Eclipse Passage Operator

User Guide

October 6, 2021

ArSysOp

Contents

Introduction

Overview

Eclipse Passage is a set of tools providing rich and easily adaptable capabilities to declare and control licensing constraints. Being an open-source project under Eclipse Foundation, it is entirely developed with Java 11 by the *ArSysOp* company.

This user guide contains a glossary, which gives you base understanding of terms used in Eclipse Passage and two sets of instructions: for developer, who wants to license some features or entire application with Passage, and for operator, who wants to manage licenses and metadata with the Eclipse Passage Operator.

Happy Reading!

Contribution

If you found a mistake in the text or just want to improve this user guide, feel free to contribute on Github repository.

Also we want to remind that Eclipse Passage is an open-source project, so you can always contribute to the project itself.

License

Components

Eclipse Passage consists of three parts:

Licensing Components

Licensing Components is a set of tools you have to include in your application to be able to declare any licensing logic (For example, what to do if a correct license was not acquired by the licensing framework). It allows you to restrict unauthorized using of a single feature, bundle or even the whole application.

Operator

Operator client is a separate application giving you a manual control under all the licenses your product can have (both personal and floating). Also, it is required in access cycle in order to define the product or/and features metadata and to generate keypair for license file encoding.

Floating License Server

Eclipse Passage FLS is a component implementing the Floating License Server concept. Works roughly in a way described below.

The licensee acquires (in any way) a finite number of licenses and these licenses are stored on the License Server. When an authorized user wants to use the application, they request a license from the server, and if the license pool is not empty (in other words, there are still licenses available), the server gives user an access to the application. When user finishes his work with the application, the license is released and can be obtained by the other authorized user.

Passage Operator User Guide

Introduction

Main concepts

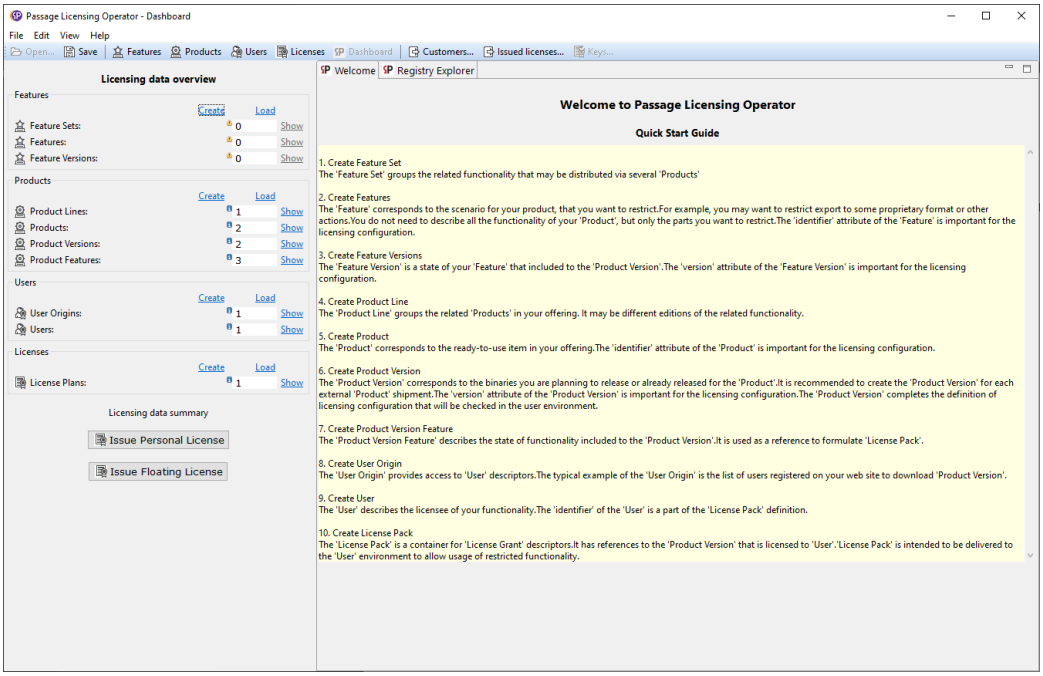
The best approach to learn any existing user interface is understanding the concepts it is based on. In Eclipse Passage worldview the product licensing sequence consits of three parts:

- **Definition** - on this step you define your product lines, products and all the features it has to be licensed with Eclipse Passage. Responsibility for this step usually lies on the marketers' shoulders.
- **Evolution** - when you defined the product and features it has, it's time to start development. New versions are being released, some functionality can be added, changed or even dropped away.
- **Licensing** - after you have at least one version of your product or feature released, you can start issuing licenses to your users.

These are three main steps to have your features properly licensed with Passage Operator.

UI Overview

Eclipse Passage Operator's initial layout is quite simple: dashboard with all products, features, users and license plans contained in your workspace; workbench part (initially filled with welcome page), where you you usually deal with some licensing data (E. g. you can define your features here), top toolbar with some useful wizards and controls, two buttons to issue a license.



How to license your product

This tutorial contains three parts mapped with three licensing steps mentioned in the section above.

Definition

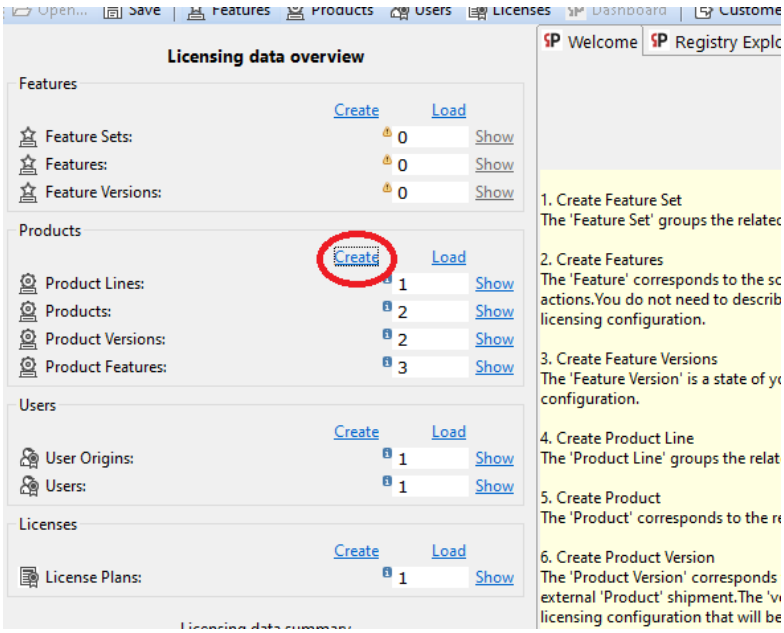
First of all, we have to define what our product is. To be more concise with the reader, tutorial at all is provided with examples based on Eclipse Passage commercial version (Cordon).

Usually some products are provided separately around the same system (library or something like that). In our case ArSysOp Cordon provides a simple product line, containing three licensable components:

- Cordon Access Cycle (Licensing Components)
- Cordon Operator (Like Passage Operator)
- Cordon Floating License Server

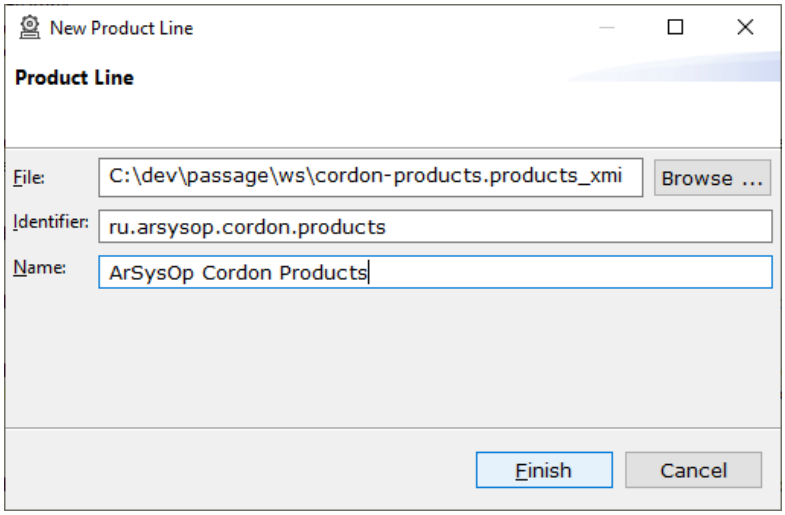
Create Product Line

Let's open product line creation dialog by pressing the *Create* button near the Products area.



In the dialog appeared we have to choose a place, where our product line model will physically exist (as .xmi model), choose unique identifier and any beautiful name.

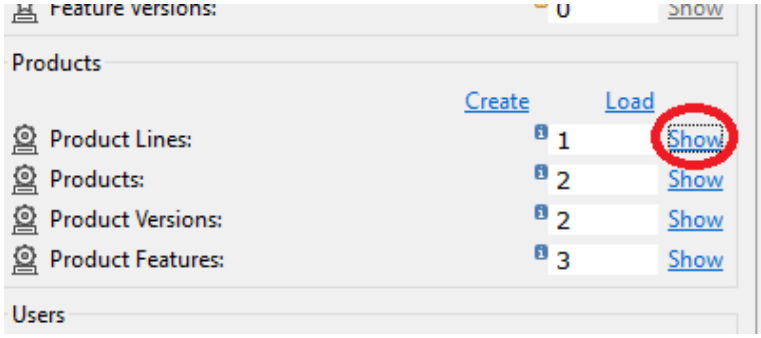
Remember location where your xmi files are saved. These model files can also be imported with *Load* button.



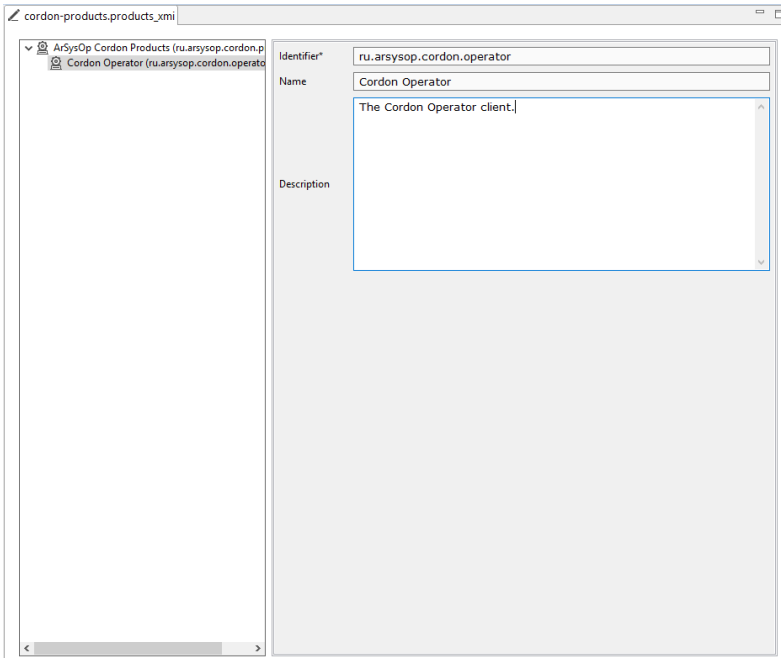
Press *Finish* and that's it! My congratulations, you have just defined your first product line with Passage! Let's fill it with some products.

Fill Product Line with Products

Press *Show* near the Product lines count. Now you see the workbench with one defined product line and no products under it.



On your product line with right click open the context menu, where you will see an option named *Product*. Pressing it you define new product under the selected product line. Let it be the Cordon Operator in our case. Fill all the required fields and optional fields if needed.



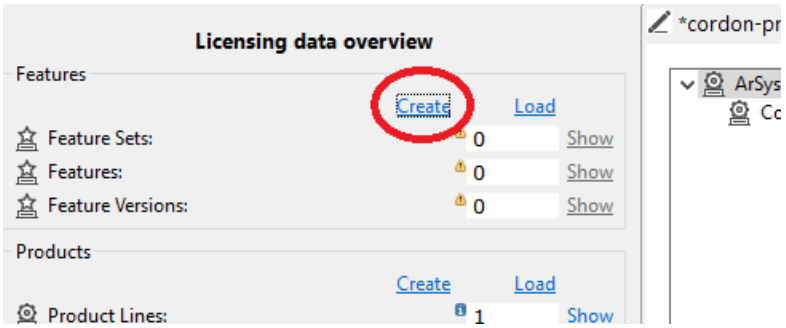
Congratulations again! The product is now created and ready to be filled with features, which we are about to define.

Define Features

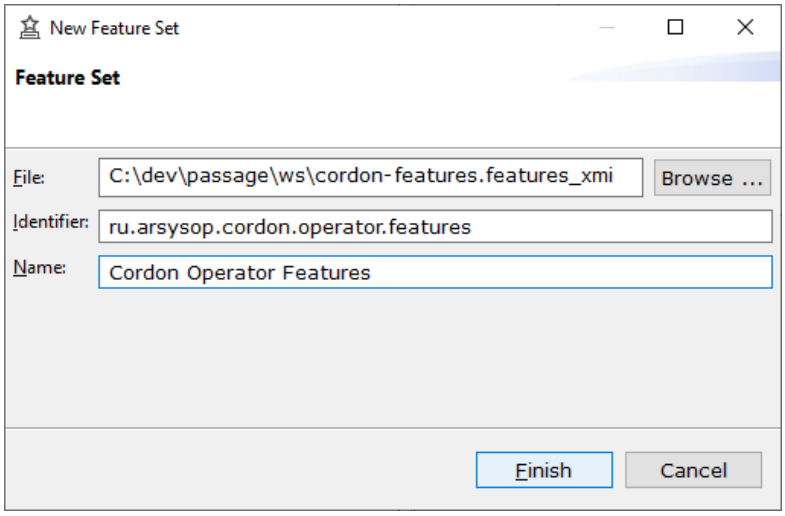
Let's imagine we have two licensing cases for our product:

- we want to check license on start for all users
- we want one feature to be licensed separately - not all users who bought Cordon Operator License are allowed to use it. Let it be license usage statistics, for example.

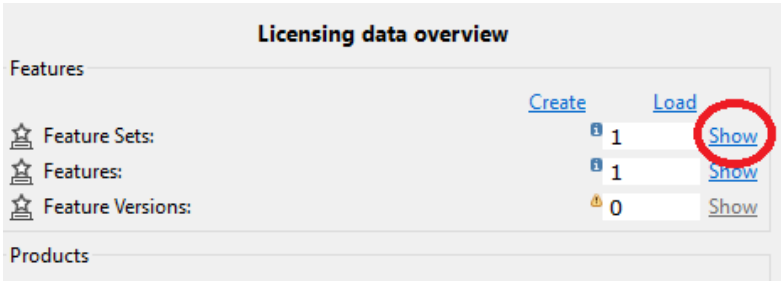
First, we have to define a feature that will represent usage statistics collection. Press *Create* button near the Features area to create Feature Set.



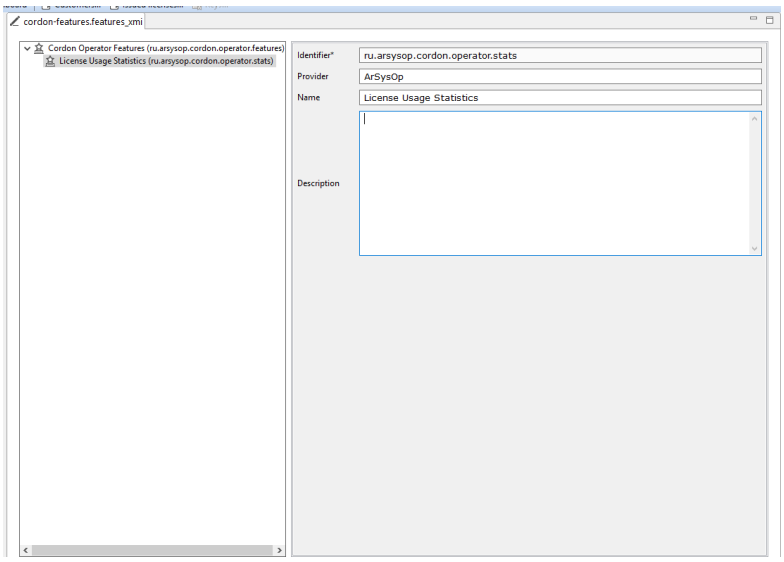
And then select model file location, feature set id and name.



Press *Finish* and then *Show* near the feature set count.



Like the product under product line, create the feature under feature set with the context menu showed on right-click, then fill identifier and name fields at least and save.



Now we successfully defined our product exceptional feature. To have our product licensed at all, we need to define another feature with **the same identifier as the product has**. Also, developer must use the same identifier when they define their product with Eclipse.

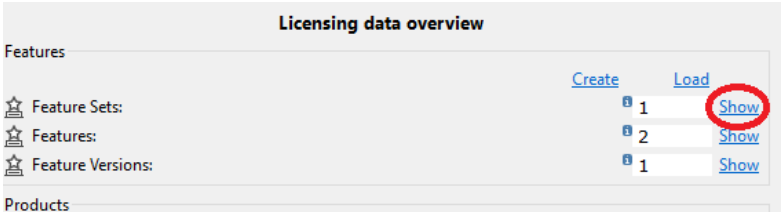
For now we have one product line, one product in it and some exceptional feature to be licensed separately. That's quite enough to proceed to the next part - **evolution**.

Evolution

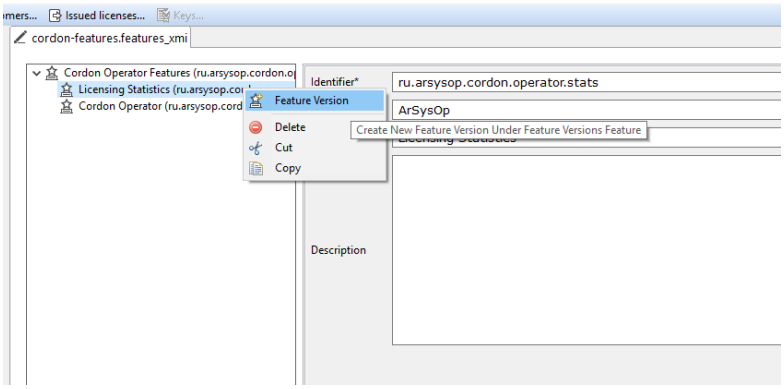
Now you defined the features product has, and developers can finally start working around it. When the first version is about to release, it's time to define a new product and feature versions.

Define Feature Version

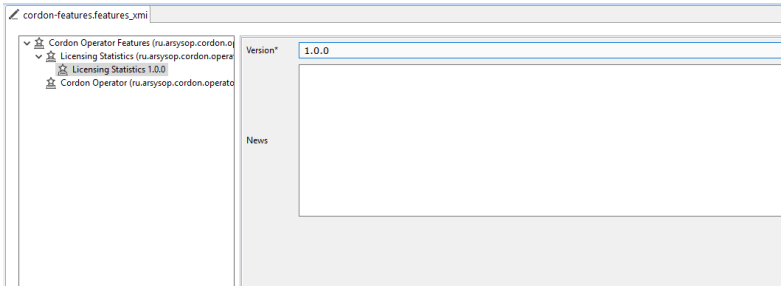
Let's define our first feature version. Press *Show* button near the Feature sets count: you will see the workspace with two features you defined earlier (if you did, of course).



Then right-click on any feature (we will use previously mentioned stats feature), and then select Feature Version in the context menu.



The only thing you have to define here is feature version itself. In ArSysOp we usually prefer semantic versioning to choose correct version names, so in this tutorial we will follow these simple rules (you can have a look at them at the provided website).



After all you will see something like this.

Not really obvious, but important remark here is the idea, that in Passage every product ***is*** *a set of features. In fact you can't license a product, but you can license a feature, that represents the product at all or the feature "to*

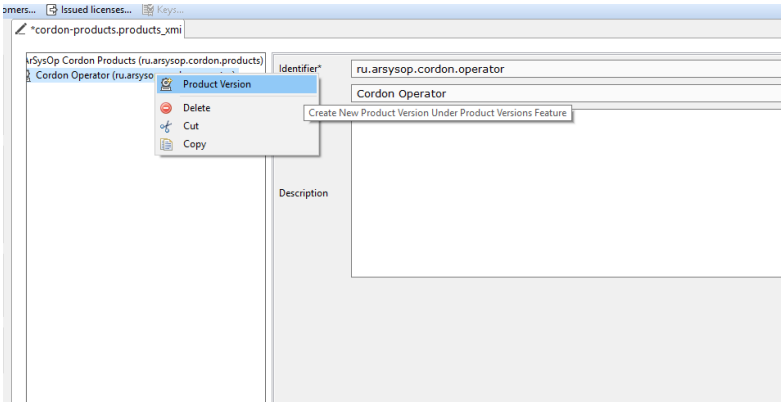
launch product" - up to you. So, let's create an .operator feature version as well.

Now we can go on to the product version definition. Product version contains a set of feature versions: a single release - a single set of feature versions in it.

Define Product Version

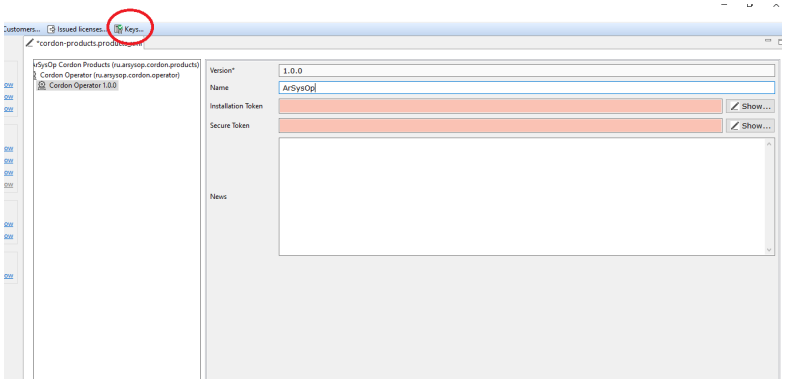
Open the products workbench with *Show* button near the product lines count.

To create new product version, right-click in the workbench on the product you want to distribute.



Generate keypair

When you have created product version, after selecting it (just left-click on it) you will see *Keys* button activated.



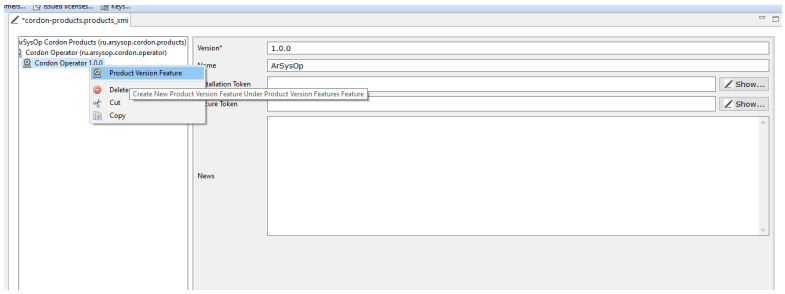
Press it to generate a keypair to encode and decode license files. Passage uses asymmetric encoding to deal with license data (this is required to protect licenses from being stolen or falsified). Public and private keys are now located in passage workspace subfolder representing your product (usually passage workspace is located under your user home folder), you have to share the public one with your developer to include it into the product. The private one remains with you to issue licenses for your product.

If you see private and public key fields colored in white (instead of red they were), you've done all right.

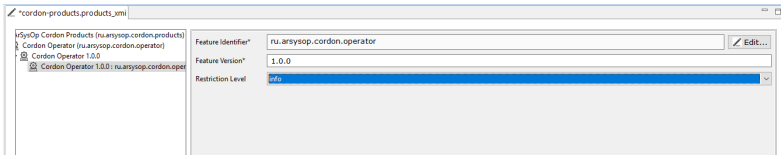
Include Features

Now let's fill our product version with feature versions (as we mentioned above, the product is a set of features).

First, create new product version feature under your product version: this node binds feature version to product version.



When the version is created, select the feature you want to include, then type in the version you created earlier and select the restriction level.



Restriction Levels

Info This restriction level means that unauthorized usage of your feature is not really bad for your business logic. If user won't have a license for the feature with this restriction level, they won't even be notified. No restriction in fact.

Warn This one means that user will be able to use feature if he has no appropriate license, but also they will be notified that it's better to obtain a license for this feature.

Error Selecting this level restricts any unauthorized usage of the feature. If the user tries to use it with no license, the framework will interrupt him and offer to obtain a license in any way.

Fatal It is generally used for product definition features to hardly restrict even a launch of the product with no license.

Add remaining features

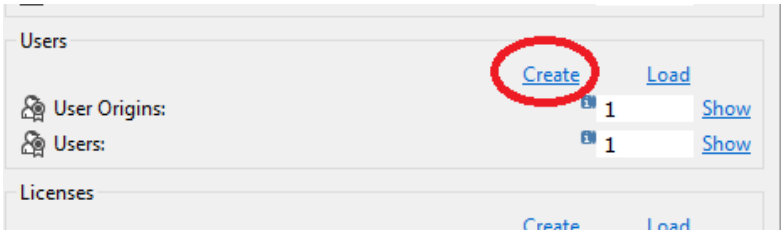
After all, add all your features you want to include in your product version and then you are ready to start issuing licenses.

Licensing

First of all, we have to tell Passage who is this lucky man about to obtain a license for your product. Inside Passage this lucky man is called *User*. Every user should have its *User Origin*. This user origin usually represents a company or any other group of users you want to work with.

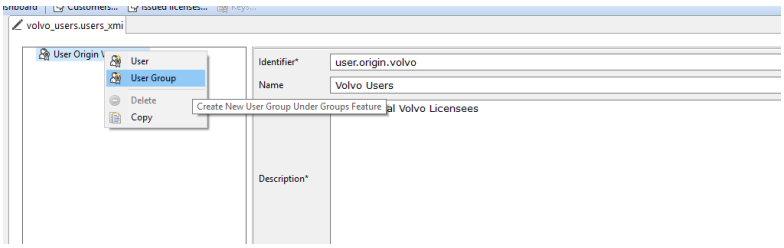
Create user origin

In the Users area press *Create* and create your first user origin. It can be your client company or any other user origin you wish. User Origin identifier and **description** here are mandatory.

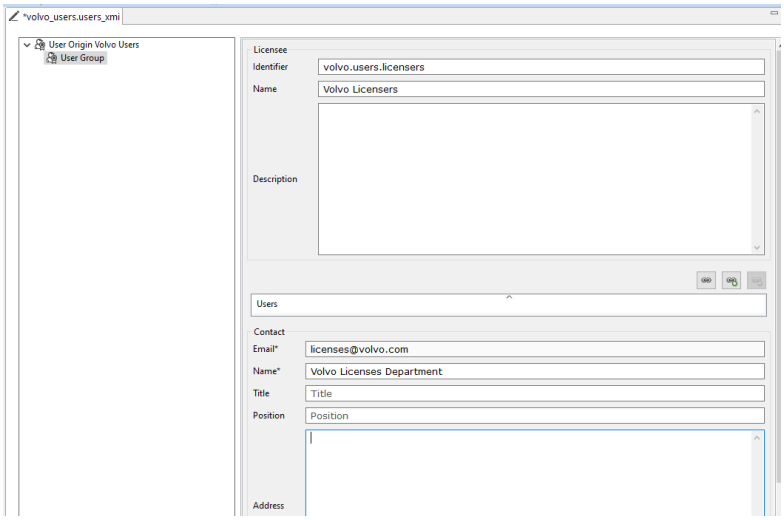


Create user group

In any company there are several groups of users with different features access. User Group represents a group of users of the same user origin. For Cordon Product sounds really well to create Licensers user group representing workers, who issue licenses for Volvo clients. Let's create it with the context menu on user origin.



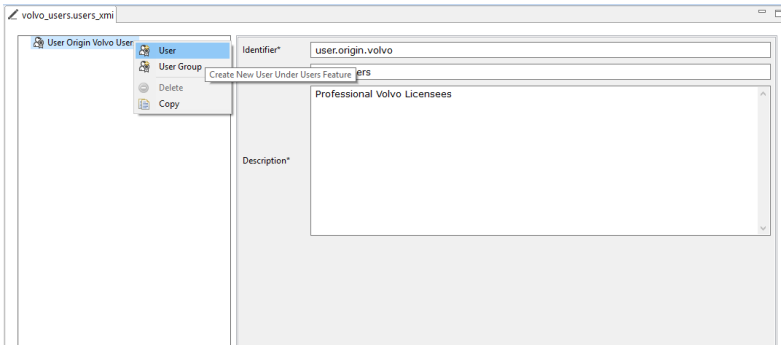
Fill all the required fields including email address. Passage uses it to automatically send licenses to licensees.



We've created a user group, but usually we don't issue a license to group of users (if it is not a floating license, of course), then we have to create at least one user.

Create user

Imagine Santa Claus working in Volvo. Let's create a user for him.



Fill identifier, name, email like in the previous part and also evaluation type.

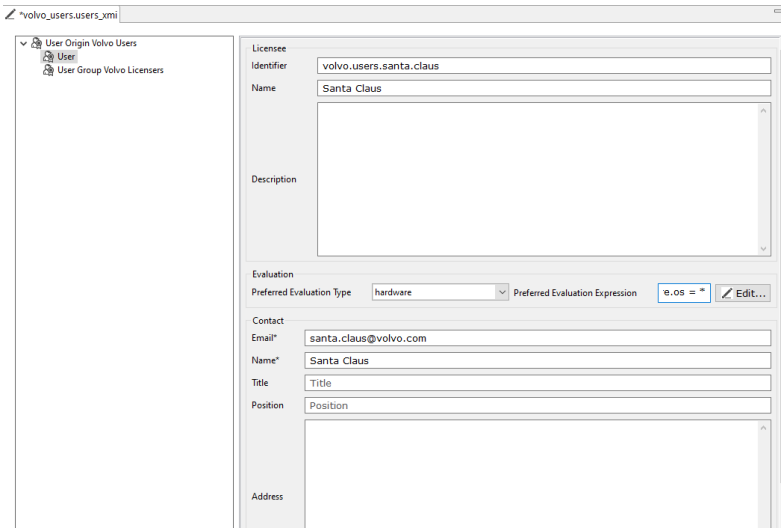
Evaluation type is a way how license is being authorized. We simply tell the framework a way to determine that it currently works on a computer of its user, not anyone else. For now there is only one option - hardware. It

means that the framework will get the parameters of your system (actually, the framework have already got, you can check them with *Inspect hardware* button under *Help* button in top toolbar) and then compare it using the **evaluation expression**, where the rule which hardware the framework should accept is located. Every rule simply looks like:

`hardware.PARAMETER = value (or * if any is appliable)`

For now let's create a rule that checks user OS and applies any of them.

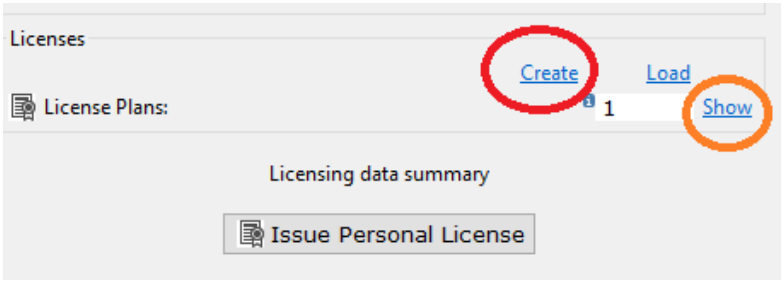
`hardware.os = *`



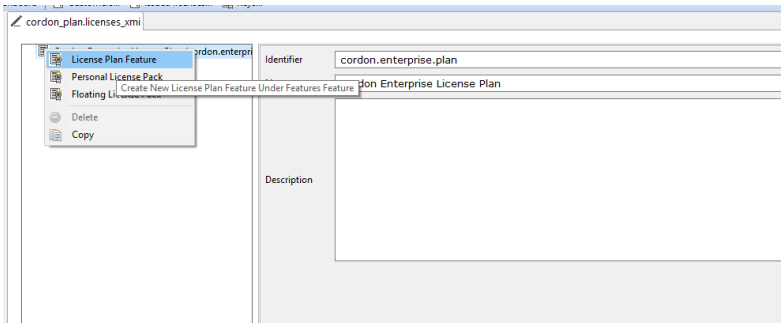
Now add our new user to user group and we are ready to create License Plan for our user.

Create license plan

First lets create a license plan - a set of licenses for your product line. Use *Create* button.



Then with *Show* button go to License Plans workbench and create new License Plan feature:



Select a feature, feature version and a versions compatibility rule.

Compatibility Rules

Compatible This rule accepts all the previous major versions and all minor versions before next major version. The example is always provided under the field.

Equivalent Accepts all the previous versions and all patches for current minor version.

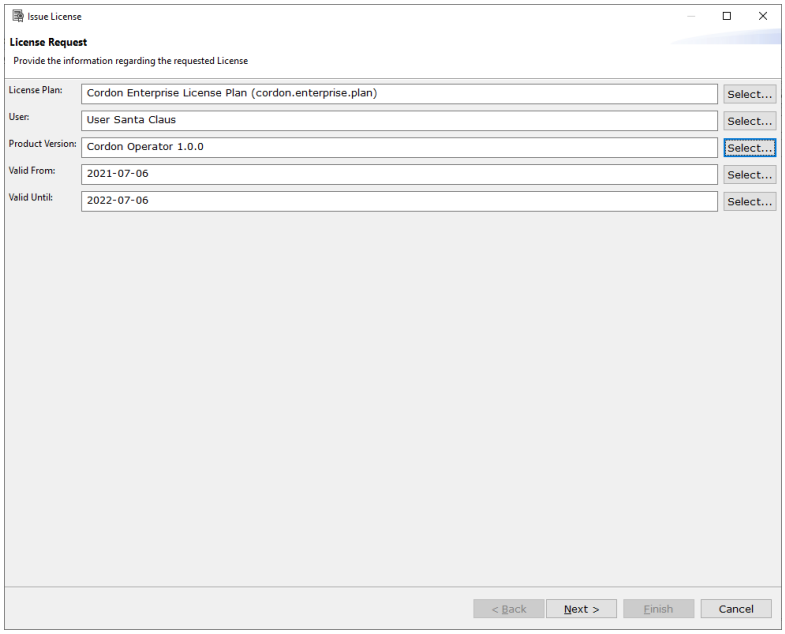
Greater or Equal In fact accepts every version.

Perfect Accepts only the version you selected.

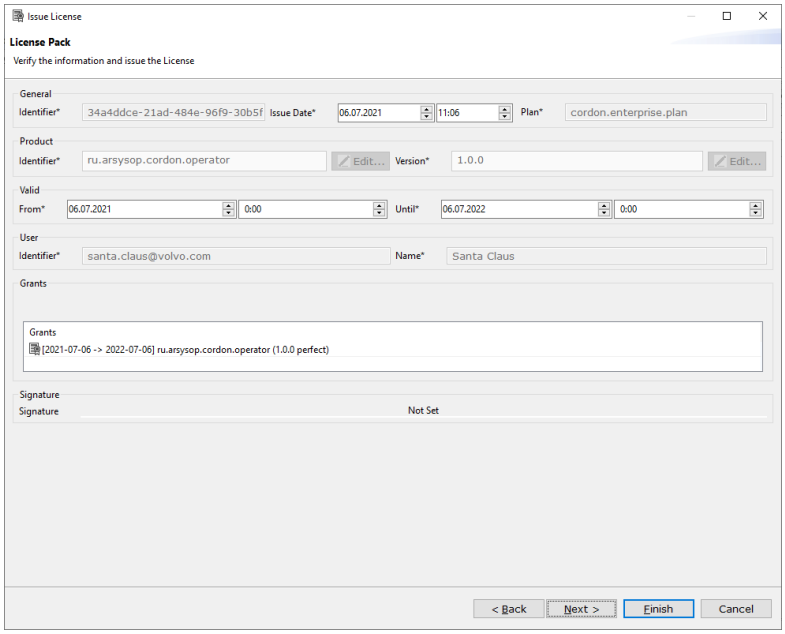
Issue a license

Now we are ready to issue a first license. Press the large *Issue a personal license* button in the left part of the window.

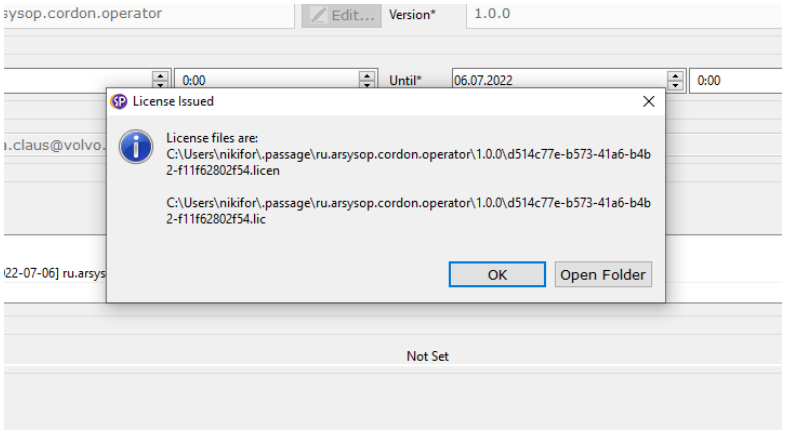
Fill the fields in the dialog with the entities we defined earlier. Select validity dates (*In a free version of Passage three months is the maximum validity period. If you have no enterprise license, Passage will automatically reduce the license validity period to three months, if it exceeds the limit.*)



Check if all the things are right and press *Finish*.



That's it! We've issued a license. Passage tells you where the license files for your user are located and the only thing you need to do here is sending .licen file to your end user.


sysop.cordon.operator
Edit...
Version*
1.0.0
0:00
Until*
06.07.2022
0:00
License Issued
License files are:
C:\Users\nikifor\.passage\ru.arsysop.cordon.operator\1.0.0\d514c77e-b573-41a6-b4b2-f11f62802f54.licen
C:\Users\nikifor\.passage\ru.arsysop.cordon.operator\1.0.0\d514c77e-b573-41a6-b4b2-f11f62802f54.lic
OK
Open Folder
Not Set

Glossary

Feature The 'Feature' corresponds to the scenario for your product, that you want to restrict. For example, you may want to restrict export to some proprietary format or other actions. You do not need to describe all the functionality of your 'Product', but only the parts you want to restrict. The 'identifier' attribute of the 'Feature' is important for the licensing configuration.

Feature Set The 'Feature Set' groups the related functionality that may be distributed via several 'Products'

Feature Version The 'Feature Version' is a state of your 'Feature' that included to the 'Product Version'.The 'version' attribute of the 'Feature Version' is important for the licensing configuration.

User The 'User' describes the licensee of your functionality. The 'identifier' of the 'User' is a part of the 'License Pack' definition.

User Origin The 'User Origin' provides access to 'User' descriptors. The typical example of the 'User Origin' is the list of users registered on your web site to download 'Product Version'.

Product The 'Product' corresponds to the ready-to-use item in your offering. The 'identifier' attribute of the 'Product' is important for the licensing configuration.

Product Feature The 'Product Version Feature' describes the state of functionality included to the 'Product Version'. It is used as a reference to formulate 'License Pack'.

Product Line The 'Product Line' groups the related 'Products' in your offering. It may be different editions of the related functionality.

Product Version The 'Product Version' corresponds to the binaries you are planning to release or already released for the 'Product'. It is recommended to create the 'Product Version' for each external 'Product' shipment. The 'version' attribute of the 'Product Version' is important for the licensing configuration. The 'Product Version' completes the definition of licensing configuration that will be checked in the user environment.