instance XML, do the following:

1. Open the *Model Instances* view.
2. Right-click the model instance and select **Export Raw Data** or **Import Raw Data** respectively.

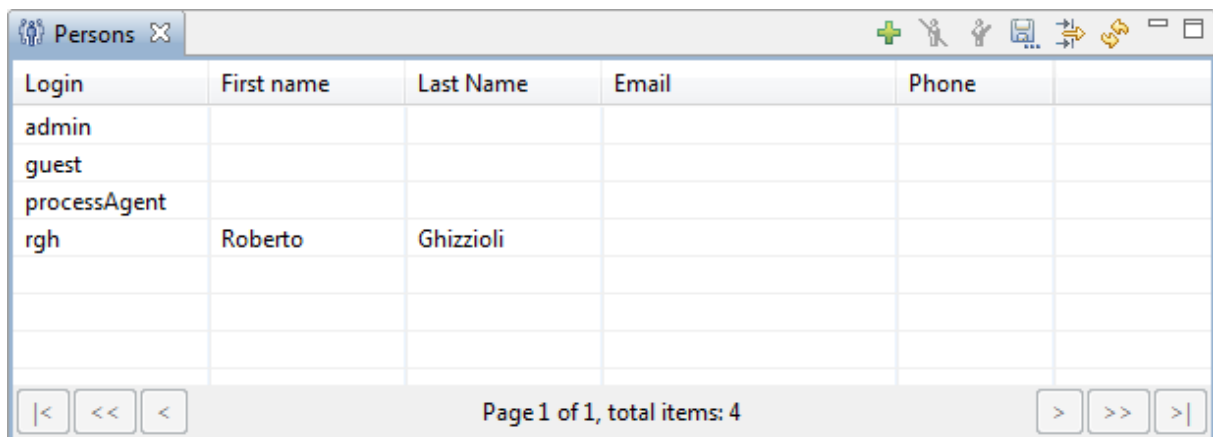
Important: Importing a corrupted model instance XML can cause that the system will fail to work with the data.

1. In the Model Instance Export dialog box, define export details.
-

2.3 User Management

Users are represented by *persons* with personal details and access rights to the application, which include appropriate [security](#) and [modeled Roles](#). Optionally they can define their [substitutes](#), that is, persons who can act on behalf of the person when the person activates their substitution (stand-ins).

List of available Persons is displayed in the Persons view. By double-clicking a person entry in the Persons view, you can display the person detail view with the person's data and settings.



Login	First name	Last Name	Email	Phone
admin				
guest				
processAgent				
rgh	Roberto	Ghizzioli		

Page 1 of 1, total items: 4

Figure 2.11 Persons view

2.3.1 Creating a Person

To create a new person, do the following:

1. Open the **Persons** view.
2. Click the **Add** button  in the view menu.
3. Enter the data in the displayed person detail.
4. Click the **Save** button .

Person: JamesD

User Information

Login: JamesD

Password:

Password confirmation:

State: Enabled

Application Data

First Name: James

Last Name: Dean

Email: example@example.sk

Phone:

Security Roles

ApplicationRoleManager [readRole, readPerso]

Manage Roles

Modeled Roles

Common::Seller

Common::Stockkeeper

Manage Roles

Substitution

☐ Substitution active

processAgent

Manage Substitutes

Figure 2.12 Person detail view

2.3.2 Disabling and Enabling Persons

Once a person is created, it is not possible to remove them. Instead of removing a person from the system, disable them: A person that was disabled, cannot access the LSPS Server.

When persons are disabled, the following applies:

- Any assigned security and modeled roles, and substitutes, if defined, are permanently removed.
- Any locked to-dos are unlocked and released, and assigned to another person with the respective roles. Note that if no such person is available, the to-do becomes orphaned.

A disabled person can be recovered (enabled). However, none of the removed data is retrieved.

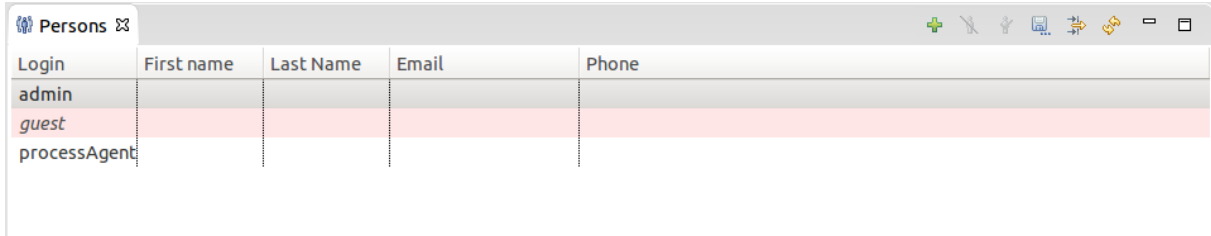
Information about person state is shown in the respective person detail view under User Information area. In the Persons view, disabled persons are shown highlighted in red.

To enable or disable a person, do the following:

1. Open the Persons view.
2. Select the person.

3. In the view toolbar, click Disable () or Enable ().

Note: The *admin* and *processAgent* cannot be disabled. In addition, the admin user is fully read-only.



Login	First name	Last Name	Email	Phone
admin				
guest				
processAgent				

Figure 2.13 Persons view with the guest person disabled

2.3.3 Exporting Persons List

To create a new person, do the following:

1. Open the **Persons** view.
2. Filter the data so that only the entries you want to export are displayed.
3. Click the **Export Displayed Content to CSV** button  .

2.3.4 Role Management

2.3.4.1 Managing Runtime Roles

To assign a runtime role to one or more persons, do the following:

1. Open the Roles view.
2. Under Runtime Roles, right-click the respective role and select Manage Persons. If the modeled role was not assigned to any persons yet, it is not available. Go to the respective person detail and assign the person at least one modeled role.
3. In the Filter box of the Persons dialog box, type the person's name or its part. To display all available persons, type * (asterisk).
4. Select a person in the box below Filter and click the right two-headed arrow to add the person to the Resulting selection.
5. Click OK.

2.3.4.2 Assigning Parametric Roles

To define a parameter value of a modeled parametric role:

1. Open the respective person detail view.
2. Under Modeled Roles, click Manage Roles.
3. In the Resulting selection area, select the parametric role.
4. Edit parameter values and click Add Role.

The added parameter value is shown in the Resulting Selection.

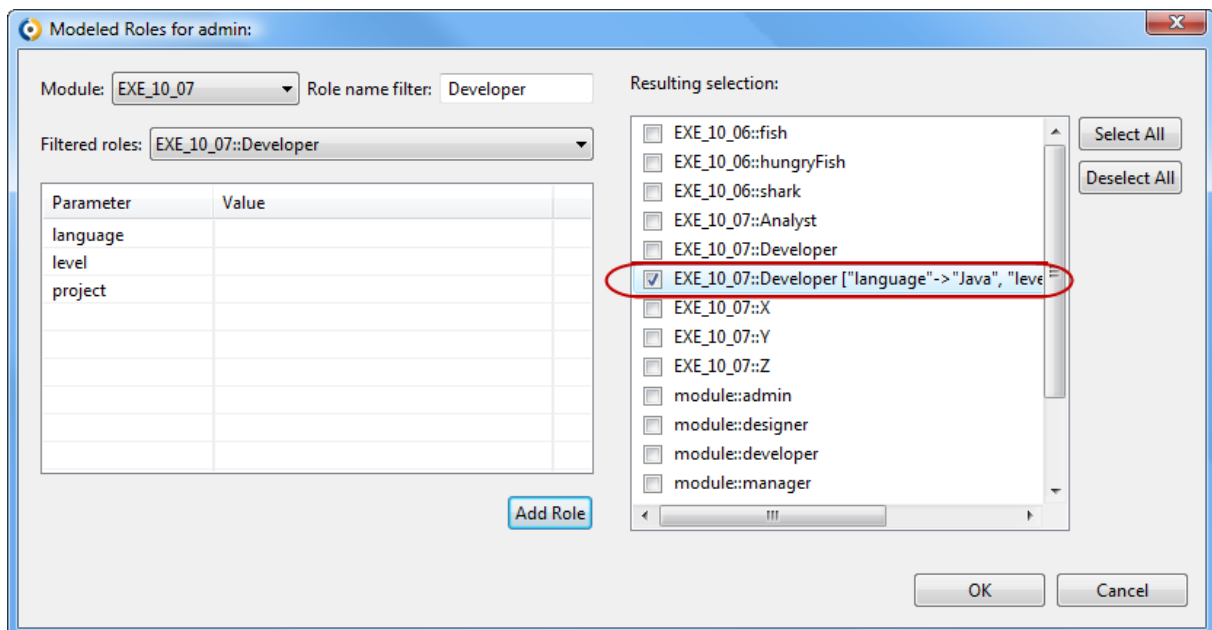
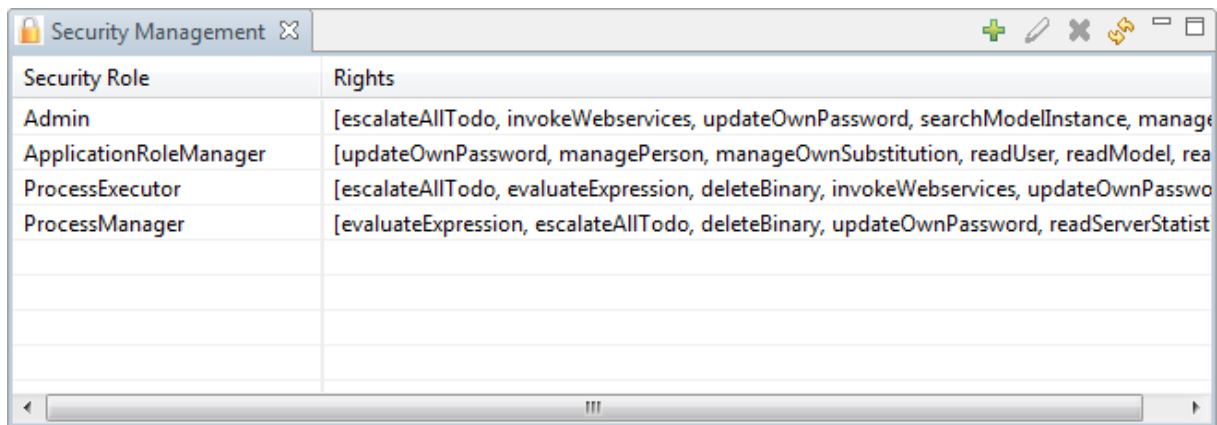


Figure 2.14 A parametric role with a parameter

2.3.5 Security Management

To manage the security rights of a person, you will create or modify a security role if necessary and assign the security role to the person.

Important: When creating a new security role or modifying the rights of an existing security role make sure to *restart the server*.



The screenshot shows a window titled "Security Management" with a toolbar containing icons for adding (+), editing (pencil), deleting (X), and other functions. The main content is a table with two columns: "Security Role" and "Rights".

Security Role	Rights
Admin	[escalateAllTodo, invokeWebservices, updateOwnPassword, searchModelInstance, manage...
ApplicationRoleManager	[updateOwnPassword, managePerson, manageOwnSubstitution, readUser, readModel, rea...
ProcessExecutor	[escalateAllTodo, evaluateExpression, deleteBinary, invokeWebservices, updateOwnPasswo...
ProcessManager	[evaluateExpression, escalateAllTodo, deleteBinary, updateOwnPassword, readServerStatist...

Figure 2.15 Security Management view

2.3.5.1 Creating or Editing Security Roles

To create a security role, do the following:

1. Open the Security Management view.
 2. In the Security Management view, click Add () or Edit () in the view toolbar.
 3. In the Role Name text box of the Security Role Definition dialog box, type the security role name.
 4. Select the security rights to assign to the role, and click OK.
-

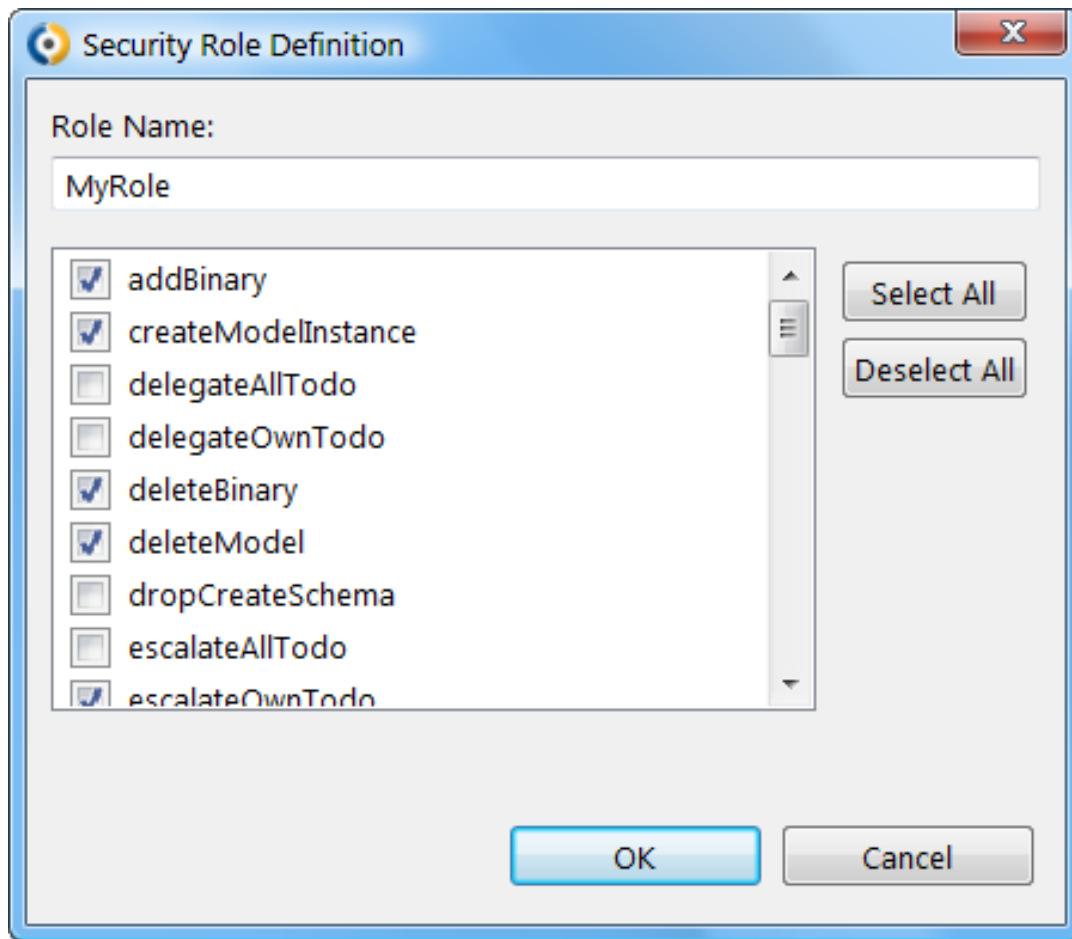


Figure 2.16 Security role definition dialog

5. Restart the server.

2.3.5.2 Managing Security Roles of a Person

To add or remove person's security roles, do the following:

1. Display the respective person detail view.
2. Under Security Roles, click Manage Roles.
3. In the displayed dialog box, select the security roles you wish to assign the person, and click OK.

2.3.6 User Management

2.3.6.1 Defining Substitutes

To define a person's substitutes, open the Person Detail view and in the Substitution section, click **Manage Substitutes**. In the pop-up define the substitutes.

Check if the Substitution is activated for the person.

2.3.6.2 Activating and Deactivating Substitution

The Substitution feature can be applied from the front-end application as well as from PDS. To activate or deactivate substitution on a person from PDS, open the Person Detail view and select or unselect the **Substitution active** option.

2.4 Document Management

The documents users have saved and are hence probably working on are available in the *Saved Documents* view. You can export the documents as XMLs and import them back after adapting the runtime data. Note that the features are subject to the security right `Document:Write`.

2.4.1 Exporting a Saved Document

To export a saved document to XML, do the following:

1. Open the *Saved Documents* view.
2. Right-click the document and select *Export Saved Document State*.

You can edit the downloaded XML and then import it back into the document.

2.4.2 Importing a Saved Document

To import a saved document XML, do the following:

1. Open the *Saved Documents* view.
2. Right-click the document and select *Import Saved Document State*.

Important: Importing a corrupt document can cause the system to fail to work with the document.

2.5 To-Do Management

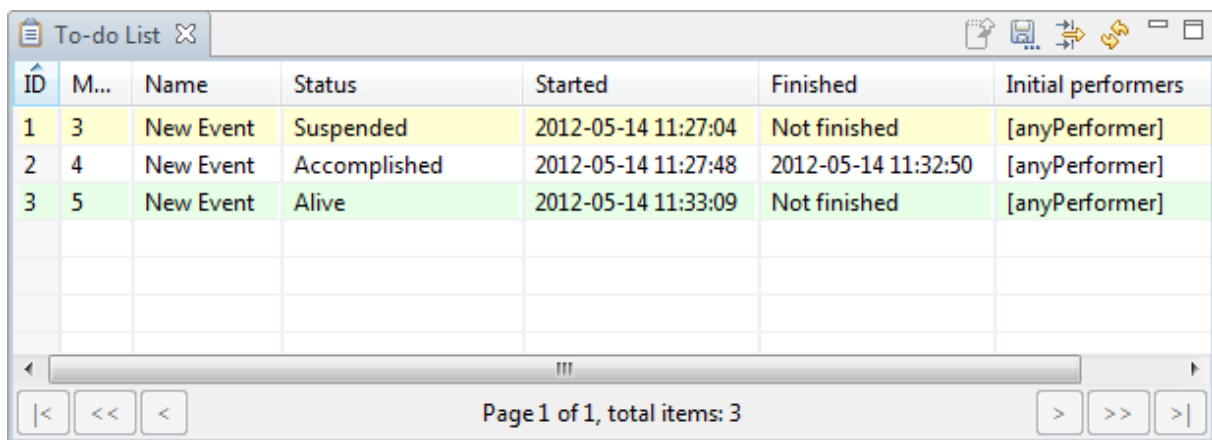
To manage To-Dos generated by the LSPS Server and to open their details use the *To-Do List* view.

The view contains the following To-Do details:

- ID: unique to-do identifier
 - Model Instance ID: identifier of the model instance the to-do was generated by
 - Name: title of the to-do as displayed in the to-do list of a front-end application
-

- **Status:** current to-do execution status (a to-do can be Alive, Accomplished, or Interrupted; Alive to-dos are highlighted)
- **Note:** If its parent model instance is suspended, a to-do is also in the suspended status; however, this is not a to-do status by itself.
- **Started:** date and time when to-do was generated
- **Finished:** date and time when to-do was submitted or its parent humanoid task finished
- **Initial Performers:** set of roles or persons, who are entitled to perform the to-do as defined for the model instance at its instantiation
- **Locked By:** person who has locked the to-do

You can copy the table rows to the clipboard by selecting them and pressing Ctrl + C and [export and import XML to-do states](#).



ID	M...	Name	Status	Started	Finished	Initial performers
1	3	New Event	Suspended	2012-05-14 11:27:04	Not finished	[anyPerformer]
2	4	New Event	Accomplished	2012-05-14 11:27:48	2012-05-14 11:32:50	[anyPerformer]
3	5	New Event	Alive	2012-05-14 11:33:09	Not finished	[anyPerformer]

Page 1 of 1, total items: 3

Figure 2.17 To-Do List view

You can view further To-Do details as well as perform actions on the to-do from its detail view: To open a to-do detail view, double-click a to-do entry in the *To-do List* view.

You can do the following when managing to-dos:

- [Delegate](#) to reversibly assign a to-do to other users
- [Escalate](#) to escalate the to-do so the underlying process processes the escalation signal
- [Cancel rejection](#) to overrule a reject from a user
- [Reset](#) allows a front-end admin user to reset to-dos
- [Reassign](#) to assign an *alive* to-do to other users, who become its initial performers

The accessibility of the actions can be restricted by user security rights.

For explanation of the concepts behind the actions, refer to [To-Dos](#).

Important: The to-do management mechanism are provided as a base for custom to-do management features and are in no way to be considered production-ready.

2.5.1 To-Do Detail View

A To-Do Detail view contains detailed information about a particular to-do and allows you to perform further actions available for the to-do, such as delegation, undo delegation, and cancel rejection.

To open a to-do detail view, click a to-do entry in the To-do List view.

A to-do detail view contains two areas:

- To-do Details: general information about the to-do
 - **To-do ID:** unique to-do ID
 - **Model instance ID:** ID of the owning model instance
 - **External link:** external link to the to-do
 - **To-do title:** title of the to-do
 - **Task name:** name of the task, which generated the to-do (its parent module, process definition, and task name)
 - **Task status:** execution status of the human task
 - **Issued:** date and time when the to-do was generated
 - **Finished:** date and time when the to-do was submitted or the respective human task finished (contains Not finished value, if the to-do is alive)
 - **Interruption reason:** reason why the to-do was finished (for example, timeout expired, cancel intermediate event was triggered, to-do was submitted, etc.)
 - **To-do notes:** notes added by front-end application users who had the to-do allocated or are processing it
- **Assignment:**
 - **Locked by:** the person who has [locked the to-do](#)
 - **Initial performers:** set of roles (or persons), who are entitled to perform the to-do as defined in the parent model
 - Note:** This set does not include substitutes or delegates, who may have the to-do assigned by initial performers.
 - **Assignees:** list of the current set of roles or persons that can perform a to-do, notes on rejection (indicated by red person icon), and the current delegation level

From the view toolbar in the upper right corner of a to-do detail view, you can perform allocation related actions (for more information refer to [Escalation](#), and [Delegation](#)).

The screenshot shows a web application window titled "To-do: New Event". It is divided into two main sections: "To-do Details" and "Assignment".

To-do Details:

- To-do ID: 3
- Model Instance ID: 5
- External link: <http://localhost:8080/lsp>
- To-do title: New Event
- Task name: Form::Form::'Task 1':Tas
- Task status: Alive
- Issued: 2012-05-14 11:33:09
- Finished: Not finished
- Interruption reason: (empty field)
- Todo notes: (empty text area)

Assignment:

- Locked By: (empty field)
- Initial performers: anyPerformer
- Assignees: delegation level 0
- Assigned users:
 - admin (admin)
 - guest (guest)
 - processAgent (processAgent)
 - rgh (Roberto Ghizzioli) (rejection reason: V
- Cancel Rejection button

Figure 2.18 To-Do Detail view of a rejected To-Do

2.5.2 Searching for Orphaned To-Dos

An orphaned to-do is a to-do, which cannot be seen by any front-end application user, possibly as a result of such actions as escalation, delegation, etc: If a to-do was rejected by all assigned persons, or delegated to persons with insufficient security rights.

You can search for such to-dos from the To-Do List view: the view toolbar, click Filter , and check Orphaned to-dos only.

2.5.3 Delegating To-Dos

You can [delegate a to-do](#) from the Management perspective: Click the Delegate () button in the toolbar of the To-Do Detail view and define the delegates. Delegates and the current delegation level are then displayed in the Assignees area of the respective to-do detail view.

To undo delegation, click the Undo Delegate  button in the toolbar of the To-Do Detail view.

2.5.4 Escalating To-Dos

Escalation sends an escalation Signal to the server while keeping the to-do locked.

Important: The underlying Model is expected to catch and process it as appropriate with a Signal Catch Intermediate Event or Signal Start Event: The LSPS Server does not handle this Escalation signal by default.

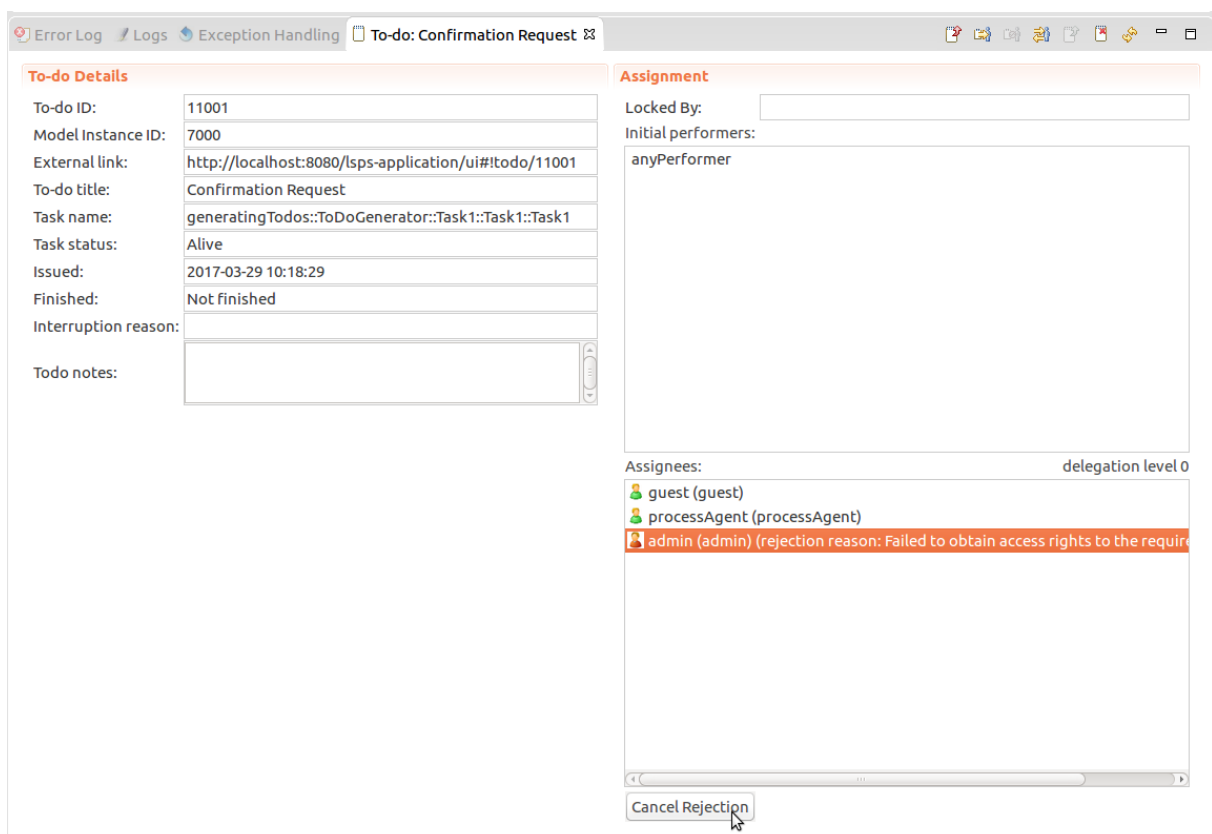
To send such an escalation signal from a to-do from the Management perspective, click the Escalate () button in the toolbar of the To-Do Detail view or select the to-dos in the To-Do List and click Escalate.

Note that this escalation mechanism is **not related** to the [Escalation of the GO-BPMN Modeling Language](#).

2.5.5 Cancelling Rejection

Rejection allows a front-end user to reject a to-do: when they do so, they are excluded from its assignees. You can cancel the rejection, so if the to-do is not locked by other users, it will reappear in their to-do list.

In the Assignees box of a to-do detail view, Assignees, who have previously rejected the to-do, are indicated by a red assignee icon. The rejection reason provided by the assignee is shown next to their icon and name. You can cancel the rejection and reallocate the to-do to the assignee from the To-Do Detail view.



To-do Details

To-do ID:	11001
Model Instance ID:	7000
External link:	http://localhost:8080/lsp-application/ui#todo/11001
To-do title:	Confirmation Request
Task name:	generatingTodos::ToDoGenerator::Task1::Task1::Task1
Task status:	Alive
Issued:	2017-03-29 10:18:29
Finished:	Not finished
Interruption reason:	
Todo notes:	

Assignment

Locked By:

Initial performers: anyPerformer

Assignees: delegation level 0

- guest (guest)
- processAgent (processAgent)
- admin (admin) (rejection reason: Failed to obtain access rights to the require...)

Cancel Rejection

2.5.6 Resetting a Locked To-Do

Click the to-do reset () button at the top of the to-do detail to [reset a to-do](#). Reset erases the data in a saved to-do but the to-do remains locked. This feature is useful if the data used by the saved to-do have changed.

2.5.7 Reassign a To-Do

Click the [to-do reassign](#)  button at the top of the to-do detail and define the roles and persons who should become new initial performers.

2.5.8 Exporting and Importing To-Dos

If you want to modify runtime data of a to-do that is not persisted in the database, you can do so directly in the raw XML of the model instance. Note that if you modify data that is persisted in the database, the data is changed only in the XML and remains unchanged in the database.

Note that the To-Do import and export features are subject to the security right `ToDo:Write_All`

To export or import the model instance XML with the to-do data, do the following:

1. Open the *To-Do List* view.
2. Right-click the to-do and select *Export To-Do State* or *Import To-Do State* respectively.

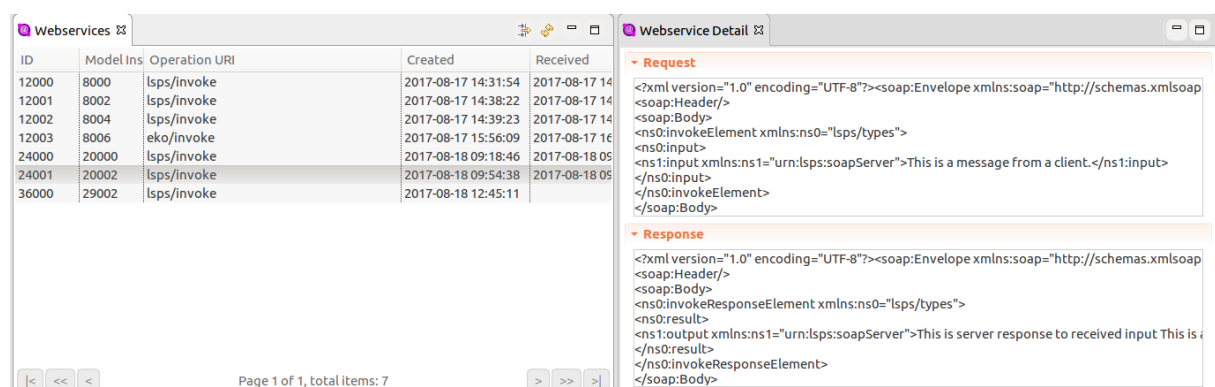
Important: Importing a corrupted XML can cause the system to fail to work with the to-do.

2.6 Web Service Management

You can check the status of calls to your **web-service server Processes** wait points in the *Web Services view*. The view contains a list of active and processed wait points waiting for a Web service request from a client.

If no model instance handled a Web service request, the wait point is highlighted in red.

Double-clicking a wait point entry, opens the respective **Webservice Detail** with the request and response on the wait point. Note that the request and response are available only if the underlying Task has the `logXMLMessage` parameter set to true since excessive request and response data could cause a significant database growth.



The screenshot displays the 'Webservices' management interface. On the left, a table lists wait points with columns for ID, Model Ins, Operation URI, Created, and Received. The table contains 7 entries, with the last one (ID 36000) highlighted in red, indicating it is not handled by a model instance.

ID	Model Ins	Operation URI	Created	Received
12000	8000	lsp/invoker	2017-08-17 14:31:54	2017-08-17 14:31:54
12001	8002	lsp/invoker	2017-08-17 14:38:22	2017-08-17 14:38:22
12002	8004	lsp/invoker	2017-08-17 14:39:23	2017-08-17 14:39:23
12003	8006	eko/invoker	2017-08-17 15:56:09	2017-08-17 15:56:09
24000	20000	lsp/invoker	2017-08-18 09:18:46	2017-08-18 09:18:46
24001	20002	lsp/invoker	2017-08-18 09:54:38	2017-08-18 09:54:38
36000	29002	lsp/invoker	2017-08-18 12:45:11	

On the right, the 'Webservice Detail' view shows the raw XML for the selected wait point (ID 36000). It displays the 'Request' and 'Response' messages. The request is a SOAP message with an input element, and the response is a SOAP message with a result element.

```
<?xml version="1.0" encoding="UTF-8"?><soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header/>
  <soap:Body>
    <ns0:invokeElement xmlns:ns0="lsp/types">
      <ns0:input>
        <ns1:input xmlns:ns1="urn:lsp:soapServer">This is a message from a client.</ns1:input>
      </ns0:input>
    </ns0:invokeElement>
  </soap:Body>
</soap:Envelope>
```

```
<?xml version="1.0" encoding="UTF-8"?><soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header/>
  <soap:Body>
    <ns0:invokeResponseElement xmlns:ns0="lsp/types">
      <ns0:result>
        <ns1:output xmlns:ns1="urn:lsp:soapServer">This is server response to received input This is a message from a client.</ns1:output>
      </ns0:result>
    </ns0:invokeResponseElement>
  </soap:Body>
</soap:Envelope>
```

2.7 Exceptions and Logs Management

2.7.1 Logs

You can display application log entries in the *Logs* view with the following details:

- Model instance ID: identifier of the model instance, which executed the Log task that generated the log message
- Timestamp: date and time when the log message was created
- Level: severity of the log message (info, debug, or error)
- Description: log message content

Note: Availability of these logs does not depend on the `CREATE_PROCESS_LOG` database setting: this setting is related to the availability of module runtime data, which is displayed, for example, in the Model Instance views.

2.7.2 Exceptions

The LSPS Server returns an exception when a component of a model instance cannot perform the required action, for example, if data is missing, or entry data does not correspond with the expected data (a model throws an exception with the `error()` call, refer to [Exception Handling](#)).

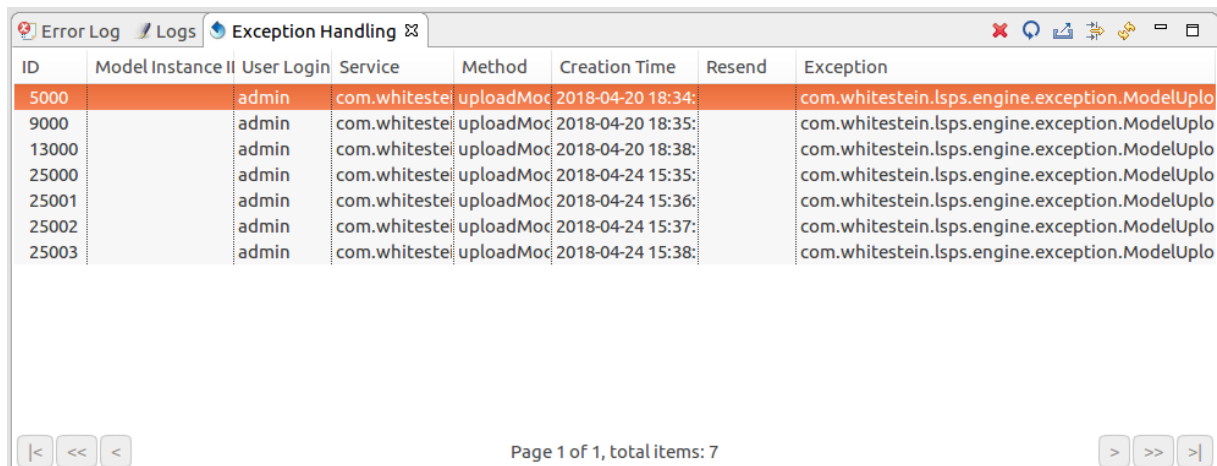
The list of such exceptions is displayed in the **Exception Handling** view with their details:

- ID: unique exception identifier
- Component: component which generated the exception
- Creation Time: time when the exception occurred
- Resend: time when the resend request was sent
- Exception: exception message

In the view toolbar in the upper-right corner, the buttons for the following actions are available:

- Remove (): removes the selected exception entry.
 - Resend (): resends the request.
 - Export to File (): saves one or multiple exceptions in a text file.
 - Filter (): filters the content of the displayed list.
 - Refresh (): refreshes the view content.
-

You can copy multiple rows to the clipboard with Ctrl + c.

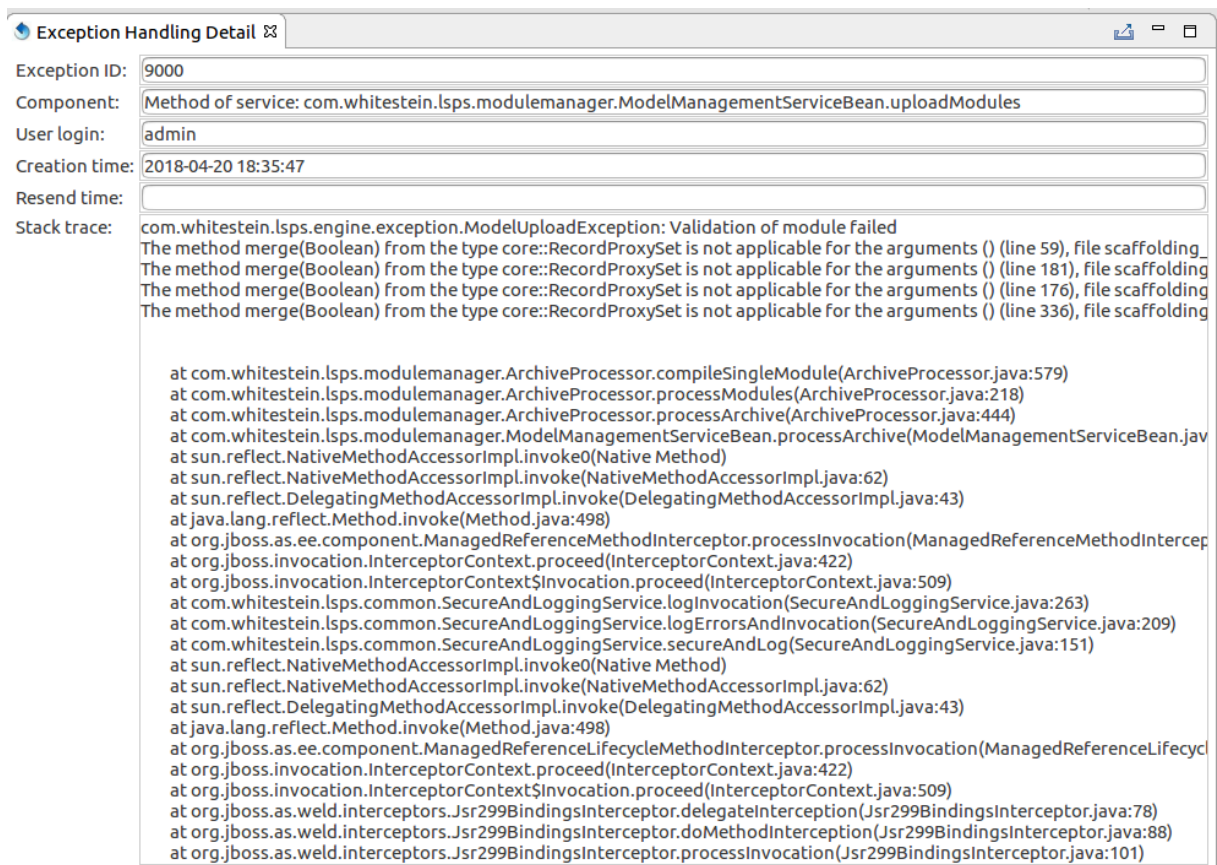


ID	Model Instance II	User Login	Service	Method	Creation Time	Resend	Exception
5000		admin	com.whitestel	uploadMod	2018-04-20 18:34:		com.whitestel.lsp.engine.exception.ModelUplo
9000		admin	com.whitestel	uploadMod	2018-04-20 18:35:		com.whitestel.lsp.engine.exception.ModelUplo
13000		admin	com.whitestel	uploadMod	2018-04-20 18:38:		com.whitestel.lsp.engine.exception.ModelUplo
25000		admin	com.whitestel	uploadMod	2018-04-24 15:35:		com.whitestel.lsp.engine.exception.ModelUplo
25001		admin	com.whitestel	uploadMod	2018-04-24 15:36:		com.whitestel.lsp.engine.exception.ModelUplo
25002		admin	com.whitestel	uploadMod	2018-04-24 15:37:		com.whitestel.lsp.engine.exception.ModelUplo
25003		admin	com.whitestel	uploadMod	2018-04-24 15:38:		com.whitestel.lsp.engine.exception.ModelUplo

Page 1 of 1, total items: 7

Figure 2.19 Exception Handling view

Double-clicking a table entry displays the respective exception detail view with detailed information about the selected exception and its complete stacktrace. From the detail view you can export the exception to a file: click the Export to File () button in the view toolbar, to do so.



Exception Handling Detail	
Exception ID:	9000
Component:	Method of service: com.whitestel.lsp.modulemanager.ModelManagementServiceBean.uploadModules
User login:	admin
Creation time:	2018-04-20 18:35:47
Resend time:	
Stack trace:	<pre>com.whitestel.lsp.engine.exception.ModelUploadException: Validation of module failed The method merge(Boolean) from the type core::RecordProxySet is not applicable for the arguments () (line 59), file scaffolding_ The method merge(Boolean) from the type core::RecordProxySet is not applicable for the arguments () (line 181), file scaffolding_ The method merge(Boolean) from the type core::RecordProxySet is not applicable for the arguments () (line 176), file scaffolding_ The method merge(Boolean) from the type core::RecordProxySet is not applicable for the arguments () (line 336), file scaffolding_ at com.whitestel.lsp.modulemanager.ArchiveProcessor.compileSingleModule(ArchiveProcessor.java:579) at com.whitestel.lsp.modulemanager.ArchiveProcessor.processModules(ArchiveProcessor.java:218) at com.whitestel.lsp.modulemanager.ArchiveProcessor.processArchive(ArchiveProcessor.java:444) at com.whitestel.lsp.modulemanager.ModelManagementServiceBean.processArchive(ModelManagementServiceBean.java: at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62) at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) at java.lang.reflect.Method.invoke(Method.java:498) at org.jboss.as.ee.component.ManagedReferenceMethodInterceptor.processInvocation(ManagedReferenceMethodIntercep at org.jboss.invocation.InterceptorContext.proceed(InterceptorContext.java:422) at org.jboss.invocation.InterceptorContext\$Invocation.proceed(InterceptorContext.java:509) at com.whitestel.lsp.common.SecureAndLoggingService.logInvocation(SecureAndLoggingService.java:263) at com.whitestel.lsp.common.SecureAndLoggingService.logErrorsAndInvocation(SecureAndLoggingService.java:209) at com.whitestel.lsp.common.SecureAndLoggingService.secureAndLog(SecureAndLoggingService.java:151) at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method) at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62) at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43) at java.lang.reflect.Method.invoke(Method.java:498) at org.jboss.as.ee.component.ManagedReferenceLifecycleMethodInterceptor.processInvocation(ManagedReferenceLifecyc at org.jboss.invocation.InterceptorContext.proceed(InterceptorContext.java:422) at org.jboss.invocation.InterceptorContext\$Invocation.proceed(InterceptorContext.java:509) at org.jboss.as.weld.interceptors.Jsr299BindingsInterceptor.delegateInterception(Jsr299BindingsInterceptor.java:78) at org.jboss.as.weld.interceptors.Jsr299BindingsInterceptor.doMethodInterception(Jsr299BindingsInterceptor.java:88) at org.jboss.as.weld.interceptors.Jsr299BindingsInterceptor.processInvocation(Jsr299BindingsInterceptor.java:101)</pre>

Figure 2.20 Exception detail

Chapter 3

Management Console

Living Systems Process Management Console is a web-based front-end application intended for management of model instances, LSPS users, and the LSPS front-end applications, that is, the Process Management Console and the Application User Interface.

The tool is primarily intended for the application and LSPS administrators. However, also process designers and application developers might benefit from its features.

3.1 Accessing Management Console

Note that as an LSPS user you need to have the access rights assigned explicitly to be able to access the Management Console.

You can access the Management Console from a supported Web browser:

1. Launch your web browser.

2. Go to the URL of the application.

By default, the URL is [http://\\$DOMAIN/lsp-management](http://$DOMAIN/lsp-management).

3. On the login page, enter your credentials.

To log out, click the drop-down icon next to your user name and in the context menu, click **Log out**.

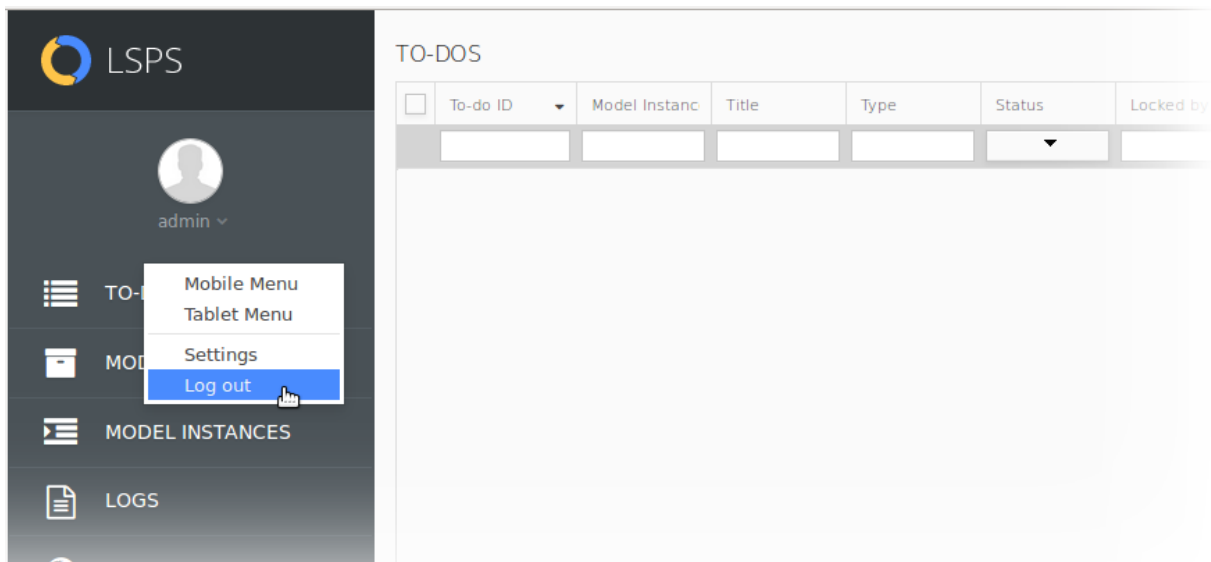


Figure 3.1 Logging out

Related topics:

- [User Interface](#)
- [To-Do Management](#)
- [Module Management](#)
- [Model Instance Management](#)
- [Saved Documents](#)
- [Person Management](#)
- [Role Management](#)
- [Security Role Management](#)
- [Log Management](#)
- [Exception Management](#)
- [Application Settings](#)

3.2 User Interface

To navigate through views of the Process Management Console environment, use the navigation toolbar located on the left: The relevant content is displayed in the area on the right.

The following views are available:

- [To-dos](#): List of all To-do in the Navigation Engine
- [Modules](#): List of Modules uploaded to the Module Repository

- [Model Instances](#): List of Model instances in the Navigation Engine
- [Logs](#): Logs created by Log tasks of Processes
- [Webservices](#): Webservices requests and responses involved in Webservices Tasks of Processes
- [Persons](#): Users with their details and rights
- [Roles](#): Roles created based on elements of Organization Models
- [Security Roles](#): Security Roles
- [Exceptions](#): Exceptions logged by the Navigation Engine
- [Dynamic Themes](#): List of available CSS themes
- [Model Update Logs](#): Logs of model update processes

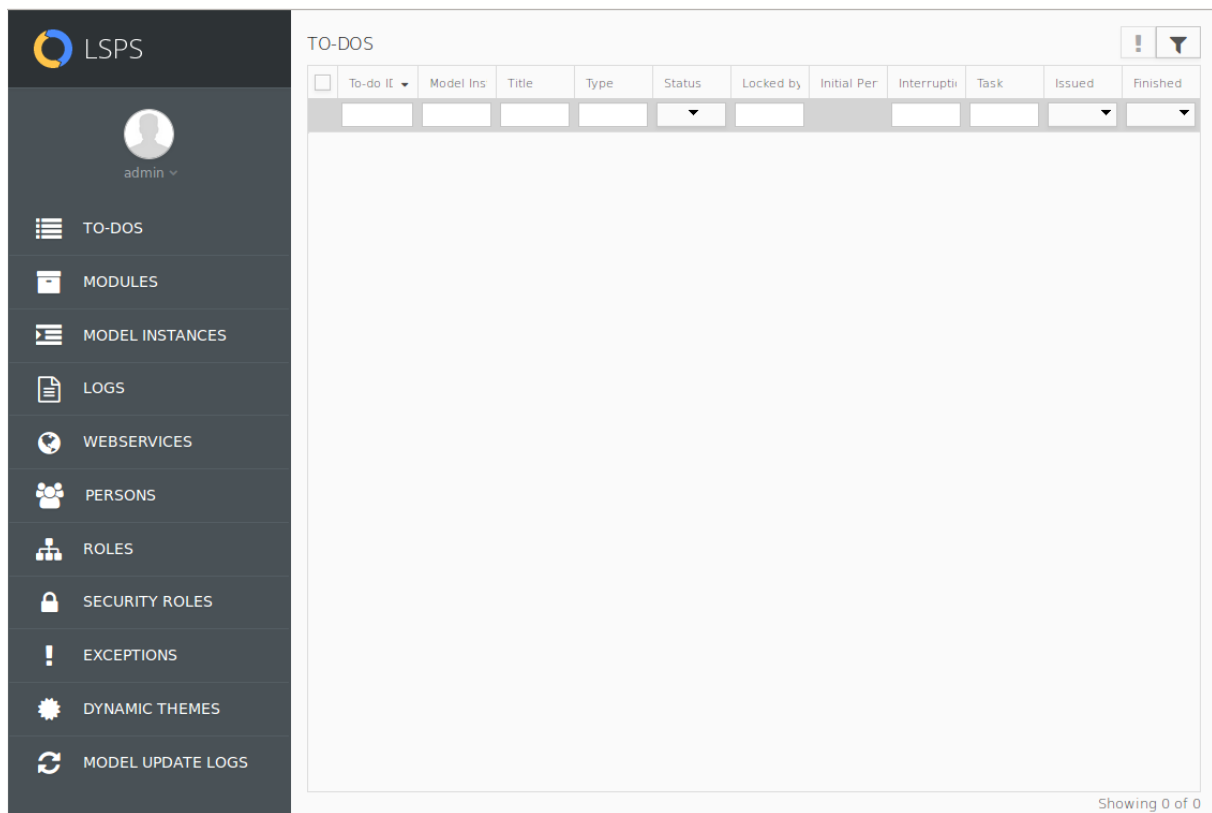


Figure 3.2 Management Console environment

3.2.1 To-dos View

The To-dos view contains a list of all to-dos dispatched by the Navigation Engine and their details.

The [to-dos](#) are listed in a table with the following data:

- To-do ID: unique to-do ID
- Model Instance ID: ID of the parent model instance (the to-do was generated by a User Task of one of the module instances of this model instance)

- Title: todo title defined by the title parameter of its User Task
- Type: name of the User Task that generated the to-do
- Status: status of the User Task (for information on the task life cycle, refer to GO-BPMN Modeling Language Guide)
- Locked by: name of the person, who has locked the to-do
- Performers: the expression used to obtain the users to whom the to-do was initially allocated
- Interruption Reason: reason why the to-do was terminated, such as, the parent model instance was terminated
- Task Namespace: full path of the User Task
- Issue Date: date when the to-do was issued
- Submitted: date when the to-do was submitted (became *finished*)

From the view, you can [search for orphaned to-dos](#), [escalate](#) and [reset](#) alive to-dos, and display [to-do details](#).

To-do ID	Model Insta	Title	Type	Status	Locked by	Initial Perf	Interruption	Task	Issued	Finished
13000	9000	Regist...	Regist...	Alive		[anyPe...		mod:p...	Sep 21...	
13001	9003	Regist...	Regist...	Alive		[anyPe...		mod:p...	Sep 22...	
13002	9005	Regist...	Regist...	Alive		[anyPe...		mod:p...	Sep 22...	

Figure 3.3 To-dos page

3.2.2 To-Do Detail

A To-do Detail appears after clicking a to-do ID or name in the To-do List or on the To-do page.

The To-do Detail page contains the following to-do information:

- To-do ID: unique to-do ID
- Model instance ID: ID of the model instance that produced the to-do
- External link: link to the to-do content
- Title: task title
- Task: namespace path and name of the to-do's user task
- Status: task execution status
- Issued: date and time, when the to-do was generated
- Initial Performs: expression used to acquire the initial potential performers for the to-do

- Assignees: users who can see the to-do
- Finished: date and time, when the to-do was finished (submitted or interrupted)
- Locked by: person who has locked the to-do
- To-do notes: notes provided by users of the Application User Interface
- Delegation level: current level of delegation

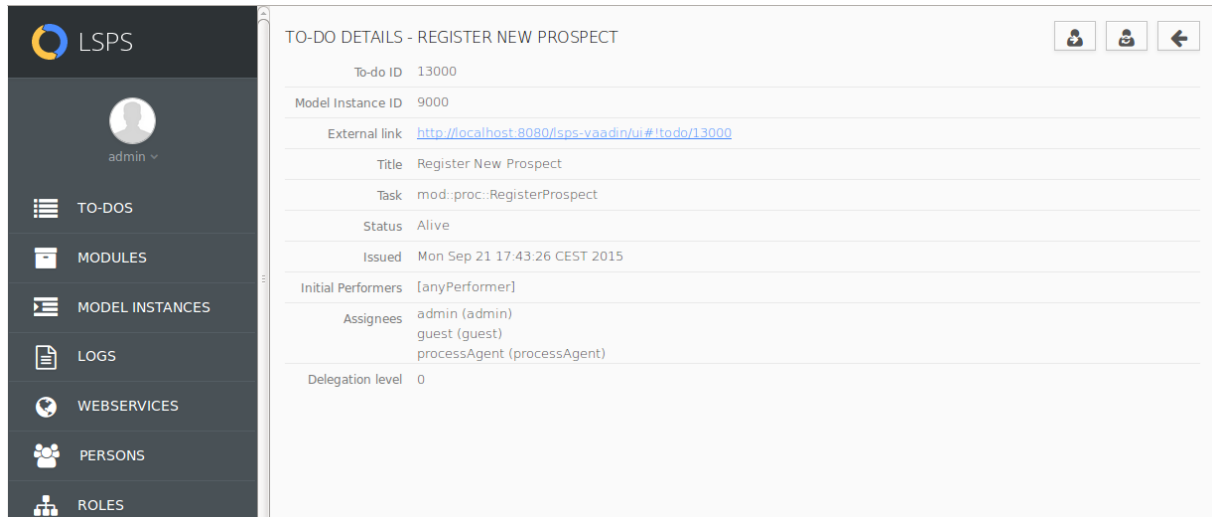


Figure 3.4 To-Do detail

From the to-do detail, you can delegate the to-do or undo a delegation and reassign to-dos.

3.2.3 Modules View

The Modules view contains a list of modules uploaded on the LSPS Server. From the view, you can perform module-related actions, including upload, unload, instantiation, and update.

You can also filter the module list based on their details.

3.2.4 Model Instances View

The Model Instances view contains a list of all model instances in the LSPS Server (Execution Engine) with their details:

- ID: unique ID of the model instance
- Model ID: ID of the underlying model
- Model name: name of the underlying model
- Model version: version of the underlying model
- Started: time when the model instance was created
- Finished: time when the model instance finished

- State: current execution *status of the model instance*

You can filter the table content by entering the filter values in the respective header cell and press ENTER.

☐							
	Id	Model ID	Model name	Model version	Started	Finished	State
☐	10000	3005	diagrams	1.0	2016-10-13 1...	Oct 13, 2016 ...	Finished
☐	10001	3006	machineModule	1.0	2016-10-13 1...	Oct 13, 2016 ...	Finished
☐	10002	3008	todoGenerator	1.0	2016-10-13 1...		Running

Showing 3 of 3

Figure 3.5 Model Instances

Selected model instances can be managed using the buttons in the view toolbar.

3.2.5 Model Instance Detail

Model instance detail view contains multiple tabs with detailed information about a model instance.

To open a detail of a model instance, click the model instance ID in the Model Instances view.

On the view toolbar, the actions related to the displayed tab are available.

The screenshot displays the LSPS user interface. On the left is a dark sidebar with the LSPS logo and a list of navigation items: TO-DOS, MODULES, MODEL INSTANCES, LOGS, WEBSERVICES, PERSONS, ROLES, SECURITY ROLES, EXCEPTIONS, DYNAMIC THEMES, and MODEL UPDATE LOGS. The main area is titled 'MODEL INSTANCE 9001 - DELIVERY' and features a 'General' tab. Below the tab, the following information is presented:

Model Instance ID	9001
Initiator	admin (admin)
Description	
Started	Tue Sep 22 08:46:48 CEST 2015
Finished	
Status	RUNNING
Module	delivery - 1.0

Figure 3.6 Model Instance Detail

General Tab

The General tab contains general information about a model instance.

- Model Instance ID: unique model instance ID
- Initiator: person, who created the model instance
- Description: free-text description
- Started: date and time of model instance creation
- Finished: date and time of model instance termination
- Status: current execution status of the model instance
- Module: name and version of the executable module that was used to create the model instance

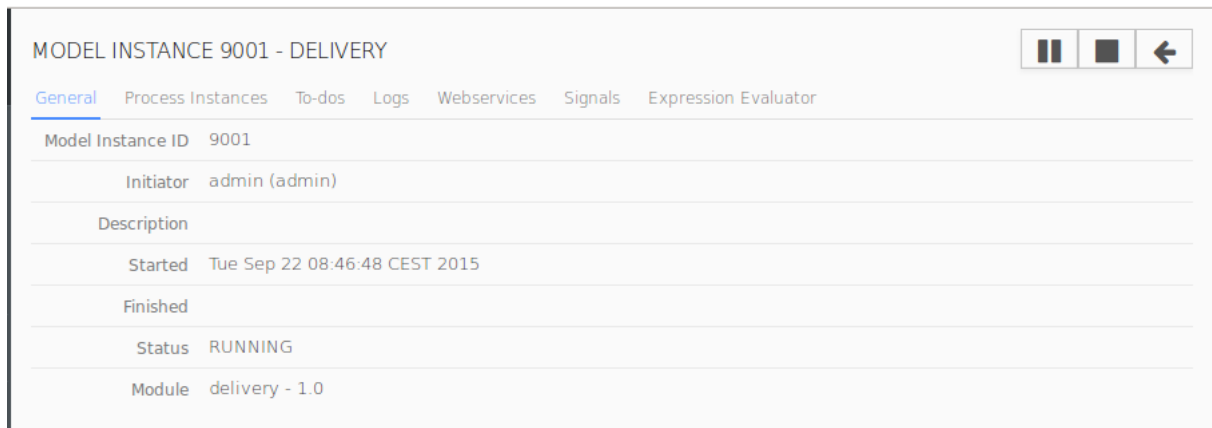


Figure 3.7 The General tab in a Model Instance detail view

Context Tab

The *Context* tab contains the runtime data of the model instance including their parameter values, and a tree structure of the model instance with details on the displayed elements:

- Parameters of the model: Initiator, Description, and other metadata provided when the instance was created
Value of the parameters is in the Value column.
- Tree with modules, their global variables, processes, and Goals, Plans and Diagrams: the details for individual elements are in the given table row.

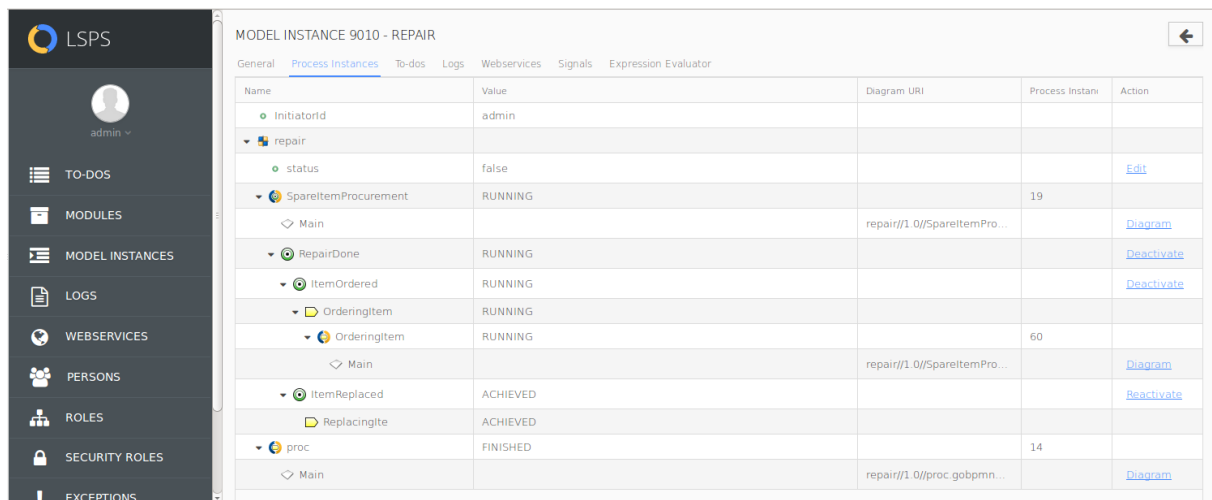


Figure 3.8 Context tab

To-dos Tab

The To-dos tab contains all to-dos generated by the respective model instance and their details.

- To-do ID: unique to-do ID number
- Title: title of the underlying user task
- Type: name of the user task
- Status: to-do status (Alive, Accomplished, or Interrupted)
- Locked By: name of the person, who has locked the to-do
- Initial Performers: names of initial performers
- Interruption reason: reason why the to-do was finished without completion if applicable (for example, due to process deactivation or model update)
- Task: task path and task name (mostly module name::process name::plan name::task name)
- Issued: date when the to-do was issued
- Finished: date when the to-do was finished

Clicking a to-do ID opens the to-do detail. Note that you can filter for any orphaned to-dos from this tab: click the Filter button in the upper right corner and define the filter properties.

To-do ID	Model Instance I	Title	Type	Status	Locked by	Initial Performer	Interruption reason	Task	Issued	Finished
13004	9012	Register Ne...	RegisterPro...	Accomplish...	admin (ad...	[anyPerfor...		mod:proc::...	Sep 22, 201...	Sep 22, 201...
13005	9012	Register Ne...	ValidatePro...	Alive		[anyPerfor...		mod:proc::...	Sep 22, 201...	
13006	9012	Register Ne...	ReviewPros...	Alive		[anyPerfor...		mod:proc::...	Sep 22, 201...	

Figure 3.9 To-dos Tab

Logs Tab

The Logs tab contains a table with messages logged by Log tasks, tasks of the Log task type, executed as part of the model instance.

For each log entry, the table shows:

- Log ID: unique log ID
- Timestamp: date when the message was logged
- Level: severity of the log message defined as an integer value
If the user defines a filter values, all logs with a value greater that the defined value are displayed.
- Description: log message content

MODEL INSTANCE 9010 - REPAIR

General

Process Instances

To-dos

Logs

Webservices

Signals

Expression Evaluator

Log ID	Model Instance ID	Timestamp	Level from	Description
	9010			
18008	9010	2015-09-22 1...	200	The item has been replaced.
18009	9010	2015-09-22 1...	200	The item has been ordered.
18010	9010	2015-09-22 1...	200	The item has been received.
18011	9010	2015-09-22 1...	200	The item has been replaced.

Figure 3.10 Logs tab

Web Services Tab

The Web Services tab contains a list of active and processed wait points (waiting for a web-service request from a client) of a model instance.

The Web Services tab contains a table with all wait points of web service clients in the model instance:

- ID: unique wait point ID
- Operation URI (Unified Resource Identifier): identifier of the resource (if several wait-point requests are processed by the same task type, value of their operation URI is identical. However, their wait-point IDs differ)
- Created: time when the wait point was created
- Request: link to the XML received from the client
- Request Received: time when the client request was received
- Response: link to the XML sent to the client and information on the response type (time-out, error, finished, terminated)
- Finished: time when the Web service response was sent
- Response timed out: if the response time was overstepped
- Terminated: if the wait point was erased due to parent element termination
- Error response: if the received response was an error response

MODEL INSTANCE 9023 - ORDERMANAGEMENT

General

Process Instances

To-dos

Logs

Webservices

Signals

Expression Evaluator

Id	Model Instance	Operation URI	Created	Request	Request received	Response	Finished	Response time	Terminated	Error response
	9023									
19001	9023	ordering/invoke	2015-09-...					false	false	false
19002	9023	ordering/invoke	2015-09-...					false	false	false
19003	9023	ordering/invoke	2015-09-...					false	false	false

Figure 3.11 Webservices Tab

Signals Tab

The Signals tab contains a list of signals waiting to be consumed generated by a model instance.

Waiting signals may be removed.

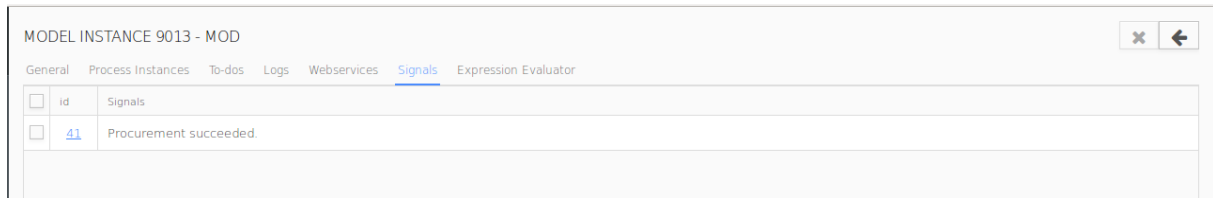


Figure 3.12 Signals tab

Expression Evaluator Tab

The Expression Evaluator tab enables you to evaluate expressions defined in the Expression Language in the context of the model instance.

By default, the evaluated expressions have no impact on the persisted data, that is, on Shared Record instances. If you want to apply the expression on Shared Records, select the Persistent option.

Note: You cannot use data from a local contexts of the model instance.

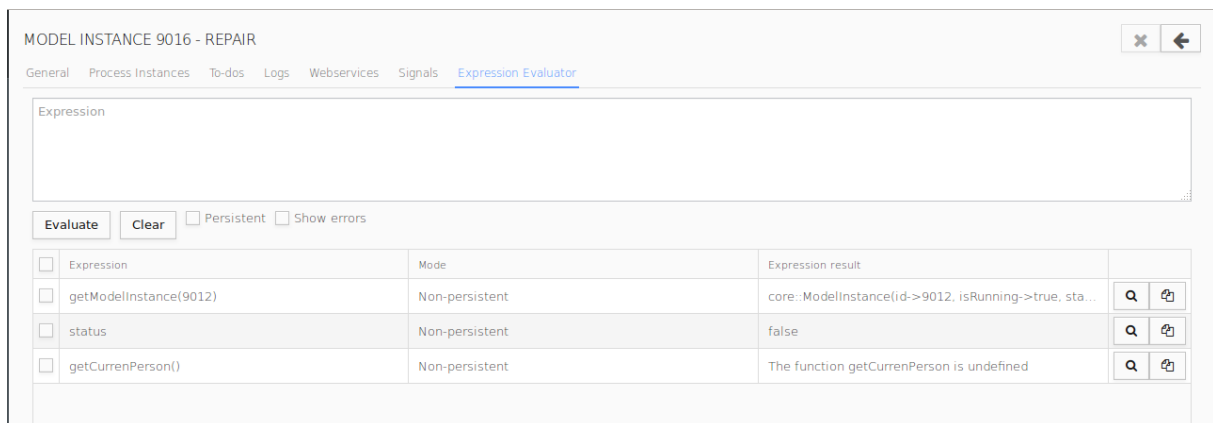


Figure 3.13 Expression Evaluator

Show Errors enables you to display detailed error messages in case some of the expression could not be evaluated.

3.2.6 Saved Documents View

The *Saved Documents* view lists the documents users have saved and are hence probably working on. You can export the documents as XMLs and import the modified documents back: this enables you the modify the runtime data of saved documents. Note that the export and import features are subject to the security right `Document<->:Write`.

3.2.7 Logs View

The Logs view contains a table with messages logged by Log Tasks, tasks of the Log task type, executed as part of the model instance.

For each log entry, the following details are displayed in the log table:

- Log ID: unique log ID
- Model Instance ID: ID of the parent model instance of the Log task
- Timestamp: date when the message was logged
- Level: severity of the log message

When filtering according to the log level, any logs with severity higher than the defined value are displayed.

- Description: log message content

Log ID	Model Instance ID	Timestamp	Level from	Description
18002	9008	2015-09-22 1...	200	The item has been replaced.
18003	9008	2015-09-22 1...	200	The item has been ordered.
18004	9008	2015-09-22 1...	200	The item has been received.
18005	9009	2015-09-22 1...	200	The item has been replaced.
18006	9009	2015-09-22 1...	200	The item has been ordered.
18007	9009	2015-09-22 1...	200	The item has been received.
18008	9010	2015-09-22 1...	200	The item has been replaced.
18009	9010	2015-09-22 1...	200	The item has been ordered.
18010	9010	2015-09-22 1...	200	The item has been received.
18011	9010	2015-09-22 1...	200	The item has been replaced.
18012	9010	2015-09-22 1...	200	The item has been replaced.

Figure 3.14 Logs page

3.2.8 Web Services View

The Web Services view contains a list of active and processed wait points (waiting for a Web service request from a client).

The Web Services view contains a table with wait points details:

- ID: unique wait point ID
- Model Instance ID: ID of the model instance processing the Web service request
- Operation URI (Unified Resource Identifier): identifier of the resource

Note: If several wait-point requests are processed by the same task type, value of their operation URI is identical. However, their wait-point IDs differ.

- Created: time when the wait point was created
- Request: link to the request XML received from the client
- Request Received: time when the client request was received
- Response: link to the response XML sent to the client and information on the response type (time-out, error, finished, terminated)
- Finished: time when the Web service response was sent
- Response timed out: if the response time was overstepped
- Terminated: if the wait point was erased due to parent element termination
- Error response: if the received response was an error response

Id	Model Instance	Operation URI	Created	Request	Request rece	Response	Finished	Response tim	Terminated	Error re
19000	9022	ordering/invoke	2015-09-...					false	false	false
19001	9023	ordering/invoke	2015-09-...					false	false	false
19002	9023	ordering/invoke	2015-09-...					false	false	false
19003	9023	ordering/invoke	2015-09-...					false	false	false

Figure 3.15 Web Services view

3.2.9 Persons View

The Persons view shows a list of persons, that is, LSPS users, as defined on the LSPS server.

☐	Id	Login	First Name	Last Name	E-mail	Phone	Enabled
☐	admin	admin					true
☐	guest	guest					false
☐	jane@	jane	Jane	Doe	jane@doe.com	908987897	true
☐	john@	john	John	Doe	john@doe.com	879087	true
☐	processAgent	processAgent					true

Figure 3.16 Persons View

By clicking a person id, you can display the person detail.

Using the buttons in the view toolbar, you can add a new person, disable selected persons, or enable selected persons.

3.2.10 Person Detail View

The Person Detail view shows details of a person including substitution setting, their security and modeled roles.

The detail appears after clicking a person id in the Persons or Roles view.

JANE (JANE DOE)

Login: jane

☒ Enabled

First Name: Jane

Last Name: Doe

E-mail: jane@doe.com

Phone: 908987897

Modeled roles

Security Roles

- ☐ Admin
- ☒ ApplicationRoleManager
- ☒ ProcessExecutor
- ☐ ProcessManager

Properties: No Properties to display.

☐ Substitution active

Figure 3.17 Person Detail view

3.2.11 Roles View

The Roles view contains lists of runtime and modeled roles defined in organization definitions of modules on the server.

If a runtime role is selected, persons with the runtime role are shown in a table displayed on the right.

ROLES		
Role name	Module	Parameters
CEO	mod	
CEO	mod	[]
Programmer	mod	department, languages
Programmer	mod	["department"->"Compliance", "languages"->...
ProjectManager	mod	projectName
ProjectManager	mod	["projectName"->"LSPS"]
Secretary	mod	department
TeamLead	mod	
Tester	mod	

Figure 3.18 Roles view with runtime and modeled roles

3.2.12 Security Roles View

The Security view contains the list of available [security roles](#).

Every active user has at least one security role, which restricts the actions they can perform in runtime; security roles define a particular set of security rights:

- Security right is a right to perform an action.
- Security role is a group of security rights.

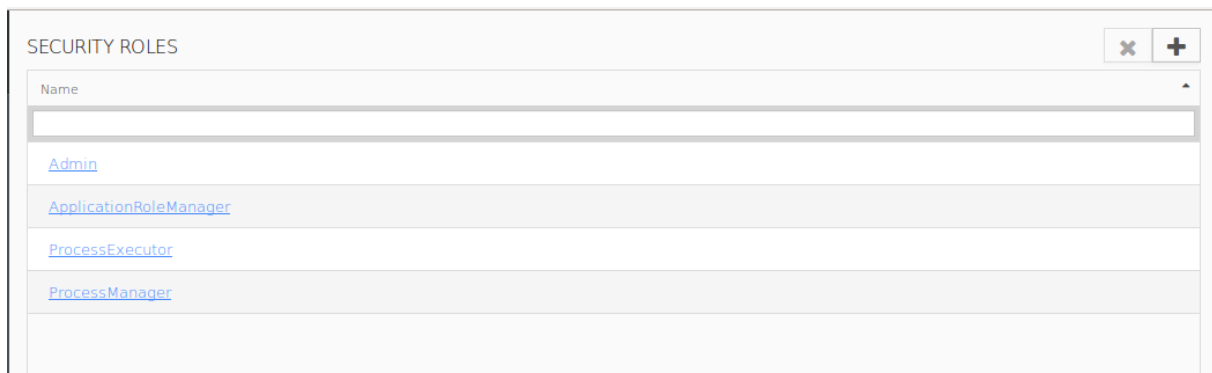


Figure 3.19 Security view

3.2.13 Exceptions View

The Exceptions view contains a list of all server exceptions and their details.

The Exceptions view contains:

- filter area to filter the exceptions listed below
- table with exception and their details:
 - ID: ID of exception
 - Exception: the first line of the exception
 - Creation Time: time when the exception occurred
 - Resend: date when the resend has successfully taken place

To resend an exception, select it in the list and click the Resend  button.

 - Invoked Component: component which generated the exception

To view the stack trace of an exception, click the exception id in the table.

EXCEPTIONS ✕ ↺					
☐	Id	Exception	Creation Time	Resend	Invoked Component
☐	17000	com.whitestein.lsp.com...	Sep 22, 2015 8:46:59 AM		com.whitestein.lsp.hum...
☐	17001	com.whitestein.lsp.com...	Sep 22, 2015 8:47:15 AM		com.whitestein.lsp.hum...
☐	17002	com.whitestein.lsp.com...	Sep 22, 2015 1:54:39 PM		com.whitestein.lsp.hum...
☐	17003	com.whitestein.lsp.com...	Sep 22, 2015 3:56:02 PM		com.whitestein.lsp.hum...
☐	17004	com.whitestein.lsp.lang...	Sep 22, 2015 4:00:46 PM		com.whitestein.lsp.engi...
☐	17005	com.whitestein.lsp.lang...	Sep 22, 2015 4:01:46 PM		com.whitestein.lsp.engi...
☐	17006	com.whitestein.lsp.lang...	Sep 22, 2015 4:05:48 PM		com.whitestein.lsp.engi...
☐	17007	com.whitestein.lsp.lang...	Sep 22, 2015 4:06:47 PM		com.whitestein.lsp.engi...
☐	17008	com.whitestein.lsp.lang...	Sep 22, 2015 4:20:48 PM		com.whitestein.lsp.engi...
☐	17009	com.whitestein.lsp.lang...	Sep 22, 2015 4:23:57 PM		com.whitestein.lsp.engi...
☐	17010	com.whitestein.lsp.lang...	Sep 22, 2015 4:24:09 PM		com.whitestein.lsp.engi...
☐	17011	com.whitestein.lsp.lang...	Sep 22, 2015 4:25:12 PM		com.whitestein.lsp.engi...
☐	17012	com.whitestein.lsp.lang...	Sep 22, 2015 4:25:22 PM		com.whitestein.lsp.engi...

Figure 3.20 Exceptions view

3.2.14 Model Update Logs View

The Model Update Logs view contains a log entry for every updated model instance. You can display the log content by clicking the log ID.

MODEL UPDATE LOGS - 33000 ←	
Id	34000
Started	Wed Sep 23 14:21:02 CEST 2015
Finished	Wed Sep 23 14:21:02 CEST 2015
State	MODEL_UPDATE_UPDATED
Source Model	oldModel - 1.0
Target Model	newModel - 1.0
Update Log	<pre> INFO Wed Sep 23 14:21:02 CEST 2015 Pre-processing started. INFO Wed Sep 23 14:21:02 CEST 2015 Pre-processing started. INFO Wed Sep 23 14:21:02 CEST 2015 Pre-processing finished. INFO Wed Sep 23 14:21:02 CEST 2015 Transforming model of model instance 28000 from model oldModel 1.0 to model newModel 1.0. INFO Wed Sep 23 14:21:02 CEST 2015 Token oldModel::oldProcess::_ (16) discarded. INFO Wed Sep 23 14:21:02 CEST 2015 All tokens discarded in process instance newModel::oldProcess (76), new token created. INFO Wed Sep 23 14:21:02 CEST 2015 Model update transformed successfully. INFO Wed Sep 23 14:21:02 CEST 2015 Post-processing started. INFO Wed Sep 23 14:21:02 CEST 2015 Post-processing started. INFO Wed Sep 23 14:21:02 CEST 2015 Post-processing finished. </pre>

Figure 3.21 Log content

3.3 To-Do Management

Features for To-dos management are accessible from the To-Do page.

Note that you can filter the table content by setting the filter value in the header of the respective cell. Filtering is by default case-sensitive and supports wildcards (*, ?); for example search for Task? * returns results containing the word Task followed by any character, a space, and any subsequent characters.

You can perform the following on to-dos:

- [reassign](#) locked to-dos to another person,
- [reset](#) the data in a saved to-do,
- [delegate](#), or [escalate](#).

As a result of such an action, a to-do might end up with no performers. Such to-dos are referred to as *orphaned* and require the attention of an administrator: in the Management Console, the administrator can [search specifically for such to-dos](#).

3.3.1 Searching for Orphaned To-Dos

To display orphaned To-dos in the table of the To-Do view, click the Filter  button in the view toolbar; in the displayed dialog, select the Orphaned to-dos option and confirm.

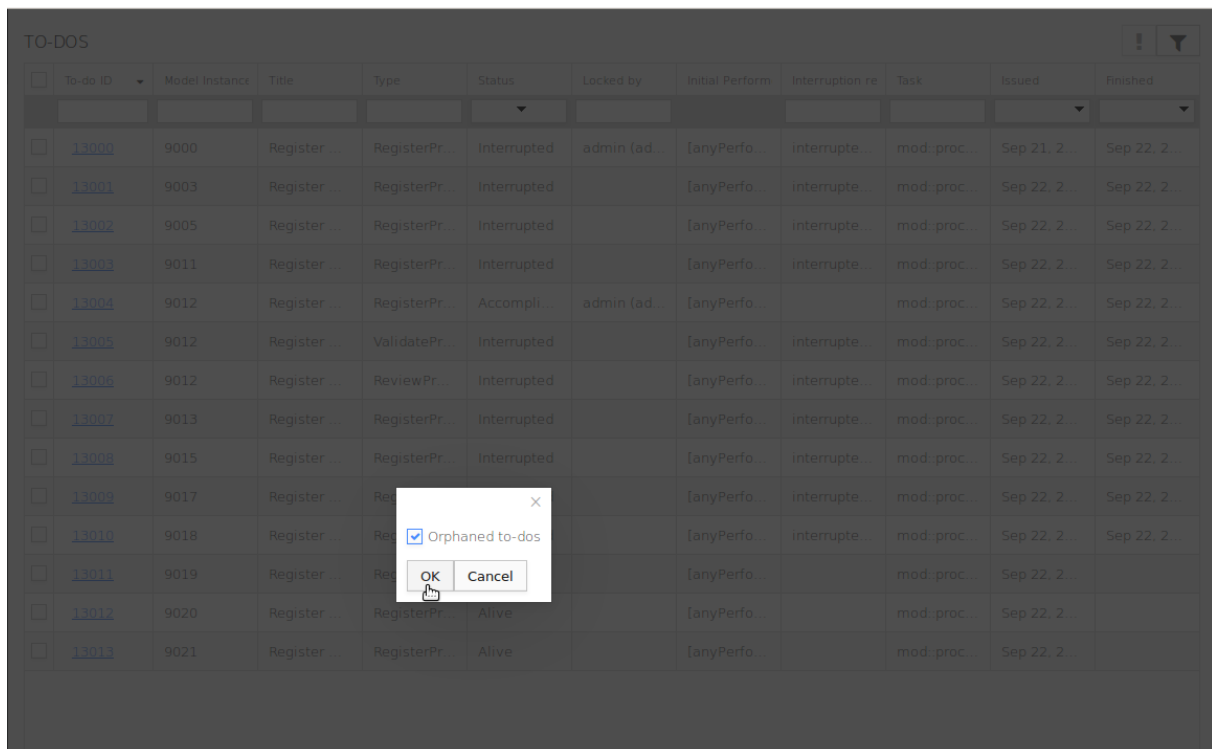


Figure 3.22 Activating orphan to-dos filtering

3.3.2 Reassigning To-Dos

[Reassignment](#) enables an administrator to assign an Alive to-do to other users, who become the initial performers.

Note: Reassignment cannot be undone.

To reassign a to-do, do the following:

1. Display the to-dos detail (go to To-do List view and click a to-do ID).
2. Click the Reassign  button in the view toolbar.
3. Select Persons to reassign the to-do to users or Roles to reassign the to-do to users with a runtime role.
4. In the table below, select the persons or roles.
5. In the view toolbar, click the confirmation button.

TO-DO DETAILS - REGISTER NEW PROSPECT

To-do ID 13013

Model Instance ID 9021

External link <http://localhost:8080/sps-vaadin/ui/#!todo/13013>

Title Register New Prospect

Task mod::proc::RegisterProspect

Status Alive

Issued Tue Sep 22 16:54:37 CEST 2015

Initial Performers [anyPerformer]

Assignees
admin (admin)
jane (Jane Doe)
john (John Doe)
processAgent (processAgent)

Delegation level 0

Choose reassign type ☒ Persons ☐ Roles

☐	Id	Login	First Name	Last Name	E-mail	Phone	Enabled
☐	admin	admin					true
☐	guest	guest					false
☒	jane0	jane	Jane	Doe	jane@doe.com	908987897	true
☒	john0	john	John	Doe	john@doe.com	879087	true
☐	processAgent	processAgent					true

Figure 3.23 Reassigning To-Do

3.3.3 Escalating To-Dos

Escalation enables users to trigger the escalation process. Escalation might be triggered automatically (for example, after a certain period has elapsed), or manually: If a to-do is escalated, a signal with a certain value is sent to the Execution Engine. The signal can be caught and processed by a catch event. If escalation handling and catching of a particular signal is not handled within the model instance, escalation takes **no effect**. Unconsumed Signals remain in the signal queue.

Important: The escalation mechanism described below is deprecated. Use the **escalation mechanism** instead; for example, call the `throwEscalation()` function on a button component and process the escalation with Catch Escalation elements.

TO-DOS											
☐	To-do ID	Model Instance	Title	Type	Status	Locked by	Initial Perform	Interruption re	Task	Issued	Escalated
☐	13000	9000	Register ...	RegisterPr...	Interrupted	admin (ad...	[anyPerfo...	interrupte...	mod.:proc...	Sep 21, 2...	Sep 22, 2...
☐	13001	9003	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13002	9005	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13003	9011	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13004	9012	Register ...	RegisterPr...	Accompli...	admin (ad...	[anyPerfo...		mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13005	9012	Register ...	ValidatePr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13006	9012	Register ...	ReviewPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13007	9013	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13008	9015	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13009	9017	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13010	9018	Register ...	RegisterPr...	Interrupted		[anyPerfo...	interrupte...	mod.:proc...	Sep 22, 2...	Sep 22, 2...
☐	13011	9019	Register ...	RegisterPr...	Alive		[anyPerfo...		mod.:proc...	Sep 22, 2...	
☒		9020	Register ...	RegisterPr...	Alive		[anyPerfo...		mod.:proc...	Sep 22, 2...	
☒		9021	Register ...	RegisterPr...	Alive		[anyPerfo...		mod.:proc...	Sep 22, 2...	

Figure 3.24 Escalating To-Dos

To escalate one or several to-dos, do the following:

1. Open the To-dos view.
2. Select the to-dos you want to escalate.
Only live to-dos can be escalated.
3. Click Escalate .
4. Type escalation reason and click Escalate.

3.3.4 Delegating To-Dos

To **delegate** a live to-do to another user, do the following:

1. Activate the respective to-do detail (To-dos To-do ID).
Only live to-dos can be delegated.
2. Click the Delegate  button.
3. Filter out the relevant users using the filter in the column of the table header.
Wildcard usage is supported; to display all possible substitutes, type *.

4. Select delegates from the table and click the Delegate button.

TO-DO DETAILS - PROCUREMENT REVIEW

✕

✓

Delegate

To-do ID

12000

Model Instance ID

8000

External link

<http://localhost:8080/sps-vaadin/ui/#!todo/12000>

Title

Procurement review

Task

debugTest::goalProc::'Plan 1'::procurementReview

Status

Alive

Issued

Tue Sep 29 09:53:55 CEST 2015

Initial Performers

[anyPerformer]

Assignees

admin (admin)
guest (guest)
processAgent (processAgent)

Delegation level

0

☐	Id	Login	First Name	Last Name	E-mail	Phone	Enabled
☐							▼
☐	admin	admin					true
☒	guest	guest					true
☐	processAgent	processAgent					true

Figure 3.25 Selecting delegates

5. Click OK.

Delegation can be undone so the to-do is assigned to the previous users.

The respective to-do disappears from the to-do lists of all previous assignees and appears in the to-do list of delegates. If locked, to-do is unlocked and delegated as well.

3.3.4.1 Undoing Delegation

Delegation can be undone only for live to-dos.

To undo delegation of a to-do, do the following:

1. Open the to-do detail (To-dos To-do ID).
2. In the view toolbar, click the Undo Delegation  button.

Delegation is undone and the delegation level decreases. Repeat the process to undo delegation on several levels.

3.3.5 Resetting To-Dos

[To-do reset](#) erases the data in a saved to-do. This feature is useful if the data used by the saved to-do have changed. Note that a reset to-do remains locked by the user.

To reset a live to-do of a user, do the following:

1. Open the To-Dos page.
2. Select the to-dos in the list: make sure the to-do is alive.
3. Click the Reset  button.

TO-DOS											!	✕	⌵
☐	To-do ID	Model Instanc	Title	Type	Status	Locked by	Initial Perform	Interruption r	Task	Issued	Finished		
☐	12000	8000	2	Task1	Alive		[anyPerf...		todosMo...	Feb 15, 2...			
☐	12001	8000	1	Task2	Alive	admin (a...	[anyPerf...		todosMo...	Feb 15, 2...			
☒	12002	8002	2	Task1	Alive	admin (a...	[anyPerf...		todosMo...	Feb 15, 2...			
☐	12003	8002	1	Task2	Alive		[anyPerf...		todosMo...	Feb 15, 2...			

3.3.6 Exporting and Importing a Saved To-Do for Editing

If you want to modify runtime data of a saved to-do, you can do so directly in its raw XML. Note that if you modify data that is persisted in the database, the data remains unchanged.

To export a to-do state, do the following:

1. Open the To-Dos page.
2. Click the to-do.
3. On the To-Do detail page, click the *Export To-Do State* or *Import To-Do State* respectively.

Important: Importing a corrupt to-do state can cause the system to fail to work with the to-do.

3.4 Module Management

Modules are parts that are either used for reuse or for creation of Model instances.

From the Management Console, you can do the following with Modules:

- [upload modules to the server](#)
- [display module details](#)
- [download modules](#)
- [unloading modules](#)

3.4.1 Uploading Modules

Important: You can upload only Modules exported with the **GO-BPMN export**.

To upload a Module with all its module imports or a Model (an executable module with any imported modules) to the LSPS Server, do the following:

1. In the Modules view, click the Upload  button.
2. Define the [database update strategy](#).
3. In the Model Upload dialog box, locate and select the model, and click Upload.
4. Click the Upload button.

3.4.2 Displaying Module Details

To view details of a Module in the Management Console, do the following:

1. Open the Modules view.
2. Click the Module ID.
3. Expand the nodes to examine the Module structure. For Goal, Plan, and Process diagram node, you can click the *Diagram* link to display the diagram content.

MODULE 2003 - DIAGRAMS ←		
Module	Diagram URI	Action
▼ Imported modules		
▶ core		
▶ human		
▶ ui		
▼ NewBookEntry.gobpmn		
▼ NewBookEntry		
◊ Main	diagrams//1.0//NewBookEntry.go...	Diagram
▶ BookEnterByHead		
▶ BookEnterbyLib		
▶ EnterBookMach1		
▼ EnterBookMach2		
◊ Main	diagrams//1.0//NewBookEntry.go...	Diagram
●		
●		
● EnteredAutomatically		
● EnteredManually		
● NewBookEntered		
diagrams.datatypes		
diagrams.orgmodel		

Figure 3.26 Details of a Module

3.4.3 Downloading a Model or Module

You can download a model or module as a zip archive. Such an archive can be imported into PDS and edited.

To download a module from the Engine repository, do the following:

1. Open the Modules view.
2. In the modules table, select the module.
3. Click the Download  button.
4. In the dialog box, click Download.

3.4.4 Unloading Modules

To unload a module, do the following:

1. On the Modules page, select the module.
2. In the view toolbar, click Unload .
3. In the displayed dialog box, click Unload.

The model is unloaded from the server and any model instances based on the model are finished. Roles and role assignments remain unchanged.

3.5 Model Instance Management

Model instances are execution entities that follow the execution logic defined in their Model and imported Modules.

From the Management Console, you can do the following:

- [create model instances of uploaded models](#)
 - [finish model instances](#)
 - [suspend and resume model instances](#)
 - [restart model instances](#)
 - [filter model instances](#)
 - [display details of model instances](#)
 - [export and import raw contexts of model instances](#)
 - [change data of model instances](#)
 - [remove a waiting signal of a model instance](#)
 - [evaluate an expression in the context of a model instance](#)
 - [update a model instance](#)
-

3.5.1 Running a Model Instance

When you run a model, a *model instance* is created. In the model instance, a module instance of the executable module and module instances of all its imported modules are created. For more details, refer to section [Model Instance in the GO-BPMN Language Guide](#).

To run a model, do the following:

1. Select the executable module in the list of modules.
2. Click Run Model  in the view toolbar.
3. In the Run Model dialog box, type the required data and add properties of the model instance if required (any key-value pairs).
4. Click Run.

A model instance is created and its detail appears.

3.5.2 Filtering Model Instances

To filter the Model Instances view, do the following:

1. Define the filtering criteria in the filtering row of the table:
 - In the Model Instance ID text box, type the model instance ID.
 - In the Status drop-down box, select the execution status.

Where applicable, filtering is by default case-sensitive and may contain wildcards (*, ?); for example, search for `Task? *` returns results containing the word `Task` followed by any character, a space, and any subsequent characters.

2. Press Enter.

3.5.3 Display Model Instance Detail

To display a model instance detail, click the model instance ID in the Model Instance view in the Model Instance ID column.

3.5.3.1 Display Live Diagrams

To display a live diagram with the execution, do the following:

1. On the menu, click Model Instances.
2. On the right, click the ID of the model instance.
3. Go to the *Context* tab.
4. Locate the diagram in the tree and click Diagram in the Action column.

Important: The diagrams in the Management console do not display hyperlink and, lane and pool diagram elements.

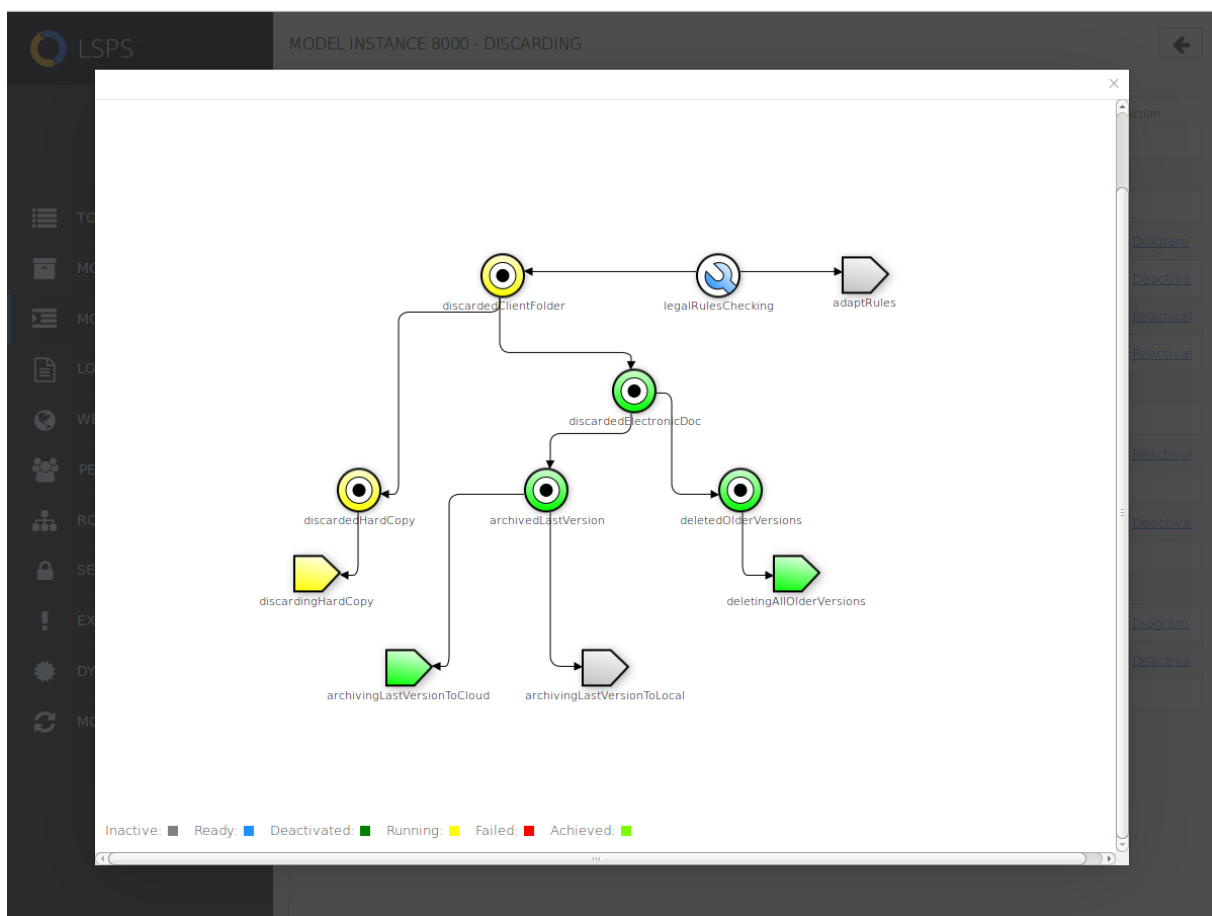


Figure 3.27 Live Goal diagram

3.5.4 Changing Model Instance Data

To change global variable values and goal status of a running model instance, do the following:

1. On the menu, click Model Instances.

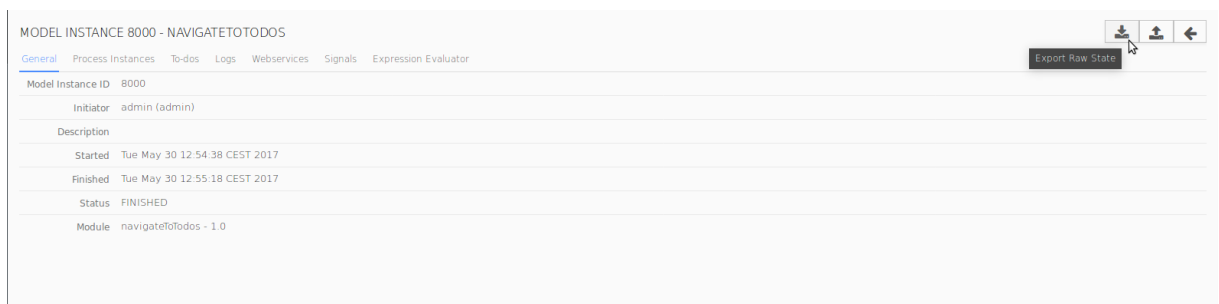
2. On the right, click the ID of the model instance.
3. Go to *Context* tab.
4. Locate the Goal or global variable:
 - For a variable, click Edit in the Action column.
 - For a Goal, click the Deactivate or Reactivate button in the Action column.

3.5.5 Exporting and Importing a Model Instance for Editing

If you want to modify runtime data of a model instance, you can do so directly in its raw XML. Such data includes values of variables, signals, etc. Note that if you modify data that is persisted in the database, the data remains unchanged.

To export a model-instance XML, do the following:

1. Open the Model Instances view.
2. Click the model instance.
3. In the model instance detail, click the *Export Raw State* or *Import Raw State* button.



Important: Importing corrupt model-instance data can cause that the system will fail to work with the data.

3.5.6 Evaluating Expressions

To evaluate an expression in the context of a model instance:

1. Display the Expression Evaluator tab (display the Model Instances view, click the desired model instance ID, and select the Expression Evaluator tab).
2. Unselect **Persistent** to evaluate the expression without changing of the data in the database.

If the Persistent flag is selected and the expression changes the value of an entity, for example, a global variable, the change is reflected on the runtime data, that is, the variable takes the newly assigned value.
3. In **Evaluation context**, select the context you want to use.
4. Type the expression in the input field of the evaluator.
5. Click Evaluate.

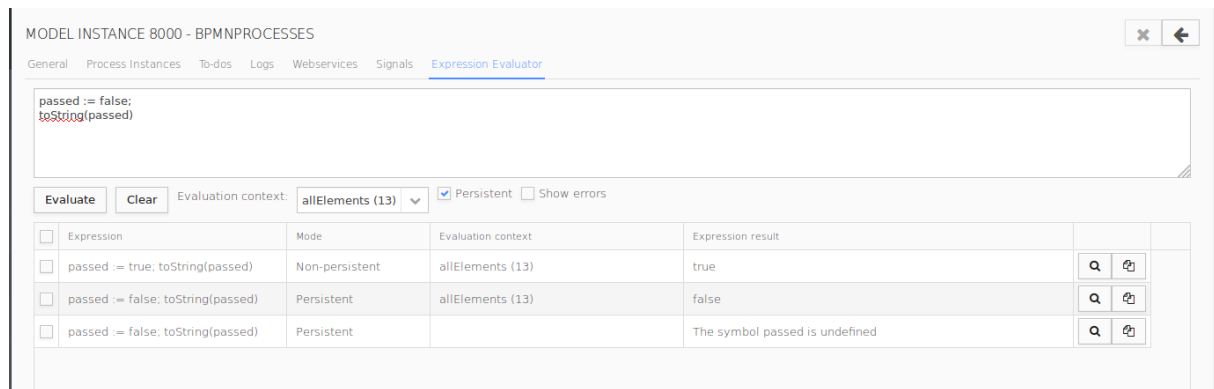


Figure 3.28 Evaluating an expression in different contexts and modes

To show errors, which occurred when attempting to evaluate expressions, select the *Show Errors* check box below the expression input field.

To remove evaluated expressions from the table, select the expressions and click the *Remove (X)* button in the upper-right corner.

3.5.7 Changing Persisted Data

To change persisted data from the Expression Evaluator, do the following:

1. Display the Expression Evaluator tab (display the Model Instances view, click the desired model instance ID, and select the Expression Evaluator tab).
2. Select Persistent to have the changes on persisted data (Shared Records) in the evaluated expression reflected on the database.
3. Type the expression in the input field of the evaluator.
4. Click Evaluate.

To display the errors that occurred when attempting to evaluate expressions, select the Show Errors check box below the expression input field.

To remove evaluated expressions from the table, select the expressions and click Remove (X) in the upper right corner.

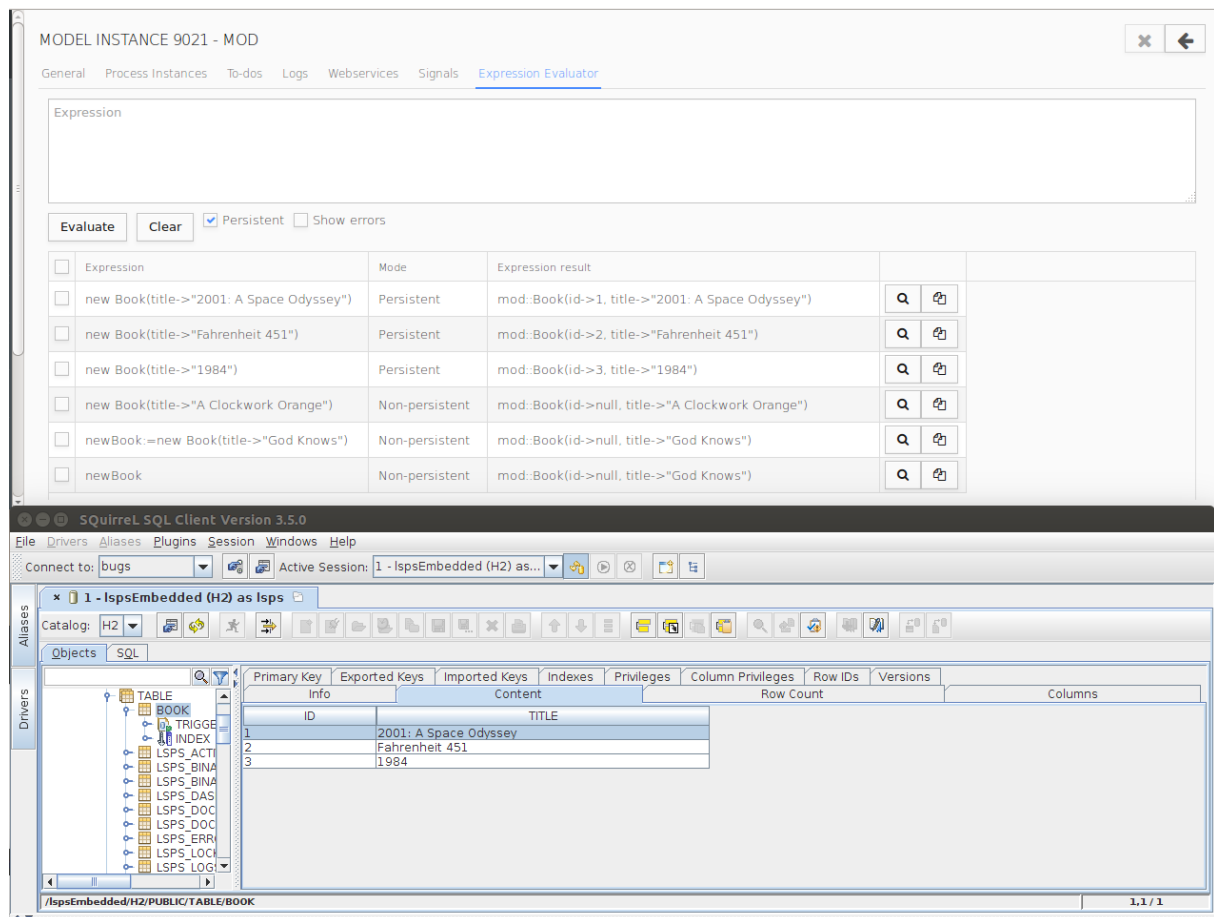


Figure 3.29 Evaluating over a shared record

3.5.8 Removing Waiting Signals

To remove a waiting signal of a model instance, do the following:

1. Display the Signal tab of a particular model instance (Model Instances model instance ID Signals).
2. Select the signals you want to remove.
3. Click the Remove (X) button in the upper right corner.

3.5.9 Filtering Wait Points

To apply a filter, do the following:

1. Open the Webservices view.
2. In the filtering row of the table, define the filtering criteria.
Filtering is case-sensitive and supports wildcards (*, ?); for example, filtering Task? * returns results containing the word Task followed by any character, a space, and any subsequent characters.
3. Press Enter.

3.5.10 Suspending and Resuming a Model Instance

A **suspended model instance** stops its execution. On resume, the instance continues from where it was suspended.

To suspend or resume a suspended model instance, do the following:

1. Open the Model Instances view.
2. In the displayed table with model instances, select the model instances.
3. In the view toolbar, click the Suspend  or Resume  command.

3.5.11 Finishing a Model Instance

When you **finish a model instance**, the running elements stop their execution immediately and the status of the model instance becomes **finished**.

To finish a model instance, do the following:

1. Open the Model Instances page.
2. In the displayed table with model instances, locate the particular model instance.
3. In the view toolbar, click the Finish  button.

MODEL INSTANCES							     
☐	Id	Model ID	Model name	Model version	Started	Finished	Status
☐	9000	3003	mod	1.0	2015-09-21 17:43	Sep 22, 2015 4:35:...	Finished
☐	9001	3004	delivery	1.0	2015-09-22 08:46	Sep 22, 2015 4:35:...	Finished
☐	9002	3004	delivery	1.0	2015-09-22 08:46	Sep 22, 2015 4:35:...	Finished
☐	9003	3003	mod	1.0	2015-09-22 08:46	Sep 22, 2015 12:0:...	Finished
☐	9004	3005	repair	1.0	2015-09-22 08:46	Sep 22, 2015 4:35:...	Finished
☐	9005	3003	mod	1.0	2015-09-22 08:47	Sep 22, 2015 4:35:...	Finished
☐	9006	3005	repair	1.0	2015-09-22 08:47	Sep 22, 2015 4:35:...	Finished
☐	9007	3006	repair	1.0	2015-09-22 13:54	Sep 22, 2015 4:35:...	Finished
☐	9008	3007	repair	1.0	2015-09-22 13:55	Sep 22, 2015 1:55:...	Finished
☐	9009	3008	repair	1.0	2015-09-22 13:57	Sep 22, 2015 1:57:...	Finished
☐	9010	3009	repair	1.0	2015-09-22 14:50	Sep 22, 2015 3:04:...	Finished
☐	9011	3010	mod	1.0	2015-09-22 15:23	Sep 22, 2015 4:35:...	Finished
☐	9012	3011	mod	1.0	2015-09-22 15:24	Sep 22, 2015 4:35:...	Finished
☐	9013	3012	mod	1.0	2015-09-22 15:54	Sep 22, 2015 4:35:...	Finished
☐	9014	3013	mod	1.0	2015-09-22 15:56	Sep 22, 2015 4:35:...	Finished
☐	9015	3014	mod	1.0	2015-09-22 15:56	Sep 22, 2015 4:35:...	Finished
☐	9016	3009	repair	1.0	2015-09-22 16:03	Sep 22, 2015 4:35:...	Finished
☐	9017	3015	mod	1.0	2015-09-22 16:19	Sep 22, 2015 4:35:...	Finished
☐	9018	3016	mod	1.0	2015-09-22 16:30	Sep 22, 2015 4:35:...	Finished
☒	9019	3018	mod	1.0	2015-09-22 16:37		Running
☐	9020	3018	mod	1.0	2015-09-22 16:41		Running

Figure 3.30 Finishing Model Instance

3.5.12 Restarting a Model Instance

When you restart a running model instance, the module instances of the model instances finish and executable module instances are created within the same model instance. The new module instances run under the same model instance so the ID and other properties of the model instance are preserved. The context of the finished module instances does not have any impact on the context of the new module instances.

To restart one or multiple model instances, do the following:

1. On the **Model Instances** page, select the Model instances.
2. Click Restart  in the page toolbar.
3. In the *Restart Model Instance* dialog box, select the Model to be used by the restarted Model instance.

Restart Model Instance
+
x

☐ Use the current model of selected model instances
☒ Select model

Id	Name	Version	Upload Date
2,003	old_ui_grid	1.0	Dec 22, 2016 4:21:41 PM
2,003	old_ui_grid	1.0	Dec 22, 2016 4:21:41 PM
2,004	old_ui_grid	1.0	Dec 22, 2016 4:22:49 PM
2,005	machineModule	1.0	Dec 22, 2016 4:50:36 PM
2,006	constraints	1.0	Dec 22, 2016 4:50:45 PM

4/4

OK
Cancel

Figure 3.31 Selecting the Model for the restarted Model instance

4. Click OK.

A model instance is created and its detail appears.

3.5.13 Updating a Model Instance

Any running or suspended model instances can be updated so their module instance can follow a new model. During model update, the model instance becomes suspended and the underlying model is substituted with a new model; the model update procedure is defined in the `model update configuration file`, which is created in PDS.

However, if you have a model update configuration at hand, you can perform the update directly from the Management Console.

To update a model instance, do the following:

1. In the Model Instances view, select the model instance and click the Model Update  button.
2. In the dialog box, enter the path to the model update MUC file and the Datamapping file and click Next.
3. Check if the correct instances are selected and click Finish
4. It is recommended to check the model update log information in the Model Update Logs view.

3.6 Saved Documents

The documents users have saved and are hence probably working on are available in the *Saved Documents* view. You can export the documents as XMLs and import them back after adapting the runtime data. Note that the features are subject to the security right `Document:Write`.

3.6.1 Exporting a Saved Document

To export a saved document to XML, do the following:

1. Open the Saved Documents view.
2. Click the saved document.
3. Click the *Export Saved Document State* button.

You can edit the downloaded XML and then import it back into the document.

3.6.2 Importing a Saved Document

To import modified data of a saved document back to the server, do the following:

1. Open the *Saved Documents* view.
2. Click the *Import Saved Document State* button.

Important: Importing corrupt document state can cause the system to fail to work with the document.

3.7 Person Management

Persons represent users that interact with the system, such as, administrator, or participate in to-do execution, hence have access to the Application User Interface. As such they define credentials, such as, name, login, etc., and have a set of rights assigned.

You can view and manage all the details in the [Person Detail View](#).

3.7.1 Filtering Persons

The Persons page is displayed.

To apply a filter, do the following:

1. In the filtering row of the table, define the filtering criteria.
Filtering is case-sensitive and supports wildcards (*, ?); for example, filtering `Task? *` returns results containing the word `Task` followed by any character, a space, and any subsequent characters.
2. Press Enter.

3.7.2 Creating Persons

To add a new person to the LSPS server, do the following:

1. Open the Persons view.
2. Click the Add  button in the view toolbar.
3. In the new view, define user details.
4. Optionally, define substitution settings: select Substitution active and select the substitutes.
5. Select the modeled roles you wish to assign the person.
Assign a person at least one security role.
6. Click Submit  in the view toolbar.

3.7.3 Editing Persons

To edit an existing person, do the following:

1. Open the Persons view.
 2. Click the login of the person you wish to edit.
 3. In person detail, click Edit .
 4. Edit the displayed user information.
 5. Optionally, [define substitution settings](#).
 6. Select the roles to be assigned to the person.
 7. Click Submit  in the view toolbar.
-

3.7.4 Disabling and Enabling Persons

When persons are disabled, they are denied access to the LSPS runtime environment, hence they cannot access the front-end clients.

The disabling mechanism is used when you want to remove a person: once persons are created, they cannot be deleted from the system.

When persons are disabled, the following applies:

- Any assigned security and modeled roles, and substitutes are permanently removed and are not renewed on enabling.
- Any locked to-dos are unlocked and released (to-dos are distributed to persons with the required rights, or become orphaned).

3.7.5 Activating Substitution

You have defined substitutes for the particular person.

To activate or deactivate substitution, do the following:

1. Open the respective person detail (Persons USER_ID)
2. Make sure you have [defined your substitutes](#).
3. Select or unselect Substitution active.
4. Click Submit.

Note: The *admin* and *processAgent* users cannot be disabled.

A disabled person can be enabled. However, the removed data, for example, their roles, are not restored.

Information about person status is shown in the respective person detail (Enabled).

To enable or disable a person, do the following:

1. Open the Persons view.
2. Select the check box of the persons you want to disable or enable.
3. Click Disable Selected Persons  or Enable Selected Persons .

Admin is a read-only person and cannot be disabled.

3.7.5.1 Defining Substitutes

To define substitutes of a person, do the following:

1. Click Persons in the navigation bar.
2. In the persons table, click the person's login.
3. On the person detail page, click the Edit  button.
4. Select the Modeled roles to assign a button.
Editable person detail appears.
5. Select *Substitution active*.
6. On the Select Substitutes for Person page, type the search criteria into the search field and click Search.
Wildcard usage is supported (to display all possible substitutes, type *).
7. Select a substitute in the text box and click Add.
8. Activate substitution if required: select Substitution active.
9. Click OK.

3.8 Role Management

After a GO-BPMN module is uploaded, the Execution Engine creates roles and organization units defined by organization schema in the module. The elements are defined by their name and parent module and are referred to as *modeled roles*.

You can assign these roles to persons as *runtime roles*, which are server versions of the modeled roles possibly with parameter values.

For information on how parametric roles work, refer to [Roles](#) and the [GO-BPMN Modeling Language Guide](#).

3.8.1 Assigning a Role to multiple Persons

To assign a role to multiple persons, do the following:

1. Open the Roles view and select one of the tabs with roles.
2. Click the role name.
3. In the displayed role detail, click Edit .
4. Select the persons and click Submit  button.

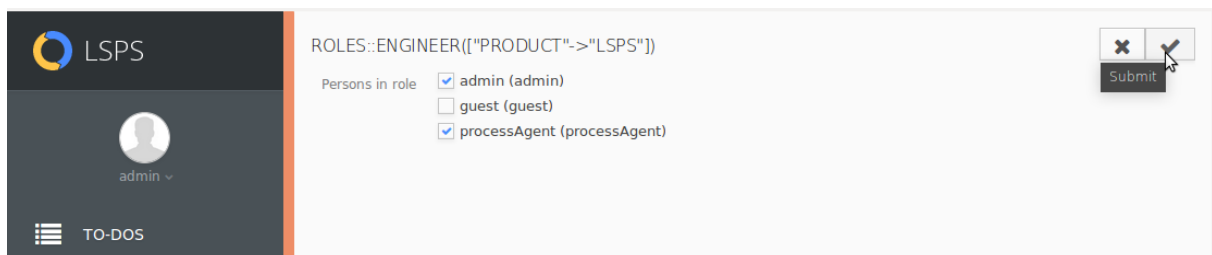


Figure 3.32 Assigning a runtime role to multiple persons

3.8.2 Assigning multiple Roles to a Person

To assign a person a role, do the following:

1. Open the Persons view.

2. Click the person id.

3. Click the Edit  button.

4. Select the Modeled roles to assign as runtime roles to the person.

To assign the modeled roles with a parameter specified, click the *Add parametric role* button and enter the details in the popup.

5. Click the Submit  button.

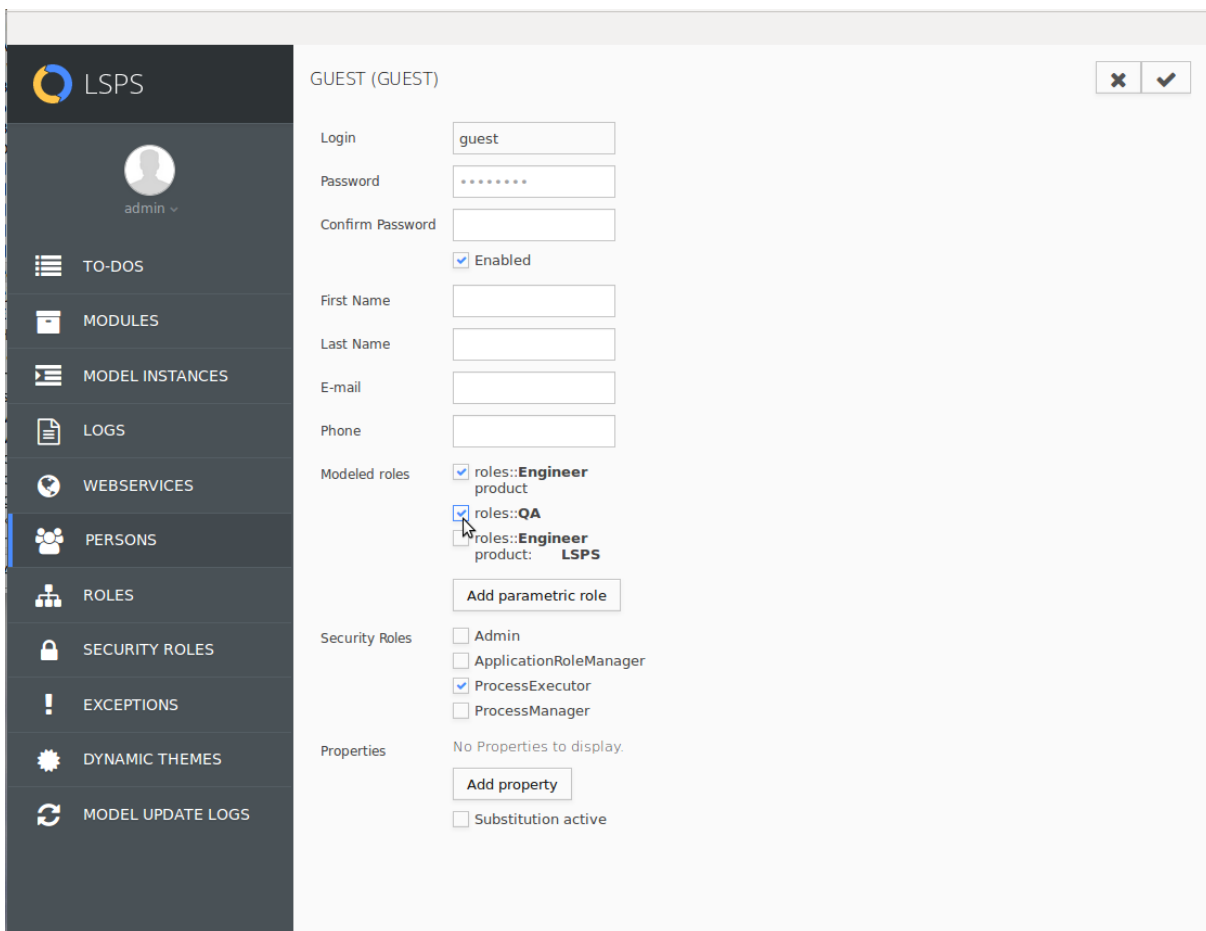


Figure 3.33 Assigning multiple roles to a person

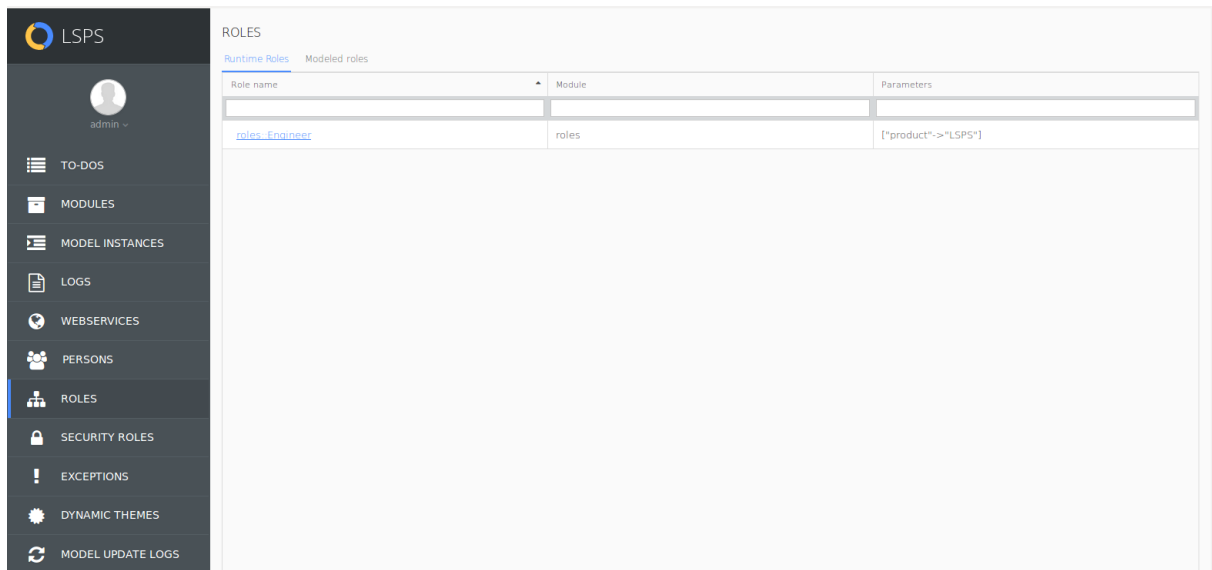


Figure 3.34 Runtime Roles tab

3.9 Security Role Management

Security roles are sets of [security rights](#) which can be assigned to users.

Important: If a person does not have a security right for an action, the result is absence of the respective GUI components in the Application User Interface; for example, if a person does not have a security role with the right `Todo:Read_Own`, the To-do List navigation item will not be displayed in the application when the person is signed in.

3.9.1 Creating Security Roles

You can create security roles with a custom set of security rights.

To create a new security role, do the following:

1. Open the Security Roles view.
2. Click Add Role  .
A new page with a list of security rights appears.
3. In the Role name text field, enter the new security role name.
4. Select security rights for the person.
5. Click the Submit  button.

The new security role is available in the list in the Security view and is ready to be assigned to a user.

3.9.2 Assigning Security Roles

To assign a person a security role, do the following:

1. On the Persons page, click person's login.
2. In the person detail view, click the Edit  button.
Editable person detail appears.
3. In the Security Roles area, select the security roles of the person.
4. Click Submit  .

3.9.3 Deleting Security Roles

To delete a security role, do the following:

1. Display the Security Roles view.
2. Select the roles to be deleted.
3. Click Delete  .

The deleted security roles disappear from the list and their assignment is discarded (the security roles are removed from all persons).

3.10 Log Management

3.10.1 Filtering Logs

The Logs page is displayed.

To apply a filter, do the following:

1. In the filtering row of the table, define the filtering criteria.
Filtering is case-sensitive and supports wildcards (*, ?); for example, filtering `Task? *` returns results containing the word `Task` followed by any character, a space, and any subsequent characters.
2. Press Enter.

The log list shows only logs meeting the defined criteria.

3.11 Exception Management

When a model instance fails with an Exception, the Execution Engine rolls back the transaction that caused the exception. Such model instances are recorded on the Exceptions page (for further information on transactions in model instance, refer to *Transactions in Model Instances* in the *Software Development Kit* guide).

From the Management Console, you can do the following over Exceptions:

- [filter the listed exceptions](#)
- [display their stacktraces](#)
- [remove exceptions from the log](#)
- [resend input to the transaction](#)

3.11.1 Filtering Exceptions

The Exceptions view is displayed.

To apply a filter, do the following:

1. In the filtering row of the table, define the filtering criteria.
Filtering is case-sensitive and supports wildcards (*, ?); for example, filtering `Task? *` returns results containing the word `Task` followed by any character, a space, and any subsequent characters.
2. Press Enter.

The table with exceptions shows exceptions containing the defined stacktrace part.

3.11.2 Displaying Exception Stacktrace

The Exceptions view is displayed.

To display an exception stack trace, click the exception id. If you need the full stacktrace, open PDS, connect to your server and inspect the exception in the Exception view of the Management perspective.

3.11.3 Removing Exceptions

To remove an exception entry, in the Exceptions view, select the exception and click the Remove button.

3.11.4 Resending Inputs on Model Instances with an Exception

Resend attempts to execute again a transaction that previously failed: on resend the system resends the inputs which caused the exception to the model instance. Before using this function you will typically fix the underlying data of the model instance or database data, or make sure that an external service was fixed, etc.

To resend the data, do the following:

1. Select the exceptions in the table.
 2. Click the Resend button in the upper right corner.
-

3.12 Application Settings

3.12.1 Switching Languages

The application can be localized into multiple languages. For the Sample Application, German and English localization is available by default. To switch the local, click the arrow next to your user name; change the locale on the Settings page and click Save.

3.12.2 Changing User Details

To edit user details, such as password, email address, phone number, and avatar, click the arrow next to your user name; change the details and click Save.

3.12.3 Dynamic Themes

In the Web Management Console, you can dynamically change the CSS theme using the Sampler available on the Dynamic Themes view: the view contains a list of custom CSS themes present on the server. If you prefer to modify the themes programmatically, download the theme: the package contains the CSS and file with external properties.

If you want to modify properties that are not externalized, modify the SCSS and generate the CSS file anew: you can then upload the archive to the Management Console.

Note: To modify themes programmatically and include them in the application, you will need the SDK and generate the LSPS sample application. For details, refer to [Developing a Custom Application User Interface](#).

3.12.3.1 Creating Themes

Important: It is not possible to change the theme of the Process Management Console. Custom themes apply only to Application User Interface.

To create a theme and upload it to the server from the Management Console, do the following:

1. In the Management Console, go to **Dynamic Themes**.

2. Click the Sampler  button in the upper right corner.

3. In the Sampler view, define the attributes of the theme:

- On the General tab, define the overall theme attributes.
- On the Menu tab, define the attributes of the side menu.
By default, the color of the menu background is computed based on the background color. To use a custom color, unselect the compute check box.
- On the Layout tab, define the margins of the layout components used in to-do and document definitions.
- On the Logo tab, upload the logo and define its properties.
- You can reset the theme to the default state with the Reset  button.

4. Click the *Preview changes* button to preview the theme.

Note that the theme preview is applied only to the Dynamic Themes view and is dropped when you leave the page. However, the page retains the most recent theme settings.

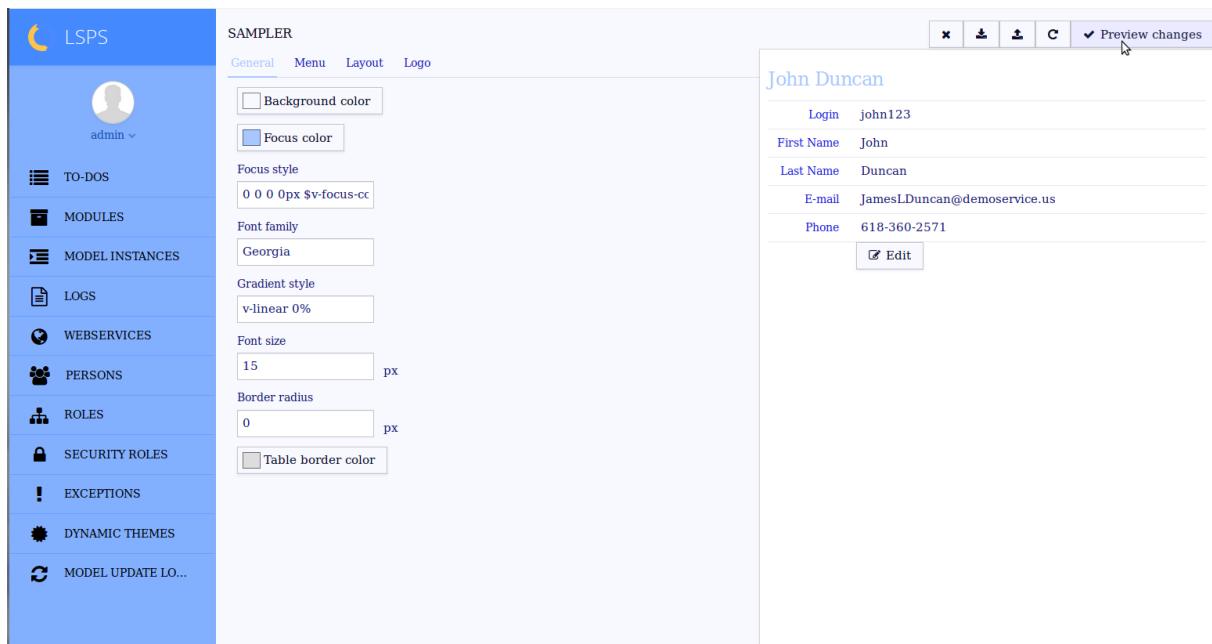


Figure 3.35 Previewing theme

5. Once you are satisfied with the changes, you can upload the theme to the server with the Upload  button.

The theme will become available in the Application Settings of the user details (click the user name in the menu bar and select Settings from the context menu). Uploaded themes are available to all users with access to their Settings.

3.12.3.2 Modifying Themes

Every user can modify only properties within Sampler and hence only the last theme from the Process Management Console. It is not possible to modify or download other uploaded themes from the Management Console: however, you can download the theme, integrate them into your Application User Interface (refer to [Custom Application Guide](#)).

3.12.3.3 Deploying Themes

Note that you need to upload a zip with CSS themes and possibly file with modified theme parameters: consider downloading the resources first and then introducing modifications to them.

To upload a zip archive with CSS themes for the Application User Interfaces, do the following:

1. In the Management Console, go to Dynamic Themes.

-
2. Click the Upload  button in the upper right corner.
3. In the popup dialog, enter the path to your CSS zip file and click Upload.

The theme is now available in the Application Settings of the user details (click the user name in the menu bar and select Settings from the context menu). Uploaded themes are available to all user with access to their Settings.

3.12.3.4 Undeploying Themes

To undeploy a themes from the server, do the following:

1. In the Management Console, go to Dynamic Themes.
2. Select the theme from the list.
3. Click the Unload  button in the upper right corner.

3.12.3.5 Downloading the Sampler Theme

From the Management Console, you can download the theme resources, which include the CSS file, the sources (SCSS, files with theme parameters), etc. For more information on how to work with Sass, refer to [Development Guide](#).

To download a zip archive with the CSS files of the last modified theme for the Application User Interfaces, do the following:

1. In the Management Console, go to Dynamic Themes:
 2. Click the Sampler button.
 3. In the toolbar above the theme details, click the Download  button in the upper right corner.
 4. In the popup dialog, enter the path to your CSS zip file and click Download.
-

Chapter 4

Command-Line Console

The Command Line Console is a command-line client that enables you to perform administrative action on the LSPS Server from command line. It is provided as batch and bash file `lspcli` in the `$LSPS_RUNTIME_HOME/cli-tools` folder of the LSPS Runtime package.

Before you can use the Command Line Console, install a **supported JDK** and set up the environment variable `JAVA_HOME` since the batch and shell scripts are wrapper scripts that run the respective java method with the LSPS libraries on the classpath and forward it the call parameters.

To send a call using the Command Line Console, do the following:

1. Open your command line interface.
2. On the prompt, run the `lspcli` script with the respective parameters. The supported parameters are described in Command Line Console Commands.

Example tool run

```
~/lsp-runtime/cli-tools$ ./lspcli.sh savedTodoExport -h http://localhost:8080/lsp-ws -u admin
```

Note that `--properties` and `--filterProps` are defined as pairs of values, for example, `property1 value property2 value2`. If the properties use whitespaces, use the quotes for the given property or value, for example, `"property 1" value property2 "value 2"`

The `lspcli` tool provides the following commands:

Important: If you require an additional command for a management action, please, contact support.

4.1 Model Management

modelList lists the models in the model repository on the LSPS Server;

Long Option	Short Option	Description
<code>--host*</code>	<code>-h</code>	host URL
<code>--username*</code>	<code>-u</code>	user name
<code>--password*</code>	<code>-p</code>	password
<code>--name</code>	<code>-n</code>	model name (supports wildcards)
<code>--version</code>	<code>-v</code>	model version (supports wildcards)

* required parameters

```
~/lsps-runtime/cli-tools$ ./lsps-cli.sh modelList --host http://localhost:8080/lsps-ws --username
Found 5 model(s).
Id      Model name    Model version
2003    15224         1.0
18000   15224         1.0
2002    ui            2.8
2000    core          2.8
2001    human         2.8
```

modelUpload uploads a model;

Important: You can upload only Modules exported with the **GO-BPMN export**.

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--model	-m	list of model filenames or directories
--dbUpdateStrategy*		strategy for database handling possible values: none, update, validate, drop

* required parameters

```
lsps-cli modelUpload -host http://localhost:8080/lsps-ws -model "C:/models/TestUploadInvalidModel"
```

```
~/lsps-runtime/cli-tools$ ./lsps-cli.sh modelUpload --host http://localhost:8080/lsps-ws --username
Info: Uploading files from /home/eko/lsps-workspace/Queries
Uploading files from /home/eko/lsps-workspace/Queries has finished.
```

modelUnload removed the module from the server; note that roles and role assignment remain unchanged.

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--name	-n	model name (supports wilcards)
--force		do not prompt on unload

* required parameters

```
~/lsps-runtime/cli-tools$ ./lsps-cli.sh modelUnload --host http://localhost:8080/lsps-ws --username
Found more models:
Are you sure to remove model '15224 (1.0)' from repository? (Y/N)y
Model 15224 (1.0) successfully unloaded.
```

4.2 Model Instance Management

When calling any of the management commands, make sure to define the model instances you perform the command on. If you want to work with one model instance with a particular id, define its id using the `--modelInstance` parameter. If you want to perform the command over multiple instances, use one of the filter parameters:

- `--modelInstanceFile` parameter with a file with the IDs
- filter parameters, `--filterStatus`, `filterFrom` and `filterTo`, `--filterProps`, etc. for multiple model instances based on their properties.

modelInstanceList returns possibly filtered list of model instances;

Long Option	Short Option	Description
<code>--host*</code>	<code>-h</code>	host URL
<code>--username*</code>	<code>-u</code>	user name
<code>--password*</code>	<code>-p</code>	password
<code>--modelInstance</code>		model instance ids
<code>--modelInstanceFile</code>		file containing the base model
<code>--filterModelId</code>		IDs of the base models
<code>--filterModelName</code>		model name pattern (wildcards supported)
<code>--filterModelVersion</code>		model version (wildcards supported)
<code>--filterStatus</code>		models in the defined execution status (CREATED, RUNNING, SUSPENDED, FINISHED)
<code>--filterFrom</code>		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
<code>--filterTo</code>		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
<code>--filterProps</code>		model instance properties defined as <code>key1 value1</code> <code>key2 value2</code>
<code>--filterExpression</code>		boolean expression; Only the model instances for which the expression evaluates to true are returned.

* required parameters

```
./lspcli.sh modelInstanceList --host http://localhost:8080/lspcli --username admin --password
```

export exports a model instance as an XML file;

Long Option	Short Option	Description
<code>--host*</code>	<code>-h</code>	host URL
<code>--username*</code>	<code>-u</code>	user name
<code>--password*</code>	<code>-p</code>	password
<code>--modelInstance</code>		model instance IDs
<code>--modelInstanceFile</code>		file with model instance IDs
<code>--location</code>		output location and file prefix (model instance IDs are appended to the prefix)

* required parameters

exportRaw exports a raw model instance state;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance IDs
--modelInstanceFile		file with model instance IDs
--location		output location and file prefix (model instance IDs are appended to the prefix)

* required parameters

importRaw import a raw model instance state;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance*		model instance ID
--file*		file with model instance

* required parameters

suspend suspends model instances;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance IDs
--modelInstanceFile		file with model instance IDs to be requested
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

```
./lsps-cli.sh suspend --testFilter --host http://localhost:8080/lsps-ws --username admin --passw
```

resume resumes model instances;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance IDs
--modelInstanceFile		file with model instance IDs to be resumed
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

finish finishes model instances

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance IDs
--modelInstanceFile		file with model instance IDs to be resumed
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

restart restarts running or suspended model instances

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password

Long Option	Short Option	Description
<code>--modelId</code>		ID of the model used for the restarted model instances (if not specified, each instance is restarted without model change)
<code>--modelInstance</code>		model instance IDs
<code>--modelInstanceFile</code>		file with model instance IDs to be restarted
<code>--filterModelId</code>		IDs of the base models
<code>--filterModelName</code>		model name pattern (wildcards supported)
<code>--filterModelVersion</code>		model version (wildcards supported)
<code>--filterFrom</code>		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
<code>--filterTo</code>		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
<code>--filterProps</code>		model instance properties defined as "key1 value1 key2 value2"
<code>--filterExpression</code>		boolean expression; Only the model instances for which the expression evaluates to true are returned.
<code>--testFilter</code>		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

4.3 Model Update

When calling any of the model update commands, the commands will be executed on all model instances of the model as defined in the provided muc file.

If you want to work only with one model instance with a particular id, define its id using the `--modelInstance` parameter. If you want to perform the command over multiple instances, use one of the filter parameters:

- `--modelInstanceFile` parameter with a file with the IDs (each ID on a new line)
- filter parameters, `--filterStatus`, `filterFrom` and `filterTo`, `--filterProps`, etc. for multiple model instances based on their properties.

Consider using the `--testFilter` parameter to trigger a dryrun and check the resulting set of model instances.

Note: To acquire the model update command, you can set up a test model update and use the [Show command line script](#) feature in PDS.

Example model update command

```
lsps-cli modelUpdate --host http://localhost:8080/lsps-ws --username admin --password admin --muc mucFile
```

modelUpdate updates models instances based on the provided muc file

Long Option	Short Option	Description
<code>--host*</code>	<code>-h</code>	host URL
<code>--username*</code>	<code>-u</code>	user name
<code>--password*</code>	<code>-p</code>	password
<code>--muc*</code>		muc file

Long Option	Short Option	Description
--datamapping		data mapping definition
--modelInstance		model instance IDs
--modelInstanceFile		file with model instance IDs to be updated
--filterStatus		only model instances in the defined execution status are updated (RUNNING, SUSPENDED, UPDATED, UPDATE_ABORTED)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

continueTransform triggers the transformation phase of update on the model instances;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance ids
--modelInstanceFile		file containing the base model
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

continuePostprocess triggers the postprocess phase of model update;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance ids
--modelInstanceFile		file containing the base model

Long Option	Short Option	Description
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

abortModelUpdate aborts update of model instances;

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelInstance		model instance ids
--modelInstanceFile		file containing the base model
--filterModelId		IDs of the base models
--filterModelName		model name pattern (wildcards supported)
--filterModelVersion		model version (wildcards supported)
--filterFrom		model instance(s) initiated after TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterTo		model instance(s) initiated before TIMESTAMP, format: yyyy-MM-dd HH:mm:ss
--filterProps		model instance properties defined as key1 value1 key2 value2
--filterExpression		boolean expression; Only the model instances for which the expression evaluates to true are returned.
--testFilter		boolean expression; displays the model instances affected by the command (the command is not executed)

* required parameters

startModelInstance instantiates and starts a model instance

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--modelName		base model name
--modelVersion		base model version
--modelId		base model ID
--properties		model instance properties defined a list of key-value pairs

* required parameters

```
./lsps-cli.sh startModelInstance --host http://localhost:8080/lsps-ws --username admin --password admin
# alternatively:
./lsps-cli.sh startModelInstance --host http://localhost:8080/lsps-ws --username admin --password admin
```

4.4 Document Management

savedDocumentExport exports a saved document

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password

* required parameters

savedDocumentImport imports a saved document

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--savedDocumentId*		saved document id
--file*		file with saved document state

* required parameters

4.5 To-Do Management

savedToDoExport exports a saved to-do

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--todoId*		to-do id
--location		output location prefix (todo id will be appended)

* required parameters

savedToDoImport imports a saved document

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name

Long Option	Short Option	Description
--password*	-p	password
--todoId*		todo ID
--file*		file with to-do state

* required parameters

4.6 Engine Management

listSettings lists the engine settings

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password

* required parameters

updateSetting updates the engine setting with the value

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--name	-n	setting name
--value	-v	setting value

* required parameters

addSetting adds a new setting with the defined value

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--name	-n	setting name
--value	-v	setting value

* required parameters

deleteSetting removes a setting

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--name	-n	setting name

* required parameters

4.7 Person Management

updatePerson updates person's password

The server must be restarted to apply any password changes.

Long Option	Short Option	Description
--host*	-h	host URL
--username*	-u	user name
--password*	-p	password
--new-password*	-np	new password

* required parameters

```
./lsps-cli.sh updatePerson -h http://localhost:8080/lsps-ws -u admin -p admin -np admin2
```


Chapter 5

Reference

5.1 Security Rights

A security role given to a person allows the person to perform actions based on the assigned security rights.

The following security rights are available:

Binary Security Rights

Binary rights govern the operations on binaries stored directly to database.

Security right	Description
Binary:Add	adding binary resources; allows the user to upload files directly to database (this is done from the ui::Upload components with <i>Upload to memory</i> set to <i>false</i>)
Binary:Delete	removing binary resources
Binary:Read	reading binary resources

Debugger Security Rights

Security right	Description
Debugger:Manage	managing debugger—setting, activating, deactivating the debug mode, and adding, updating and removing breakpoints

Document Security Rights

Document rights govern the access to the [Documents](#) uploaded as part of modules.

Security right	Description
Document:Read	reading documents
Document:Submit	editing and submitting documents
Document:Write	import document state

Exception Security Rights

Exception rights restrict the access to [exceptions in model instance execution](#).

Security right	Description
Exception:Read	retrieving rollback information
Exception:Remove	removing rollback info
Exception:Resend	resending task data from the Exception Handling view

Expression Evaluation Security Rights

Security right	Description
Expression:Evaluate	evaluating expressions in runtime (in the Expression Evaluator)

Form Preview Security Rights

Form preview right serve to restrict access to [Form preview](#)

Mind that this feature should be [disabled completely on other than development servers](#).

Security right	Description
Form:Preview	running form previews (note that you need the Model:Manage right to be able to preview forms)

Goal State Security Rights

Security right	Description
GoalState:Update	changing states of goals

Management Security Rights

Security right	Description
Management:Login	access to the management tools, the Management Console and Management perspective

Model Security Rights

Security right	Description
Model:Delete	deleting model instances
Model:Manage	adding and uploading models to the repository and exporting uploaded models to archive files
Model:Read	finding, reading, and updating models and GO-BPMN modules in the repository and reading resources (files) for a given model

Model Instance Security Rights

Security right	Description
ModelInstance:Create	creating model instances
ModelInstance:Notify	notifying model instance from a Web Service
ModelInstance:Read	retrieving model instance related information
ModelInstance:Suspend	suspending running model instances
ModelInstance:Terminate	finishes running model instances
ModelInstance:Update_Model	changing the model for running model instances
ModelInstance:Write	import XML model instance state

Person Security Rights

Security right	Description
Person:Change_Own_Password	changing own password
Person:Manage	managing general information and associations to modeled roles of persons
Person:Read	retrieving person related information

Report Security Rights

Security right	Description
Report:Dashboard_Management	enables the definition of common dashboard tabs
Report:Read_All	reading all available reports (this right overrides the security setting defined for the report)
Report:Read_Own	reading reports, which the user is entitled to

Role Security Rights

Security right	Description
Role:Manage	managing role
Role:Read	reading roles (read-only access)

Upload Schema Security Rights

Security right	Description
Schema:DropCreate	availability of drop-create strategy for business objects
Schema:Update	updating strategy of business objects
Schema:Validate	validating schema of business objects

Security Role Security Rights

Security right	Description
SecurityRole:Manage	adding, renaming, and removing security roles to/from the persistent storage and assigning rights to security roles and acquiring a set of all rights
SecurityRole:Read	retrieving security roles and associated users

Settings Security Rights

Security right	Description
Settings:Manage	access to the Settings of the Management Console and Application User Interface

Signal Security Rights

Security right	Description
Signal:Remove	removing a signal from the model instance queue
Signal:Send	sending signals to model instances

Todo Security Rights

Security right	Description
Todo:Delegate_All	delegating any to-dos
Todo:Delegate_Own	delegating to-dos of the particular person
Todo:Escalate_All	escalating any to-dos
Todo:Escalate_Own	escalating to-dos assigned to the particular user
Todo:Process	getting, submitting and canceling a to-do
Todo:Read	acquiring the list of to-dos meeting the given criteria (disables access to the Todo List)
Todo:Read_All	reading all available to-dos (this right overrides the security setting defined for the to-do)
Todo:Read_Assignees	displaying to-do assignees (in to-do details)
Todo:Read_Own	reading to-dos, which the user is entitled to
Todo:Reassign	reassigning to-dos
Todo:Reject	rejecting to-dos
Todo:Undo_Reject	cancelling to-do rejection
Todo:Write_All	import XML to-do state

Context Security Rights

Security right	Description
Variables:Update	changing values of variables of the given context

Web Service Security Rights

Web Service rights serve to restrict access to data about the web services provided by the modules.

Security right	Description
Webservice:Invoke	platform specific right
Webservice:Read	monitoring of current modeled Web Services

Technical and Deprecated Security Rights

Security right	Description
Lock:Manage	acquiring or releasing data locks
Testing:All	internal security right; Do not assign this right to any users.
