

TEXTUAL DSL FOR ARCHITECTURE DESCRIPTION

Author: Benoît Langlois – benoit.langlois@thalesgroup.com

Kitalpha version: 1.0.0

Document version: 1.0.0

1. INTRODUCTION

Kitalpha is a modeling component which implements the ISO/IEC 42010 standard for system description in system and software engineering. It provides both a development and runtime environment to create and execute rich MBE workbenches.

The development environment provides a DSL to facilitate the description of architecture framework and viewpoint. The purpose of this document is to provide the textual grammar of this DSL. Each DSL is described with the following structure:

- the grammar to textually edit and describe an architecture framework and a viewpoint,
- templates,
- helpers,
- commands

Acronyms

AF	Architecture Framework
DSL	Domain-Specific Language
MBE	Model-Based Environment
UI	User Interface

Table of Contents

1. Introduction.....	1
Acronyms.....	1
2. Textual Architecture Framework DSL	3
Model / text equivalence and synchronization	4
Description of the textual grammar.....	5
Architecture Framework description	5
3. Textual Viewpoint DSL.....	7

Overview	7
Executable viewpoint artifacts from DSL-based viewpoint description	7
Model / text equivalence and synchronization	8
Description of the textual grammar.....	10
Common expressions	10
Viewpoint description	11
Data description	13
UI description	17
Diagram description	19
Activity Explorer	36
Services description.....	38
Continuous integration description	40
Configuration description	43
Templates.....	45
Data templates	45
UI templates	46
Diagram templates	48
Activity Explorer templates	58
Rules templates.....	64
Services templates	64
Configuration templates	66
Helpers	67
For any editor	67
Viewpoint data description – [viewpoint name].data.vptext editor	67
Commands.....	68
For any editor	68
Main viewpoint editor - [viewpoint name].spec.vptext editor.....	68
Sample Activity explorer preview – [viewpoint name].activityexplorer.vptext editor	69

2. TEXTUAL ARCHITECTURE FRAMEWORK DSL

The Architecture Framework DSL enables to edit the architecture framework properties and to define the aggregation of viewpoints. It offers the ability to generate and package architecture framework artefacts for their deployment.



Figure 1 - Architecture Framework Development process

At the DSL level, there exist:

- [Abstract syntax] A model which is an architecture framework description independent of any representation,
- [Concrete syntax] A set of text files, each with its own grammar for description of an architecture framework aspect.

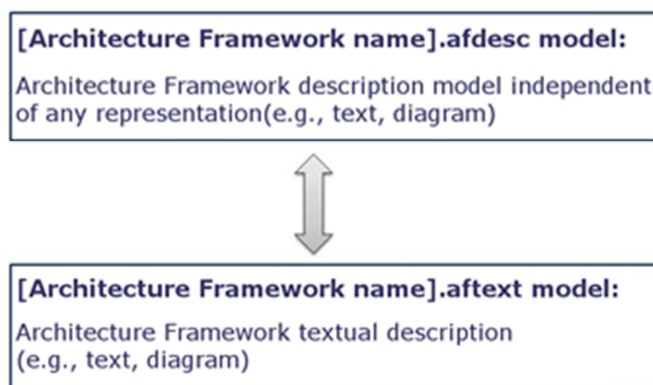


Figure 2 - Model and text equivalence

Model / text equivalence and synchronization

It is possible to create text files from a model.

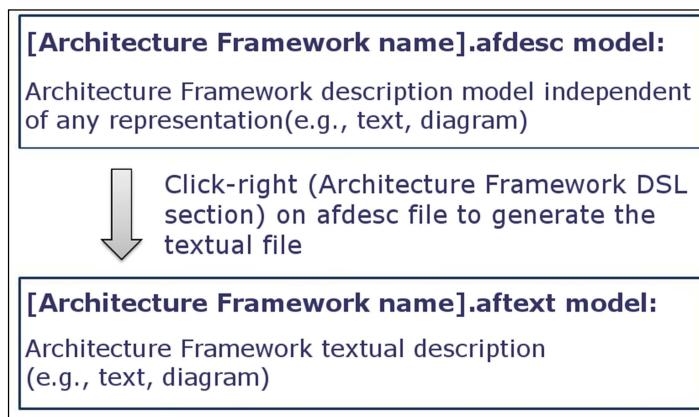


Figure 3 - Creation of text file from a model

In reverse,

- 1) when the architecture framework DSL with a textual editor is saved, the model is updated,
- 2) On a text file, it is possible to update the model.

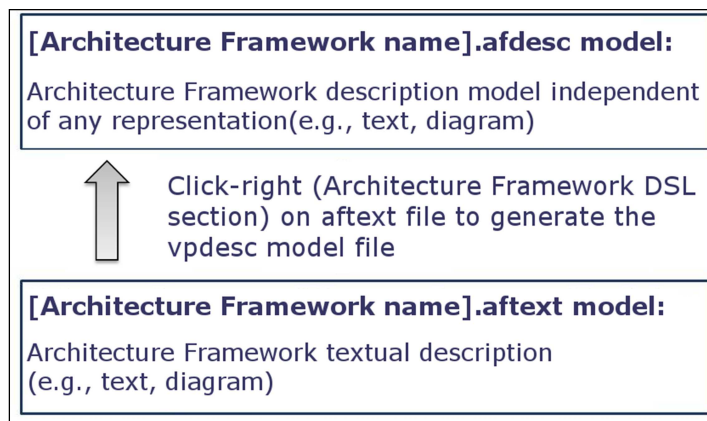


Figure 4 - Update of model from a text file

Description of the textual grammar

The following sections detail the grammar to describe architecture frameworks.

ARCHITECTURE FRAMEWORK DESCRIPTION

Unlike a viewpoint description divided into several grammars (aspects), an Architecture Framework description is reduced to one grammar.

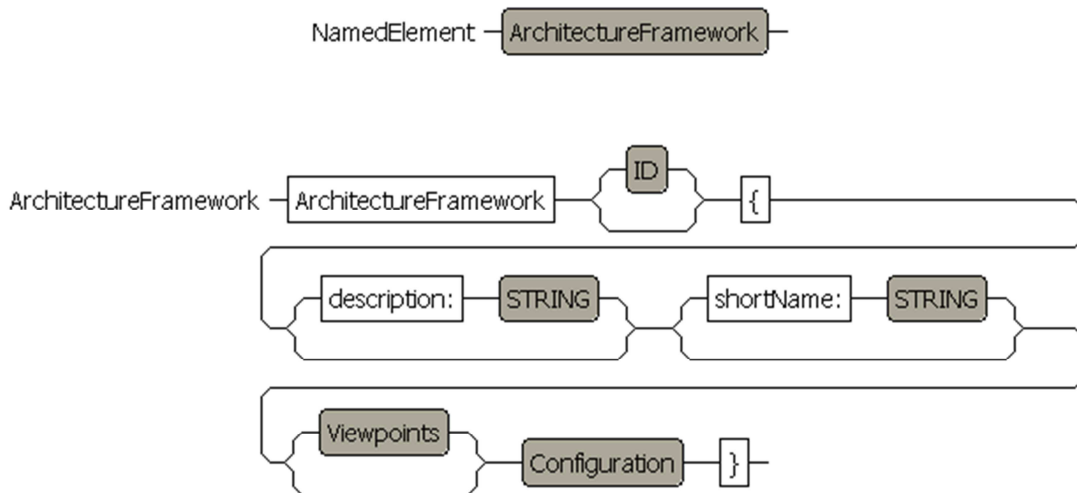
File name extension: aftext

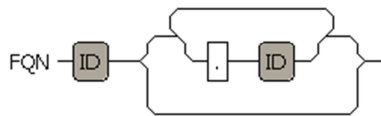
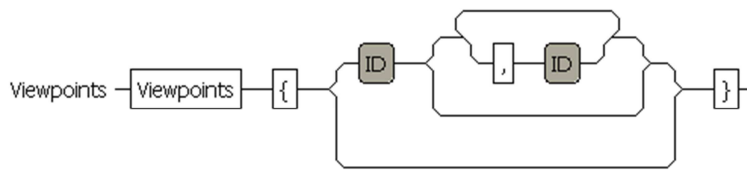
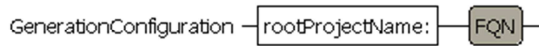
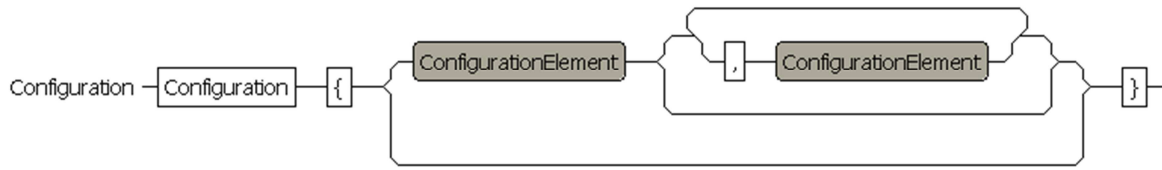
Example

```
ArchitectureFramework ComponentSampleFramework {
  shortName: "ComponentSampleFramework"
  Viewpoints {
    ComponentSample, ComponentSamplePerformance, ComponentSampleQualityAssessment,
    ComponentSampleSafety, ComponentSampleSafetyPattern
  }
  Configuration {
    rootProjectName: org.polarsys.kitalpha.af.componentsampleframework
  }
}
```

Figure 5 - Example of aftext file

Grammar





3. TEXTUAL VIEWPOINT DSL

Overview

EXECUTABLE VIEWPOINT ARTIFACTS FROM DSL-BASED VIEWPOINT DESCRIPTION

A viewpoint DSL has been developed in order to facilitate the definition of viewpoint. From this DSL, executable viewpoint artefacts are generated for deployment in a MBE workbench such as Capella.

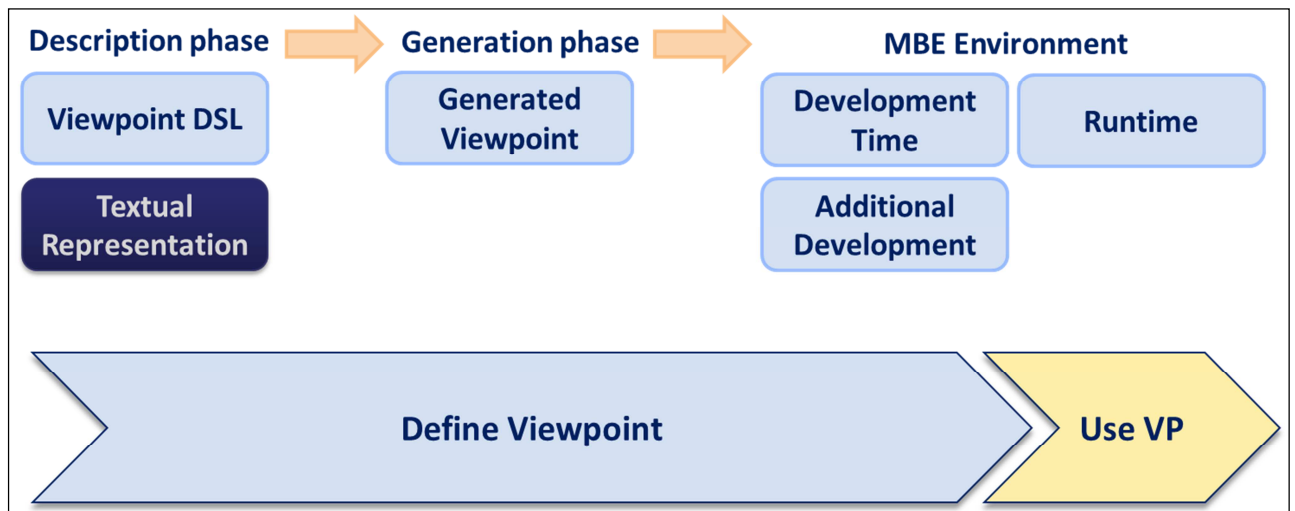


Figure 6 - Position of the viewpoint DSL in the viewpoint development chain

The artifacts produced from this viewpoint DSL are for instance:

- Artifacts produced from the data aspect:
 - 1) The EMF viewpoint metamodel, materialized by an EMF ecore file,
 - 2) The EMF API produced from an EMF Generation.
- Artifacts produced from the user interface aspect: in context of Capella, the user interfaces to manipulate Capella model properties.
- Artifacts produced from the diagram aspect: the odesign files which describe the Sirius diagrams.
- Artifacts produced from the build aspect:
 - 1) an EGF build model,
 - 2) The continuous integrations scripts produced from the build model.

MODEL / TEXT EQUIVALENCE AND SYNCHRONIZATION

At the DSL level, there exists:

- [Abstract syntax] A model which is a viewpoint description independent of any representation,
- [Concrete syntax] A set of text files, each with their own grammar for the description of a viewpoint aspect, such data or diagram.

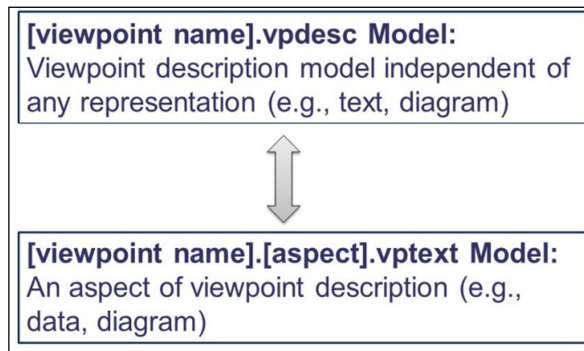


Figure 7 - Model and text equivalence

It is possible to create text files from a model.

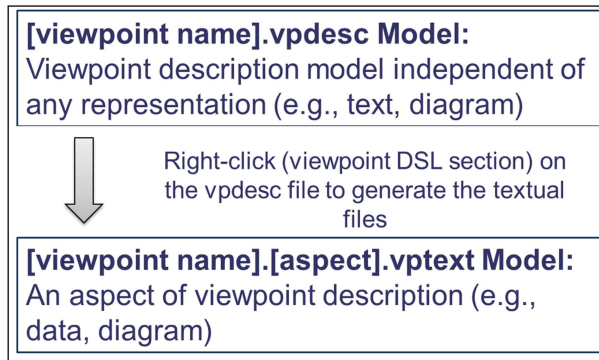


Figure 8 - Creation of text file from a model

In reverse,

- 1) when an aspect of the viewpoint DSL with a textual editor is saved, the model is updated,
- 2) On a text file, it is possible to update the model.

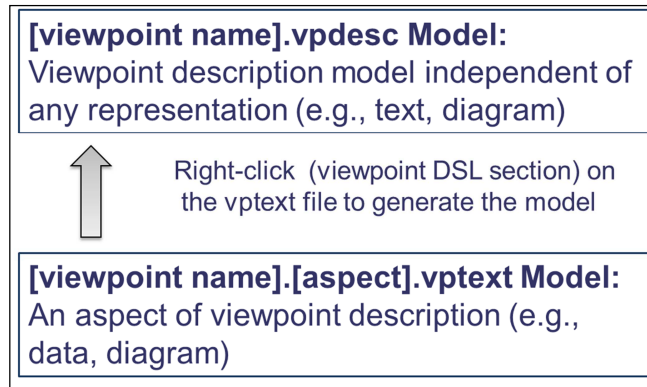


Figure 9 - Update of model from a text file

Type of editors

For reasons of decoupling and scalability, a textual edition of viewpoint is split by aspects, as depicted in the following picture.

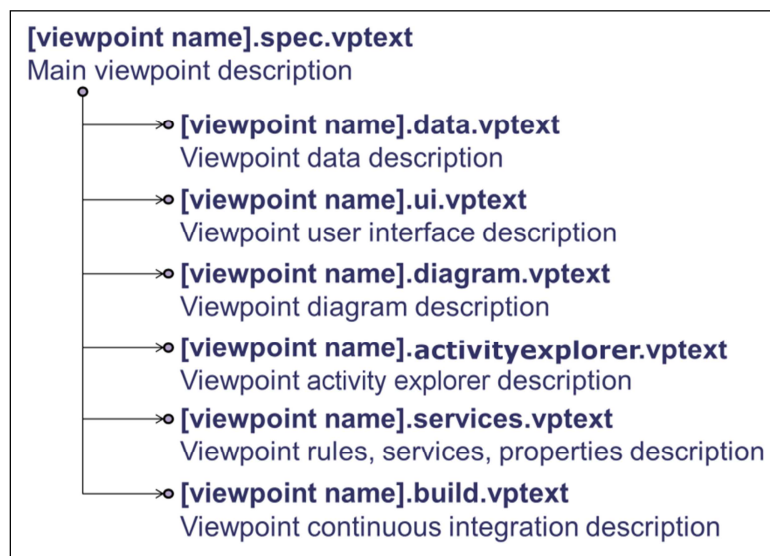


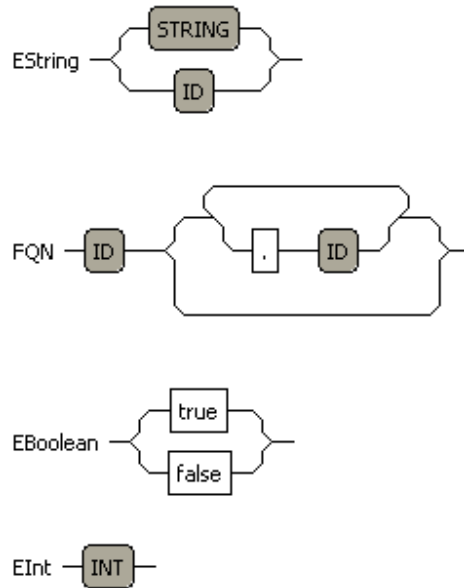
Figure 10 - Viewpoint aspects

Description of the textual grammar

The following sections detail the grammar to describe viewpoints.

COMMON EXPRESSIONS

This grammar contains the set of terminals, typically basic types: Boolean, String, FQN, Integer.



VIEWPOINT DESCRIPTION

Purpose: entry point for viewpoint description

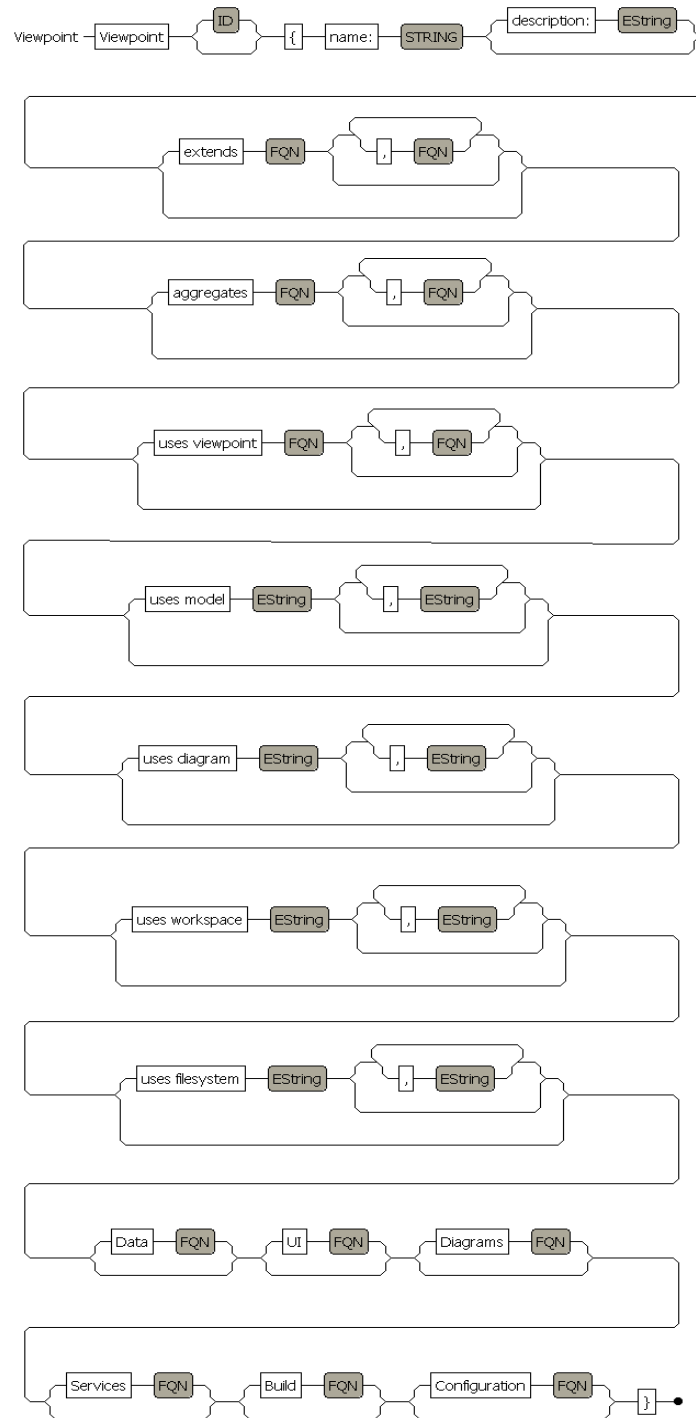
Extension file name: spec.vptext

Example

```
Viewpoint ComponentSampleQualityAssessment {
  name: "ComponentSampleQualityAssessment"
  uses viewpoint: ComponentSample
  uses model "platform:/plugin/org.eclipse.emf.ecore/model/Ecore.ecore"
  usesdiagram
  "platform:/plugin/org.eclipse.emf.ecoretools.design/description/ecore.odesign"
  uses workspace "/org.polarsys.kitalpha.vp.componentsample.af/icons"
  uses filesystem "/home/bob/extendedEcore.ecore"
  Data ComponentSampleQualityAssessment.data
  Diagrams ComponentSampleQualityAssessment.diagram
  Activity-Explorer ComponentSampleQualityAssessment.activityexplorer.vptext
  Configuration ComponentSampleQualityAssessment.conf
}
```

Figure 11 - Example of spec.vptext file

Grammar



DATA DESCRIPTION

Purpose: viewpoint data description

Extension file name: data.vptext

Example

```

Import external
"platform:/resource/org.polarsys.kitalpha.vp.componentsampe/model/componentSample.ecore"

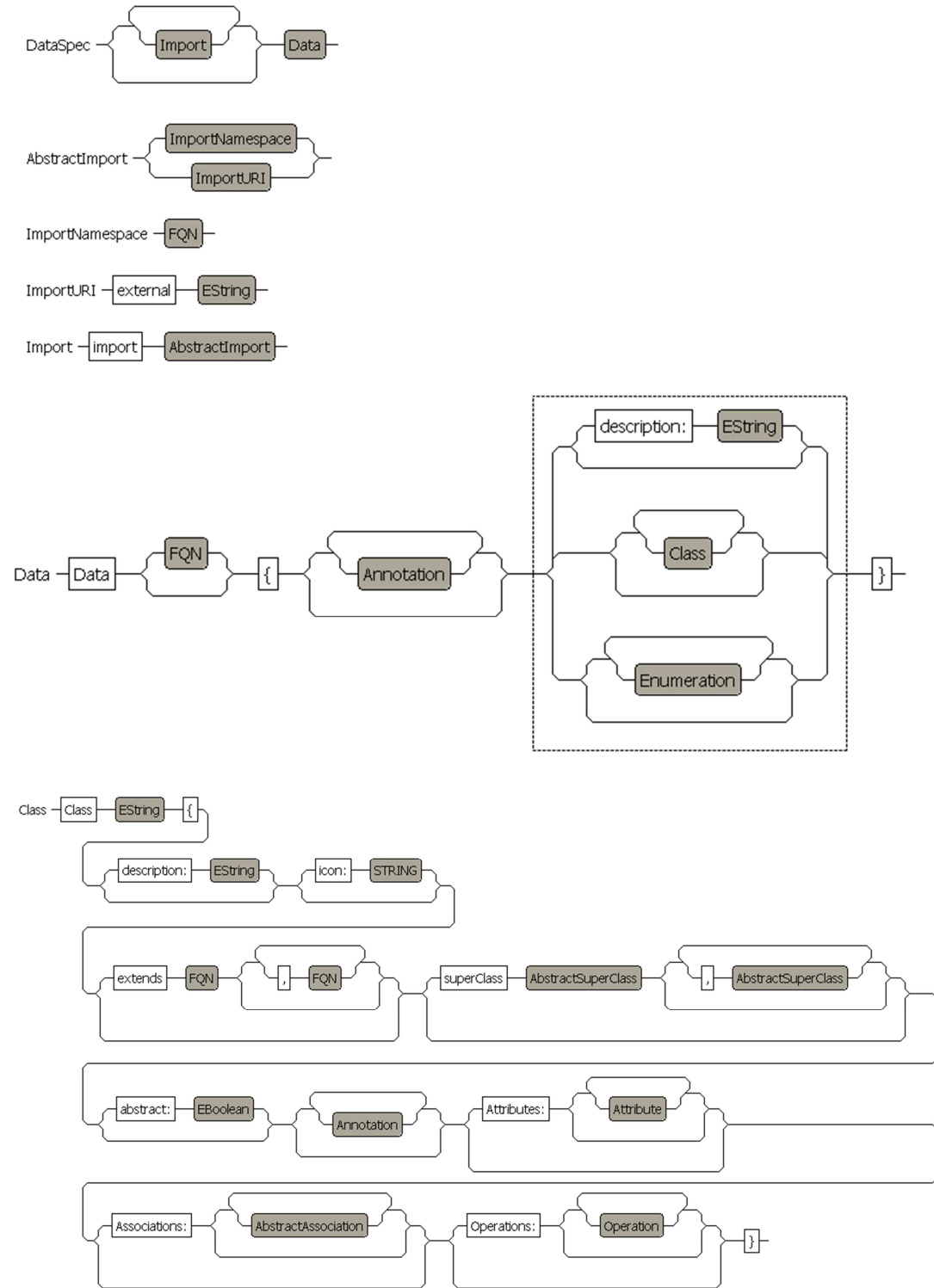
Data ComponentSampleQualityAssessment.data {
    Class QualityAssessment {
        description: "Quality Assessment"
        icon: "QualityAssessment.png"
        extends ComponentSample.AbstractComponent
        superClass external ComponentSample.ComponentElement
        Attributes:
            maturityLevel type.ecore.EString
            confidenceLevel type.ecore.EEnumerator values ( "Not Assessed" , Low ,
Medium , High )
            assessed type.ecore.EBoolean
        Associations:
            basedOn refers [0,*] QualityAssessment
            context refers [0,*] external ComponentSample.ComponentElement
            measures contains [0,*] QualityMeasure
    }

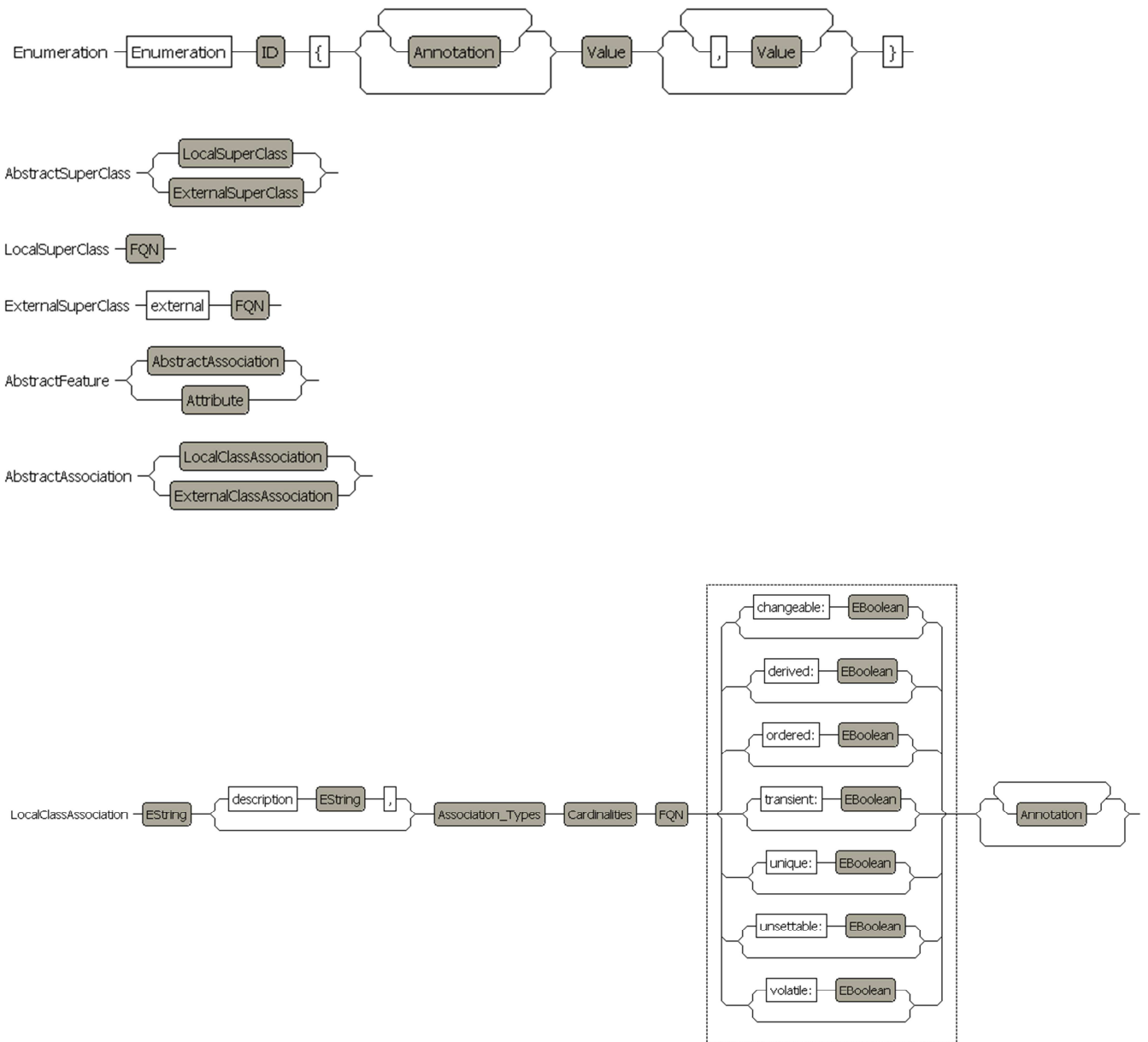
    Class QualityMeasure {
        icon: "QualityMeasure.png"
        superClass external ComponentSample.ComponentElement
        Attributes:
            criterion type.ecore.EString
            measureValue type.ecore.EInt
    }
}

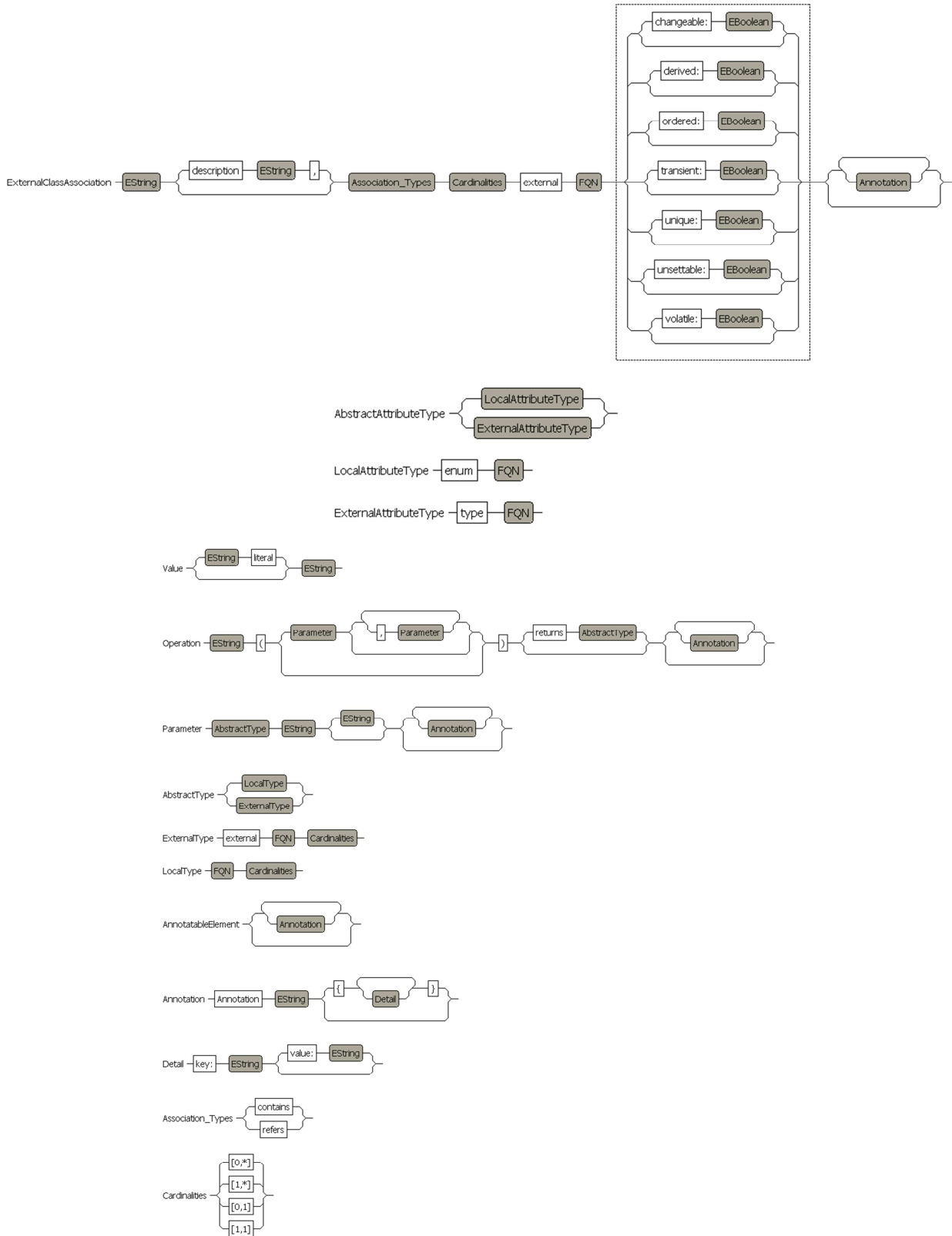
```

Figure 12 - Example of data.vptext file

Grammar







UI DESCRIPTION

Purpose: viewpoint user interface description

Extension file name: ui.vptext

Example

```

UIDescription ComponentSampleQualityAssessment.ui {
  UI ComponentSampleQualityAssessment_QualityAssessment {
    label: "Component Sample Quality Assessment"
    Container ComponentSampleQualityAssessment_QualityAssessment_Section {
      Container ComponentSampleQualityAssessment_QualityAssessment_AttributeGroup {
        label: "Quality Assessment Attributes"
        Field maturityLevelField label: "Maturity Level" type text , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.maturityLevel
        Field confidenceLevelField label: "Confidence Level" type radiobox , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.confidenceLevel
        Field assessedField label: "Assessed" type checkbox , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.assessed
      }
      Container ComponentSampleQualityAssessment_QualityAssessment_AssociationGroup {
        label: "Quality Assessment Associations"
        Field basedOnAssociation label: "Based On" type multipleChoiceList , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.basedOn
        Field contextAssociation label: "Context" type multipleChoiceList , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.context
        Field measuresAssociation label: "Measures" type multipleChoiceList , mapped-to
ComponentSampleQualityAssessment.data.QualityAssessment.measures
      }
    }
  }
  UI ComponentSampleQualityAssessment_QualityMeasure {
    label: "Component Sample Quality Assessment"
    Container ComponentSampleQualityAssessment_QualityMeasure_Section {
      Container ComponentSampleQualityAssessment_QualityMeasure_AttributeGroup {
        label: "Quality Measure Attributes"
        Field criterionField label: "Criterion" type text , mapped-to
ComponentSampleQualityAssessment.data.QualityMeasure.criterion
        Field measureValueField label: "Measure Value" type text , mapped-to
ComponentSampleQualityAssessment.data.QualityMeasure.measureValue
      }
    }
  }
}

```

Figure 13 - Example of ui.vptext file

Grammar

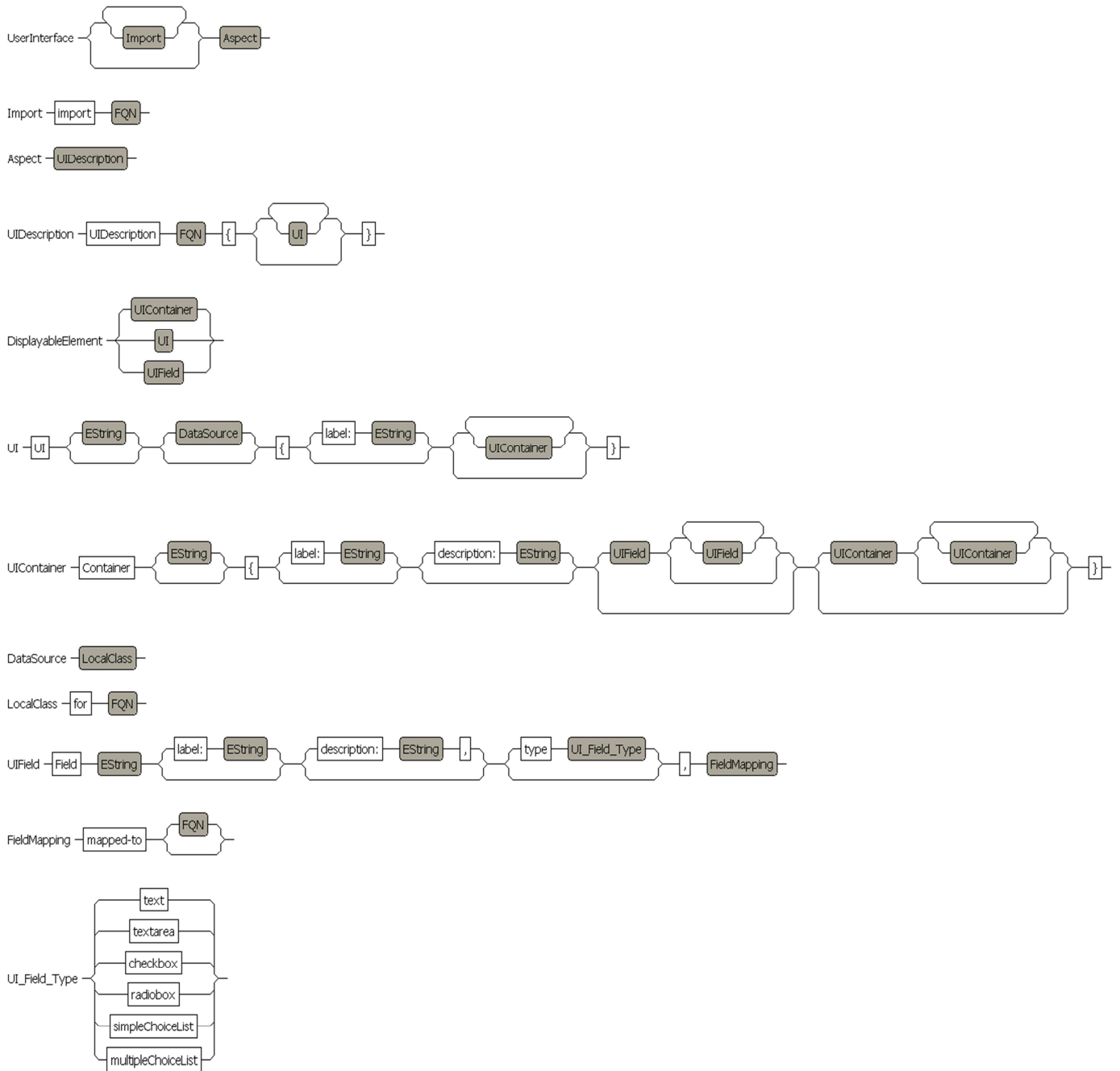


DIAGRAM DESCRIPTION

Purpose: viewpoint diagram description

Extension file name: diagram.vptext

Example

```

import external
"platform:/resource/org.polarsys.kitalpha.vp.componentsample.design/description/ComponentSample.odesign"

Diagrams ComponentSampleQualityAssessment.diagram {
  DiagramExtension "ComponentSampleQualityAssessmentDiagram" {
    extended-diagram: ComponentSampleDiagram
    Mapping {
      Container QualityAssessment_ComponentSample {
        import: HardwareComponentContainer_CM Contains {
          Node QualityAssessmentNode {
            domain-context: ComponentSampleQualityAssessment.data.QualityAssessment provided-by
            association external emde.ExtensibleElement.ownedExtensions
            Representation {
              Label { content: "Quality: " +
ComponentSampleQualityAssessment.data.QualityAssessment.confidenceLevel police: black alignment: center }
              Style {BasicStyle { border-color: black background: chocolate form: Square } }
            }
          }
          Contains {
            BorderedNode QualityMeasureBorderedNode {
              domain-context: ComponentSampleQualityAssessment.data.QualityMeasure provided-by
              association ComponentSampleQualityAssessment.data.QualityAssessment.measures
              Representation {
                Label { content:
ComponentSampleQualityAssessment.data.QualityMeasure.measureValue position: node alignment: center }
                Style {BasicStyle { background: light_chocolate form: Square } }
              }
            }
          }
        }
      }
    }
  }
  Actions {
    // QualityAssessmentNode actions
    Create QualityAssessmentNodeCreate { label: "Quality Assessment" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode}
    Drop QualityAssessmentNodeDrop { label: "Drop Quality Assessment" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode}
    Delete QualityAssessmentNodeDelete { label: "Delete Quality Assessment" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode}
    // QualityMeasureBorderedNode actions
    Create QualityMeasureBorderedNodeCreate { label: "Quality Measure" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode.QualityMeasureBorderedNode}
    Delete QualityMeasureBorderedNodeDelete { label: "Delete Quality Measure" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode.QualityMeasureBorderedNode}
    Drop QualityMeasureBorderedNodeDrop { label: "Drop Quality Measure" action-for:
QualityAssessment_ComponentSample.QualityAssessmentNode.QualityMeasureBorderedNode}
  }
}

```

Figure 14 - Example of diagram.vptext file

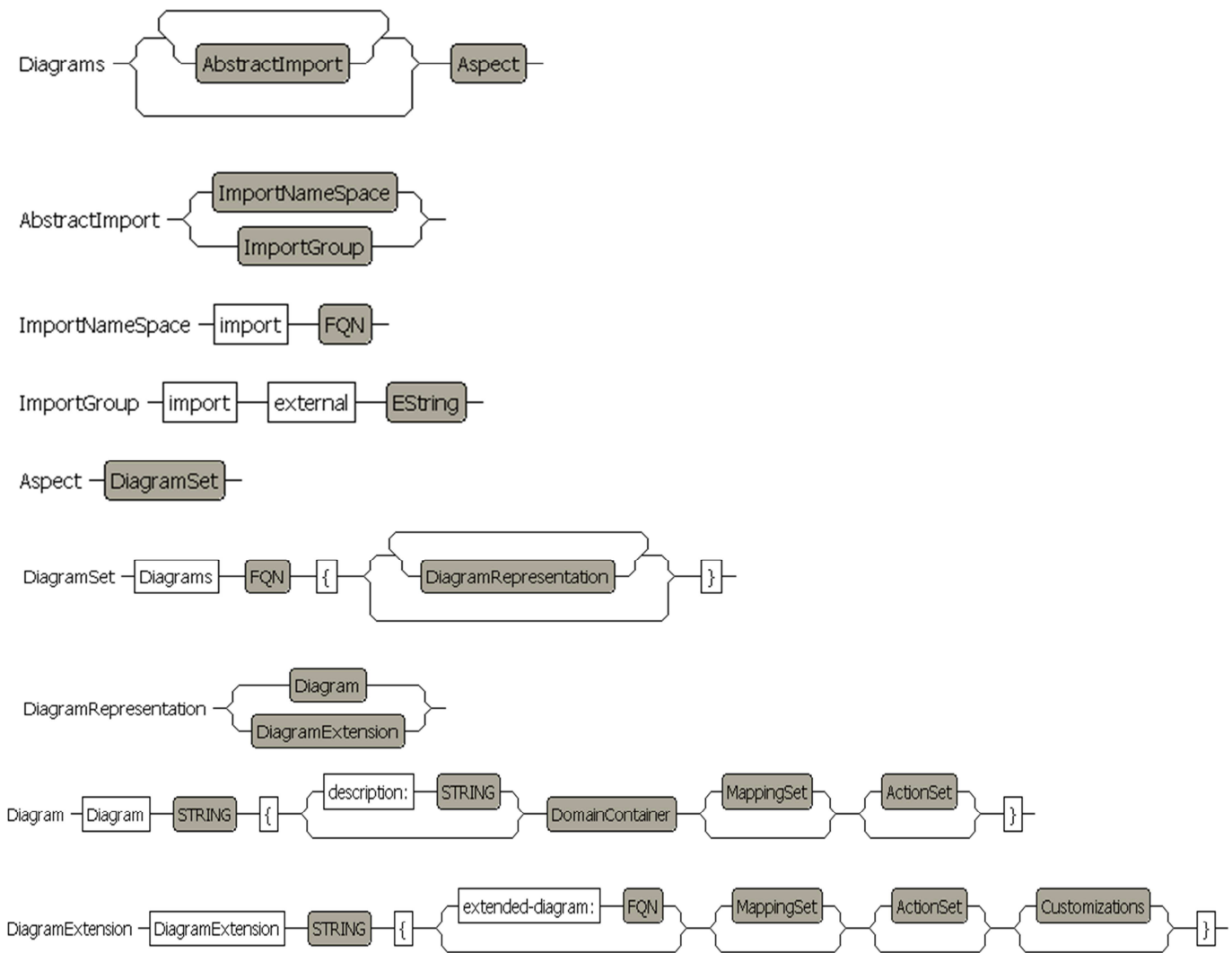
Grammar

Diagram Description structure

A general structure of diagram description is:

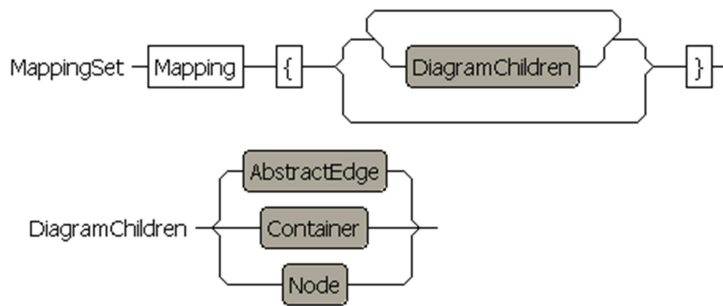
- A set of imports (eventually empty)
 - Import data
 - Import descriptions
- A set of diagrams description (eventually empty)
 - Description of new diagrams
 - Description of new diagram extension

The syntax above illustrates the global structure of a diagram description.

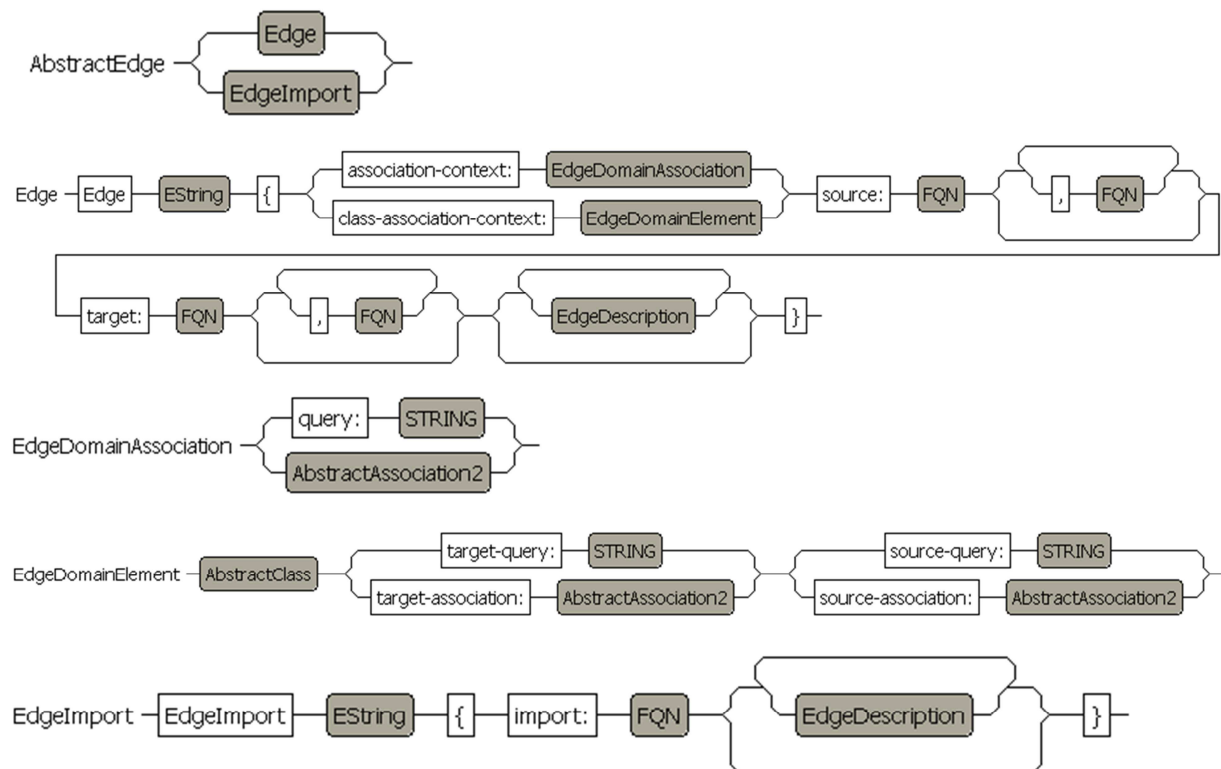


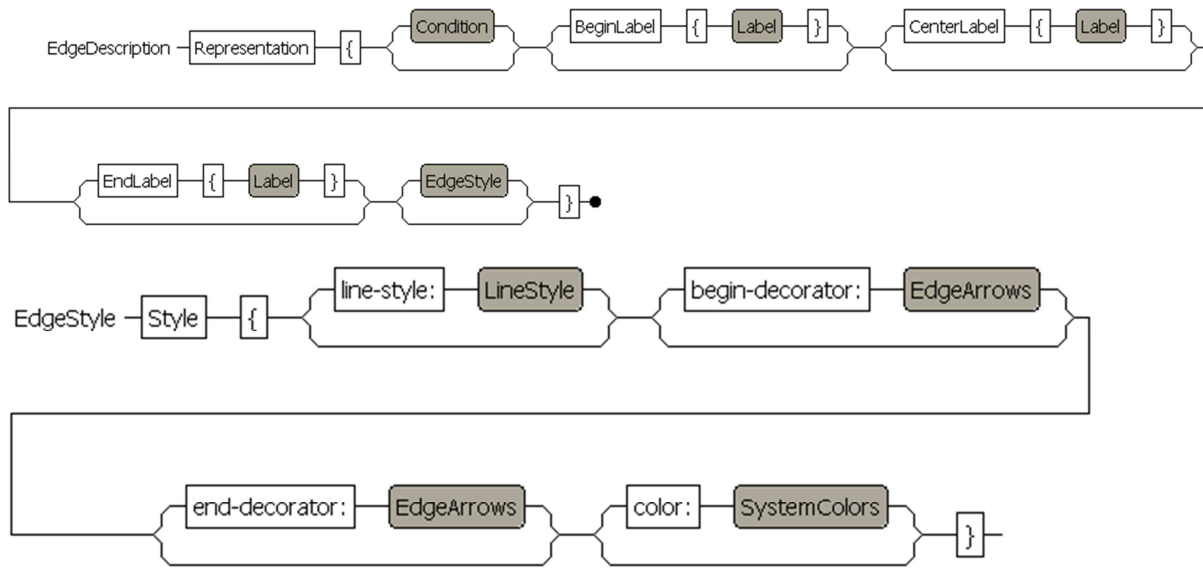
Mapping Set

Describing a set of visual descriptions associated to semantic element. Both new diagrams and diagram extensions define these descriptions.

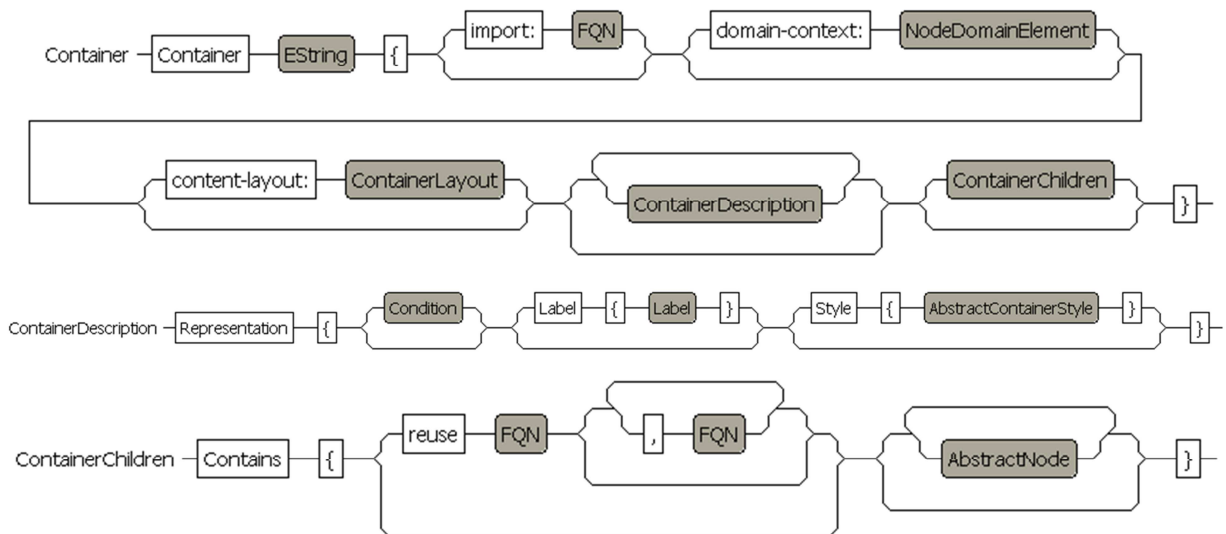


Edge description

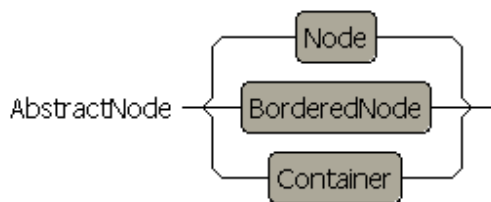


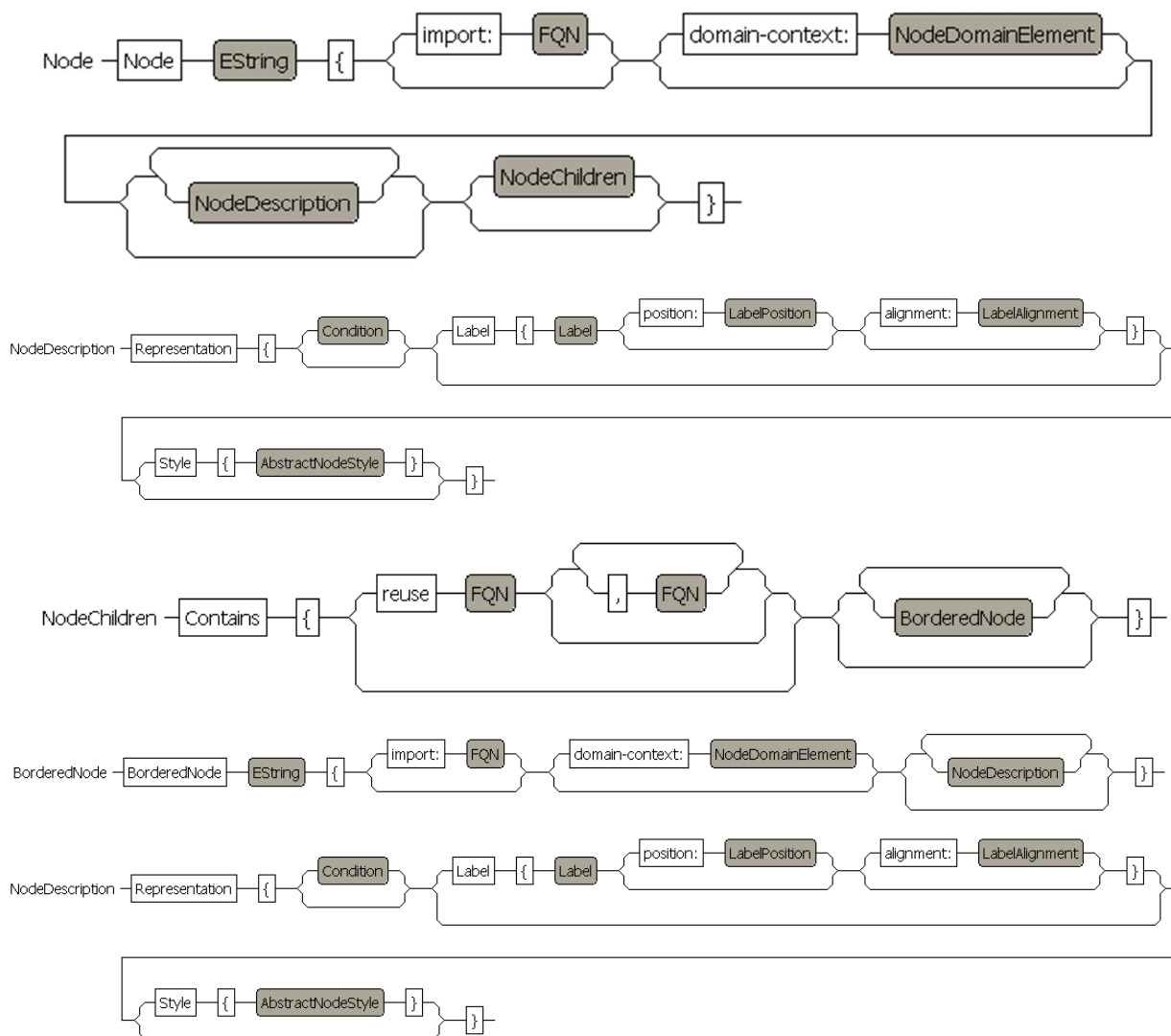


Container description



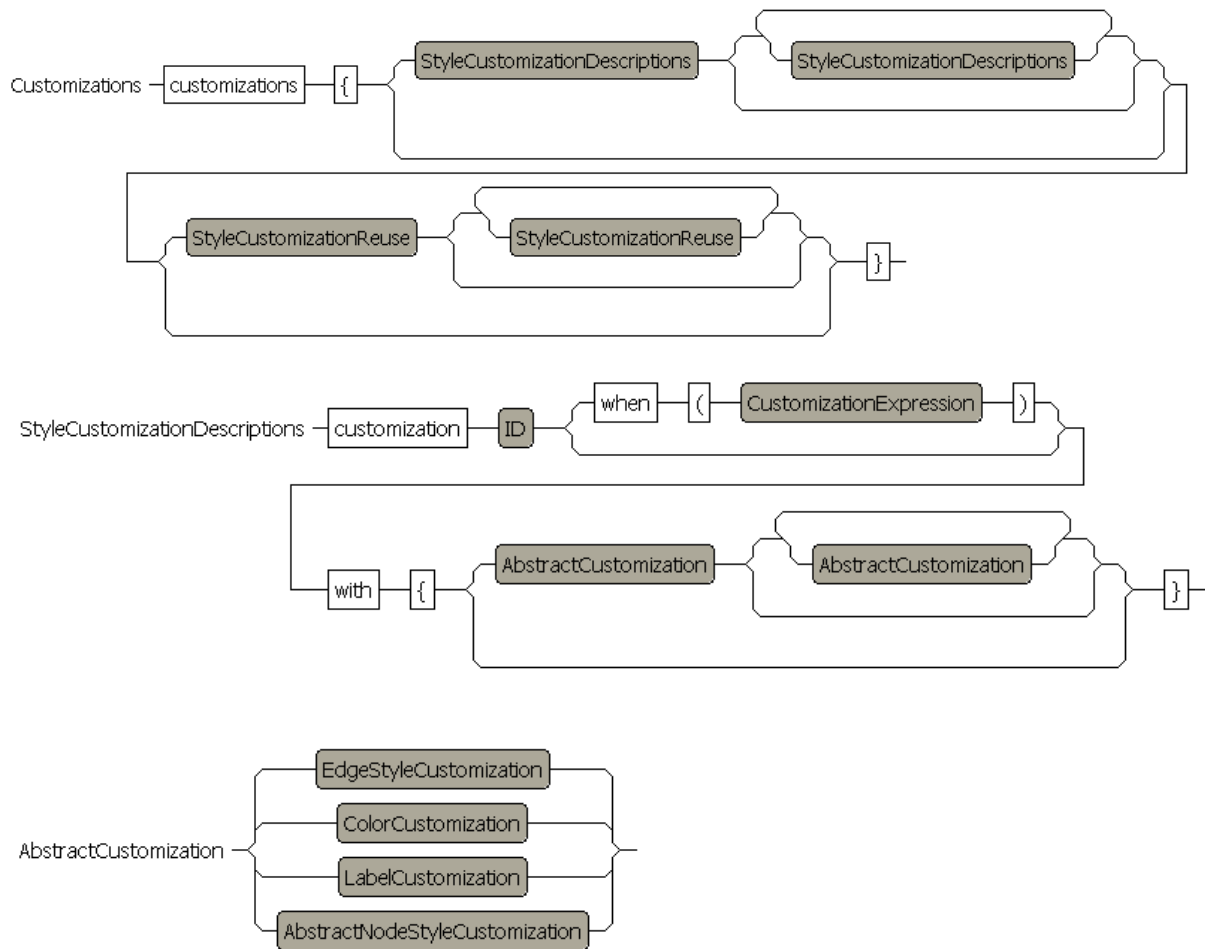
Node Description

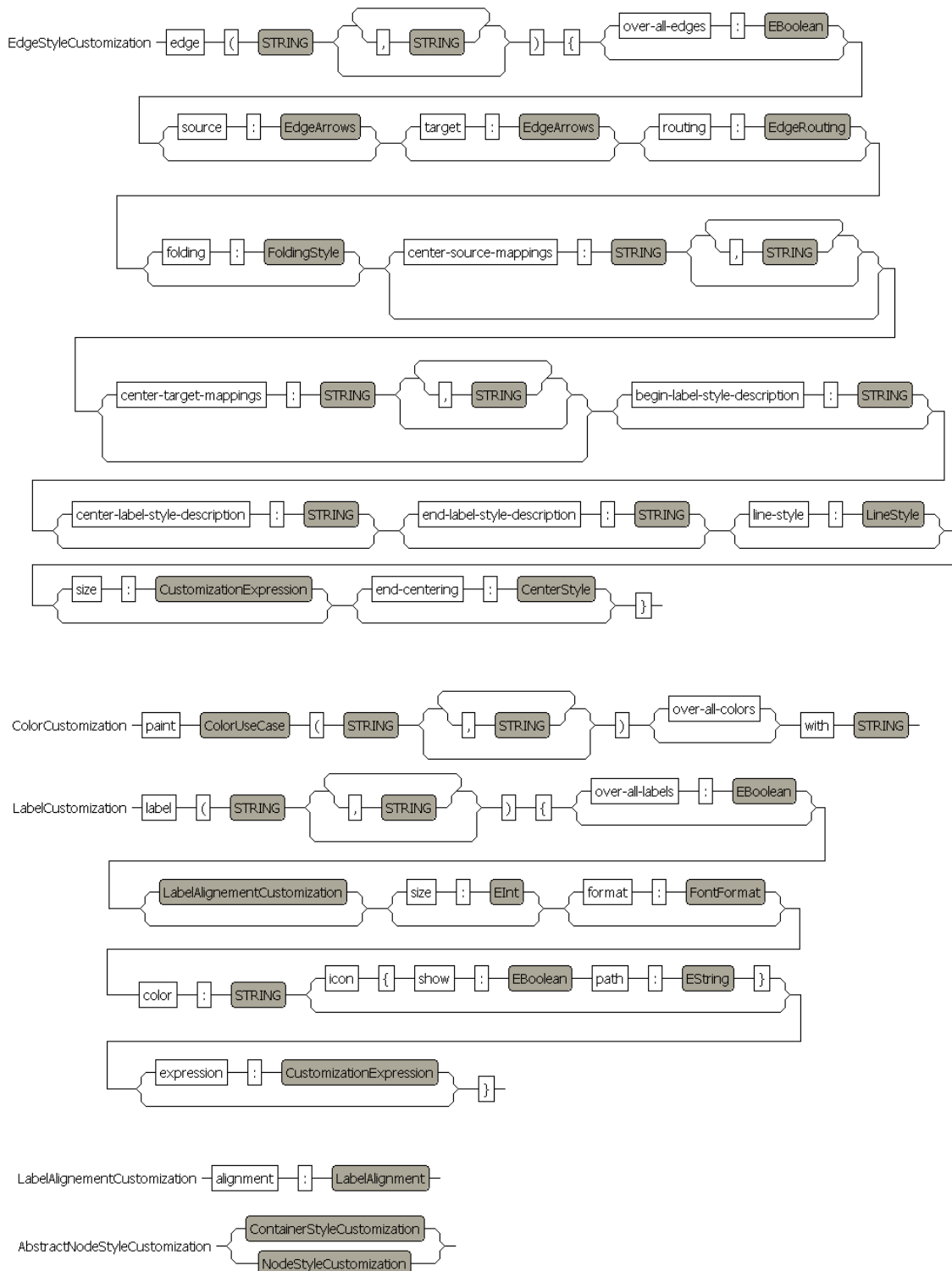


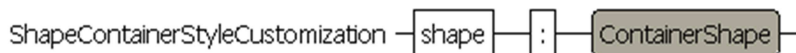
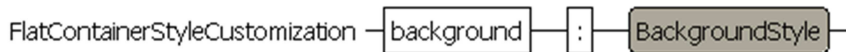
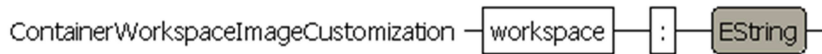
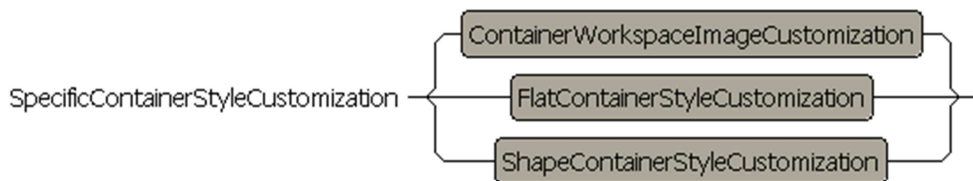
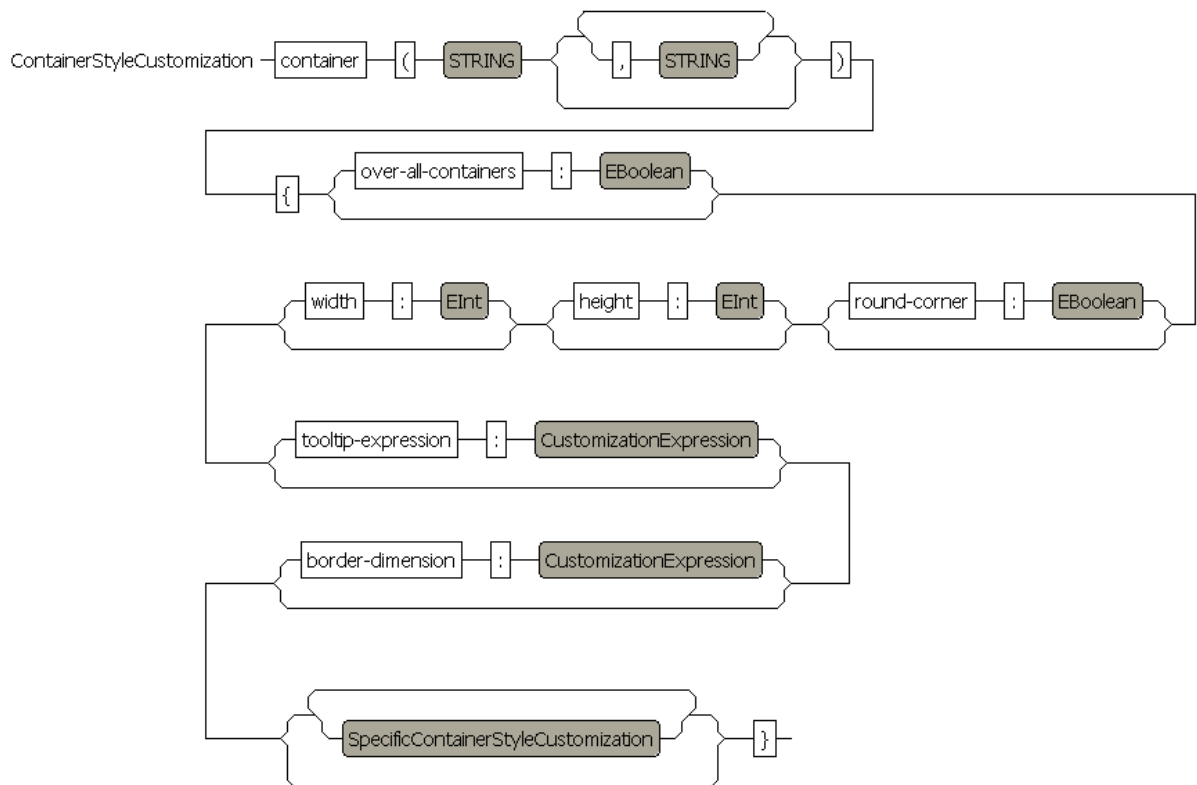


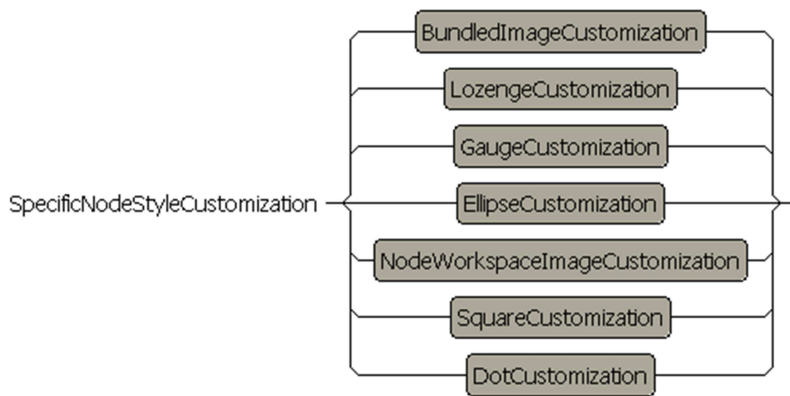
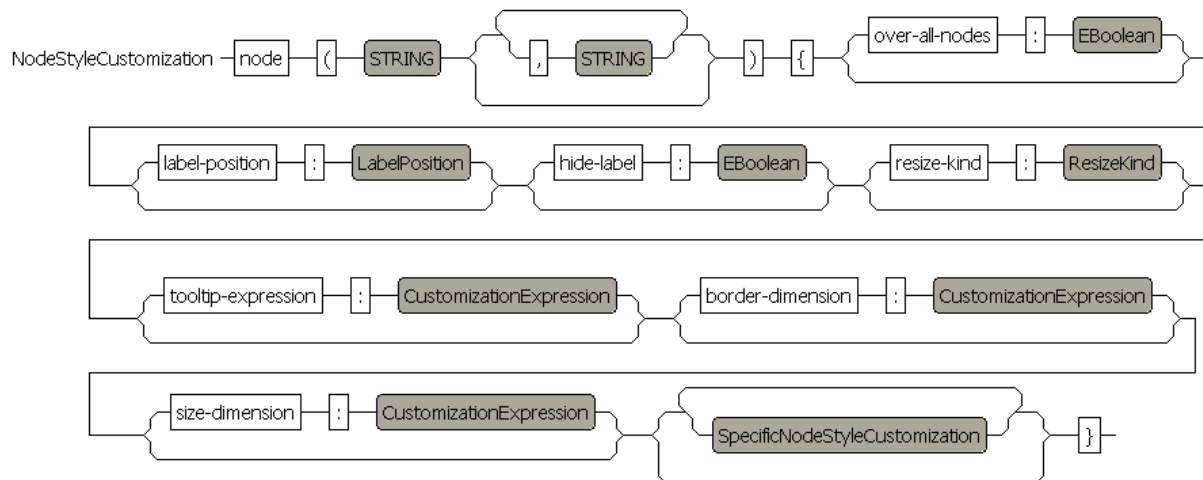
Style Customization

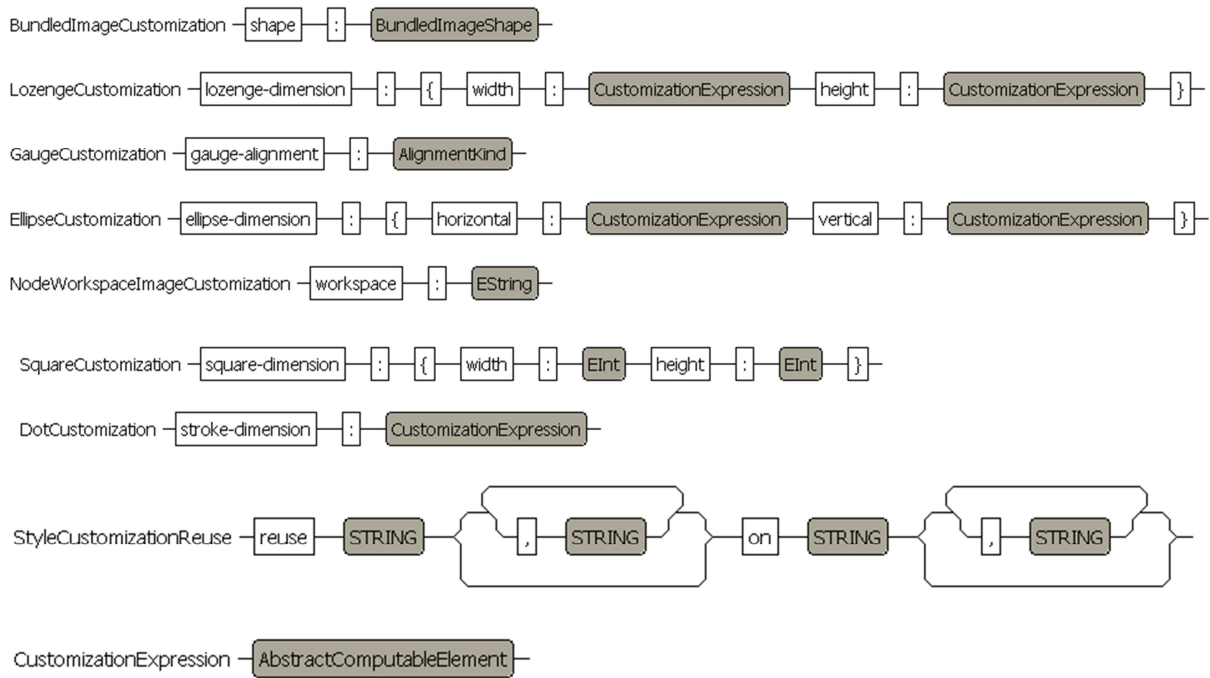
Style customizations are available in diagram extensions. They allow customizing or reusing existing styles which are defined in others Sirius representations.



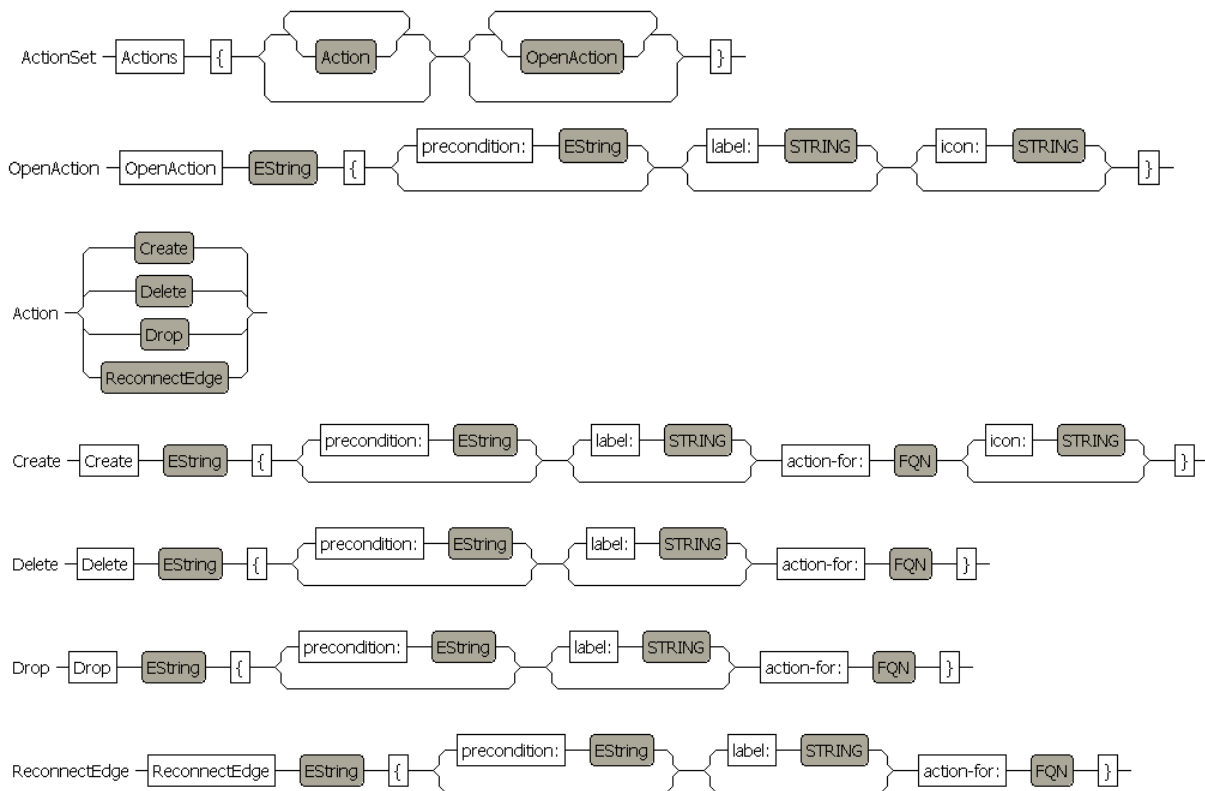






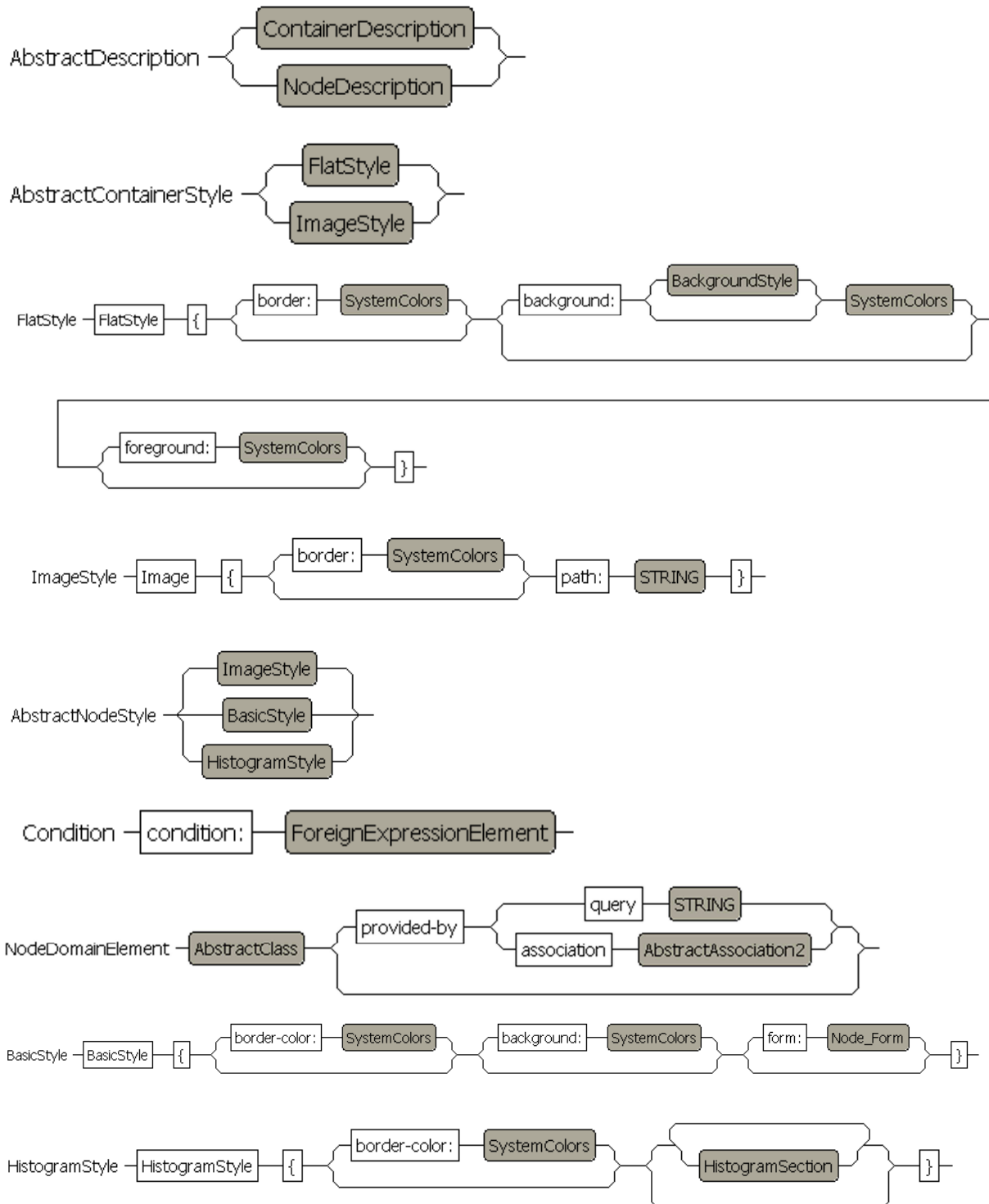


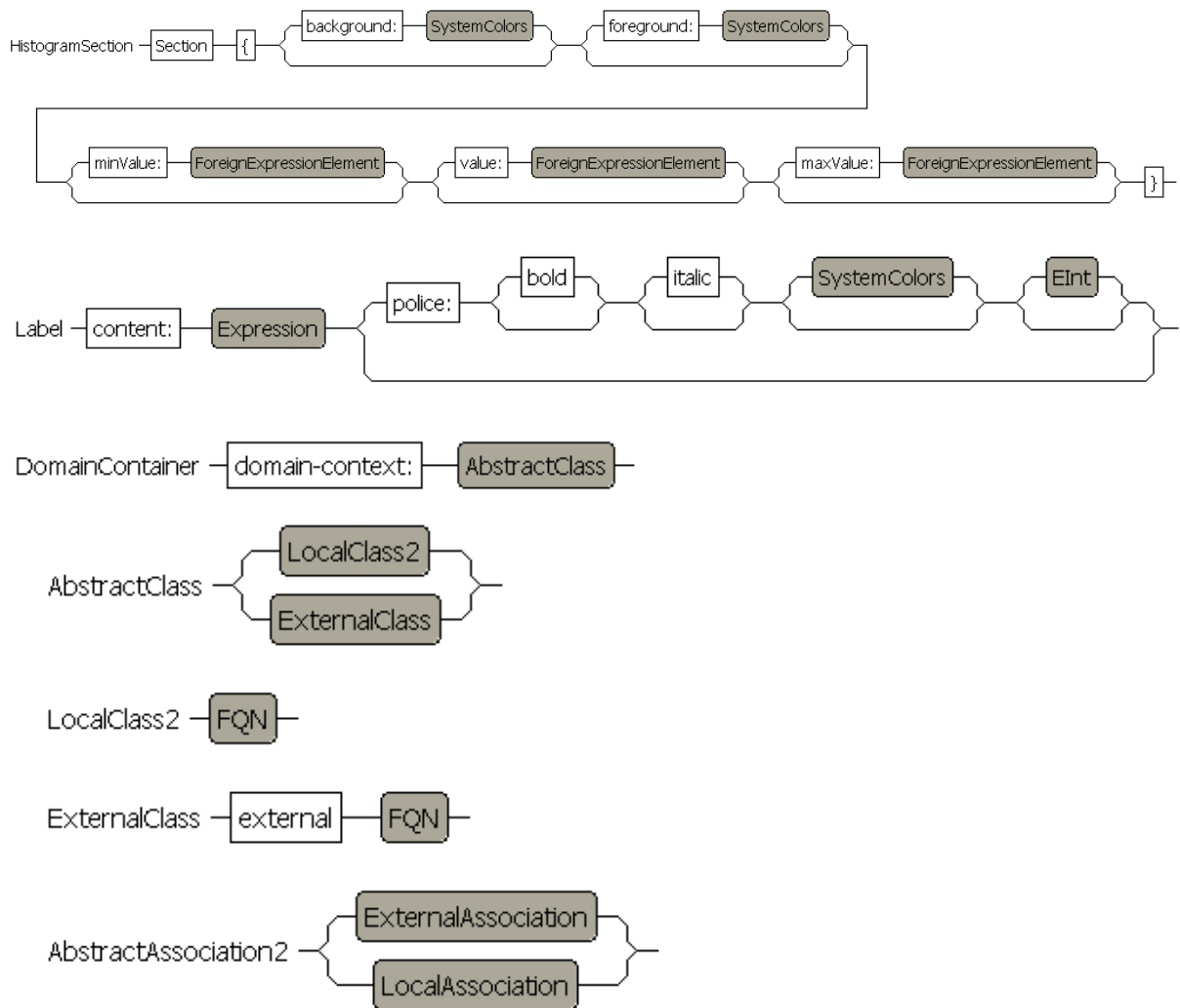
Actions Description

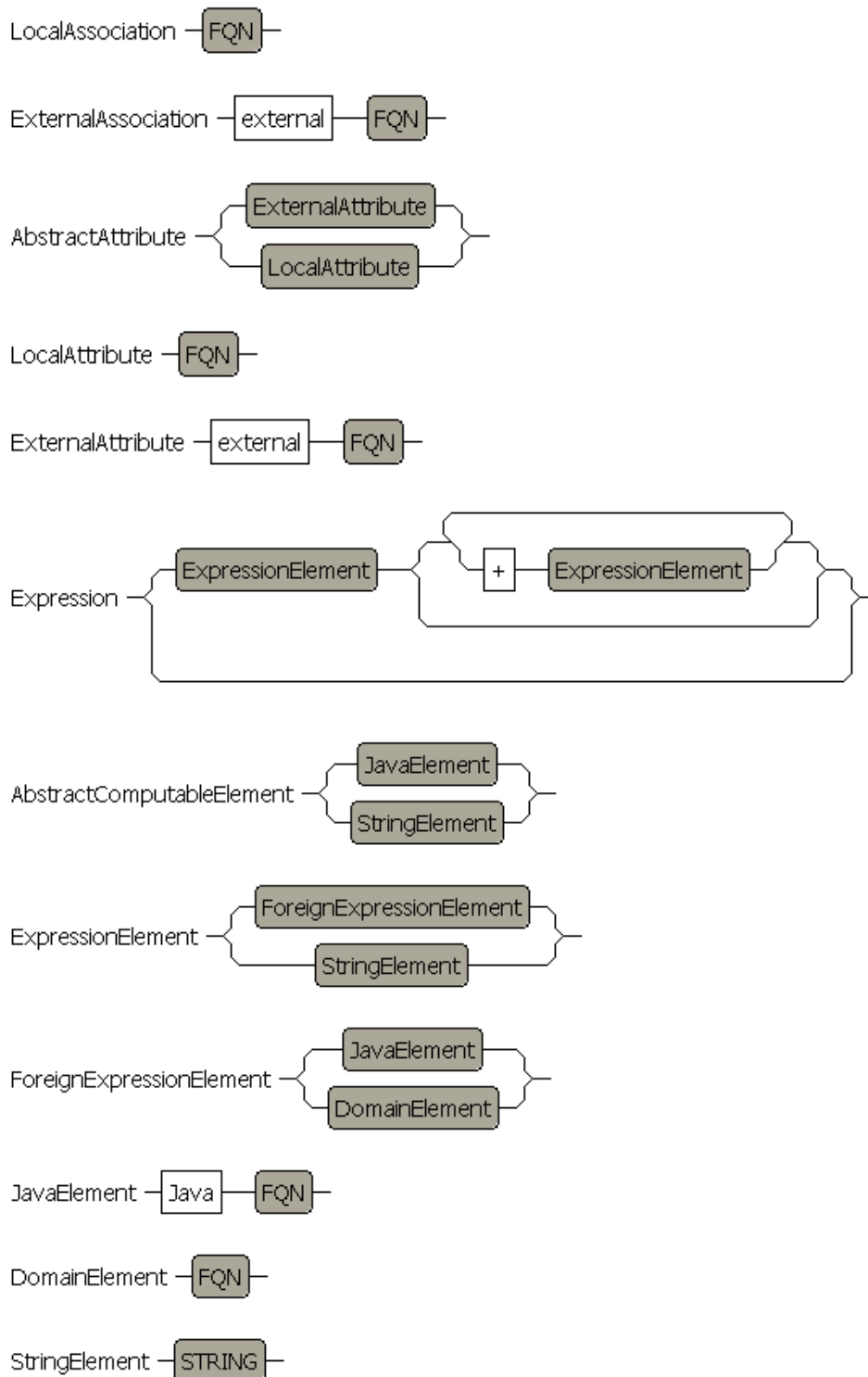


Common grammar fragments

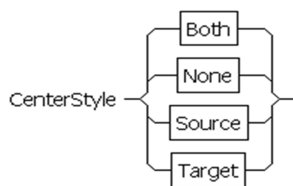
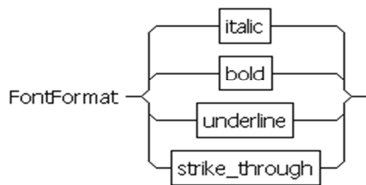
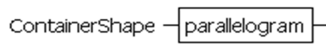
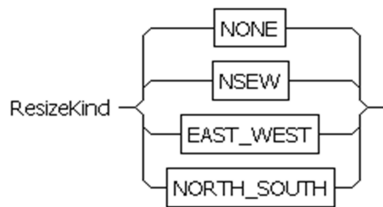
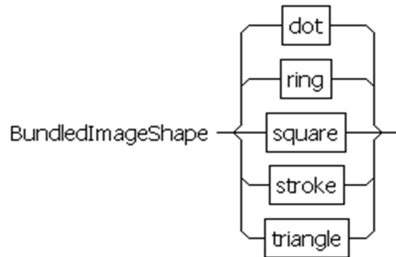
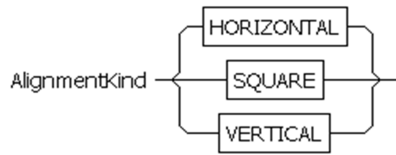
This part contains a shared between others diagram descriptions seen above

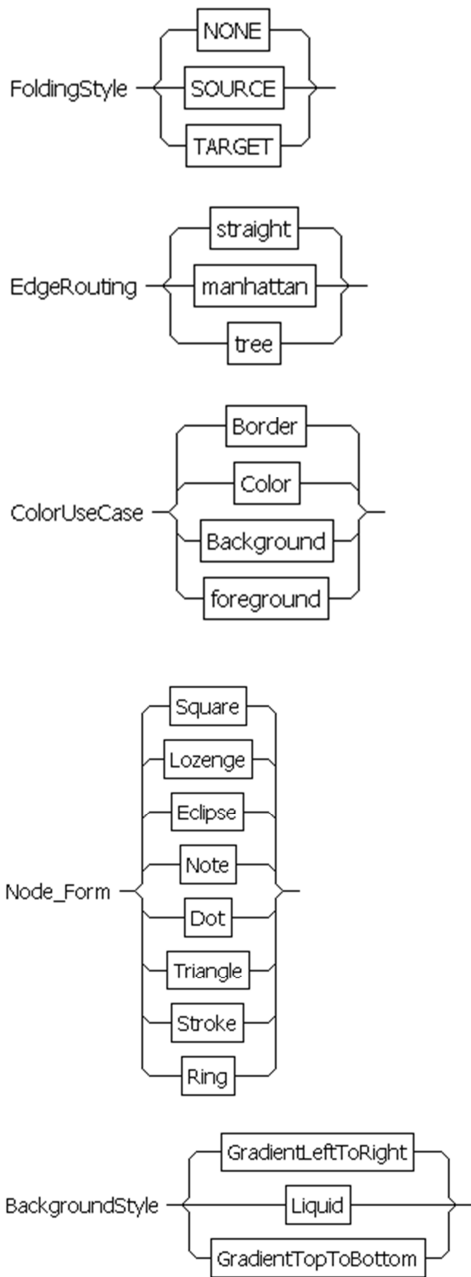


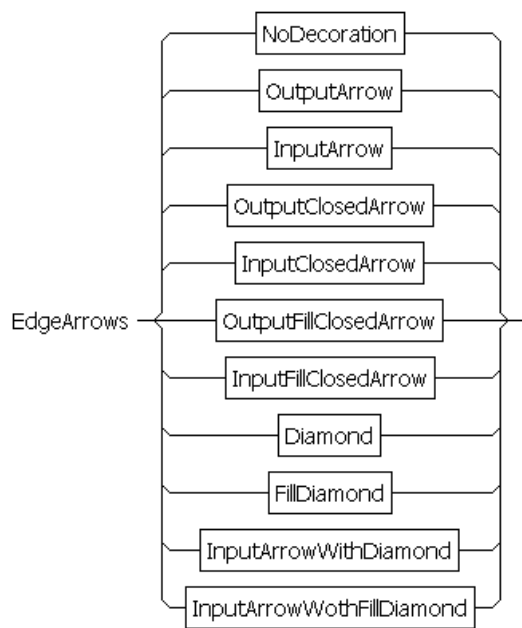
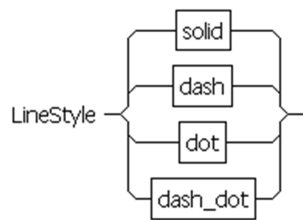
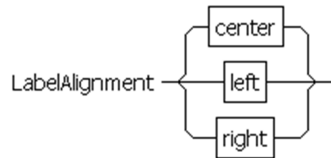
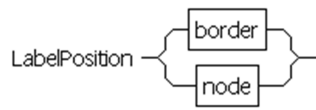


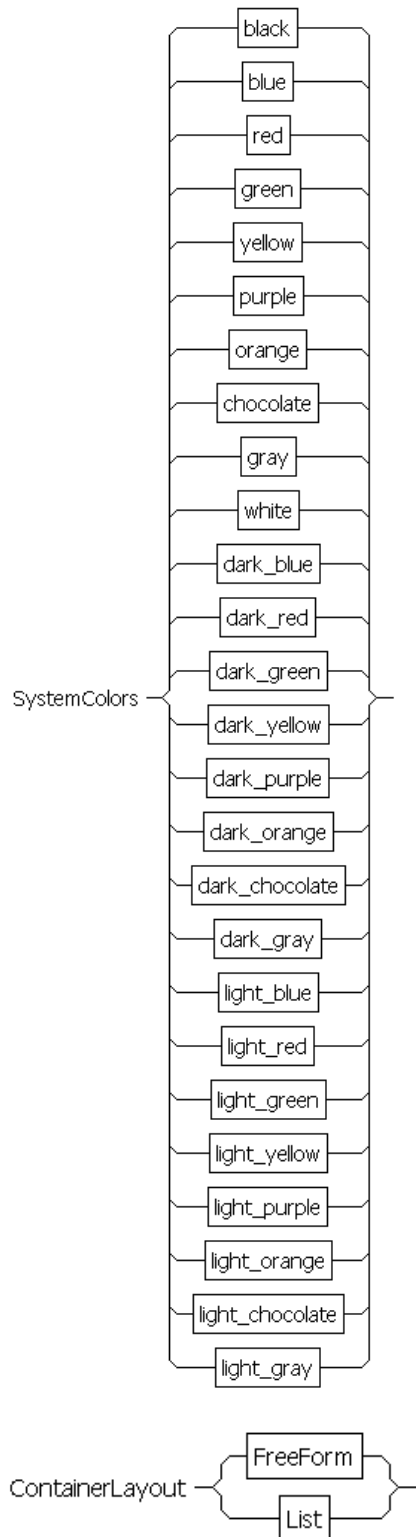


Enumerations

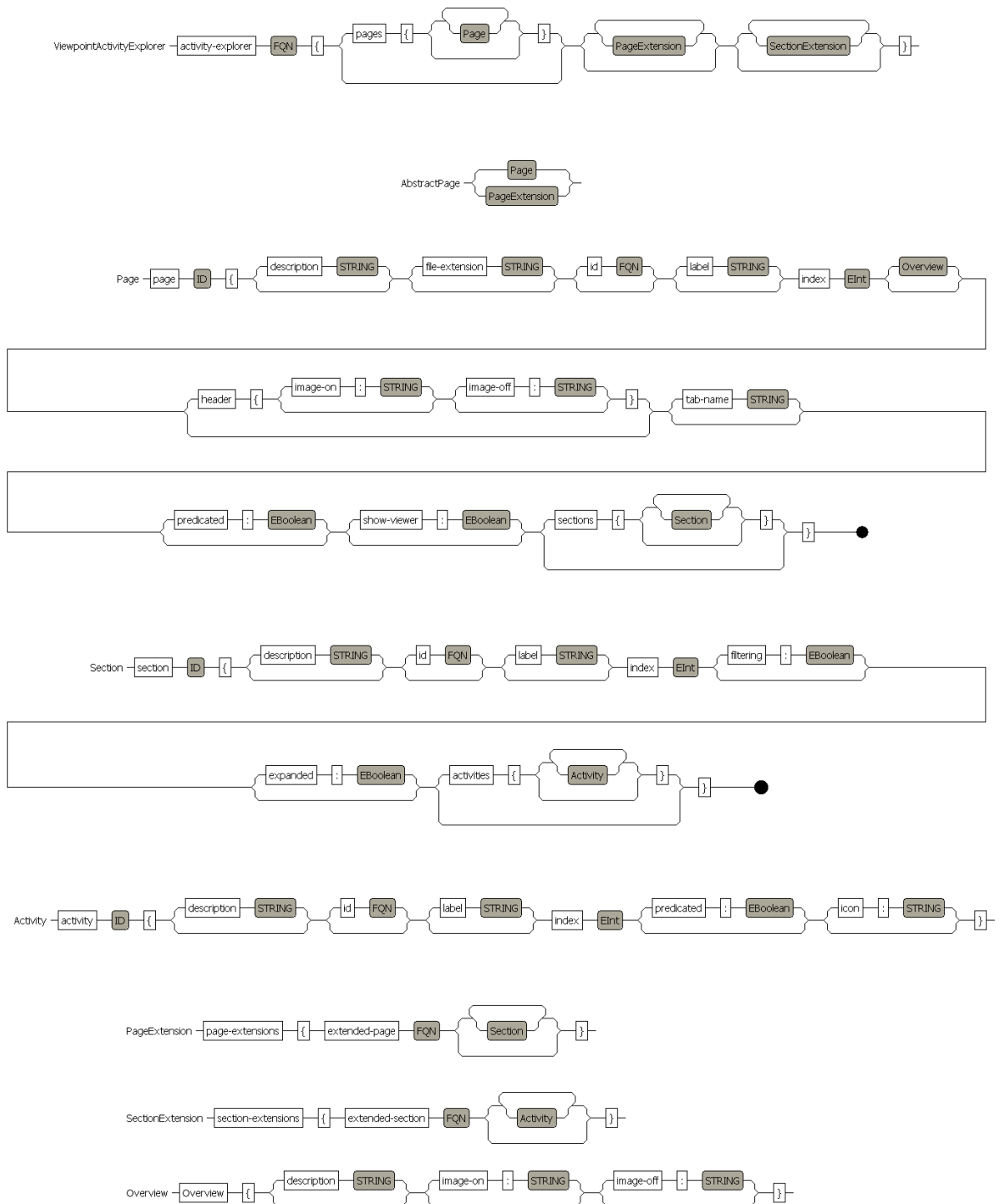








Grammar



SERVICES DESCRIPTION

Purpose: viewpoint services description

Extension file name: services.vptext

Example

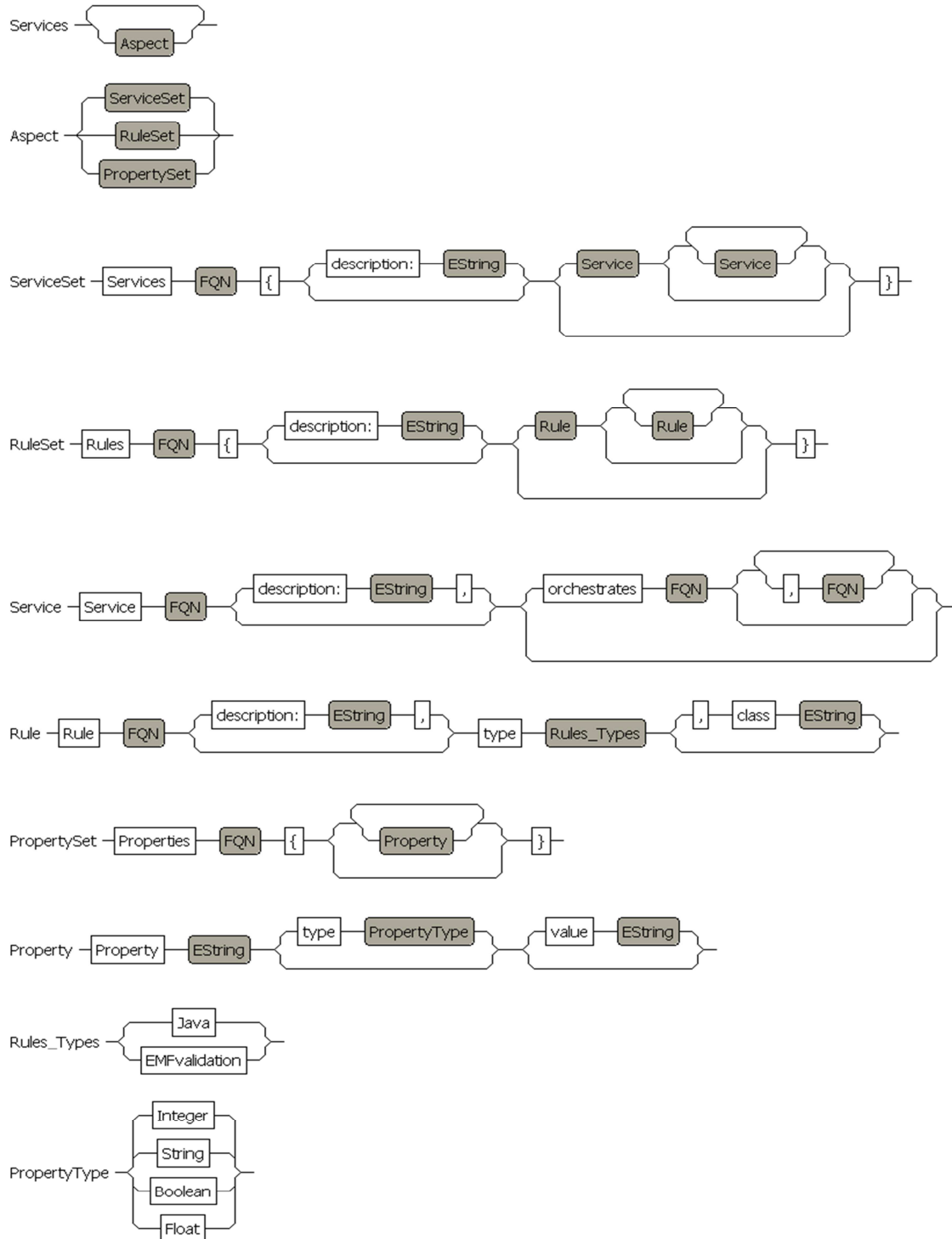
```

Rules ComponentSampleQualityAssessment.rules {
    Rule RuleOne type Java
    Rule RuleTwo type Java
}

Services ComponentSampleQualityAssessment.services {
    Service MyService orchestrates RuleOne
}
    
```

Figure 16 - Example of services.vptext file

Grammar



CONTINUOUS INTEGRATION DESCRIPTION

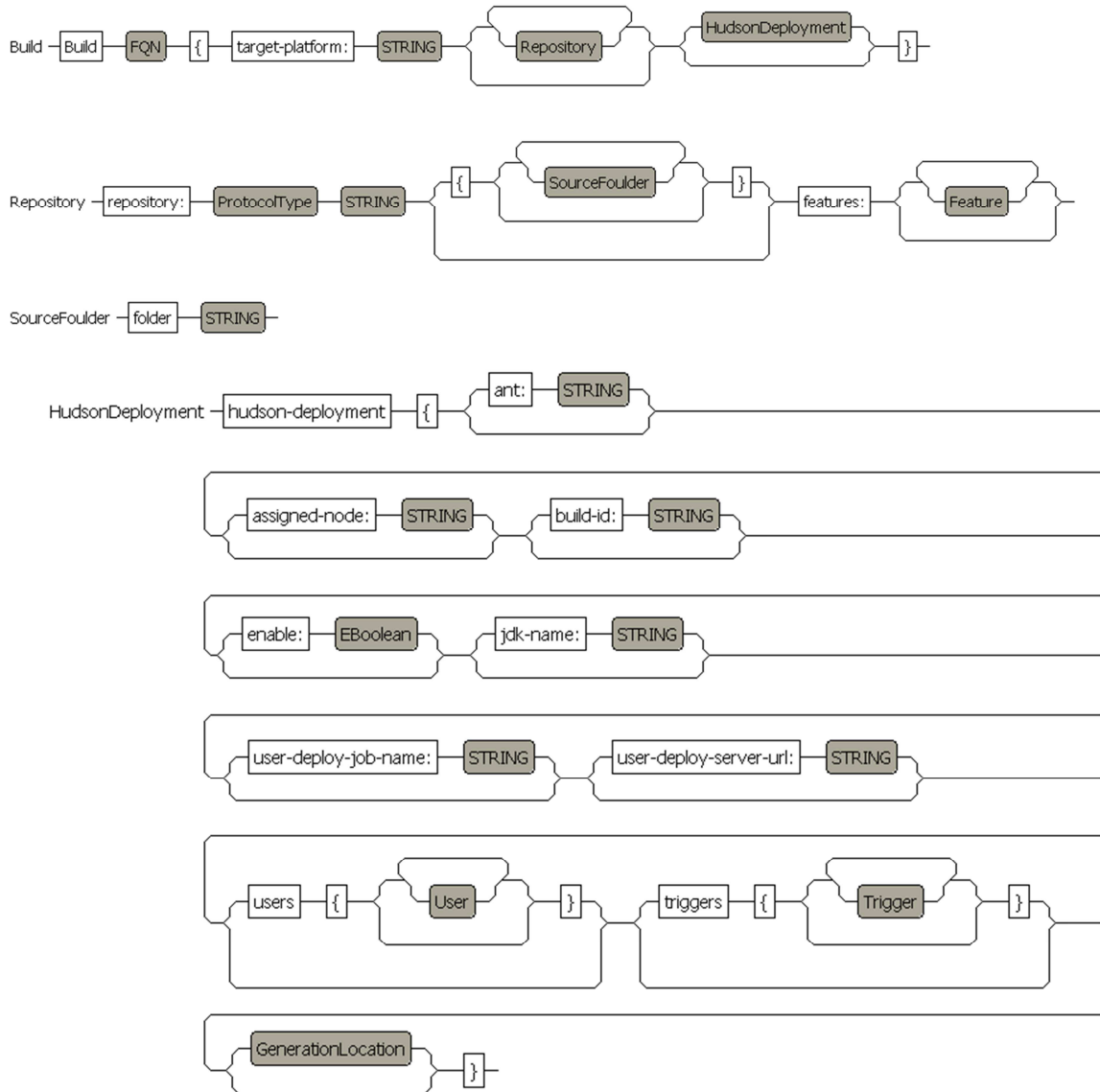
Purpose: viewpoint continuous integration description

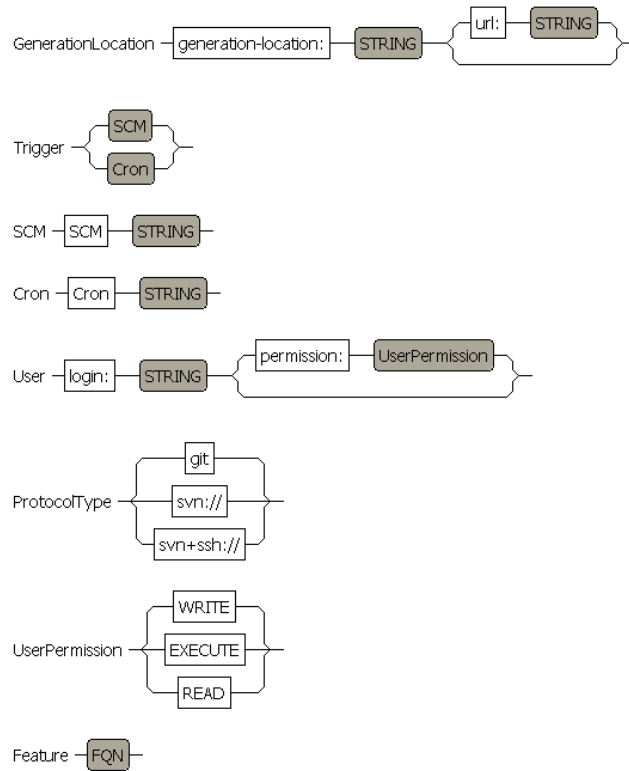
Extension file name: build.vptext

Example

```
Build ComponentSampleQualityAssessment.build {
  target-platform: "D:/ kitalpha_juno_0.5.0/eclipse.exe"
  repository: svn+ssh:// "ComponentSampleQualityAssessment/csqa"
  features: org.polarsys.kitalpha.vp.ComponentSampleQualityAssessment.feature
  hudson-deployment {
    ant: "script"
    assigned-node: "TC8"
    build-id: "ComponentSampleQualiteAssessment_20141016"
    enable: true
    jdk-name: "JavaSE-1.6"
    user-deploy-server-url: "http://ci.orchestra.ors.services.thales/"
    triggers {
      Cron "00 00 00"
      SCM "** * * *"
    }
    generation-location: "src/csqa_vp"
    url: "http://www.csqa.org"
  }
}
```

Figure 17 - Example of build.vptext file

Grammar




CONFIGURATION DESCRIPTION

Purpose: viewpoint configuration description

Extension file name: configuration.vptext

Example

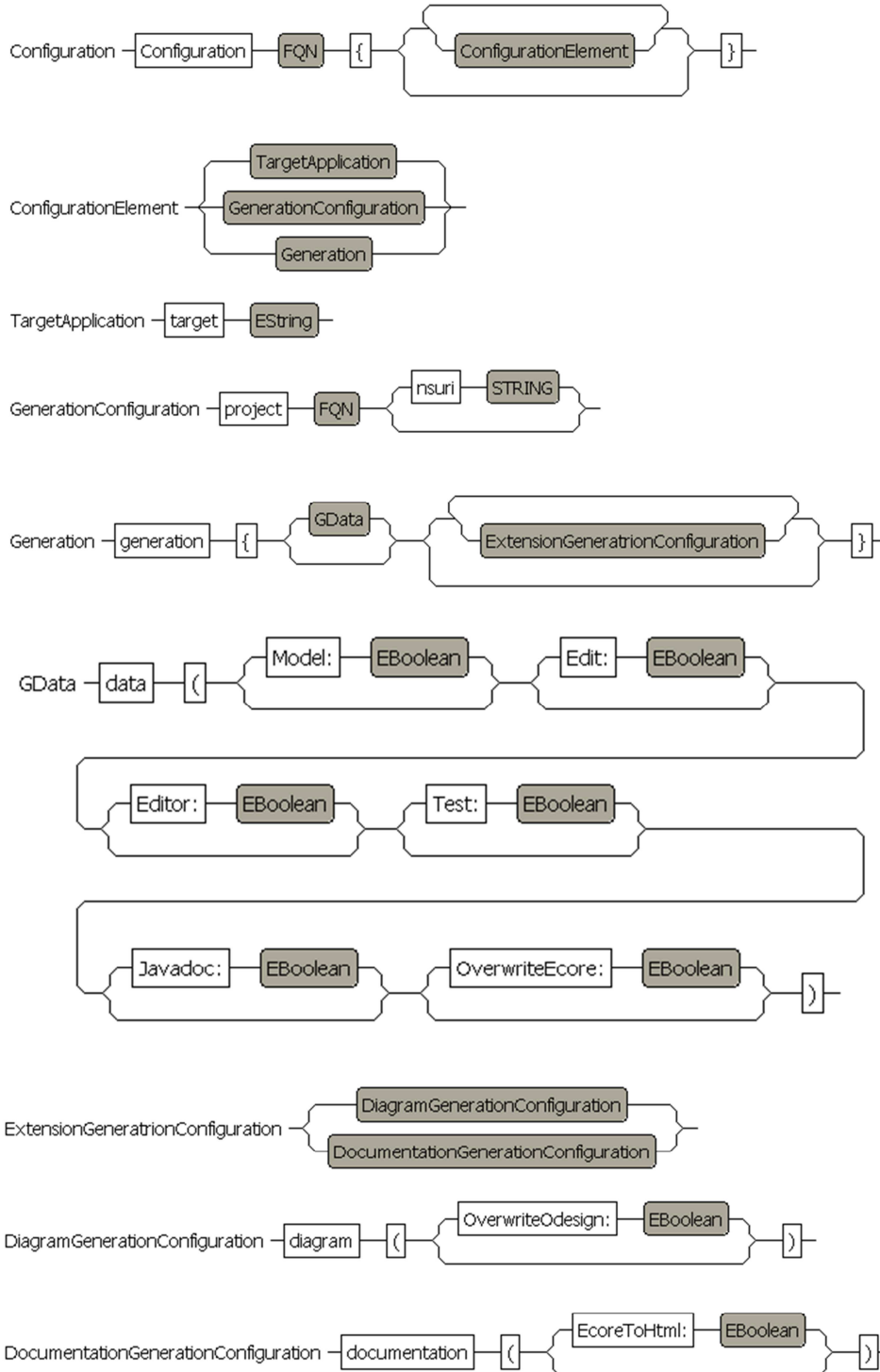
```

Configuration ComponentSampleQualityAssessment.conf {
  target EMF
  project org.polarsys.kitalpha.vp.componentsamplequalityassessment
  nsuri "http://www.polarsys.org/kitalpha/ComponentSampleQualityAssessment/1.0.0"
  generation {
    data (
      Model: true
      Edit: false
      Editor: false
      Test: true
      Javadoc: false
      OverwriteEcore: true
    )
    diagram (
      OverwriteOdesign: false
    )
    documentation (
      EcoreToHtml: false
    )
  }
}

```

Figure 18 - Example of conf.vptext file

Grammar



Templates

A template enables to insert formatted text (a sequence of text) adaptable to a context (some formatted pieces of text are computed with contextual description values).

DATA TEMPLATES

“import external – Import External Model” template

Purpose	Import an external model.
Parameters	{nsuri} entre the representation or select it with the content assist
Template	<code>import external {uri}</code>

“Class – New empty class” template

Purpose	Creation of an empty class structure.
Parameters	-
Template	<code>Class empty { }</code>

“Class – New class with attributes” template

Purpose	Creation of a class with one attribute.
Parameters	-
Template	<code>Class WithAttributes { Attributes: anAttribute type EString }</code>

“Class – New class with attributes and associations” template

Purpose	Creation of a class with one attribute and one association.
Parameters	Selection of the associated class among the existing classes of the viewpoint.
Template	<code>Class WithAttributesAndAssociations { Attributes: anAttribute type EString Associations: anAssociation contains[1,*] {Class} }</code>

“Class – New class with operations” template

Purpose	Creation of a class with one operation.
Parameters	-
Template	<pre>Class WithOperations { Operations: anOperation(external ecore.EInt [0,*] inputs) }</pre>

“Annotation – New annotation” template

Purpose	Creation of one annotation with one key/value.
Parameters	-
Template	<pre>Annotation "http://" { key: "key1" value: "value1" }</pre>

“Enumeration – New Enumeration” template

Purpose	Creation of new enumeration.
Parameters	Set the name and two default constant with the cursor
Template	<pre>Enumeration name { constant1, constant2 }</pre>

UI TEMPLATES

This part is dependent of Capella.

“UI – Generate User Interface for all classes” template

Purpose	Creation of a complete UI for all the classes of the Data aspect
Parameters	Viewpoint short name to initialize the UI name Classes with their attributes defined in the Data aspect
Template	This template is similar to the “UI – New User Interface” template, but declined for each Data classes and their attributes

“UI – New User Interface” template

Purpose	Creation of a UI preformatted for Capella.
Parameters	Viewpoint short name to initialize the UI name Viewpoint name to initialize the UI viewpoint label Selection of an attribute for a first field / attribute binding
Template	<pre> UI {name} { title: "{title}" Container {name}Section1 { Field MyFieldSection1 type text, mapped-to {class name}.{attribute name} Container {name}Group1 { Field MyFieldGroup1 type text, mapped-to {class name}.{attribute name} } } } </pre>

“UI – New User Interface” template

Purpose	Creation of new User Interface for specific class.
Parameters	{name} generate as short name of the viewpoint {class} selection of the specific class {label} generate as short name of viewpoint as Java string {section} generated with concatenate the short name, the word “section” and un number {fieldname} generated as concatenation of “MyFieldSection” with the number of the section {classAttr} select the attribute of specific class to map with the field
Template	<pre> UI {name} for {Class} { label: {label} Container {section} { Field {fieldname} type text, mapped-to {classAttr} } } </pre>

DIAGRAM TEMPLATES

“Diagram - New Diagram” template

Purpose	Creation of new Diagram.
Parameters	{class} select the class of the domain context of the diagram
Template	<pre>Diagram "newDiagramName" { domain-context: {class} Mapping { // [Container, Node or Edge] } Actions{ // [Actions] } }</pre>

“Diagram Extension - New Diagram” template

Purpose	Creation of new Diagram Extension.
Parameters	{diagram} select the diagram to extend.
Template	<pre>DiagramExtension "New Diagram Extension" { extended-diagram: {diagram} Mapping { // [Container, Node or Edge] } Actions{ // [Actions] } }</pre>

“Container – New Container provided by local association” template

Purpose	Creation of new container with provided-by local association
Parameters	<p>{localClass} select a local class</p> <p>{localAssociation} select a local association which type is {localClass}</p> <p>{attribute or expression} select one of proposed attributes or enter an expression</p>
Template	<pre> Container newContainerName { domain-context: {localClass} provided-by association {localAssociation} Representation { Label { content: {attribute or expression} police: black } Style { FlatStyle { border: blue background: light_gray } } } Contains { //[Container or Node] } } </pre>

“Container – New Container provided by external association” template

Purpose	Creation of new container with provided-by external association
Parameters	<p>{externalClass} select an external class</p> <p>{externalAssociation} select a local association which type is {externalClass}</p> <p>{attribute or expression} select one of proposed attributes</p>
Template	<pre> Container newContainerName { domain-context: external {externalClass} provided-by association external {externalAssociation} Representation { Label { content: {attribute or expression} police: black } Style { FlatStyle { border: blue background: light_gray } } } Contains { //[Container or Node] } } </pre>

“Node – New Node provided by local association” template

Purpose	Creation of new Node with provided-by local association
Parameters	{localClass} select an local class {localAssociation} select a local association which type is {localClass} {attribute or expression} select one of proposed attributes or entre an expression
Template	<pre> Node newNode { domain-context: {localClass} provided-by association {localAssociation} Representation { Label { content: {attribute or expression} police: black alignment: center } Style { BasicStyle { border-color: black background: light_gray form: Square } } } Contains { //[Container or Node] } } </pre>

“Node - New Node provided by external association” template

Purpose	Creation of new Node with provided-by external association
Parameters	<p>{externalClass} select an external class</p> <p>{externalAssociation} select a local association which type is {externalClass}</p> <p>{attribute or expression} select one of proposed attributes or enter an expression</p>
Template	<pre> Node newNode { domain-context: {externalClass} provided-by association external {externalAssociation} Representation { Label { content: {attribute or expression} police: black alignment: center } Style { BasicStyle { border-color: black background: light_gray form: Square } } } Contains { //[Container or Node] } } </pre>

“import external – Import External Representation” template

Purpose	Import external representation
Parameters	{representation} entre the representation or select it with the content assist
Template	<code>import external {representation}</code>

“Edge – New Edge provided by external association” template

Purpose	Creation of new Edge with provided-by external association
Parameters	{externalAssociation} select an external association {sourceCrossReference} select the cross reference to the source edge {targetCrossReference} select the cross reference to the target edge
Template	<pre> Edge newEdge { association-context: external {externalAssociation} source: {sourceCrossReference} target: {targetCrossReference} Representation { Style { end-decorator: OutputArrow color: black } } } </pre>

“BorderedNode – New BorderedNode by local association” template

Purpose	Creation of new BorderedNode with provided-by local association
Parameters	<p>{localClass} select an local class</p> <p>{localAssociation} select a local association which type is {localClass}</p> <p>{attribute or expression} select one of proposed attributes or enter an expression</p>
Template	<pre> BorderedNode borderedNodeName { domain-context: {localClass} provided-by association {localAssociation} Representation { Label { content: {attribute or expression} position: border alignment: center } Style { BasicStyle { border-color: light_green } } } } </pre>

“BorderedNode – New BorderedNode by external association” template

Purpose	Creation of new BorderedNode with provided-by external association
Parameters	<p>{externalClass} select an external class</p> <p>{externalAssociation} select a local association which type is {externalClass}</p> <p>{attribute or expression} select one of proposed attributes or enter an expression</p>
Template	<pre> BorderedNode borderedNodeName { domain-context: external {externalClass} provided-by association external {externalAssociation} Representation { Label { content: {attribute or expression} position: border alignment: center } Style { BasicStyle { border-color: light_green } } } } </pre>

“Action – New Create Action” template

Purpose	Creation of new create action
Parameters	{diagramNode} select the node which the action will be applied
Template	<pre> Create createActionName { label: "Create tool example" action-for: {diagramNode} } </pre>

“Action – New Delete Action” template

Purpose	Creation of new delete action
Parameters	{diagramNode} select the node which the action will be applied
Template	<pre> Delete deleteActionName { label: "Delete tool example" action-for: {diagramNode} } </pre>

“Action – New Drop Action” template

Purpose	Creation of new drop action
Parameters	-
Template	<pre> OpenAction openActionName { precondition: openActionPrecondition label: "Open Action Label" icon: "Open Action Icon" } </pre>

“OpenAction – New Open Action” template

Purpose	Creation of new open action
Parameters	{diagramNode} select the node which the action will be applied
Template	<pre> Drop dropActionName { label: "Drop tool example" action-for: {diagramNode}} </pre>

“Edge Style Customization – New Edge Style Customization” template

Purpose	Creation of new edge customization
Parameters	{edgeCustomization} select edges to customize
Template	<pre> edge ("edgeCustomization") { over-all-edges : false source : Diamond target : Diamond routing : manhattan folding : TARGET center-source-mappings : "centerSourceMappings" center-target-mappings : "centerTargetMapping" begin-label-style-description : "beginLabelStyleDescription" center-label-style-description : "centerLabelStyleDescription" end-label-style-description : "endLabelStyleDescription" line-style : solid size : "Size Expression" end-centering : Both } </pre>

“Container Style Customization – New Container Style Customization” template

Purpose	Creation of new Container customization
Parameters	{containerStyle} select containers to customize
Template	<pre> container ("containerStyle"){ over-all-containers : false width : 12 height : 12 round-corner : true tooltip-expression : "tooltip computing expression" border-dimension : "border computing expression" //add specific style customization // workspace, background and shape } </pre>

“Label Style Customization – New Label Style Customization” template

Purpose	Creation of new Label customization
Parameters	{labelStyle} select labels to customize
Template	<pre> label ("labelStyle"){ over-all-labels : false alignment : left size : 12 format : bold color : "blue" icon { show : true path : "/path/" } expression : "label expression" } </pre>

“Node Style Customization – New Node Style Customization” template

Purpose	Creation of new Node customization
Parameters	{nodeStyle} select nodes to customize
Template	<pre> node ("nodeStyle"){ over-all-nodes : false label-position : border hide-label : false resize-kind : NONE tooltip-expression : "tooltip computing expression" border-dimension : "border computing expression" size-dimension : "size computing expression" //add Specific node style // shape, lozenge dimension, gauge alignment, ellipse dimension // workspace, square dimension and stroke dimension } </pre>

“Color Style Customization – New Color Style Customization” template

Purpose	Creation of new Color customization
Parameters	<p>{colorStyle} select colors to customize</p> <p>{color} the Color</p> <p>Color Use Case: (After paint keyword)</p> <ul style="list-style-type: none"> • Color • Background • Foreground • Border
Template	paint Color ("colorStyle") with "color"

“Reuse Style Customization – New Reuse Style Customization” template

Purpose	Creation of new reuse style customization
Parameters	{reusedCustomization} select reused styles customization

	{onStyle} select the target styles on which the reused ones will be applied
Template	reuse "reusedCustomization" on "onStyle"

ACTIVITY EXPLORER TEMPLATES

“pages - New pages” template

Purpose	Creation of new pages part.
Parameters	<p>{pageName} the name of the page</p> <p>{fileExtension} the extension of the resource where the page will appear</p> <p>{pageId} an unique identifier for the page</p> <p>{pageIndex} the index of the page within the Activity Explorer</p> <p>Overview:</p> <ul style="list-style-type: none"> • Description: the description of the page within the Overview page of the Activity Explorer • {OverviewImagePathOn} icon page when the cursor is over within the Overview page • {OverviewImagePathOff} icon page when the cursor is outside within the Overview page <p>Header</p> <ul style="list-style-type: none"> • {pageImagePathOn} header icon page which appear when the page selected • {pageImagePathOff} header icon page which appear when the page not selected <p>{tabName} the name of the page tab within the Activity Explorer editor</p> <p>{show-viewer} indicate if the page owned a viewer</p>
Template	<pre> pages { page {pageName} { file-extension "{fileExtension}" id pageName_id label "Page Label" index {pageIndex} Overview { description "<p>title</p><p>description</p>" image-on : "{OverviewImagePathOn}" image-off : "{OverviewImagePathOff}" } header { image-on : "{PageImagePathOn}" image-off : "{PageImagePathOff}" } } tab-name "{tabName}" show-viewer : true sections { section {sectionName} { description "<p>description</p>" id sectionName_id label "Section Label" index {sectionIndex} filtering : true expanded : false activities { activity {activityName} { description "description" id activityName_id } } } } } </pre>

	<pre> label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" </pre>
--	--

“page - New page” template

Purpose	Creation of new page within pages part.
Parameters	<p>{pageName} the name of the page</p> <p>{fileExtension} the extension of the resource where the page will appear</p> <p>{pageId} an unique identifier for the page</p> <p>{pageIndex} the index of the page within the Activity Explorer</p> <p>Overview:</p> <ul style="list-style-type: none"> • Description: the description of the page within the Overview page of the Activity Explorer • {OverviewImagePathOn} icon page when the cursor is over within the Overview page • {OverviewImagePathOff} icon page when the cursor is outside within the Overview page <p>Header</p> <ul style="list-style-type: none"> • {pageImagePathOn} header icon page which appear when the page selected • {pageImagePathOff} header icon page which appear when the page not selected <p>{tabName} the name of the page tab within the Activity Explorer editor</p> <p>{show-viewer} indicate if the page owned a viewer</p>
Template	<pre> page {pageName} { file-extension "{fileExtension}" id pageName_id label "Page Label" index {pageIndex} Overview { description "<p>title</p><p>description</p>" image-on : "{OverviewImagePathOn}" image-off : "{OverviewImagePathOff}" } header { image-on : "{PageImagePathOn}" image-off : "{PageImagePathOff}" } tab-name "{tabName}" show-viewer : true sections { section {sectionName} { description "<p>description</p>" } } } </pre>

	<pre> id sectionName_id label "Section Label" index {sectionIndex} filtering : true expanded : false activities { activity {activityName} { description "description" id activityName_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } </pre>
--	---

“sections - New sections” template

Purpose	Creation of new sections part within page.
Parameters	-
Template	<pre> Sections { section {sectionName} { description "<p>description</p>" id section_id label "Section Label" index {sectionIndex} filtering : true expanded : false activities { activity {activityName} { description "description" id activityName_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } } } </pre>

“section - New section” template

Purpose	Creation of new section within page.
Parameters	{sectionName} the name of the section

	<p>{description} the description of the section</p> <p>{sectionId} an unique identifier for the section</p> <p>{sectionIndex} the index of the section within the page</p> <p>{filetring} indicate if the section filter the viewer</p> <p>{expanded} indicate if the section is expanded at the activation of the page</p>
Template	<pre> section {sectionName} { description "<p>description</p>" id section_id label "Section Label" index {sectionIndex} filtering : true expanded : false activities { activity {activityName} { description "description" id activityName_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } } </pre>

“activities – New activities” template

Purpose	Creation of new activities part within the section.
Parameters	-
Template	<pre> activities { activity {activityName} { description "description" id activityName_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } </pre>

“activity – New activity” template

Purpose	Creation of new activity within the sections part.
Parameters	<p>{activityName} the name of the activity</p> <p>{description} the description of the activity</p> <p>{activity_id} a unique id for the activity</p>

	<p>{activity label} the label of the activity</p> <p>{activityIndex} the index of the activity</p> <p>{predicated} indicate if the activity is predicated (shown if a certain guard is checked)</p> <p>{iconPath} the icon of the activity</p>
Template	<pre> activity {activityName} { description "description" id activity_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } </pre>

“page-extensions - New page-extensions” template

Purpose	Creation of new page extensions.
Parameters	<p>{pageId} the id of the extended page</p> <p>section: the section added to the extended page</p>
Template	<pre> page-extensions { extended-page {pageId} section {sectionName} { description "<p>description</p>" id sectionName_id label "Section Label" index {sectionIndex} filtering : true expanded : false activities { activity {activityName} { description "description" id activityName_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } } } </pre>

“section-extensions – New section-extensions” template

Purpose	Creation of new section extensions.
Parameters	{sectionId} the id of the extended section
Template	<pre> section-extensions { extended-section {sectionId} activity {activityName} { description "description" id activity_id label "Activity Label" index {activityIndex} predicated : false icon : "{iconPath}" } } </pre>

RULES TEMPLATES

“Rules – New rule set” template

Purpose	Creation of a rule set with one Java rule.
Parameters	-
Template	<pre>Rules { Rule RuleOne type Java }</pre>

SERVICES TEMPLATES

“Services – New service set” template

Purpose	Creation of a service set with one service.
Parameters	{shortname} the shortname of the viewpoint {relatedRule} Selection of the first rule among the existing viewpoint rules
Template	<pre>Services {shortname}.services { Service MyService orchestrates {relatedRule} }</pre>

“Service – New service” template

Purpose	Creation of a new service within service set.
Parameters	{relatedRule} Selection of the first rule among the existing viewpoint rules
Template	<pre>Service MyService orchestrates {relatedRule}</pre>

“Rules – New rule set” template

Purpose	Creation of a rule set.
Parameters	{shortname} the shortname of the viewpoint
Template	<pre>rules {shortname}.rules{ rule RuleOne type Java }</pre>

“Rule – New rule” template

Purpose	Creation of a new rule within rule set.
Parameters	-
Template	<pre>Rule RuleOne type Java</pre>

“Properties – New property set” template

Purpose	Creation of a new property set.
Parameters	{shortname} the shortname of the viewpoint
Template	<pre>Properties {shortname}.properties { Property MyProperty type String value "MyProperty" }</pre>

“Property – New property” template

Purpose	Creation of a new property.
Parameters	{shortname} the shortname of the viewpoint
Template	<pre>Property MyProperty type String value "MyProperty"</pre>

CONFIGURATION TEMPLATES

“Configuration – New Configuration” template

Purpose	Creation of a new configuration.
Parameters	{shortname} the shortname of the viewpoint
Template	<pre> Configuration {shortname}.conf { target EMF project org.polarsys.kitalpha.vp.{shortname} nsuri "http://www.polarsys.org/kitalpha/{shortname}" } </pre>

“Data Configuration – New Data Configuration” template

Purpose	Creation of a new data configuration.
Parameters	-
Template	<pre> data (Model: true Edit: true Editor: false Test: false Javadoc: false OverwriteEcore: true) </pre>

HELPERS

FOR ANY EDITOR

Text completion

Purpose	Contextual text completion
Actions	CTRL+SPACE

VIEWPOINT DATA DESCRIPTION – [VIEWPOINT NAME].DATA.VPTEXT EDITOR

Class list selection

Purpose	Selection of one class among a class list.
Format	Class name, package + "." + class name, and a definition when available.
Examples	See the following figure.

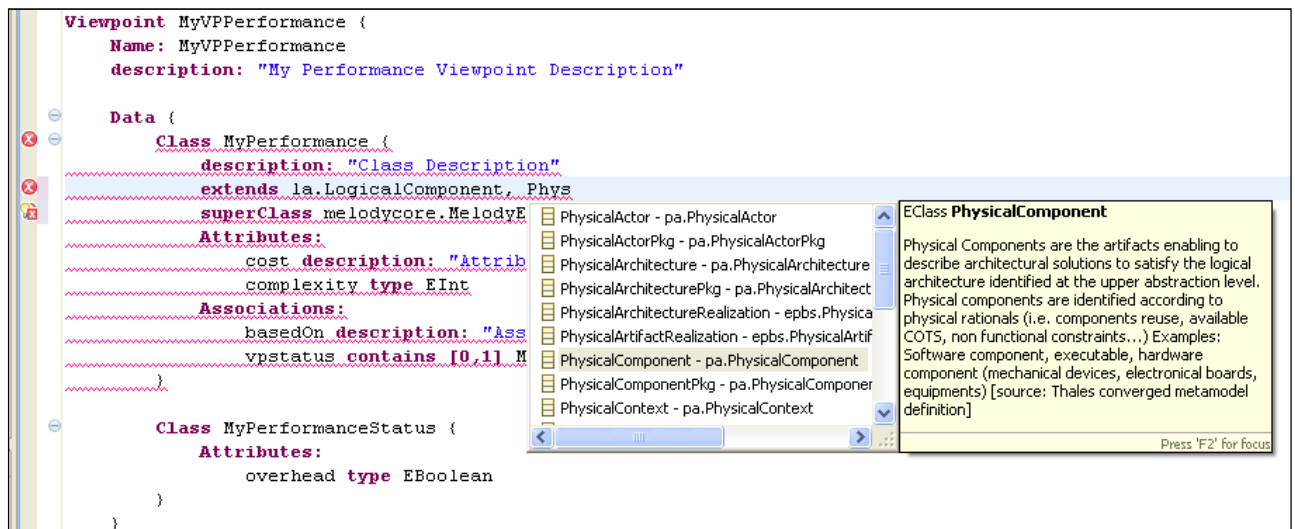


Figure 19 - Helper for class list selection

Commands

A command enables to execute a set of actions.

FOR ANY EDITOR

Text completion command

Purpose	Text completion
Actions	CTRL+SPACE

Text format command

Purpose	Reformatting the edited text
Actions	Shift+CTRL+F

Save command

Purpose	Save of the current text.
Actions	CTRL+S The text is translated into a model which conforms to the Viewpoint DSL metamodel.

MAIN VIEWPOINT EDITOR - [VIEWPOINT NAME].SPEC.VPTEXT EDITOR

Open file command

Purpose	Open/Edit a type of viewpoint description
Actions	F3 on a name after the aspect name

File selection command

Purpose	Selection of description file
Actions	CTRL+SPACE after the aspect name

« Generate viewpoint » command

Purpose	Generation of the viewpoint implementation.
Actions	Right-Click in the editor and Generation Viewpoint Generation of the viewpoint artifacts (e.g., models, diagrams, EMF plugins).

« Generate and package viewpoint » command

Purpose	Generation and packaging of the viewpoint implementation.
Actions	Right-Click in the editor and Generate and Package Viewpoint This corresponds to a command of viewpoint generation followed by a packaging of this generation.

SAMPLE ACTIVITY EXPLORER PREVIEW – [VIEWPOINT NAME].ACTIVITYEXPLORER.VPTEXT EDITOR

When cursor is on same keywork, a hover appears which display basic result of the current activity explorer description (current activity explorer description only)

Preview page

Purpose	Get Preview of a page
Actions	Cursor on page keyword


Example	<pre> @author: suuzouur * @date: 14/09/15 * */ activity-explorer ComponentSample.activityexplorer { pages { page ComponentSampleArchitecture { Component Sample Architecture (Page Name) Diagrams of Component Sample Architecture (Section Name) Set Name To Component Sample Model (Activity Name) Create a new Diagram show-viewer : true sections { section DiagramsComponentSample { description "<p>Create a new Component Sample Architecture Diagram</p>" id org.polarsys.kitalpha.vp.componentsample.^activity.^explorer.architecture.^section label "Diagrams of Component Sample Architecture" index 1 filtering : true expanded : false activities { activity SetModelName { </pre>
----------------	---

Overview preview page

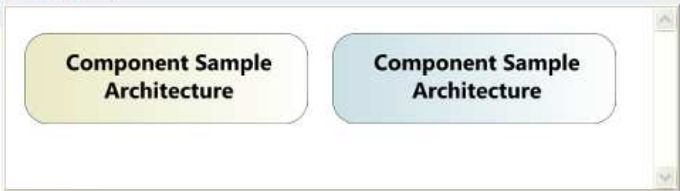
Purpose	Get Preview of the overview of the page
Actions	Cursor on overview keyword
Example	<pre> pages { page ComponentSampleArchitecture { file-extension "componentsample" id org.polarsys.kitalpha.vp.componentsample.^activity.^explorer.architecture label "Component Sample Architecture" index 1 overview { Component Sample Architecture Create a new Component Sample Architecture (Image On) (Image Off) (Description) } sections { section DiagramsComponentSample { </pre>

Description preview

Purpose	Get Preview of description of the page
Actions	Cursor on description keyword

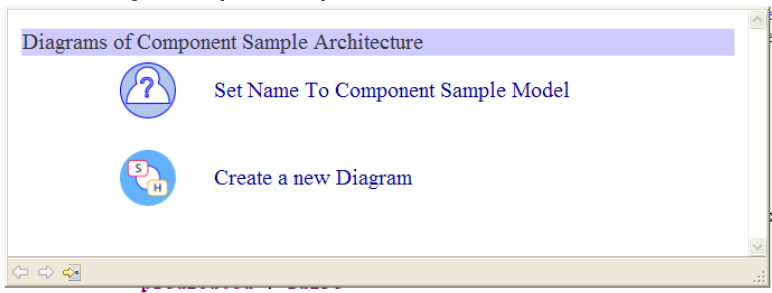
<p>Example</p>	<pre> id org.polarsys.kitalpha.vp.componentsample.^activity.^explorer.. label "Component Sample Architecture" index 1 Overview { description "<p>Component Sample Architecture</p><p>C } head Create a new Component Sample Architecture } tab- show sections { section DiagramsComponentSample { </pre> 
-----------------------	--

Page header preview

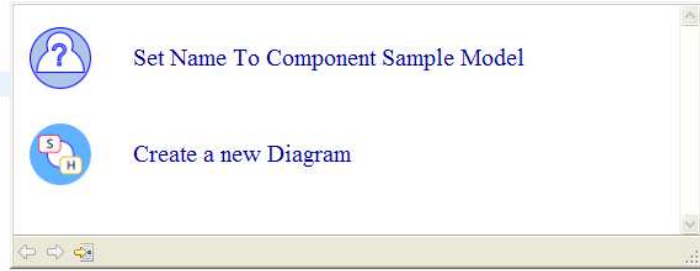
<p>Purpose</p>	<p>Get Preview of the header of the page</p>
<p>Actions</p>	<p>Cursor on header keyword</p>
<p>Example</p>	<pre> Overview { description "<p>Component Sample Architecture</p><p>Cr image-on : "csa_on.png" image-off : "csa_off.png" } header { </pre>  <pre> } sample Architectur id org.polarsys.kitalpha.vp.componentsample.^activity.^ex label "Diagrams of Component Sample Architecture" </pre>

Section preview Page


<p>Purpose</p>	<p>Get Preview of section of page</p>
<p>Actions</p>	<p>Cursor on section keyword</p>

<p>Example</p>	<pre> show-viewer : true sections { section DiagramsComponentSample { Diagrams of Component Sample Architecture Set Name To Component Sample Model Create a new Diagram } } icon : "set_name2.png" </pre> 
-----------------------	--

Activities preview page

<p>Purpose</p>	<p>Get Preview of the activities of page</p>
<p>Actions</p>	<p>Cursor on activities keyword</p>
<p>Example</p>	<pre> expanded : raise activities { Set Name To Component Sample Model Create a new Diagram } label "Create a new Diagram" index 2 </pre> 

Activity preview

<p>Purpose</p>	<p>Get Preview of an activity</p>
<p>Actions</p>	<p>Cursor on activity keyword</p>
<p>Example</p>	<pre> activity SetModelName { Set Name To Component Sample Model } </pre> 

Icon preview

Purpose	Get Preview of an icon
Actions	Cursor on icon keyword
Example	<pre> } } } icon : "create_diagram.png" } } </pre> 