Факультет ИУ "Информатика и системы управления"

Кафедра ИУ-3 "Информационные системы и телекоммуникации"

Отчет к лабораторной работе №3

по курсу "**Разработка программного обеспечения**"

направления 2304000062

" **Работа с плагинами и Apache Jena**"

Продолжительность 4 часа.

Выполнил:

студент группы ИУ3-62

Даварашвили И.Б.

Проверил:

Иванов А.М.

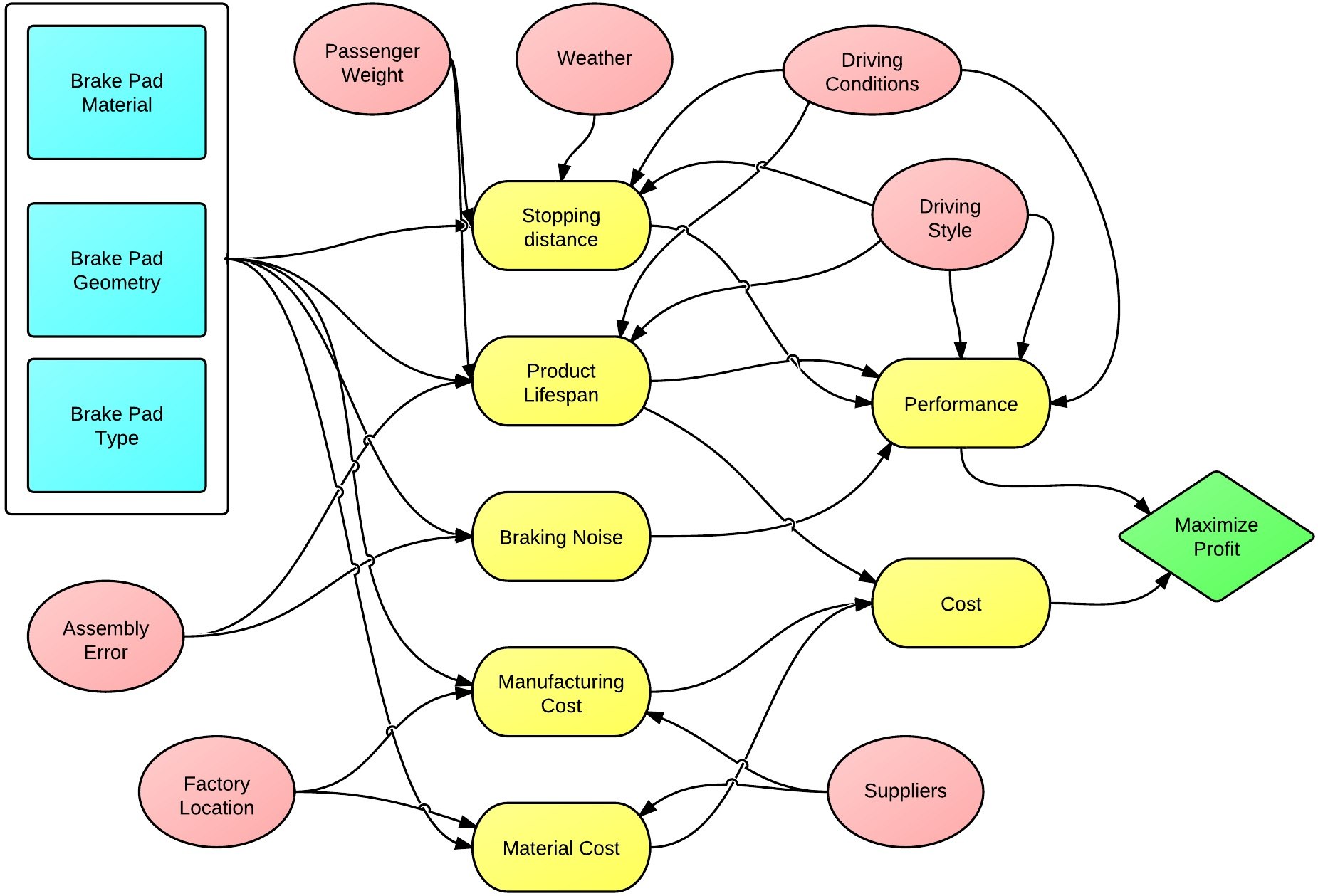
# Цели лабораторной работы

* Углубление навыков работы с системой контроля версий
* Ознакомление на практике с основами графовых БД и технологий Semantic Web

# Задание

Создать тестовый Eclipse-проект, в котором происходит формирование информационной модели данных диаграммы и работа с данными в соответствии со своим вариантом задания и требованиями.

Добавить созданные проект в репозитарий системы контроля версий.



Исходный код программы:

**package** JenaPruebas;

**import** java.io.FileWriter;

**import** java.io.IOException;

**import** com.hp.hpl.jena.ontology.Individual;

**import** com.hp.hpl.jena.ontology.ObjectProperty;

**import** com.hp.hpl.jena.ontology.OntClass;

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

**package ru.agentlab.jfxed.diagramms.influencedia;**

**import java.io.FileWriter;**

**import java.io.IOException;**

**import com.hp.hpl.jena.ontology.Individual;**

**import com.hp.hpl.jena.ontology.ObjectProperty;**

**import com.hp.hpl.jena.ontology.OntClass;**

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

**import com.hp.hpl.jena.rdf.model.ModelFactory;**

**//public class AirlineSecurity**

**public class influencedia {**

**static final String SCHEMA = "http://www.agentlab.ru/jfxed/onto/influencediagram";**

**static final String NS = SCHEMA + "#";**

**public static void main (String[] args) {**

**OntModel m = ModelFactory.createOntologyModel();**

**//super class**

**OntClass concept = m.createClass(NS + "Concept");**

**OntClass relation = m.createClass(NS + "Relation");**

**//type of concept**

**OntClass chanceVariable = m.createClass(NS + "chanceVariable");//pink**

**chanceVariable.addSuperClass(concept);**

**OntClass generalVariable = m.createClass(NS + "generalVariable");//yellow**

**generalVariable.addSuperClass(concept);**

**OntClass decision = m.createClass(NS + "decision");//bol'shoy**

**decision.addSuperClass(concept);**

**OntClass objective = m.createClass(NS + "objective");//green**

**objective.addSuperClass(concept);**

**//type of relation**

**OntClass linkClass = m.createClass(NS + "linkClass");**

**linkClass.addSuperClass(relation);**

**// domain - from; range- kuda какие могут быть связаны**

**ObjectProperty proplink = m.createObjectProperty(NS + "proplink");**

**proplink.addDomain(chanceVariable);**

**proplink.addRange(generalVariable);**

**ObjectProperty proplink1 = m.createObjectProperty(NS + "proplink1");**

**proplink1.addDomain(generalVariable);**

**proplink1.addRange(generalVariable);**

**ObjectProperty proplink2 = m.createObjectProperty(NS + "proplink2");**

**proplink2.addDomain(generalVariable);**

**proplink2.addRange(objective);**

**ObjectProperty proplink3 = m.createObjectProperty(NS + "proplink3");**

**proplink3.addDomain(decision);**

**proplink3.addRange(generalVariable);**

**// Individual имя переменной = m.createIndividual(NS + "имя с картинки", к какому классу типов относится(по цветам));**

**Individual weather = m.createIndividual(NS + "weather", chanceVariable);**

**Individual passWeight = m.createIndividual(NS + "passWeight", chanceVariable);**

**Individual drivingCond = m.createIndividual(NS + "drivingCond", chanceVariable);**

**Individual drivingStyl = m.createIndividual(NS + "drivingStyl", chanceVariable);**

**Individual assemblyError = m.createIndividual(NS + "assemblyError", chanceVariable);**

**Individual factoryLocation = m.createIndividual(NS + "factoryLocation", chanceVariable);**

**Individual suppliers = m.createIndividual(NS + "suppliers", chanceVariable);**

**Individual stoppingDistance = m.createIndividual(NS + "stoppingDistance", generalVariable);**

**Individual productLifespan = m.createIndividual(NS + "productLifespan", generalVariable);**

**Individual brakingNoise = m.createIndividual(NS + "brakingNoise", generalVariable);**

**Individual manufacturingCost = m.createIndividual(NS + "manufacturingCost", generalVariable);**

**Individual materialCost = m.createIndividual(NS + "materialCost", generalVariable);**

**Individual performance = m.createIndividual(NS + "performance", generalVariable);**

**Individual cost = m.createIndividual(NS + "cost", generalVariable);**

**Individual maximizeProfit = m.createIndividual(NS + "maximizeProfit", objective);**

**//Individual brakeMaterial = m.createIndividual(NS + "brakeMaterial", decision);**

**//Individual brakeGeometry = m.createIndividual(NS + "brakeGeometry", decision);**

**//Individual brakeType = m.createIndividual(NS + "brakeType", decision);**

**Individual brakePad = m.createIndividual(NS + "brakePad", decision);**

**/\*Individual имя связи = m.createIndividual(NS + "куда приходит", linkClass);**

**имя связи.addProperty(тип связи из какого в какой, индивидул переменная1 от куда исходит стрелка);**

**имя связи.addProperty(тип связи из какого в какой, индивидул переменная2 от куда исходит стрелка);**

**имя связи.addProperty(тип связи из какого в какой, индивидул переменная3 от куда исходит стрелка);**

**имя связи.addProperty(тип связи из какого в какой, индивидул переменная4 от куда исходит стрелка);**

**имя связи.addProperty(тип связи из какого в какой(мб proplink1,proplink2,proplink3 и пр), индивидул переменная куда исходит стрелка);**

**\*/**

**Individual link1 = m.createIndividual(NS + "stoppingdistance", linkClass);**

**link1.addProperty(proplink, weather);**

**link1.addProperty(proplink, passWeight);**

**link1.addProperty(proplink, drivingCond);**

**link1.addProperty(proplink, drivingStyl);**

**link1.addProperty(proplink, stoppingDistance);**

**link1.addProperty(proplink3, stoppingDistance);**

**link1.addProperty(proplink3, brakePad);**

**Individual link2 = m.createIndividual(NS + "productlifespan", linkClass);**

**link2.addProperty(proplink, productLifespan);**

**link2.addProperty(proplink3, productLifespan);**

**link2.addProperty(proplink, passWeight);**

**link2.addProperty(proplink, drivingCond);**

**link2.addProperty(proplink, assemblyError);**

**link2.addProperty(proplink, drivingStyl);**

**link2.addProperty(proplink3, brakePad);**

**Individual link3 = m.createIndividual(NS + "brakingnoise", linkClass);**

**link3.addProperty(proplink, brakingNoise);**

**link3.addProperty(proplink3, brakingNoise);**

**link3.addProperty(proplink, assemblyError);**

**link3.addProperty(proplink3, brakePad);**

**Individual link4 = m.createIndividual(NS + "manufacturingcost", linkClass);**

**link4.addProperty(proplink, manufacturingCost);**

**link4.addProperty(proplink3, manufacturingCost);**

**link4.addProperty(proplink, factoryLocation);**

**link4.addProperty(proplink, suppliers);**

**link4.addProperty(proplink3, brakePad);**

**Individual link5 = m.createIndividual(NS + "materialcost", linkClass);**

**link5.addProperty(proplink, materialCost);**

**link5.addProperty(proplink3, materialCost);**

**link5.addProperty(proplink, factoryLocation);**

**link5.addProperty(proplink, suppliers);**

**link5.addProperty(proplink3, brakePad);**

**Individual link6 = m.createIndividual(NS + "performance", linkClass);**

**link5.addProperty(proplink1, performance);**

**link5.addProperty(proplink, performance);**

**link6.addProperty(proplink, drivingStyl);**

**link6.addProperty(proplink, drivingCond);**

**link6.addProperty(proplink1, stoppingDistance);**

**link6.addProperty(proplink1, productLifespan);**

**link6.addProperty(proplink1, brakingNoise);**

**Individual link7 = m.createIndividual(NS + "cost", linkClass);**

**link7.addProperty(proplink1, cost);**

**link7.addProperty(proplink1, productLifespan);**

**link7.addProperty(proplink1, manufacturingCost);**

**link7.addProperty(proplink1, materialCost);**

**Individual link8 = m.createIndividual(NS + "maximizeprofit", linkClass);**

**link8.addProperty(proplink2, maximizeProfit);**

**link8.addProperty(proplink2, cost);**

**link8.addProperty(proplink2, performance);**

**//UMLUseCase.run();**

**//создание файла .owl**

**try {**

**m.write(new FileWriter("D:\\infuencediagram.owl"), "RDF/XML");**

**} catch (IOException e) {**

**e.printStackTrace();**

**}**

**}**

**}**