Министерство науки и высшего образования Российской Федерации

ФГБОУ ВО Ульяновский государственный технический университет

Кафедра «Вычислительная техника»

Лабораторная работа №3

«Построение онтологической модели топологии компьютерной сети»

Вариант №8

Выполнил студент

группы ИВТИИбд-11

Даниленко М.В.

Преподаватель

Хайрулин И.Д.

Ульяновск, 2025

\*\*1. Структурное описание онтологической модели\*\*

Онтологическая модель описывает топологию корпоративной компьютерной сети и формализует ключевые сетевые устройства и их взаимосвязи. Основные сущности (далее классы) модели включают:

- \*\*Сетевые устройства (NetworkDevice):\*\* базовый класс для всех устройств в сети

- \*\*Серверы (Server):\*\* устройства, предоставляющие различные сетевые услуги

- \*\*Компьютеры (Computer):\*\* рабочие станции пользователей

- \*\*Маршрутизаторы (Router):\*\* устройства для межсетевого взаимодействия

- \*\*Коммутаторы (Switch):\*\* устройства для соединения устройств в сети

- \*\*Межсетевые экраны (Firewall):\*\* устройства для обеспечения безопасности

- \*\*Сети (Network):\*\* сетевые сегменты и подсети

- \*\*Отделы (Department):\*\* организационные единицы компании

Для реализации модели использованы два инструмента:

1. \*\*Protégé\*\* --- применяется для построения OWL-онтологии, добавления аннотаций и формулировки логических правил с использованием SWRL.

2. \*\*Neo4j\*\* --- используется для создания графовой модели с узлами и связями, где логические зависимости реализуются с помощью языка запросов Cypher.

\*\*2. Перечень классов, отношений и аксиом\*\*

Онтологическая модель включает несколько типов элементов: классы (сущности), свойства данных и свойства объектов, а также логические правила (аксиомы), описывающие взаимосвязи между объектами.

\*\*2.1 Классы (Classes)\*\*

Модель содержит следующие основные классы:

- \*\*NetworkDevice (Сетевое устройство):\*\* базовый класс для всех устройств в сети

- \*\*Server (Сервер):\*\* устройства, предоставляющие сетевые услуги

- \*\*Computer (Компьютер):\*\* рабочие станции пользователей

- \*\*Router (Маршрутизатор):\*\* устройства для маршрутизации между сетями

- \*\*Switch (Коммутатор):\*\* устройства для соединения устройств в сети

- \*\*Firewall (Межсетевой экран):\*\* устройства для обеспечения безопасности

- \*\*Network (Сеть):\*\* сетевые сегменты и подсети

- \*\*Department (Отдел):\*\* организационные единицы компании

Каждый класс является фундаментальной единицей модели и служит основой для определения свойств и отношений.

\*\*2.2 Свойства данных (Data Properties)\*\*

Свойства данных описывают характеристики отдельных объектов (индивидуумов) классов:

| \*\*Свойство\*\* | \*\*Принадлежит классу\*\* | \*\*Описание / Комментарий\*\* |

|---------------------|------------------------|----------------------------|

| hasName | NetworkDevice | Имя устройства |

| hasIP | NetworkDevice | IP-адрес устройства |

| hasOS | Server, Computer | Операционная система |

| hasSubnet | Network | Подсеть |

| hasLocation | Department | Местоположение отдела |

| hasVLAN | Network | VLAN идентификатор |

Эти свойства позволяют хранить конкретные данные для каждого объекта и использовать их при поиске и логическом выводе.

\*\*2.3 Свойства объектов (Object Properties)\*\*

Свойства объектов определяют связи между различными объектами модели:

| \*\*Свойство\*\* | \*\*Описание\*\* |

|----------------|---------------------------------------------------------|

| connectedTo | Устройство подключено к сети или другому устройству |

| belongsTo | Устройство принадлежит отделу |

| uses | Устройство использует сервис или ресурс |

| routesThrough | Трафик проходит через устройство |

| dependsOn | Устройство зависит от другого устройства |

| manages | Устройство управляет другим устройством |

Эти свойства обеспечивают \*\*логическую структуру модели\*\* и позволяют выводить новые связи между объектами.

\*\*2.4 Аксиомы (SWRL-правила) для Protégé\*\*

Аксиомы формализуют правила логического вывода в модели OWL:

1. \*\*Связь компьютера с сервером через сеть:\*\*

Computer(?c) ^ Server(?s) ^ Network(?n) ^ connectedTo(?c, ?n) ^ connectedTo(?s, ?n) → canAccess(?c, ?s)

Если компьютер и сервер подключены к одной сети, то компьютер может получить доступ к серверу.

2. \*\*Определение сетевого маршрута:\*\*

Computer(?c) ^ Router(?r) ^ Network(?n) ^ connectedTo(?c, ?n) ^ connectedTo(?r, ?n) → routesThrough(?c, ?r)

Если компьютер и маршрутизатор подключены к одной сети, то трафик компьютера проходит через маршрутизатор.

\*\*2.5 Аналог аксиом для Neo4j (Cypher-запросы)\*\*

Для графовой модели логические выводы реализуются с помощью запросов:

1. \*\*Связь компьютеров с серверами через сеть:\*\*

MATCH (c:Computer)-[:CONNECTED\_TO]->(n:Network)<-[:CONNECTED\_TO]-(s:Server)

MERGE (c)-[:CAN\_ACCESS]->(s);

2. \*\*Связь отделов с используемыми серверами:\*\*

MATCH (d:Department)<-[:BELONGS\_TO]-(c:Computer)-[:CAN\_ACCESS]->(s:Server)

MERGE (d)-[:USES\_SERVER]->(s);

3. \*\*Автоматическое определение сетевых маршрутов:\*\*

MATCH (c:Computer)-[:CONNECTED\_TO]->(net:Network)<-[:CONNECTED\_TO]-(r:Router)

MERGE (c)-[:ROUTES\_THROUGH]->(r);

Эти запросы автоматически создают новые связи в графовой базе, аналогично SWRL-правилам в Protégé, что позволяет выполнять логический вывод и анализ сетевой топологии.

\*\*3. Реализация в Protégé\*\*

Для построения онтологической модели использовался инструмент \*\*Protégé\*\*, который позволяет создавать OWL-онтологии, задавать свойства и правила, а также выполнять логический вывод.

- Отображает иерархию всех классов модели: NetworkDevice, Server, Computer, Router, Switch, Firewall, Network, Department

- Класс NetworkDevice является базовым для всех сетевых устройств

- Видно, как классы структурированы и какие объекты могут быть связаны между собой

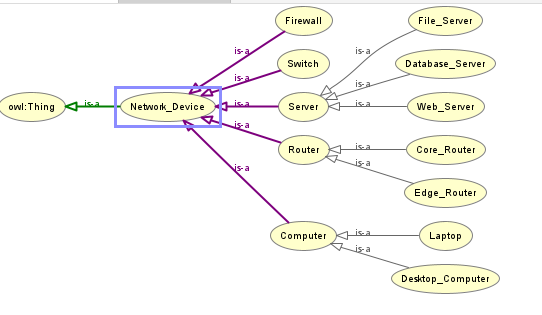


Рисунок 1 -- Дерево классов

\*\*Рисунок 2 -- Object Properties\*\*

Показывает все свойства объектов и их направление, включая:

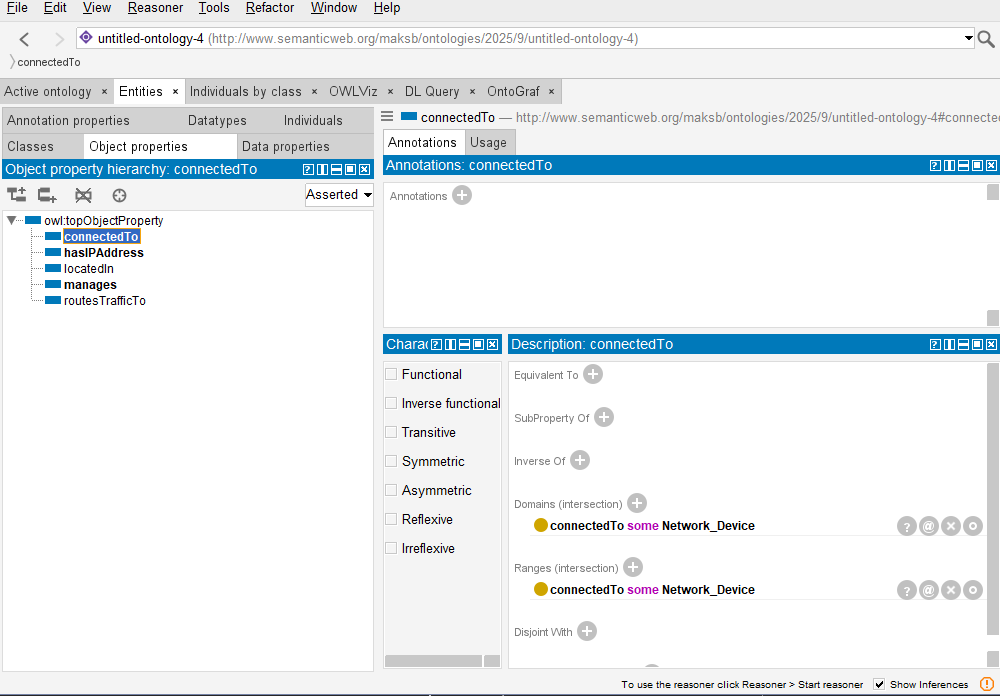
- connectedTo

- hasIPAddress

- locatedin

- manages

- routesTrafficTo



\*\*Рисунок 4 -- SWRLTab с правилами\*\*

Показывает правила логического вывода, позволяющие автоматически создавать новые связи:

- Компьютер ↔ сервер

- Компьютер ↔ маршрутизатор

Благодаря этим правилам Protégé автоматически выводит логические отношения, которые не были явно заданы.

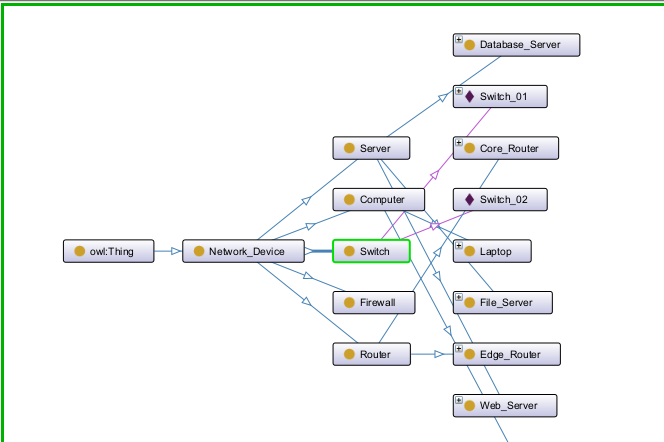


Рисунок 4 -- SWRLtab с правилами

\*\*4. Реализация в Neo4j\*\*

Для построения графовой модели использовалась \*\*Neo4j\*\*, позволяющая хранить данные в виде узлов и связей, а также выполнять логические запросы через Cypher.

Создание узлов и связей (Cypher):

CREATE

// Создание сетей

(mainNetwork:Network {name: "Main Corporate Network", subnet: "192.168.1.0/24", vlan: "10"}),

(serverNetwork:Network {name: "Server Network", subnet: "10.0.1.0/24", vlan: "20"}),

// Создание отделов

(it:Department {name: "IT Department", location: "Floor 3", head: "John Smith"}),

(sales:Department {name: "Sales Department", location: "Floor 2", head: "Maria Ivanova"}),

// Создание серверов

(web1:WebServer {name: "WEB-SRV-01", ip: "10.0.1.10", os: "Ubuntu 20.04", status: "Online"}),

(db1:DatabaseServer {name: "DB-SRV-01", ip: "10.0.1.20", os: "CentOS 8", status: "Online"}),

// Создание компьютеров

(pc1:Computer {name: "PC-IT-01", ip: "192.168.1.10", user: "john.smith", os: "Windows 11"}),

(pc2:Computer {name: "PC-SALES-01", ip: "192.168.1.20", user: "maria.ivanova", os: "Windows 10"}),

// Создание сетевого оборудования

(router:Router {name: "ROUTER-MAIN", ip: "192.168.1.1", model: "Cisco 2901"}),

(switch:Switch {name: "SW-CORE-01", ip: "192.168.1.2", model: "Cisco Catalyst 3560"}),

// Связи с сетями

(pc1)-[:CONNECTED\_TO {interface: "eth0"}]->(mainNetwork),

(pc2)-[:CONNECTED\_TO {interface: "eth0"}]->(mainNetwork),

(web1)-[:CONNECTED\_TO {interface: "eth0"}]->(serverNetwork),

(db1)-[:CONNECTED\_TO {interface: "eth0"}]->(serverNetwork),

(router)-[:CONNECTED\_TO {interface: "GigabitEthernet0/0"}]->(mainNetwork),

// Связи с отделами

(pc1)-[:BELONGS\_TO]->(it),

(pc2)-[:BELONGS\_TO]->(sales),

// Физические соединения

(router)-[:CONNECTED\_TO {cable: "Fiber", port: "GigabitEthernet0/1"}]->(switch);

- \*\*Сети:\*\* Main Corporate Network, Server Network - основные сетевые сегменты

- \*\*Серверы, компьютеры, сетевое оборудование:\*\* создаются как отдельные узлы с характеристиками

- \*\*Связи:\*\* CONNECTED\_TO, BELONGS\_TO отражают реальные отношения между объектами

- Модель позволяет выполнять дальнейший анализ и логический вывод через Cypher-запросы

\*\*Рисунок 5 -- Демонстрация связей\*\*

- Визуализация графа показывает все узлы и их связи

- Легко видеть, какие устройства подключены к каким сетям, какие отделы используют какие устройства

- Сетевые сегменты обеспечивают наглядность и удобство анализа структуры модели

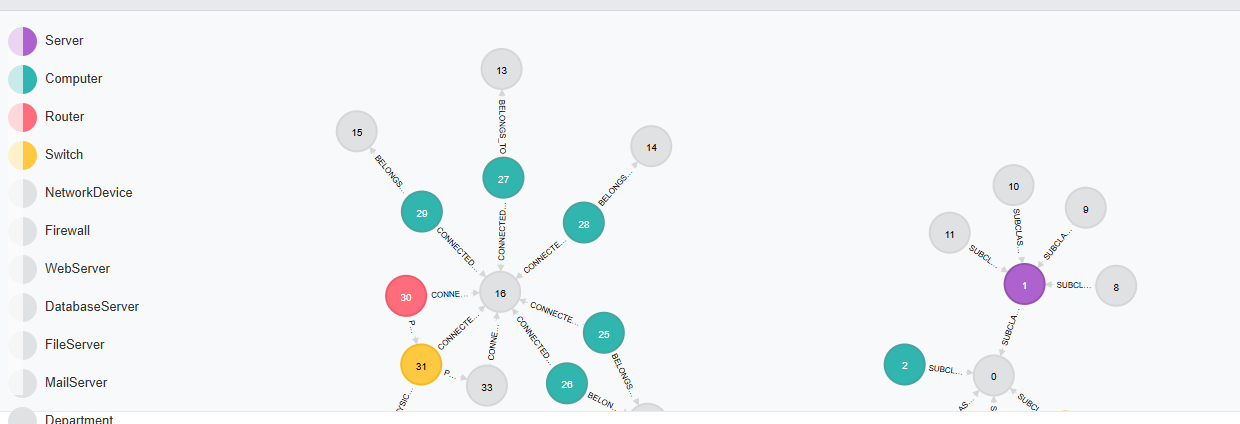


Рисунок 5 -- Демонстрация связей

\*\*5. Вывод\*\*

В работе создана онтологическая модель топологии компьютерной сети, объединяющая \*\*сетевые устройства, серверы, компьютеры, маршрутизаторы и организационные единицы\*\*.

Базовый класс \*\*NetworkDevice\*\* обеспечивает централизацию модели, связывая все сетевые сущности и упрощая анализ данных.

\*\*SWRL-правила\*\* и \*\*Cypher-запросы\*\* позволяют автоматически выводить косвенные связи:

- компьютеры ↔ серверы через общие сети

- отделы ↔ используемые серверы через принадлежащие компьютеры

- компьютеры ↔ маршрутизаторы через сетевые подключения

Модель наглядно реализована в \*\*Protégé\*\* и \*\*Neo4j\*\*, что подтверждается визуализацией и скриншотами.

Практическая ценность: отслеживание сетевых подключений, анализ доступности ресурсов, идентификация зависимостей между устройствами, а также возможность расширения модели для мониторинга и управления корпоративной сетевой инфраструктурой.