



---

**SysML**  
version 0.10.1-SNAPSHOT  
Developer Guide



## Table of Contents

---

1. <b>Table of Contents</b> .....	<b>i</b>
2. <b>Introduction</b> .....	
3. <b>Developer</b> .....	<b>1</b>
4. <b>SOP1 Norm requirement extraction</b> .....	<b>4</b>
5. <b>SOP2 Generation from Norm to code and configuration files</b> .....	<b>5</b>
6. <b>SOP3 Adding operations to the norm</b> .....	<b>8</b>
7. <b>SOP4 Requirement</b> .....	<b>10</b>
8. <b>SOP5 Release</b> .....	<b>12</b>
9. <b>FAQ Developer</b> .....	<b>15</b>



# 1 Developer

---

## 1.1 Download

The developer guide could be downloaded as a pdf [here](#)

## 1.2 Requirements

### 1.2.1 Eclipse

Use Eclipse, at least Neon version

### 1.2.2 Maven

Use Maven 3.3.1 at least

### 1.2.3 Eclipse plugins

Install M2e plugin in your Eclipse

Install Tycho Configurator as an additional maven connector

No specific extra from papyrus

### 1.2.4 SysML

Have a look to the normative document of OMG: [Embedded norm](#)

### 1.2.5 Check your installation by a basic checkout, compilation

- Clone the sysml 14 git repository git clone <https://git.eclipse.org/r/papyrus/org.eclipse.papyrus-sysml>.
- Run maven at the root of the repo: mvn clean install; it should pass
- Get sysml14 plugins in your eclipse workspace
- Et “Voila” you are good to go.

### 1.2.6 Target Environment

We have developed a target-platform-configuration artifact located at `/org.eclipse.papyrus-sysml/org.eclipse.papyrus.sysml14.targetdef/org.eclipse.papyrus.sysml14.targetdef.target`

Open it and click at the upper right corner to set it has the target platform.

## 1.3 Product Life Management

The PLM is Maven with Tycho plugins for OSGI.

### 1.3.1 Run a default installation

```
mvn clean install
```

### 1.3.2 To build also the modules relatives to RCP and Product, please activate the following profile:

```
mvn clean install -Pproduct
```

It could be necessary to use the following workaround to ensure the version of Neon plugins, you used to build against:

```
mvn clean install -Pproduct -Dtycho.localArtifacts=ignore
```

Sometimes when Ecclipse realse train is on the move, you will need to add the following option, to force to download directly from Eclipse main download site:

```
-Dtycho.disableP2Mirrors=true
```

### 1.3.3 Generate and deploy the web site

```
mvn clean site site:stage-deploy scm-publish:publish-scm -Pdocumentation
```

#### 1.3.4 A minimal iteration

This section describes the different steps relative to the integration of a new feature or to the correction of a bug: from bug definition, to requirement, to code integration through gerrit review with the continuous integration system.

- Describe in Bugzilla the bug, feature you are working on. Please use the component SysML. And begins with [SysML 1.4] for Summary.
- Get the code from the master with git, `git clone https://git.eclipse.org/r/papyrus/org.eclipse.papyrus-sysml`, and work on a local branch,
- Add the new requirement in the different sysml 1.1 models located in the doc folder of the relevant plugin. have a look to [SOP4: Requirement](#)
- Modify the code
- Push on Gerrit `ssh://{ECLISPEUSERLOGIN}@git.eclipse.org:29418/papyrus/org.eclipse.papyrus-sysml`
- Ask for a review `https://git.eclipse.org/r/#/dashboard/self`
- After a few iteration, you code should be merged and accessible in the master.

Have a look to default rules [miscellaneous.html](#)

#### 1.3.5 Tips

If you are working with multiple version of Papyrus, it is possible that Tycho do not pull the right one. You can force it by using the following parameter in the build command. More details [here](#)

```
mvn clean install -Dtycho.localArtifacts=ignore
```

Please find additional information here: [Miscellaneous](#)

## 1.4 Standard Operating Procedures and FAQ

### 1.4.1 SOPS

- [SOP1: OMG Norm requirements automanual extraction](#)
- [SOP2: From a OMG profile to a dedicated Papyrus application \(elementype, palette etc...\)](#)

- [SOP3](#): Extended the norm by adding a new operation.
- [SOP4](#): Add a new requirement.
- [SOP5](#): Make the release.

#### **1.4.2 Dev FAQ**

- [FAQ](#).

### **1.5 Miscellaneous**

#### **1.5.1 Libraries**

[QUDV and others](#)

## 2 SOP1 Norm requirement extraction

---

### 2.1 Pre-requisit

install pdfminer <http://www.unixuser.org/~euske/python/pdfminer/>

### 2.2 Extract the TOC

#### 2.2.1 Extract the toc from the pdf with pdfminer

```
dumppdf.py -T foo.pdf > toc.xml]]</source>
</subsection>
<subsection
name="Extract basic data from the generated toc file">
<source><![CDATA[ cat toc.beta.xml | egrep "<outline|pageno" | tr '\012' "@" | sed
```

#### 2.2.2 Layout the data

Generate the elements for each requirements

Create a basic maven java project with the following dependencies

```
<dependency>
<groupId>org.eclipse.emf</groupId>
<artifactId>org.eclipse.emf.ecore</artifactId>
<version>2.11.0-v20150123-0347</version>
</dependency>
<dependency>
<groupId>org.eclipse.emf</groupId>
<artifactId>org.eclipse.emf.common</artifactId>
<version>2.11.0-v20150123-0347</version>
</dependency>
```

Make a main class based upon the following code: [RequirementGenerator.java](#)

#### 2.2.3 Create a sysml model with the requirements

Create a sysml project and add the requirement data in it

Reference this new model/library from your MDE model project



## 3 SOP2 Generation from Norm to code and configuration files

---

### 3.1 Pre-requisit

Have your profile myprofile.profile.uml, in our case SysML14.profile.uml

Have a Papyrus instance with UML2Extension and Developer tools installed

### 3.2 Step 1: generation of model,edit plugins code

#### 3.2.1 1.0: Purpose

Generate the "static profile" (the java code related to the UML profile)

#### 3.2.2 1.1: Create the EMF generator Model

right-click on the profile, select new EMF Generator Model

Options

- load
- skip warning
- next
- select SysML

#### 3.2.3 1.2: Generate the code

Modify the options

- switch the default output directory from src to src-gen
- for the EditPluginClass remove intermediate package name, to reflect `.provider.SysmlEditPlugin`
- switch NLS on

Right click generate Model and Edit code

#### 3.2.4 1.x: Validation

Code should compile at this step

### 3.3 Step 2: generation of ElementType configuration files

#### 3.3.1 2.0: Purpose

Generate semantic and diagramatics elementtypes related to the profile

#### 3.3.2 2.1: Semantic Elementtypeconfiguration

- Open the profil with Papyrus
- Select generate Tooling Model Elementtype
- base: `uml o.e.p.uml.servicetype`
- identifier `o.e.p.sysML14`
- file name: `SysML.elemeznntypeconfiguration`

Move it to the right plugin place

### 3.3.3 2.1: UI Elementtypeconfiguration

- Open the profil with Papyrus
- Select generate Tooling Model> Elementtype
- base: uml o.e.p.uml.uml.diagramclass and composite
- identifier o.e.p.sysML14
- file name: SysML.elemeznntypeconfiguration

Move it to the common plugin place and do it foreach root diagrams

### 3.3.4 2.x: Validation

to be completed...

## 3.4 Step 3: generation of New Child Menu

### 3.4.1 3.0: Purpose

Generate the new child menu present in the model explorer

### 3.4.2 3.1: Action

to be completed...

### 3.4.3 3.x: Validation

to be completed...

## 3.5 Step 4: generation of Property view

### 3.5.1 4.0: Purpose

Generate the default property views (one context .ctx file and multiple xwt files)

### 3.5.2 4.1: Action

- Select New...> Other
- Select Papyrus Category
- Select Property view Configuration
- Choose create from UML Profile
- Source: /org.eclipse.papyrus.sysml14/resources/profile/SysML.profile.uml
- Choose Standard Layout Generator
- Target: /org.eclipse.papyrus.sysml14.ui/resources/properties/SysML1.4.ctx

### 3.5.3 4.x: Validation

Should generate one \*.ctx file and many \*.xwt files

## **3.6 Step 5: generation of Assistant**

### **3.6.1 5.0: Purpose**

Generate the tooling to have assistants in customized diagrams

### **3.6.2 5.1: Action**

to be completed...

### **3.6.3 5.x: Validation**

to be completed...

## 4 SOP3 Adding operations to the norm

---

### 4.1 Contexte

Sometimes the sysml profile norm do not reflect the text/pdf norm document. It is pertinent to modify the sysml profile to add new operations.

### 4.2 SOP

#### 4.2.1 Documentation

Reference your new operations here, in order to be able to process the model automatically in a near future.: /org.eclipse.papyrus.sysml14/doc/SysML-Extension.di

Open it with UML Editor to access UML type and cardinality

Add your requirements here by creating a new derive requirement with Papyrus Req plugin: /org.eclipse.papyrus.sysml14/doc/org.eclipse.papyrus.sysml14.di

You should export the org.eclipse.papyrus.sysml14 plugin to have access to the pathmap for the different UML models, such as:

- OMG norm: resources/doc/omg.sysml.uml
- MDE model: resources/doc/org.eclipse.papyrus.sysml14.uml
- SysML 1.4 extension model: resources/doc/SysML-Extension.uml

#### 4.2.2 SysML profile: semantic upgrade

Modify /org.eclipse.papyrus.sysml14/resources/profile/SysML.profile.uml to add a new operation, with its parameters.

Open the genmodel, right click on ti and click on Reload the UML profile and regenerate the Model and Edit code.

Validate your enhancement by looking in /org.eclipse.papyrus.sysml14/src-gen

Implement the function in the /org.eclipse.papyrus.sysml14/src; do not forget to reference the requirement

You have to provide eventually a new XFactoryCustomImpl that has to be referenced in the plugin.xml

#### 4.2.3 SysML diagram upgrade

Modify the Expansion diagram of your /org.eclipse.papyrus.sysml14.diagram.blockdefinition/resources/configuration/blockDefinitionDiagramConfig.expansionmodel

Add new representation, a new induced representation and do not forget to add it to the context.

Be careful to replace with text editor the kind href=" ../../../../plugin/org.eclipse.papyrus.uml.diagram.common/" with kind href="platform:/plugin/

#### 4.2.4 SysML element type

/org.eclipse.papyrus.sysml14.diagram.common/resources/SysML14ClassDiagram-extension.elementtypesconfigurations

Add a new Specialization type configuration and eventually its edit helper

#### 4.2.5 Palette

Add your element in the palette: `/org.eclipse.papyrus.sysml14.diagram.blockdefinition/resources/palette/blockDefinitionDiagram.paletteconfiguration`

#### 4.2.6 CSS

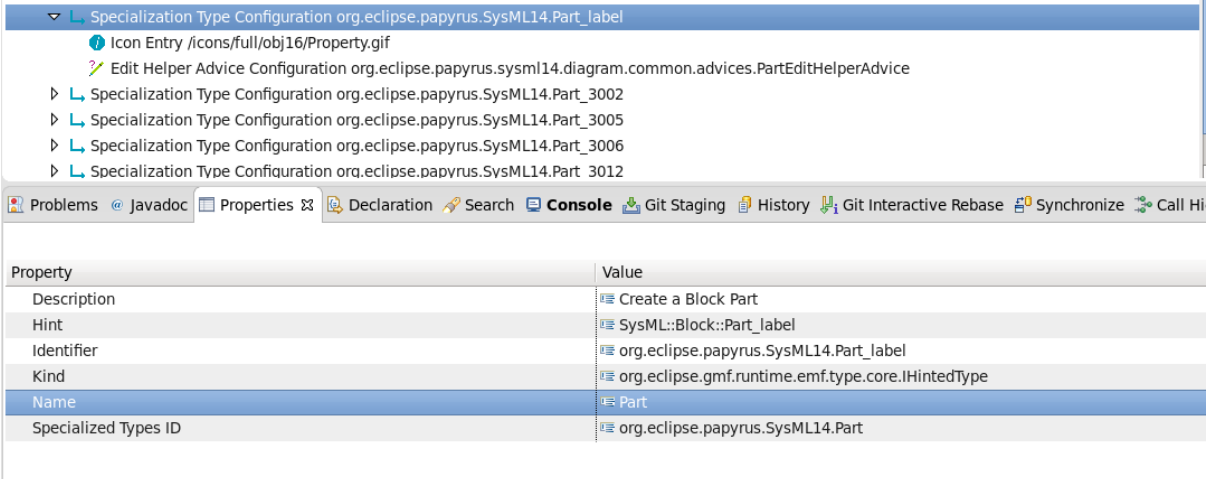
Modify the CSS to display your compartment: `/org.eclipse.papyrus.sysml14.diagram.blockdefinition/resources/style/blockDefinitionDiagram.css`

#### 4.2.7 Test

Implement your test in `/org.eclipse.papyrus.sysml14.tests/src/org/eclipse/papyrus/sysml14/tests/blocks/BlockTest.java`

Open a runtime, you should be able to do the move from the palette to your compartment

#### 4.2.8 todo



The screenshot shows the Eclipse IDE interface. The top part displays a tree view of the project structure, with the following items expanded:

- Specialization Type Configuration org.eclipse.papyrus.SysML14.Part\_label
  - Icon Entry /icons/full/obj16/Property.gif
  - Edit Helper Advice Configuration org.eclipse.papyrus.sysml14.diagram.common.advice.PartEditHelperAdvice
  - Specialization Type Configuration org.eclipse.papyrus.SysML14.Part\_3002
  - Specialization Type Configuration org.eclipse.papyrus.SysML14.Part\_3005
  - Specialization Type Configuration org.eclipse.papyrus.SysML14.Part\_3006
  - Specialization Type Configuration org.eclipse.papyrus.SysML14.Part\_3012

The bottom part of the screenshot shows the Properties view for the selected element. The view is divided into two columns: Property and Value.

Property	Value
Description	Create a Block Part
Hint	SysML::Block::Part_label
Identifier	org.eclipse.papyrus.SysML14.Part_label
Kind	org.eclipse.gmf.runtime.emf.type.core.IHintedType
Name	Part
Specialized Types ID	org.eclipse.papyrus.SysML14.Part

## 5 SOP4 Requirement

### 5.1 Context

In order to develop the SysML 1.4 application, the developer team has chosen to use a model based approach. At least, it is used to gather all requirements in different SysML 1.1 models.

A first model of requirement has been made by extracting all basic requirements from the Norm. Each section of the norm described in the table of content has been extracted as a requirement.

This model is located here `core/org.eclipse.papyrus.sysml14/doc/omg.sysml.uml`

New requirements, through the use of papyrus Req plugin, are created in each SysML 1.1 model located in the doc folder of each module. They should use the "DeriveReq" link.

Finally you could reference your requirement anywhere in your code, document by using the following convention:

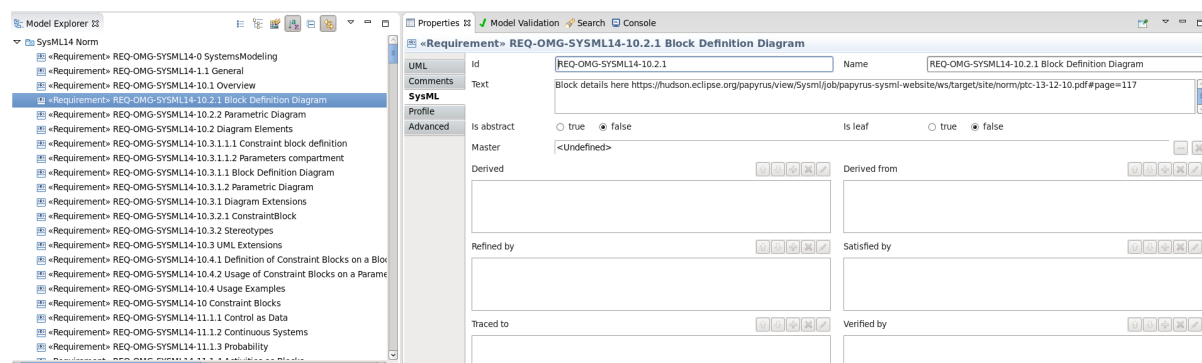
```
@pap.req ${Model.ID}#{Req.ID}
```

The goal is to have a tool, plugin that will be able to parse this snippet of code to link it to the model, and eventually open the model.

### 5.2 Example

#### 5.2.1 Zoom on OMG SYSML Requirement Model

Pay attention to the ID, NAME and the TEXT, that have been normalized.



#### 5.2.2 Zoom on a Derivereq Link

Pay attention to the ID, it has to be uniq, it is generated by the Papyrus Req plugin: /  
org.eclipse.papyrus/extraplugins/req/org.eclipse.papyrus.reqreqif

Papyrus Req plugin is an extra plugin of Papyrus project, you have to download it and install it in your own Eclipse installation. Then when opening a requirement diagram, you will have a new right-click menu that allows you to create directly new derived requirement from an initial requirement. It will compute automatically the identifier and pre-fill the text.

The screenshot displays the Eclipse IDE interface with several panels:

- Project Explorer:** Shows a project structure for 'org.eclipse.papyrus.sysml14' with folders like 'diagram', 'resources', and 'doc'. The 'diagram.blockdefinition' folder is selected.
- Diagram View:** Displays a SysML diagram with two requirement elements: 'REQ-OMG-SysML14-8.2.1 Block Definition Diagram' and 'Requirement Block display'. A dashed arrow labeled '<DeriveReq>' points from the lower requirement to the upper one.
- Model Explorer:** Shows a tree view of the model elements, including 'Requirements' and 'Requirement' elements.
- Properties View:** Shows the properties of the selected 'Requirement' element. The 'UML' tab is active, showing fields for 'id' (Req001), 'Name' (Block display), and 'Comments' (The block can display constraints, operations, parts, references, values, properties compartments). The 'SysML' tab is also visible, showing 'Is abstract' (false), 'Is leaf' (false), and 'Derived from' (REQ\_001).
- Palette:** Shows a list of SysML stereotypes and relationships, including 'Requirement', 'DeriveReq', and 'InformationFlow link'.

## 6 SOP5 Release

---

### 6.1 sop5-Release

#### 6.1.1 Who is in charge of?

Committer + a basic developer in order to transfer knowledge and improve the process.

#### 6.1.2 When?

when you need it, on a regular basis, every two month.

#### 6.1.3 How long?

It last at least 120mn

### 6.2 Prerequisite

#### 6.2.1 Access

Have the Hudson access

#### 6.2.2 General release information

You will find the general information on How to make a release for a Papyrus specialization [here](#)

### 6.3 Pre-Actions

You can send an email to the developers mailing list.

### 6.4 Steps

#### 6.4.1 Create a bugzilla ticket.

#### 6.4.2 No critical bugs open for the release you are targeting

#### 6.4.3 Jobs (master,website,quality,deploy) are green

#### 6.4.4 Initial Version should match qualifier and -SNAPSHOT

You must fill the changelog to describe the new version feature.

```
/org.eclipse.papyrus-sysml/src/changes/changes.xml
```

Details <https://maven.apache.org/plugins/maven-changes-plugin/changes.html>

#### 6.4.5 Upgrade the version of application

use tycho-versions plugin to switch from qualifier to release, and then back from release to qualifier



```
mvn org.eclipse.tycho:tycho-versions-plugin:set-version -DnewVersion=X.Y.Z-SNAPSHOT
```

check the different pom.xml and MANIFEST.MF

You have to manually update the category.xml at /org.eclipse.papyrus-sysml/releng/  
org.eclipse.papyrus.sysml14.p2/category.xml

Update also the pom

```
mvn org.eclipse.tycho:tycho-versions-plugin:update-pom -Dtycho.localArtifacts=ignore
```

Upgrade also the target platform version in the main pom

#### 6.4.6 Push on Gerrit the different modifications

Who: non committer action

git or others.

#### 6.4.7 Accept the changes

Who: committer action

Gerrit web action.

#### 6.4.8 Rerun the job Master and Website

Who: any

Goal is to use this job version as data for the promotion.

#### 6.4.9 Tag the release

Who: committer action

how

#### 6.4.10 Deploy the product (update site and the rcp)

Who: non-committer action

how: job on Hudson, fill the fields

#### 6.4.11 Deploy the web site

Who: non-committer action + committer review

Generate the new web site

```
mvn install site site:stage-deploy -Pproduct,documentation,documentation-pdf,quality
```

Copy/paste it in the git repository dedicated to the web, in its own version folder

Update the root index.html

```
https://git.eclipse.org/c/www.eclipse.org/papyrus-sysml.git/
```

Ask for validation by adding Remi SCHNEKENBURGER as reviewer.

### 6.4.12 Upgrade to the next snapshot version of the application

use tycho-versions plugin

```
mvn org.eclipse.tycho:tycho-versions-plugin:set-version -DnewVersion=X.Y.Z-SNAPSHOT
```

You have to manually update the category.xml at /org.eclipse.papyrus-sysml/releng/  
org.eclipse.papyrus.sysml14.p2/category.xml

check the different pom.xml and MANIFEST.MF

## 6.5 Web-site release

You need also to deploy the new web site matching your release product version.

By going to <https://hudson.eclipse.org/papyrus/view/Sysml/job/papyrus-sysml-eclipse-web-deploy/>,  
you could retrieve the zip of the site under the workspace ws/site-staging/3DIGITVERSION

Then you have to submit a gerrit patch on the Papyrus git repository, located here: [www.eclipse.org/papyrus.git](http://www.eclipse.org/papyrus.git) modeling.mdt.papyrus project website

You have also to modify the main index.html file to update the current website version and to add your version in the menu.

The final result should be available after a few minutes here <https://www.eclipse.org/papyrus/components/sysml/3DIGITVERSION/>

## 6.6 Post-Actions

You can send an email to the developers mailing list

## 7 FAQ Developer

---

### 7.1 Frequently Asked Questions

#### General

1. [Where to retrieve the code?](#)
2. [Where is the website of the project?](#)
3. [Where is the javadoc of the project?](#)
4. [Where are the continuous integration jobs?](#)
5. [Where is the review server?](#)

#### Installation

1. [How do I install SysML?](#)

### 7.2 General

#### Where to retrieve the code?

You have to clone the git repository

```
git clone https://git.eclipse.org/r/papyrus/org.eclipse.papyrus-sysml
```

---

#### Where is the website of the project?

You have to go <https://hudson.eclipse.org/papyrus/view/Sysml/job/papyrus-sysml-website/ws/site-staging/index.html>

---

#### Where is the javadoc of the project?

You have to go <https://hudson.eclipse.org/papyrus/view/Sysml/job/papyrus-sysml-website/ws/site-staging/apidocs/index.html>

---

#### Where are the continuous integration jobs?

You have to go <https://hudson.eclipse.org/papyrus/view/Sysml/>

---

#### Where is the review server?

You have to go to gerrit: <https://git.eclipse.org/r/#/q/status:open+project:papyrus/org.eclipse.papyrus-sysml>

### 7.3 Installation

#### How do I install SysML?

SysML 1.4 is an eclipse project and follow the standard recommendation of the Eclipse Foundation for installation.