DyMMer

Software Architecture Document

Revision History

Data	Versão	Descrição	Autor	Revisor
23/07/2015	0.1	Creation of template document	Jefferson Silva Barbosa	Jefferson Silva Barbosa
23/07/2015	0.2	Development document introduction	Jefferson Silva Barbosa	
23/07/2015	0.3	Definition application architecture	Jefferson Silva Barbosa	
24/07/2015	0.5	Definition of Architecture Logic Vision and their respective UMLS	Jefferson Silva Barbosa	
27/07/2015	0.8	Definition of the class diagrams in different types of abstraction	Jefferson Silva Barbosa	
19/06/2016	1.0	Document Revision	Jefferson Silva Barbosa	Jefferson Silva Barbosa

Summary

Software Architecture Document

1. Introduction

This document presents the architecture of DyMMer system. The architecture is presented through a set of views that together are intended to cover the main technical aspects of the development and deployment of the system in question. The goal is to capture points of improvement in the system.

This document is organized into topics related to the different architectural views. Section 2 presents an overview of how architecture is currently organized. Section 3 illustrates the relationship between system components in different views of abstraction. The Section 4 shows different diagrams for an implementation view.

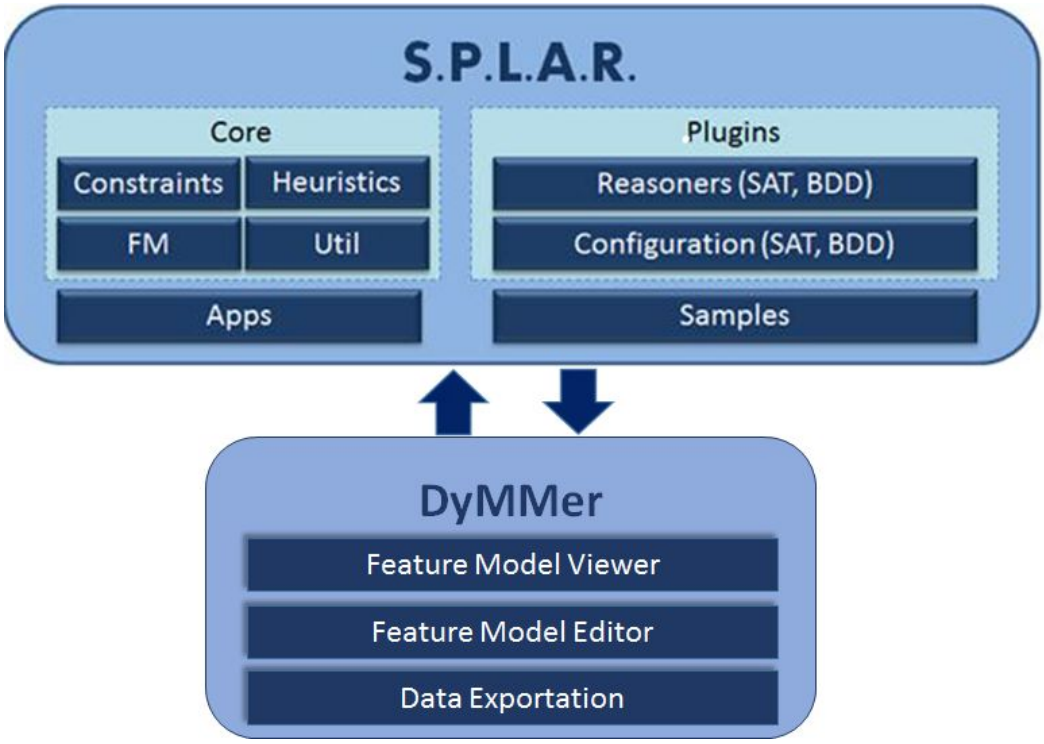
2. Architectural Representation

Currently the components of DyMMer tool are organized similarly to the pattern Model View Controller (MVC). The system is divided into an interface component, a feature models manager, a component to export and import models and another component to edit models.

3. Logical View

This section presents different types of logics views with different abstraction levels to have a visibility of communication between layers, packages and classes.

3.1 Overview DyMMer tool.



Data exportation

DyMMer makes it possible to export in an automatic manner the values of the quality measures for a spreadsheet in the Microsoft Office Excel format. The domain engineer can export all measures at once for one or more features model. This is a great advantage for tool enables analysis of various features models at the same time. This exportation of the measures can also be separated by measures of SPLs and DSPLs.

Feature Model Viewer

DyMMer makes it possible to visualize and analyze a specific feature model, previously imported. In DyMMer tool we can visualize for each context adaptation of the features model the activated and deactivated features according to its constraints. In visualization of the model features can also select individual measures and collect the results of these measurements and display the model visualization screen in the tab measures. Such measures may be features models of traditional SPLs or DSPLs.

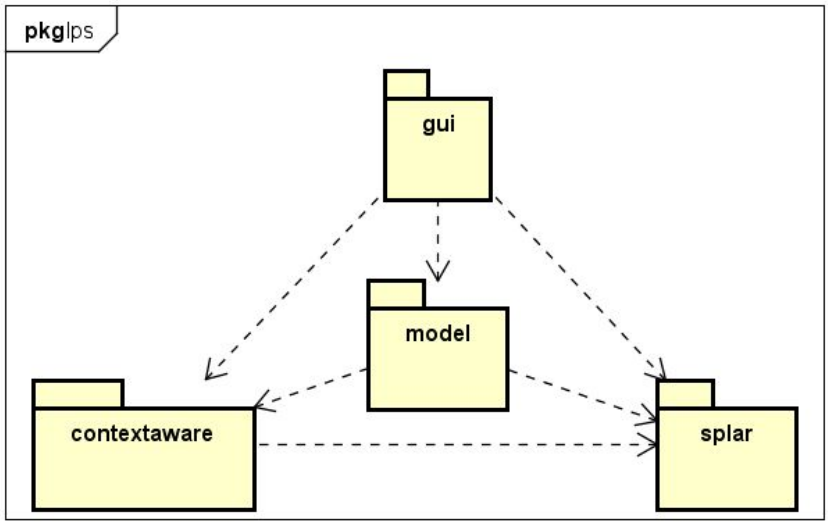
Feature Model Editor

DyMMer makes it possible to edit a specific feature model, adding context adaptations (in feature models without context), adding or removing features, adding or removing constraints, etc. For each context adaptation added the domain engineer must activate or deactivate features and include constraints if necessary. DSPLs features model may have one or more context adaptations. This functionality enables the domain engineer can handle DSPLs features model, since the S.P.L.O.T does not support modeling DSPLs.

S.P.L.A.R

The SPLAR (Software Product Lines Automated Reasoning library) is a library that represents the application core, where from this library is possible to read files that represent the features models.

3.2 View Package



Package GUI

Responsible for receiving user input and display different data according to the received action.

Package Model

Responsible for defining the structure of the feature model. In it are contained subpackages:

Context – Responsible for defining different types of measures according to the context of the models.
Normal – Responsible for defining a interface to models without context.
Xml – Responsible for interpreting XML files that are received as input.

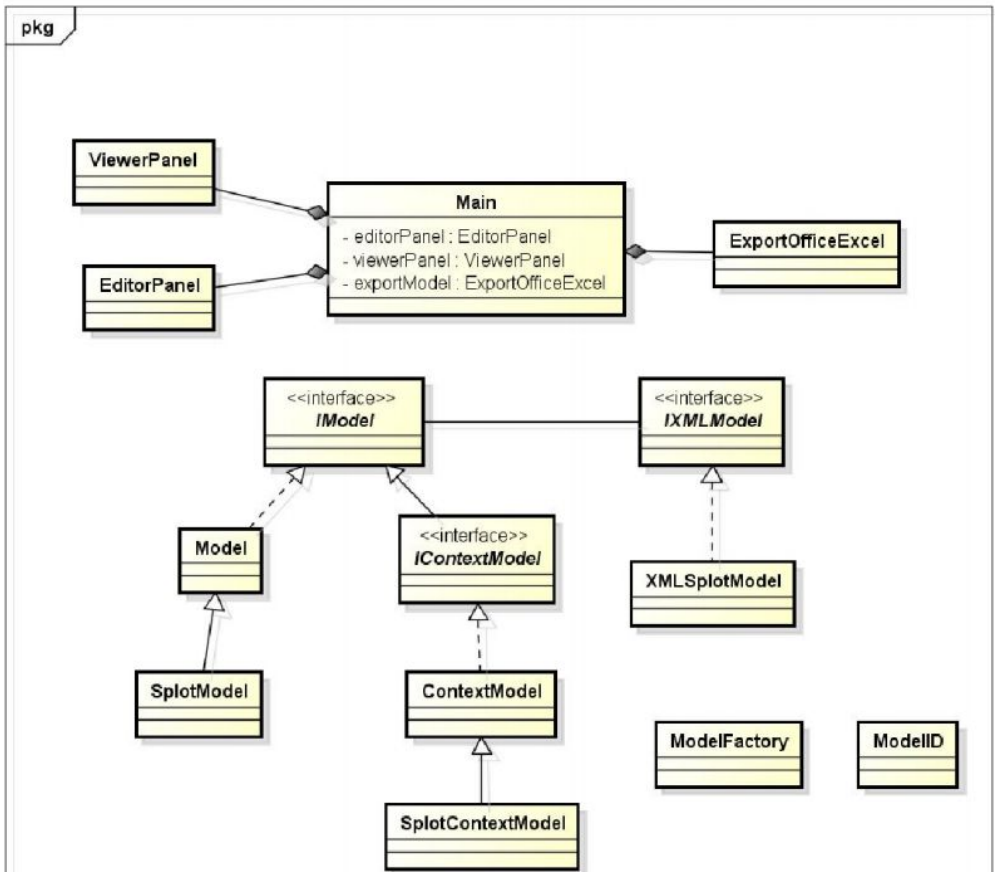
Package ContextAware

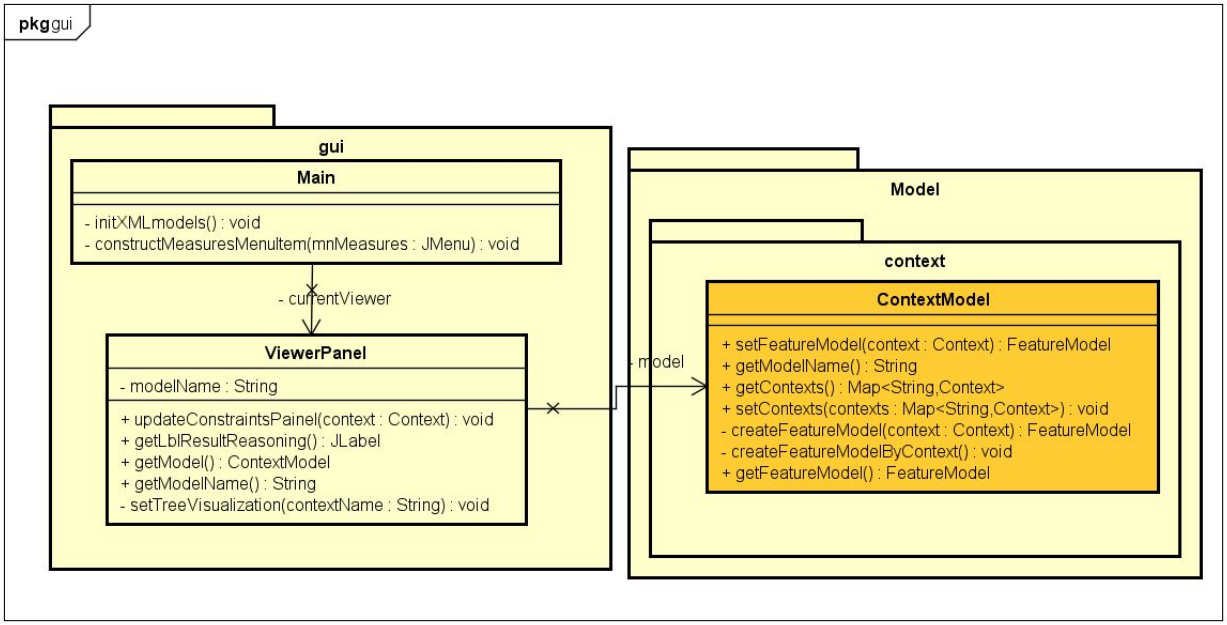
Responsible for managing the states of feature models according to selected context.

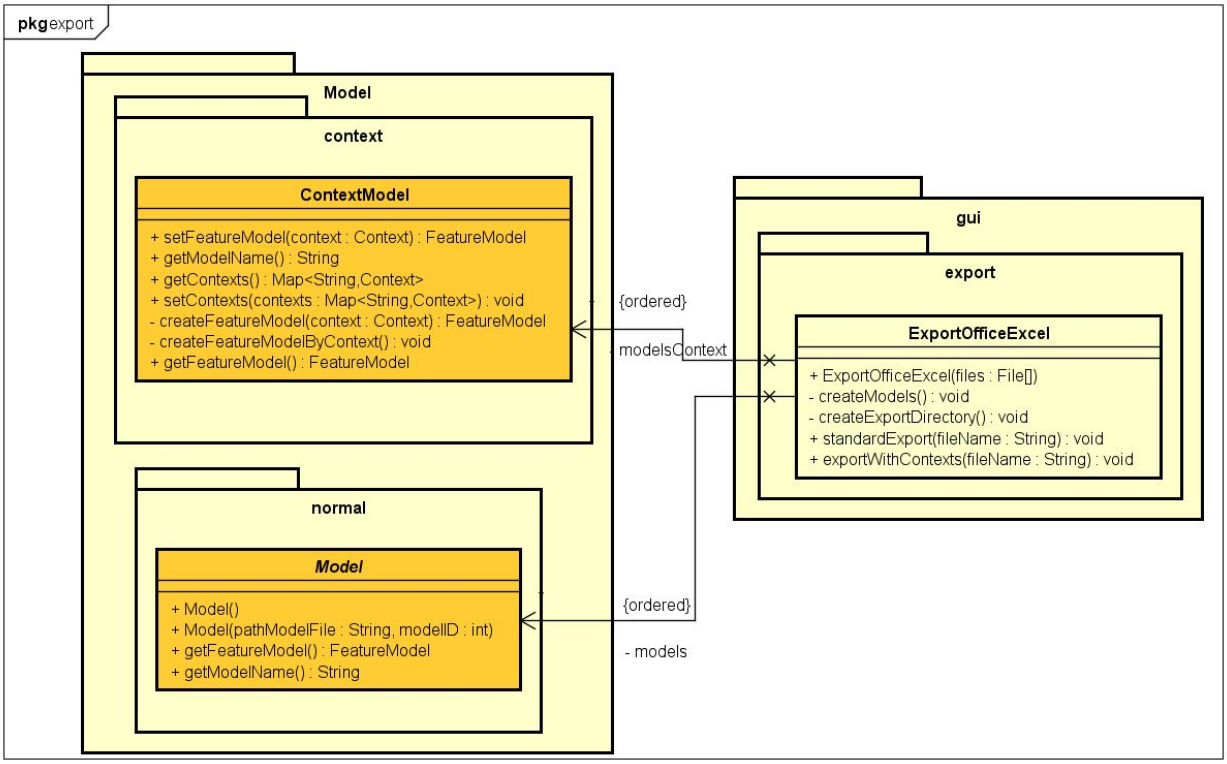
Package Splar

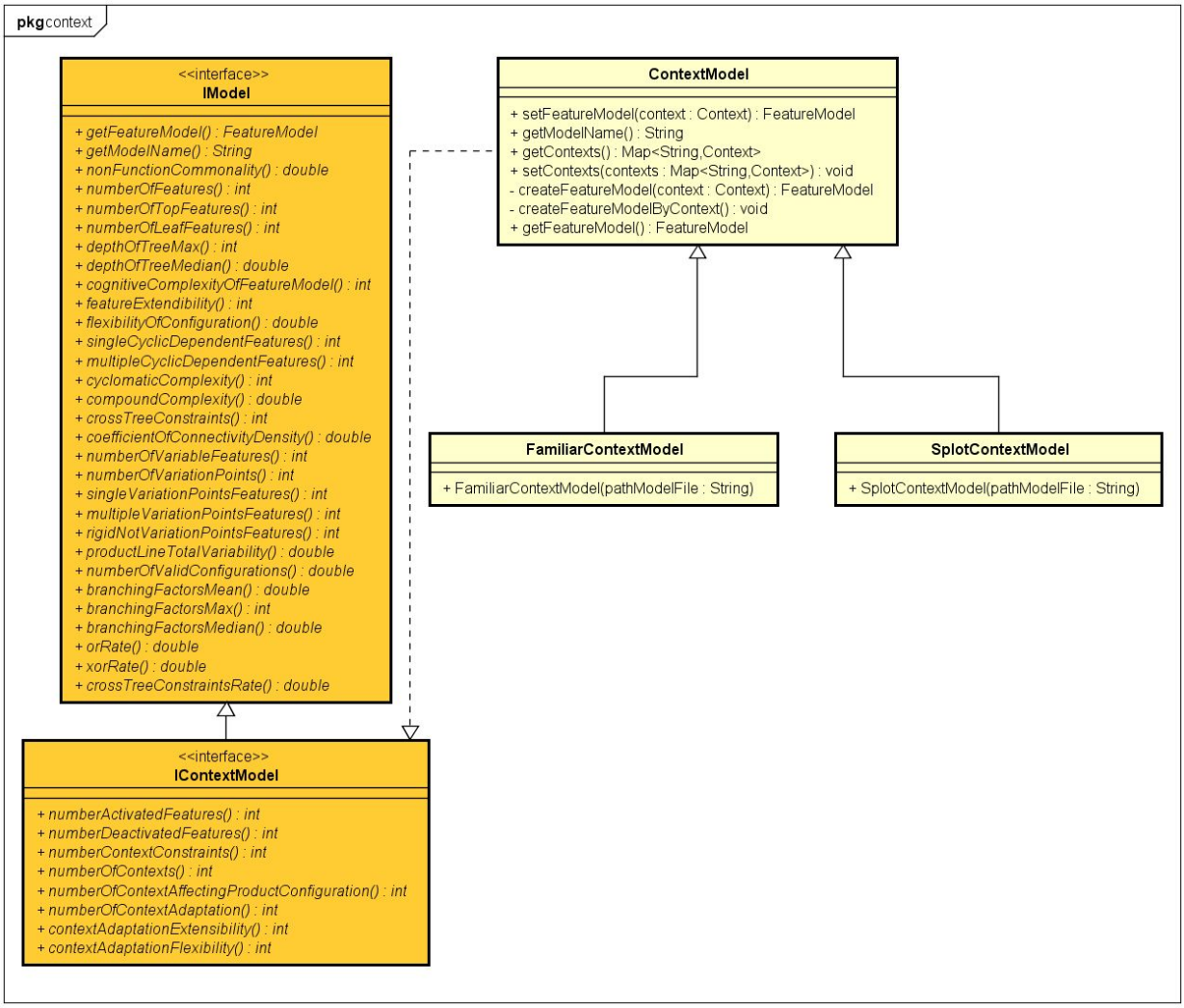
Management operations related to manipulation of the feature model structure.

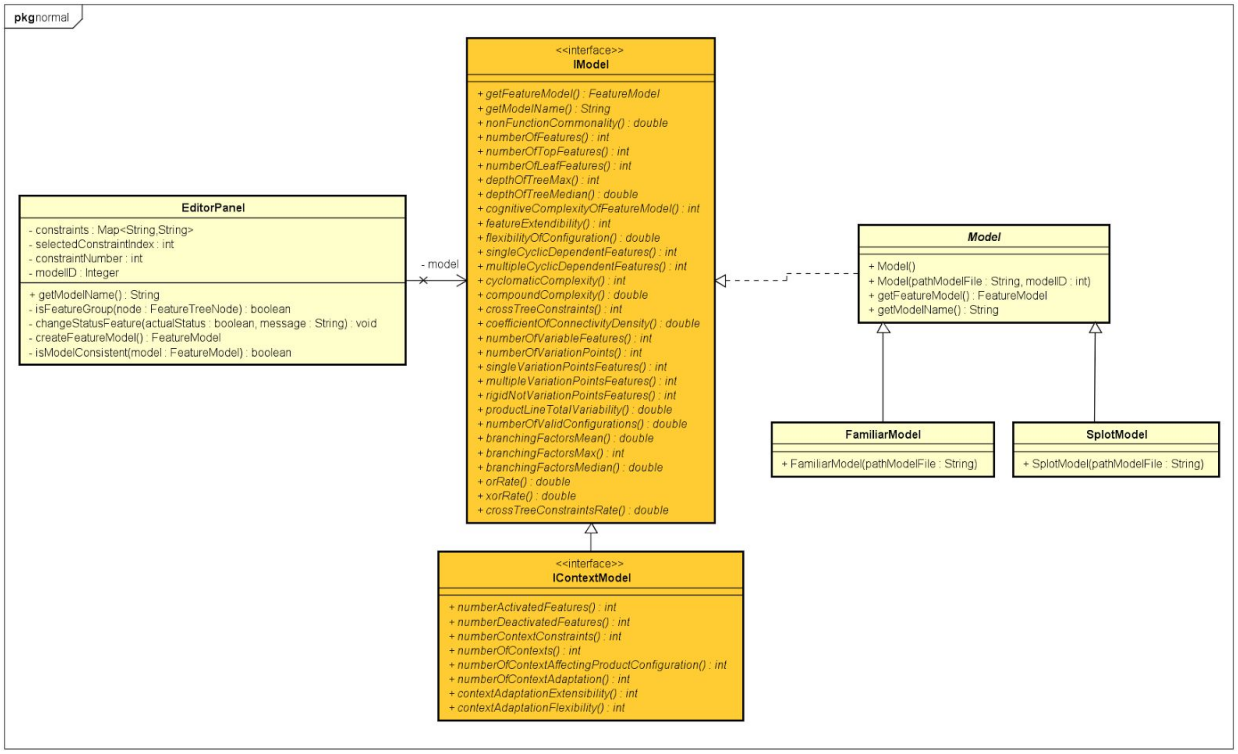
4. Implementation View

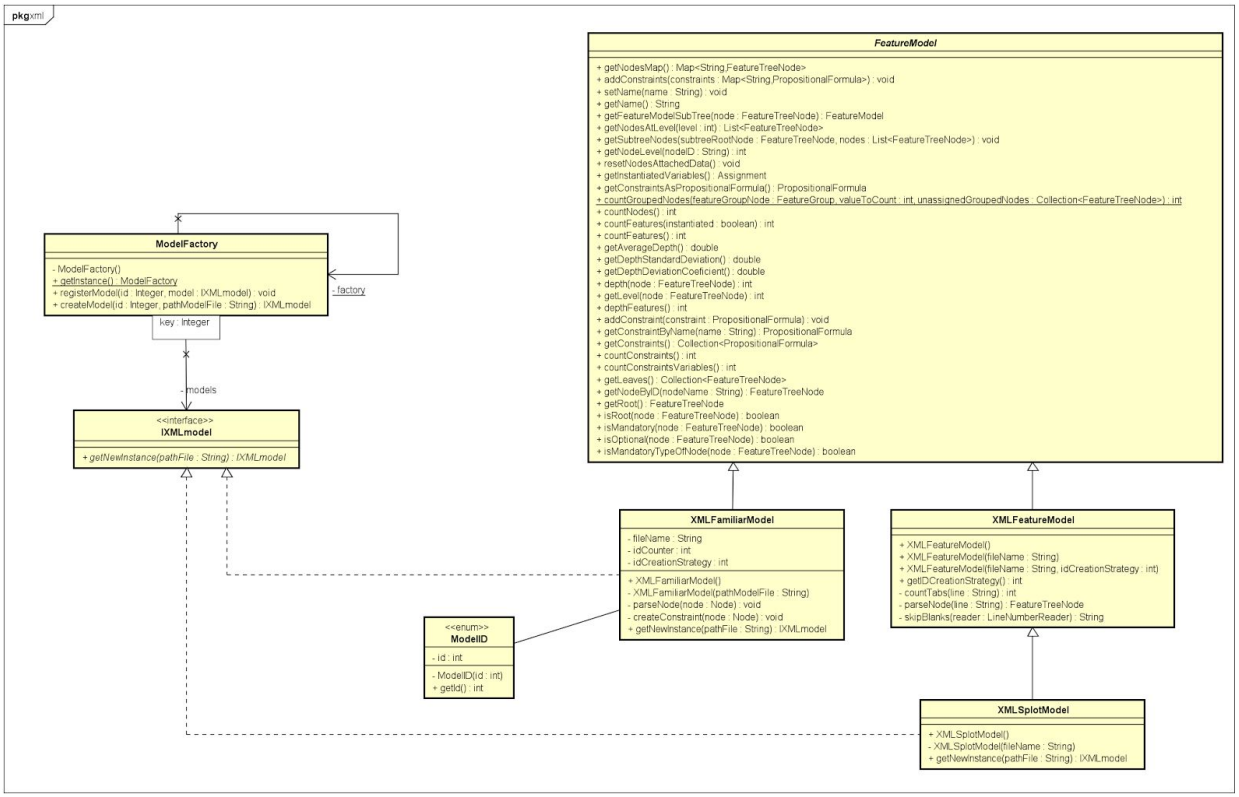


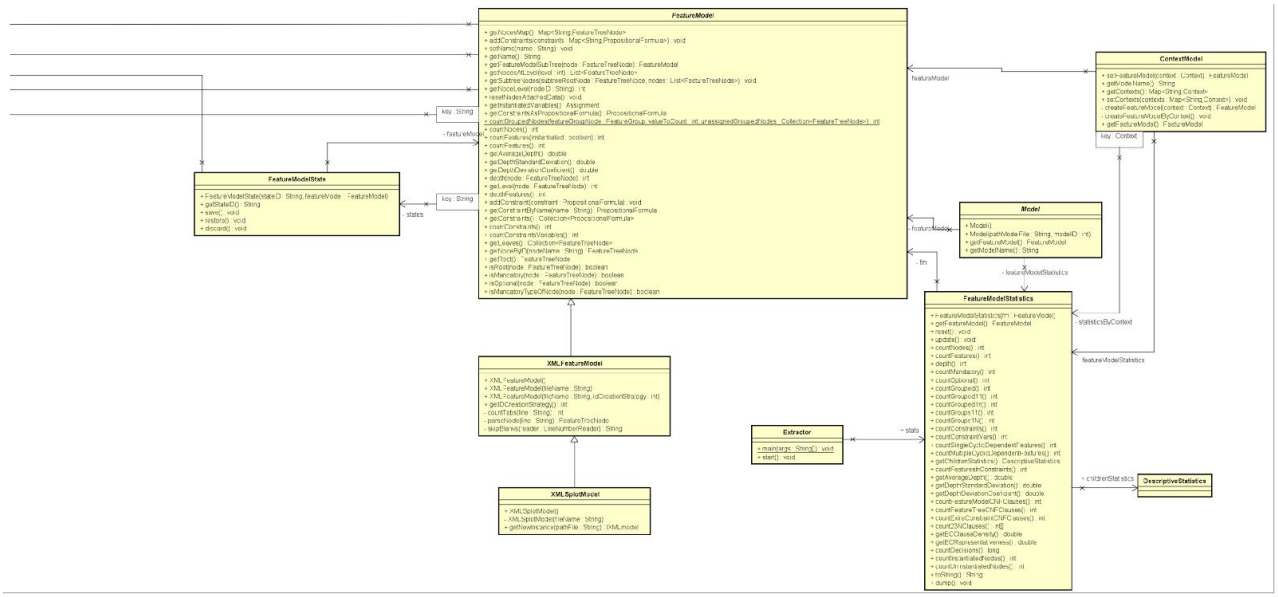

pkg gui
gui
Main
- initXMLmodels() : void
- constructMeasuresMenuItem(mnMeasures : JMenu) : void
- currentViewer
ViewerPanel
- modelName : String
+ updateConstraintsPainel(context : Context) : void
+ getLblResultReasoning() : JLabel
+ getModel() : ContextModel
+ getModelName() : String
- setTreeVisualization(contextName : String) : void
- model
Model
context
ContextModel
+ setFeatureModel(context : Context) : FeatureModel
+ getModelName() : String
+ getContexts() : Map<String,Context>
+ setContexts(contexts : Map<String,Context>) : void
- createFeatureModel(context : Context) : FeatureModel
- createFeatureModelByContext() : void
+ getFeatureModel() : FeatureModel

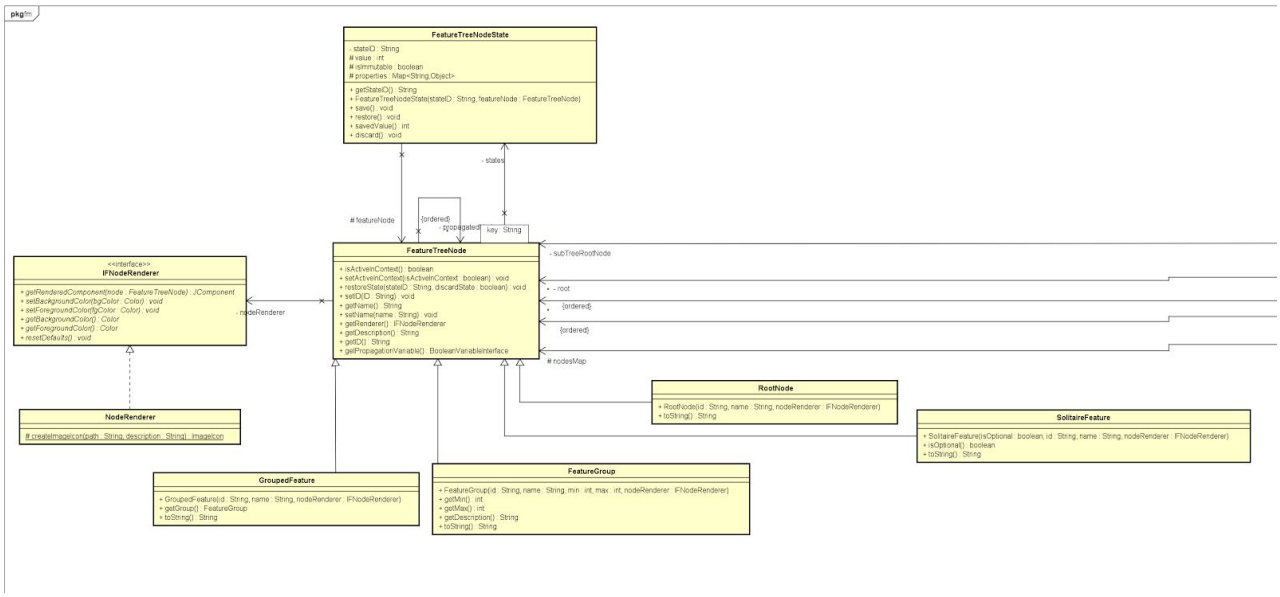

pkgexport
Model
context
ContextModel
+ setFeatureModel(context : Context) : FeatureModel
+ getModelName() : String
+ getContexts() : Map<String,Context>
+ setContexts(contexts : Map<String,Context>) : void
- createFeatureModel(context : Context) : FeatureModel
- createFeatureModelByContext() : void
+ getFeatureModel() : FeatureModel
normal
Model
+ Model()
+ Model(pathModelFile : String, modelID : int)
+ getFeatureModel() : FeatureModel
+ getModelName() : String
{ordered}
- modelsContext
{ordered}
- models
gui
export
ExportOfficeExcel
+ ExportOfficeExcel(files : File[])
- createModels() : void
- createExportDirectory() : void
+ standardExport(fileName : String) : void
+ exportWithContexts(fileName : String) : void

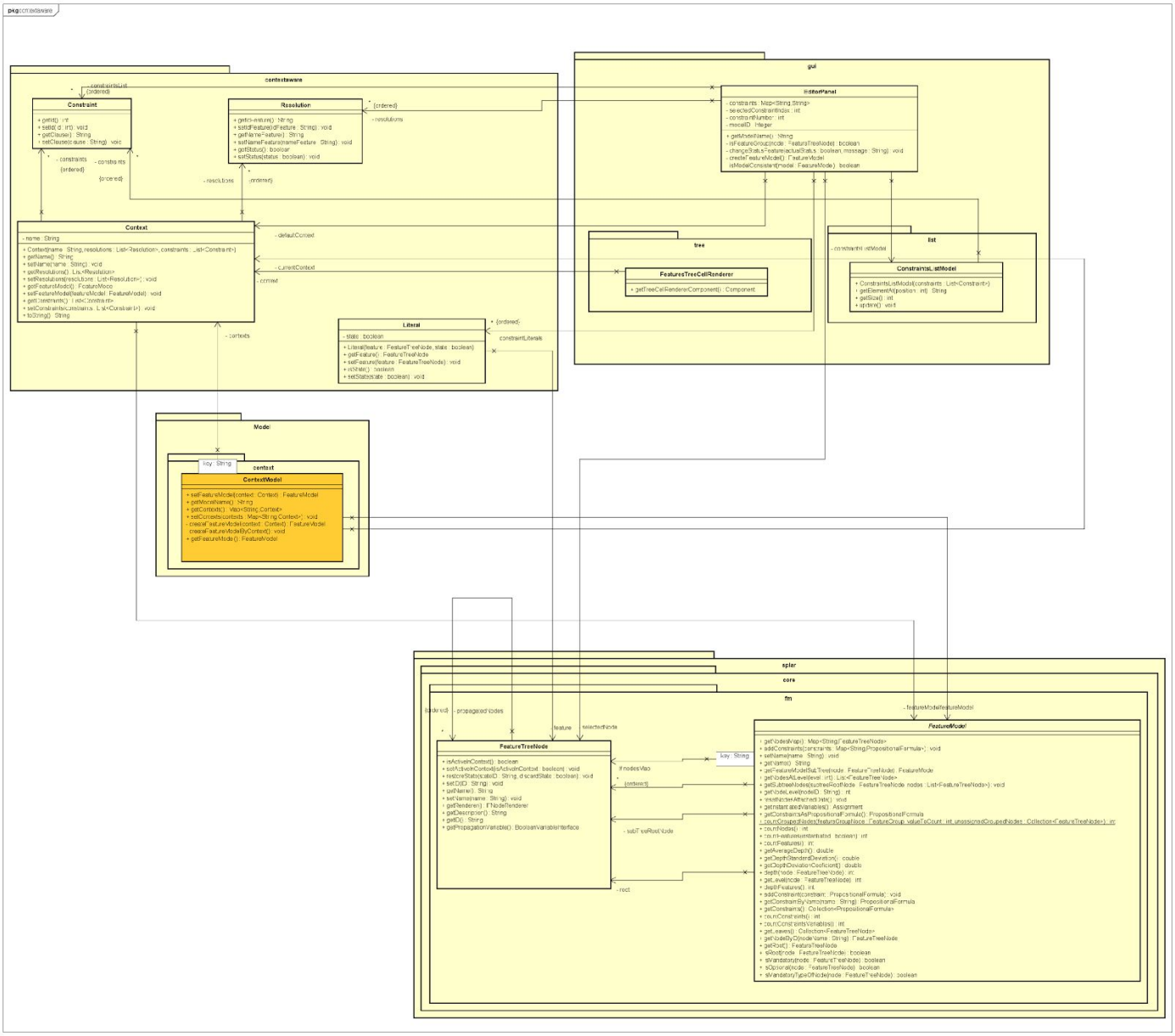

pkg context
<<interface>>
IModel
+ getFeatureModel() : FeatureModel
+ getModelName() : String
+ nonFunctionCommonality() : double
+ numberOfFeatures() : int
+ numberOfTopFeatures() : int
+ numberOfLeafFeatures() : int
+ depthOfTreeMax() : int
+ depthOfTreeMedian() : double
+ cognitiveComplexityOfFeatureModel() : int
+ featureExtendibility() : int
+ flexibilityOfConfiguration() : double
+ singleCyclicDependentFeatures() : int
+ multipleCyclicDependentFeatures() : int
+ cyclomaticComplexity() : int
+ compoundComplexity() : double
+ crossTreeConstraints() : int
+ coefficientOfConnectivityDensity() : double
+ numberOfVariableFeatures() : int
+ numberOfVariationPoints() : int
+ singleVariationPointsFeatures() : int
+ multipleVariationPointsFeatures() : int
+ rigidNotVariationPointsFeatures() : int
+ productLineTotalVariability() : double
+ numberOfValidConfigurations() : double
+ branchingFactorsMean() : double
+ branchingFactorsMax() : int
+ branchingFactorsMedian() : double
+ orRate() : double
+ xorRate() : double
+ crossTreeConstraintsRate() : double
ContextModel
+ setFeatureModel(context : Context) : FeatureModel
+ getModelName() : String
+ getContexts() : Map<String,Context>
+ setContexts(contexts : Map<String,Context>) : void
- createFeatureModel(context : Context) : FeatureModel
- createFeatureModelByContext() : void
+ getFeatureModel() : FeatureModel
FamiliarContextModel
+ FamiliarContextModel(pathModelFile : String)
SplotContextModel
+ SplotContextModel(pathModelFile : String)
<<interface>>
IContextModel
+ numberActivatedFeatures() : int
+ numberDeactivatedFeatures() : int
+ numberContextConstraints() : int
+ numberOfContexts() : int
+ numberOfContextAffectingProductConfiguration() : int
+ numberOfContextAdaptation() : int
+ contextAdaptationExtensibility() : int
+ contextAdaptationFlexibility() : int


pkg normal
<<interface>>
IModel
+ getFeatureModel() : FeatureModel
+ getModelName() : String
+ nonFunctionCommonality() : double
+ numberOfFeatures() : int
+ numberOfTopFeatures() : int
+ numberOfLeafFeatures() : int
+ depthOfTreeMax() : int
+ depthOfTreeMedian() : double
+ cognitiveComplexityOfFeatureModel() : int
+ featureExtendibility() : int
+ flexibilityOfConfiguration() : double
+ singleCyclicDependentFeatures() : int
+ multipleCyclicDependentFeatures() : int
+ cyclomaticComplexity() : int
+ compoundComplexity() : double
+ crossTreeConstraints() : int
+ coefficientOfConnectivityDensity() : double
+ numberOfVariableFeatures() : int
+ numberOfVariationPoints() : int
+ singleVariationPointsFeatures() : int
+ multipleVariationPointsFeatures() : int
+ rigidNotVariationPointsFeatures() : int
+ productLineTotalVariability() : double
+ numberOfValidConfigurations() : double
+ branchingFactorsMean() : double
+ branchingFactorsMax() : int
+ branchingFactorsMedian() : double
+ orRate() : double
+ xorRate() : double
+ crossTreeConstraintsRate() : double
EditorPanel
- constraints : Map<String,String>
- selectedConstraintIndex : int
- constraintNumber : int
- modelID : Integer
+ getModelName() : String
- isFeatureGroup(node : FeatureTreeNode) : boolean
- changeStatusFeature(actualStatus : boolean, message : String) : void
- createFeatureModel() : FeatureModel
- isModelConsistent(model : FeatureModel) : boolean
- model
Model
+ Model()
+ Model(pathModelFile : String, modelID : int)
+ getFeatureModel() : FeatureModel
+ getModelName() : String
FamiliarModel
+ FamiliarModel(pathModelFile : String)
SplotModel
+ SplotModel(pathModelFile : String)
<<interface>>
IContextModel
+ numberActivatedFeatures() : int
+ numberDeactivatedFeatures() : int
+ numberContextConstraints() : int
+ numberOfContexts() : int
+ numberOfContextAffectingProductConfiguration() : int
+ numberOfContextAdaptation() : int
+ contextAdaptationExtensibility() : int
+ contextAdaptationFlexibility() : int


pkg xml
FeatureModel
+ getNodesMap() : Map<String,FeatureTreeNode>
+ addConstraints(constraints : Map<String,PropositionalFormula>) : void
+ setName(name : String) : void
+ getName() : String
+ getFeatureModelSubTree(node : FeatureTreeNode) : FeatureModel
+ getNodesAtLevel(level : int) : List<FeatureTreeNode>
+ getSubtreeNodes(subtreeRootNode : FeatureTreeNode, nodes : List<FeatureTreeNode>) : void
+ getNodeLevel(nodeID : String) : int
+ resetNodesAttachedData() : void
+ getInstantiatedVariables() : Assignment
+ getConstraintsAsPropositionalFormula() : PropositionalFormula
+ countGroupedNodes(featureGroupNode : FeatureGroup, valueToCount : int, unassignedGroupedNodes : Collection<FeatureTreeNode>) : int
+ countNodes() : int
+ countFeatures(instantiated : boolean) : int
+ countFeatures() : int
+ getAverageDepth() : double
+ getDepthStandardDeviation() : double
+ getDepthDeviationCoefficient() : double
+ depth(node : FeatureTreeNode) : int
+ getLevel(node : FeatureTreeNode) : int
+ depthFeatures() : int
+ addConstraint(constraint : PropositionalFormula) : void
+ getConstraintByName(name : String) : PropositionalFormula
+ getConstraints() : Collection<PropositionalFormula>
+ countConstraints() : int
+ countConstraintsVariables() : int
+ getLeaves() : Collection<FeatureTreeNode>
+ getNodeByID(nodeName : String) : FeatureTreeNode
+ getRoot() : FeatureTreeNode
+ isRoot(node : FeatureTreeNode) : boolean
+ isMandatory(node : FeatureTreeNode) : boolean
+ isOptional(node : FeatureTreeNode) : boolean
+ isMandatoryTypeOfNode(node : FeatureTreeNode) : boolean
ModelFactory
- ModelFactory()
+ getInstance() : ModelFactory
+ registerModel(id : Integer, model : IXMLmodel) : void
+ createModel(id : Integer, pathModelFile : String) : IXMLmodel
- factory
key : Integer
- models
<<interface>>
IXMLmodel
+ getNewInstance(pathFile : String) : IXMLmodel
XMLFamiliarModel
- fileName : String
- idCounter : int
- idCreationStrategy : int
+ XMLFamiliarModel()
- XMLFamiliarModel(pathModelFile : String)
- parseNode(node : Node) : void
- createConstraint(node : Node) : void
+ getNewInstance(pathFile : String) : IXMLmodel
XMLFeatureModel
+ XMLFeatureModel()
+ XMLFeatureModel(fileName : String)
+ XMLFeatureModel(fileName : String, idCreationStrategy : int)
+ getIDCreationStrategy() : int
- countTabs(line : String) : int
- parseNode(line : String) : FeatureTreeNode
- skipBlanks(reader : LineNumberReader) : String
<<enum>>
ModelID
- id : int
- ModelID(id : int)
+ getId() : int
XMLSplotModel
+ XMLSplotModel()
- XMLSplotModel(fileName : String)
+ getNewInstance(pathFile : String) : IXMLmodel


FeatureModel
FeatureModelState
XMLFeatureModel
XMLSplotModel
ContextModel
Model
FeatureModelStatistics
Extractor
DescriptiveStatistics


pkg
FeatureTreeNodeState
- stateID : String
value : int
isImmutable : boolean
properties : Map<String,Object>
+ getStateID() : String
+ FeatureTreeNodeState(stateID : String, featureNode : FeatureTreeNode)
+ save() : void
+ restore() : void
+ savedValue() : int
+ discard() : void
- states
featureNode
{ordered}
- propagated
key : String
<<interface>>
IFNodeRenderer
+ getRenderedComponent(node : FeatureTreeNode) : JComponent
+ setBackgroundColor(bgColor : Color) : void
+ setForegroundColor(fgColor : Color) : void
+ getBackgroundColor() : Color
+ getForegroundColor() : Color
+ resetDefaults() : void
- nodeRenderer
FeatureTreeNode
+ isActiveInContext() : boolean
+ setActiveInContext(isActiveInContext : boolean) : void
+ restoreState(stateID : String, discardState : boolean) : void
+ setID(ID : String) : void
+ getName() : String
+ setName(name : String) : void
+ getRenderer() : IFNodeRenderer
+ getDescription() : String
+ getID() : String
+ getPropagationVariable() : BooleanVariableInterface
- subTreeRootNode
- root
{ordered}
{ordered}
nodesMap
NodeRenderer
createImageIcon(path : String, description : String) : ImageIcon
RootNode
+ RootNode(id : String, name : String, nodeRenderer : IFNodeRenderer)
+ toString() : String
SolitaireFeature
+ SolitaireFeature(isOptional : boolean, id : String, name : String, nodeRenderer : IFNodeRenderer)
+ isOptional() : boolean
+ toString() : String
GroupedFeature
+ GroupedFeature(id : String, name : String, nodeRenderer : IFNodeRenderer)
+ getGroup() : FeatureGroup
+ toString() : String
FeatureGroup
+ FeatureGroup(id : String, name : String, min : int, max : int, nodeRenderer : IFNodeRenderer)
+ getMin() : int
+ getMax() : int
+ getDescription() : String
+ toString() : String


pkg[contextaware]
contextaware
Constraint
Resolution
Context
Literal
gui
HistoryPanel
tree
FeaturesTreeCellRenderer
list
ConstraintsListModel
Model
ContextModel
splar
core
fm
FeatureTreeNode
FeatureModel