Федеральное государственное бюджетное образовательное учреждение высшего профессионального образования

Московский Государственный Технический Университет имени Н.Э. Баумана

факультет “Информатика и системы управления”

Лабораторная работа №3

Стасенко Н.В.

Группа ИУ 3 - 61

Вариант 14

Проверил:

Иванов А.М.

Москва 2014

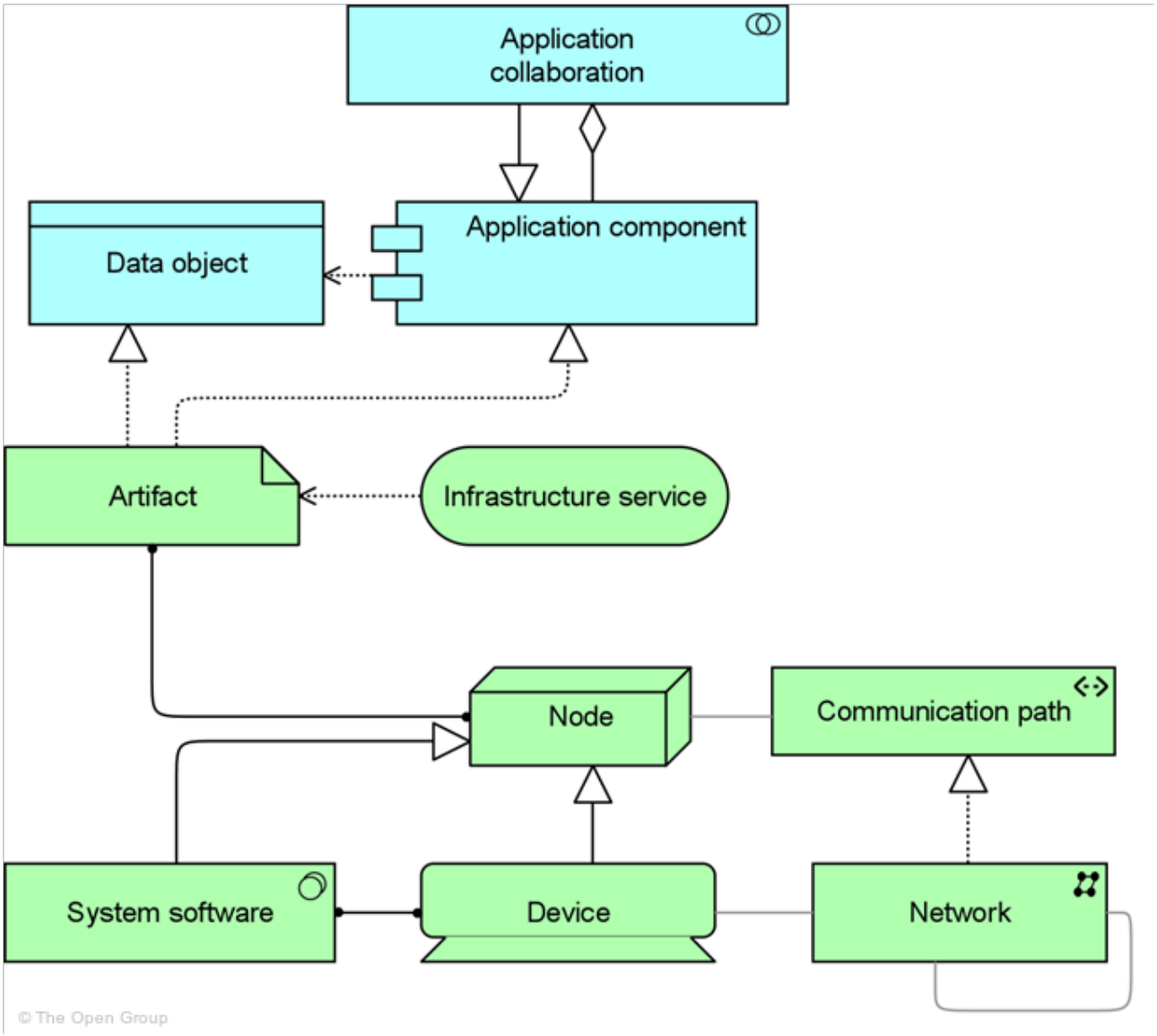
Цели лабораторной работы

- 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.



Листинг программы

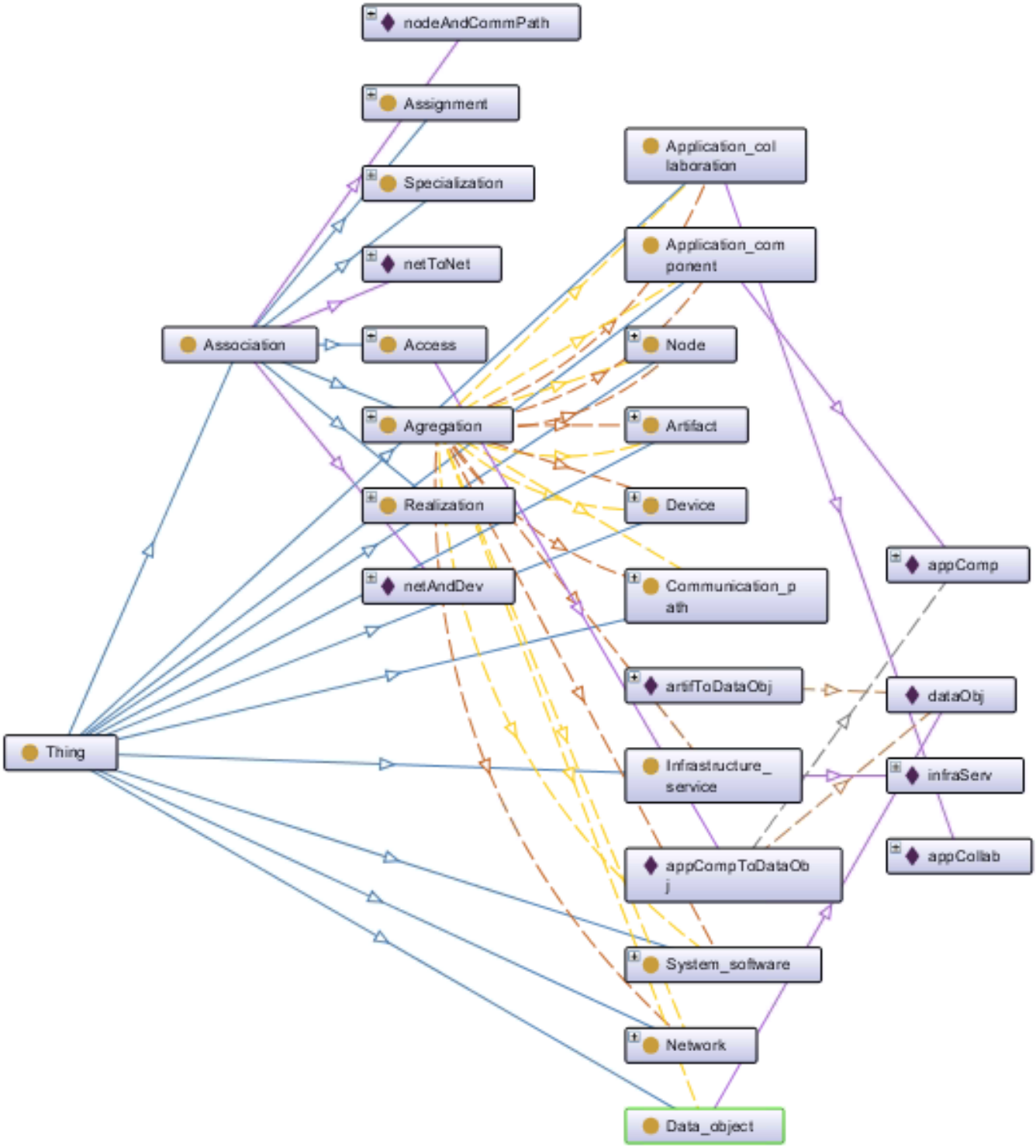
```
package ru.agentlab.jfxed.diagramms.productviewpoint;
import java.io.FileWriter;
import java.io.IOException;

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 ProductViewpointTest {

    static String SOURCE = "http://www.eswc2006.org/technologies/ontology";
    static String NS = SOURCE + "#";

    public static void main (String[] args){

        OntModel m = ModelFactory.createOntologyModel();
        ObjectProperty propTo = m.createObjectProperty(NS + "To");//объекты - свойства
        ObjectProperty propFrom = m.createObjectProperty(NS + "From");

        /*
         * Вершины графа
         * */
        OntClass applicationCollaborationClass = m.createClass(NS + "Application_collaboration");
        OntClass applicationComponentClass = m.createClass(NS + "Application_component");

        OntClass dataObjectClass = m.createClass(NS + "Data_object");
        OntClass artifactClass = m.createClass(NS + "Artifact");
        OntClass infrastructureServiceClass = m.createClass(NS + "Infrastructure_service");
        OntClass nodeClass = m.createClass(NS + "Node");
        OntClass communicationPathClass = m.createClass(NS + "Communication_path");
        OntClass networkClass = m.createClass(NS + "Network");
        OntClass deviceClass = m.createClass(NS + "Device");
        OntClass systemSoftwareClass = m.createClass(NS + "System_software");
        /*
         * end
         * */

        /*
         * Объявление связей
         * */
        OntClass associationClass = m.createClass(NS + "Association");
        OntClass assignmentClass = m.createClass(NS + "Assignment");//линия с точками
        OntClass specializationClass = m.createClass(NS + "Specialization");//связь треугольник,
линия целая
        OntClass realizationClass = m.createClass(NS + "Realization");// связь треугольник,
линия пунктир
        OntClass aggregationClass = m.createClass(NS + "Agregation");//ромб пустой
        OntClass accessClass = m.createClass(NS + "Access");//стрелка, пунктир

        aggregationClass.addSuperClass(associationClass);//родительская связь
        assignmentClass.addSuperClass(associationClass);//родительская связь
```

```
        specializationClass.addSuperClass(associationClass);//родительская связь
        realizationClass.addSuperClass(associationClass);//родительская связь
        accessClass.addSuperClass(associationClass);

        /*
         * end
         * */
        //свойства
        propTo.addDomain(aggregationClass);
        propFrom.addDomain(aggregationClass);

        propFrom.addRange(applicationCollaborationClass);
        propFrom.addRange(applicationComponentClass);
        propFrom.addRange(dataObjectClass);
        propFrom.addRange(artifactClass);
        propFrom.addRange(nodeClass);
        propFrom.addRange(communicationPathClass);
        propFrom.addRange(networkClass);
        propFrom.addRange(deviceClass);
        propFrom.addRange(systemSoftwareClass);

        propTo.addRange(applicationCollaborationClass);
        propTo.addRange(applicationComponentClass);
        propTo.addRange(artifactClass);
        propTo.addRange(deviceClass);
        propTo.addRange(infrastructureServiceClass);
        propTo.addRange(networkClass);
        propTo.addRange(nodeClass);
        propTo.addRange(communicationPathClass);
        propTo.addRange(systemSoftwareClass);


        //создание экземпляров
        Individual applicationCollaborationIndividual = m.createIndividual(NS + "appCollab",
applicationCollaborationClass);
        Individual applicationComponentIndividual = m.createIndividual(NS + "appComp",
applicationComponentClass);
        Individual dataObjectIndividual = m.createIndividual(NS + "dataObj", dataObjectClass);
        Individual artifactIndividual = m.createIndividual(NS + "artif", artifactClass);
        Individual infrastructureServiceIndividual = m.createIndividual(NS +
"infraServ",infrastructureServiceClass);
        Individual nodeIndividual = m.createIndividual(NS + "nod",nodeClass);
        Individual communicationPathIndividual = m.createIndividual(NS +
"commun",communicationPathClass);
        Individual networkIndividual = m.createIndividual(NS + "net",networkClass);
        Individual deviceIndividual = m.createIndividual(NS + "dev",deviceClass);
        Individual systemSoftwareIndividual = m.createIndividual(NS +
"sysSoft",systemSoftwareClass);

        //создание связей
        Individual appCollabToAppComp = m.createIndividual(NS + "appCollabToAppComp",
specializationClass);
        appCollabToAppComp.addProperty(propTo, applicationCollaborationIndividual);
        appCollabToAppComp.addProperty(propFrom, applicationComponentIndividual);
```

```
        Individual appCompToAppCollab = m.createIndividual(NS + "appCompToAppCollab",
aggregationClass);
        appCompToAppCollab.addProperty(propTo, applicationComponentIndividual);
        appCompToAppCollab.addProperty(propFrom, applicationCollaborationIndividual);

        Individual appCompToDataObj = m.createIndividual(NS + "appCompToDataObj",
accessClass);
        appCompToDataObj.addProperty(propTo, applicationComponentIndividual);
        appCompToDataObj.addProperty(propFrom, dataObjectIndividual);

        Individual artifToAppComp = m.createIndividual(NS + "artifToAppComp", realizationClass);
        artifToAppComp.addProperty(propTo, artifactIndividual);
        artifToAppComp.addProperty(propFrom, applicationComponentIndividual);

        Individual artifToDataObj = m.createIndividual(NS + "artifToDataObj", realizationClass);
        artifToDataObj.addProperty(propTo, artifactIndividual);
        artifToDataObj.addProperty(propFrom, dataObjectIndividual);

        Individual infraServToArtif = m.createIndividual(NS + "infraServToArtif", accessClass);
        infraServToArtif.addProperty(propTo, infrastructureServiceIndividual);
        infraServToArtif.addProperty(propFrom, artifactIndividual);

        Individual ArtifAndNode = m.createIndividual(NS + "ArtifAndNode", assignmentClass);
        ArtifAndNode.addProperty(propTo, artifactIndividual);
        ArtifAndNode.addProperty(propFrom, artifactIndividual);
        ArtifAndNode.addProperty(propTo, nodeIndividual);
        ArtifAndNode.addProperty(propFrom, nodeIndividual);

        Individual nodeAndCommPath = m.createIndividual(NS + "nodeAndCommPath",
associationClass);
        nodeAndCommPath.addProperty(propTo, communicationPathIndividual);
        nodeAndCommPath.addProperty(propFrom, communicationPathIndividual);
        nodeAndCommPath.addProperty(propTo, nodeIndividual);
        nodeAndCommPath.addProperty(propFrom, nodeIndividual);

        Individual netToCommPath = m.createIndividual(NS + "netToCommPath", realizationClass);
        netToCommPath.addProperty(propTo, networkIndividual);
        netToCommPath.addProperty(propFrom, communicationPathIndividual);

        Individual netAndNet = m.createIndividual(NS + "netToNet", associationClass);
        netAndNet.addProperty(propTo, networkIndividual);
        netAndNet.addProperty(propFrom, networkIndividual);

        Individual netAndDev = m.createIndividual(NS + "netAndDev", associationClass);
        netAndDev.addProperty(propTo, networkIndividual);
        netAndDev.addProperty(propFrom, networkIndividual);
        netAndDev.addProperty(propTo, deviceIndividual);
        netAndDev.addProperty(propFrom, deviceIndividual);

        Individual devToNode = m.createIndividual(NS + "devToNode", specializationClass);
        devToNode.addProperty(propTo, deviceIndividual);
        devToNode.addProperty(propFrom, nodeIndividual);

        Individual devAndSysSoft = m.createIndividual(NS + "devAndSysSoft", assignmentClass);
        devAndSysSoft.addProperty(propTo, systemSoftwareIndividual);
        devAndSysSoft.addProperty(propFrom, systemSoftwareIndividual);
```

```
        devAndSysSoft.addProperty(propTo, deviceIndividual);
        devAndSysSoft.addProperty(propFrom, deviceIndividual);

        Individual sysSoftToNode = m.createIndividual(NS + "sysSoftToNode", specializationClass);
        sysSoftToNode.addProperty(propTo, nodeIndividual);
        sysSoftToNode.addProperty(propFrom, systemSoftwareIndividual);

        //m.write(System.out);//и в консоль
        try {
            m.write(new FileWriter("Smosia.owl"), "RDF/XML");
        } catch (IOException e) {
            e.printStackTrace();
        }
        }
}
```

Граф


nodeAndCommPath
Assignment
Specialization
netToNet
Association
Access
Agregation
Realization
netAndDev
Thing
Application_collaboration
Application_component
Node
Artifact
Device
Communication_path
artifToDataObj
Infrastructure_service
appCompToDataObj
System_software
Network
Data_object
appComp
dataObj
infraServ
appCollab