*Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования*

***«Московский государственный технический университет имени Н.Э. Баумана»   
(МГТУ им. Н.Э. Баумана)***

***Кафедра ИУ3, 3 курс, 6 семестр.***

**Отчёт**

**по лабораторной работе №5**

**“ Расширение Eclipse своими плагинами ”**

**по курсу**

**“Разработка ПО”**

Выполнила: Крайникова А.В.

Группа: ИУ3-62

Проверил: Иванов А. М.

Москва, 2014

Оглавление

[Цель работы 3](#_Toc391374699)

[Задание 3](#_Toc391374700)

[Исходные данные 4](#_Toc391374701)

[Структура проекта 5](#_Toc391374702)

[plugin.xml 5](#_Toc391374703)

[MANIFEST.MF 6](#_Toc391374704)

[Исходный код 6](#_Toc391374705)

[Результат выполнения 8](#_Toc391374706)

# Цель работы

* Углубление навыков работы с системой контроля версий
* Научиться разрабатывать и встраивает свои плагины в eclipse IDE
* Запуск редактора диаграмм Jfxed

# Задание

1. Чтобы познакомиться с технологией выполните указания обучающего материала

Extending the Eclipse IDE - Plug-in development

2. Откройте Eclipse и обновите локальную копию репозитария

* В перспективе Git, надо сделать Pull

3. Переключитесь на стандартную целевую платформу Eclipse

* Настройки Eclipse > Plug-in Development > Target Platform, установить Running Platform

4. Импортируйте в Workspace проекты:

* ru.agentlab.jfxed
* ru.agentlab.jfxed.figures
* ru.agentlab.jfxed.figures.class

4. Запустите редактор

* Конфигурация запуска Eclipse Jfxed

## Исходные данные

* **Диаграмма из лабораторной работы №3** (т. е., описывающий её OWL)

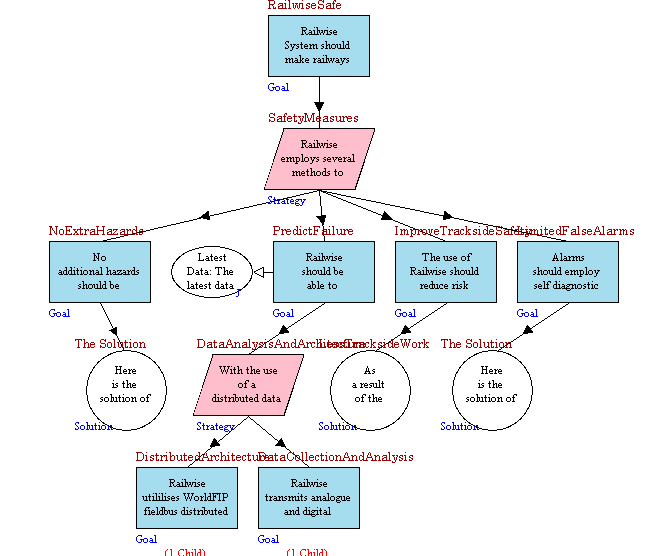


Рис.1 SafetyCaseModelDiagram

* **Фигурка из лабораторной работы №2**

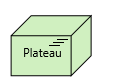


Рис.2 Plateau

# Структура проекта

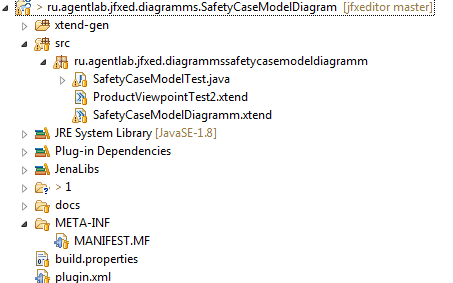


Рис.3

## plugin.xml

<?xml version="1.0" encoding="UTF-8"?>

<?eclipse version="3.0"?>

<plugin>

<extension

point="ru.agentlab.jfxed.diagram">

<diagram name="Nastya`s Diagramm" class="ru.agentlab.jfxed.diagrammssafetycasemodeldiagramm.SafetyCaseModelDiagramm" ontoUri="http://www.agentlab.ru/jfxed/onto/safetycasemodel"/>

</extension>

</plugin>

## MANIFEST.MF

Manifest-Version: 1.0

**Bundle-ManifestVersion**: 2

**Bundle-Name**: SafetyCaseModelDiagramm

**Bundle-SymbolicName**: ru.agentlab.jfxed.diagramms.safetyCaseModelDiagramm;*singleton*:=true

**Bundle-Version**: 1.0.0.qualifier

**Require-Bundle**: org.apache.jena.core,

org.apache.jena.iri,

org.apache.jena.sdb,

org.apache.jena.tdb,

org.slf4j.api,

org.slf4j.ext,

org.slf4j.log4j,

org.apache.xerces;*bundle-version*="2.11.0",

org.eclipse.core.runtime,

org.eclipse.emf.ecore;*visibility*:=reexport,

org.eclipse.xtext.xbase.lib;*visibility*:=reexport,

com.google.guava,

org.eclipse.fx.core,

org.eclipse.fx.javafx,

org.eclipse.fx.osgi.util,

de.fxdiagram.lib,

de.fxdiagram.core,

de.fxdiagram.annotations,

ru.agentlab.jfxed,

ru.agentlab.jfxed.figures.class,

ru.agentlab.jfxed.figures.plateau2,

org.apache.jena.core;*bundle-version*="2.11.0",

org.apache.jena.iri;*bundle-version*="0.9.5"

**Import-Package**: com.hp.hpl.jena.ontology

**Bundle-RequiredExecutionEnvironment**: JavaSE-1.8

**Export-Package**: ru.agentlab.jfxed.diagrammssafetycasemodeldiagramm

# Исходный код

package ru.agentlab.jfxed.diagrammssafetycasemodeldiagramm

import com.hp.hpl.jena.ontology.OntModel

import com.hp.hpl.jena.query.Query

import com.hp.hpl.jena.query.QueryExecution

import com.hp.hpl.jena.query.QueryExecutionFactory

import com.hp.hpl.jena.query.QueryFactory

import com.hp.hpl.jena.query.QuerySolution

import com.hp.hpl.jena.query.ResultSet

import com.hp.hpl.jena.rdf.model.Resource

import de.fxdiagram.core.XDiagram

import de.fxdiagram.core.XNode

import ru.agentlab.jfxed.IDiagram

import ru.agentlab.jfxed.figures.clazz.ClassFigure

import ru.agentlab.jfxed.figures.plateau.Plateau

public class SafetyCaseModelDiagramm implements IDiagram {

static String SOURCE = "http://www.agentlab.ru/jfxed/onto/safetycasemodel"

Query query

QueryExecution qe

ResultSet results

String queryString

QuerySolution soln

Resource x

XNode target

override createJfx(OntModel jenaModel, XDiagram jfxDiagram) {

makeQuery(jenaModel, jfxDiagram, "Goal")

makeQuery(jenaModel, jfxDiagram, "Strategy")

makeQuery(jenaModel, jfxDiagram, "J")

makeQuery(jenaModel, jfxDiagram, "Solution")

}

def makeQuery(OntModel jenaModel, XDiagram jfxDiagram, String className) {

target = null

queryString ='''

PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>

select ?uri

where {

?uri rdf:type <«SOURCE»#«className»>

}

'''

query = QueryFactory.create(queryString)

// Execute the query and obtain results

qe = QueryExecutionFactory.create(query, jenaModel)

results = qe.execSelect()

for ( ; results.hasNext() ; )

{

soln = results.nextSolution()

x = soln.getResource("uri") // Get a result variable by name.

println(x)

jfxDiagram => [

target = new Plateau() => [

layoutX = 280

layoutY = 280

name = x.localName

]

nodes += target

]

}

qe.close()

}

}

# Результат выполнения

