FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

Zadání bakalářské práce

Název:	Modelování a řízení dat v doméně online vzdělávání
Student:	Filip Sikora
Vedoucí:	Ing. Michal Valenta, Ph.D.
Studijní program:	Informatika
Obor / specializace:	Webové a softwarové inženýrství, zaměření Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	do konce letního semestru 2021/2022

Pokyny pro vypracování

Online doporučovací systémy pro optimální volbu materiálů a kurzů vzhledem k vytýčenému cíli (znalosti, dovednosti) tzv. learning paths jsou zajímavou aplikační doménou a výzvou z hlediska datového modelování, řízení, sběru a správy dat o kurzech a poptávce po konkrétních pracovních pozicích a také z hlediska využití moderních technologií jako jsou grafové databáze.

Cílem bakalářské práce je vytvoření konceptuálních a databázových modelů dat a tvorba a aplikace postupů pro řízení a správu dat v této doméně.

1. Seznamte se s podrobnostmi společného projektu FIT a firmy Learneron.
2. Ve spolupráci s vedoucím BP vytvořte modely jednotlových datových zdrojů a integrovaný model datových zdrojů.
3. Ve spolupráci s vedoucím BP vypracujte postupy pro řízení a správu dat v tomto projektu.
4. Postupy ověřte na datech projektu a dokumentujte je.

Výstupem práce budou datové modely, postupy pro správu a řízení dat, jejich dokumentace a zpráva o aplikaci v projektu s firmou Learneron.

Elektronicky schválil/a Ing. Michal Valenta, Ph.D. dne 11. listopadu 2020 v Praze.

Bakalářská práce

MODELOVÁNÍ A ŘÍZENÍ DAT V DOMÉNĚ ONLINE VZDĚLÁVÁNÍ

Filip Sikora

Fakulta informačních technologií ČVUT v Praze
Katedra softwarového inženýrství
Vedoucí: Ing. Michal Valenta Ph.D.
11. května 2021

Obsah

Seznam obrázků

Seznam tabulek

Chtěl bych poděkovat celému týmu Learneron, se kterým jsem spolupracoval během vytváření práce a jeho vedoucímu Vladislavovi Severovi, ale především vedoucímu práce Ing. Michalu Valentovi, Ph.D. za jeho pomoc na projektu, který jako odborník v tomto oboru přinesl do výsledné práce spoustu praktických rad a zkušeností.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací. Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 citovaného zákona.

V Praze dne 20. dubna 2021 .....

Abstrakt

Tato bakalářská práce se zabývá tvorbou konceptuálních a logických (relačních a grafových) modelů dat v doméně doporučovacích systémů pro vzdělávání. Dále pak tvorbou a aplikací postupů pro řízení a správu těchto dat. V práci je využita notace OntoUML pro tvorbu a údržbu domény, grafová databáze Neo4j a relační databáze MySQL pro uložení dat. Součástí práce je původní návrh postupu transformace OntoUML schématu do grafové databáze Neo4j.

Klíčová slova doporučovací systém, návrh databáze, ontologie, konceptuální modelování, transformace modelu, grafová databáze, Neo4J

Abstract

This bachelor thesis deals with the creation of conceptual and logical (relational and graph) data models in the domain of recommendation systems for education. Furthermore, it deals with the creation and application of procedures for the management and administration of this data. The thesis uses the OntoUML notation to create and maintain the domain and the Neo4j graph database together with the MySQL relational database for data storage. The work's original proposal is design of the transformation process of OntoUML schema into the Neo4j graph database.

Keywords recommender system, database design, ontology, conceptual modelling, model transformation, graph database, Neo4J

Seznam zkratek

EA	Enterprise Architect
UFO	Unified Foundational Ontology

Kapitola 1
Úvod

Doporučovací systémy jsou dnes běžnou praxí od doporučování přátel na Facebooku po další video ke zhlédnutí na YouTube. Personalizovaný obsah je velice efektivní a není divu, že jej společnosti jednotlivci využívají. Většinou se jedná o řešení, ze kterého mají užitek jak producenti tak konzumenti dat. Relevance obsahu totiž konzumenta často přiměje k větší konzumaci obsahu.

Jeden z takových systémů je vytvářen společností Learneron. Tento systém má za cíl pomáhat lidem při jejich profesním rozvoji. Doporučováním správných zdrojů může pomáhat v karierním růstu jednotlivcům, kteří chtějí přejít od jednoho zaměstnavatele k jinému, tak také v kariérním růstu v rámci jedné organizace. Vybral jsem si toto téma jelikož upřímně věřím v jeho dobrý přínos společnosti, které může pomoci se seberozvojem, ale také protože se zajímám o konceptuální přístupy, které v této práci mohu zhodnotit.

V dnešní době se trh neustále mění a vyvíjí, proto je přístup k systému doporučujícímu relevantní obsah pro sebevzdělávání dává lidem velkou výhodu nestagnovat a požadavkům trhu se přizpůsobovat.

Pro implementaci rozšířeného datového modelu, který je dílčím výsledkem této práce, byla zvolena grafová databáze Neo4j. Grafové databáze jsou poslední dobou hojně využívány právě pro doporučovací úlohy v mnoha různých doménách. Grafová databáze nemá žádné pevné schéma, na druhou stranu je velmi důležité udržet si představu o struktuře ukládaných dat a zajistit tak rozvoj a řízení dat, využíváme konceptuální model, konkrétně notaci OntoUML.

V první části představím potřebnou teorii pro pochopení praktické části. Tato část popisuje projekt Learneron, konceptuální modelování a grafovou databázi. Ve druhé části je popsána analýza problematiky spolu s návrhem ontologie, implementací grafové databáze a metodika pro další rozvoj databáze.

Kapitola 2
Cíl práce

Hlavním cílem této bakalářské práce je analýza, návrh a implementace datové části pro doporučovací systém. Pro pochopení všech částí praktické sekce práce je nutné uvést teoretický základ daných problematik. Při analýze je potřeba zpracovat návrhy zadavatele se stavem jeho existujícího řešení. Součástí návrhu datové části je ontologický OntoUML model pro zachycení modelu a jednoznačnou dokumentaci. Pro implementaci je nutné zpracovat již existující data s novými do potřebné formy vyhovující cílům projektu. A pro zpracování dat zdokumentovat metodiky potřebné k takovému postupu.

Kapitola 3
Literární rešerše

V této kapitole přiblížím teoretická i praktická východiska mé bakalářské práce. První sekce popisuje projekt s firmou Learneron, do jehož řešení jsem byl zapojen a kde vzniklo téma mé bakalářské práce. Druhá sekce se věnuje vysvětlení pojmů ontologie, konceptuální modelování a zmiňuje konkrétní prostředky pro modelování ontologií a jejich částí. Třetí sekce představuje grafové databáze a jednu jejich konkrétní implementaci - Neo4J, která byla použita pro implementaci nového datového modelu.

3.1 O projektu Learneron

Tato práce probíhá v rámci společného projektu firmy Learneron, SE a Fakulty informačních technologií ČVUT. Financování této spolupráce je podpořeno Pražským voucherem na inovační projekty ve výzvě s číslem tři. Předmětem řešení je *Pokročilé inovativní datové struktury v podpoře celoživotního vzdělávání*. Tato žádost byla vytvořena podnikatelským záměrem[1] předaném hlavnímu městu Praha. Kde je mimojiné popisováná konkurenceschopnost, tržní potenciál i jeho příspěvek společnosti.

Firma Learneron, SE podniká dle NACE kategorizace v podpůrné činnosti ve vzdělávání. Byla založena v roce 2017. Hlavní podnikatelskou aktivitou společnosti je poskytovat zájemcům o celoživotní vzdělávání návodné a personalizované vedení při volbě cesty vzdělání (tzv. Learning Path) a v pozdější fázi i cílů vzdělání (Learning Goal), a to v online prostředí.

„Služba společnosti Learneron, SE byla již funkční a v minulé době a provozovala sociální webové fórum pro otázky zájemců o celoživotní vzdělávání a specializovanou vyhledávací funkcionalitu pokrývající přibližně sedmnáct tisíc tzv. MOOC online kurzů (další jsou průběžně přidávány) a několik milionů knih.“ [1]

Cílemi této spolupráce jsou:

1. Datová ontologie pro definici a modelování souvislostí a propojení datových entit
2. Datový model a konverze z částečných datových modelů
3. Konkrétní vzorová implementace výsledného datového modelu
4. Testování algoritmické funkcionality
5. Vzorová / testovací implementace. Vzorové end-to-end propojení a implementace v rámci GRANDstack
6. Průběžné konzultace, manuály

Tabulka 3.1 Tabulka technologií

	Původní	Nové
Front-end	PHP, React	GRANDstack, React Native (pro mobilní aplikaci), MySQL, Souborový systém Hadoop, PHP, mikroslužby: - Elasticsearch / FAISS - Crawling - Datové roury v Python-u - Cypher skripty
Back-end	MySQL, PHP, mikroslužby: - Elasticsearch - Crawling - Datové roury v Python-u	

Má práce se týká prvních tří bodů zadání projektu. Zbytek byl řešen ostatními spolupracovníky zastupující ČVUT v projektu.

3.1.1 Použité technologie v aplikaci

Seznam použití technologií v rámci celého stacku je v tabulce 3.1. Ze sloupce *Nové* je viditelné, že díky společnému projektu se ČVUT, aplikace znamenala velký modernizační posun v technologiích.

V průběhu prací došlo k architektonickému rozhodnutí a pokynu vedoucích projektu Learneron, že část dat zůstane i nadále v relačním databázovém stroji MySQL, část bude převedena do Neo4j a některé datové entity budou z výkonových a jiných důvodů replikovány v obou úložištích.

Důvod proč je MySQL mezi plánovanými technologiemi k použití je:

- větší důvěra v odezvu klasické databáze,
- současné řešení a zřejmě i budoucí frontend (minimálně zpočátku) využívá (pro vyhledávání a našeptávání) přídavnou indexaci v Elastic search.

3.1.1.1 GRANDstack

GRANDstack je kombinace technologií pro vytvoření full-stack datově orietovaných aplikací.

Těmito komponenty jsou:

GraphQL je paradigma pro volání API. Vyhledávání funguje na bázi odesílání dotazů typicky viděných u databázových strojů. Tedy varianta klasického REST API,

React je JavaScript knihovna pro front-end, založena na znovu použitelnosti bloků nazývané jako komponenty,

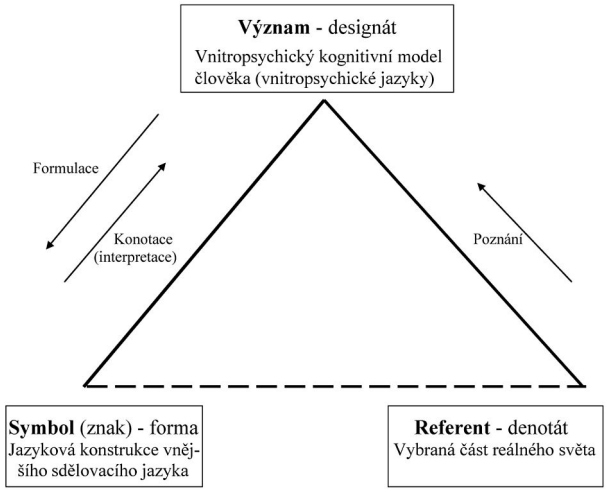
Apollo je sada nástrojů pro práci s GraphQL,

Neo4j Databáze je NoSQL databáze reprezentující data jako graf .

3.2 Ontologie a jejich modelování

Jedná se původně filozofický přístup zabývající se základními pojmy našeho světa, jejich vlastnostmi, strukturou a vzájemnými vztahy. Dnes se tento pojem používá v IT k definici zkoumané subdomény.

Toto jasné definovaní je velice důležité, jelikož vnímání skutečnosti různými lidmi se může rozcházet. Ontologie má za cíl jasně definovat chápanou problematiku pro všechny zúčastněné k docílení *sémantické interoperability*. Neboli sjednotit pohled na realitu.



Obrázek 3.1 Sémiotický (sémantický) trojúhelník [3]

3.2.1 Konceptuální modelování

„Konceptuální modelování je aktivita popisující aspekty fyzického a sociálního světa s cílem pro porozumění a komunikaci… adekvátnost notace konceptuálního modelování leží ve schopnosti sdílet porozumění o popisovaném světě pro jejich uživatele.“ [a] Konceptuální modely jsou obecné modely zachycované skutečnosti, používající pojmy jako entita, atributy a vztahy k tomuto popisu [2].

Někdy se k pochopení používá *Sémiotický trojúhelník* jako na obrázku 3.1, kde referentem je skutečnost, významem je naše chápání skutečnosti a symbolem je zápis v tomto případě v podobě konceptuálního modelu.

Namísto klasické konceptuální modelovací techniky — také TCM techniky stojící pro *Traditional conceptual modelling* — padlo rozhodnutí aplikovat techniku zakládanou na ontologii — také ODCM jako *Ontology-driven conceptual modelling*. Jelikož dle studie [4] je přístupnější pro nové uživatele, kteří v porovnání s první variantou jsou schopni vytvářet kvalitnější modely. Výhodou je také, že s ní již mám zkušenosti z výuky. Díky skutečnosti, že se vyučuje na Fakultě informačních technologií ČVUT mám taky blízký přístup ke lidem zkušeným této techniky.

3.2.2 UFO

UFO (Unified Foundational Ontology) je soubor postupů pro modelování ontologií. Tyto postupy jsou inspirovány formální ontologií, filozofickou logikou, lingvistikou ale i kognitivní psychologií. Dle svého tvůrce jménem Giancarlo Guizzardi — který je popsal ve své práci [5] — jsou tyto postupy rozděleny na 3 vrstvy, v rámci této práce se využívá pouze první vrstva A.

Všechny vrstvy jsou následující:

UFO-A se zaměřuje na persistentní objekty, vztahy a jejich struturu

UFO-B popisuje události a procesní akce

UFO-C pohlíží na sociální aspekty

[a] John Mylopoulos, Conceptual Modeling and Telos, 1992

3.2.3 UML

Jedná se o grafickou modelovací notaci založenou na jednom meta-modelu. Slouží k popisu SW a SW systémů především těch založených na OOP. Takový model slouží jako adekvátní abstrakce pro diskutování designu v porovnání se zápisy v programovacích jazycích samotných. Notace UML je řízena skupinou OMG, která se sformovala pro vytvoření obecného standartu mezi OOP systémy [6].

Celkem je v době psaní této zprávy 14 typu UML diagramů, mezi nejznámější pro ukázku patří následující:

- Strukturální (diagramy)
  - Třídní
  - Balíčkové
  - ...
- Behaviorální (diagramy)
  - Diagram aktivit
  - Use Case diagram
  - ...

Je potřeba zdůraznit, že právě Třídní model — doplněný o stereotypy — je model, na kterém staví OntoUML model.

3.2.4 OntoUML

OntoUML je implementací konceptuální modelovacího jazyka, které je rozšířením (profilem) základního UML. Z ontologického hlediska budu modelovat pouze vrstvu UFO-A. Pro tuto vrstvu se používá *Třídní model* jazyka UML, ten popisuje entity s jejich asociacemi a vlastnostmi.

Využívá pouze konstrukty UML, kterým dává nový smysl. Navíc dává velký významový smysl popisům stávajících konstruktů, které nazývá stereotypy. Jakým způsobem je detailně popsáno zde [7], [5]. Je důležité poznamenat, že se jedná pouze o způsob, jak zapsat ontologický model a UML je využit jako prostředek pro notaci, který je ve společnosti velice rozšířený.

Následně představím základní typy entit a stereotypů potřebných především pro pochopení mých modelů, které popisuji v obrázcích v následujících sekcích. Především stereotypy dělají z modelu ontologický strukturální popis, proto budou zmíněny především ony. Tento krátký popis je reinterpretován dle ČVUT materiálů zabývajících se danou problematikou [8], [9], [10].

Podobně jako v OOP tak i v konceptuálním modelování jsou vnímány třídy a instance. Ale od OOP se liší pár věcmi. Instance dat mohou dědit (specializovat se) z jakéhokoliv počtu tříd. Dědění je označováno šipkou od dědice k rodiči. Z hlediska dědění vnímáme také navíc existenci dvou meta-atributů.

complete : tedy každá instance nadtypu je také instancí alespoň jednoho z podtypů. Neboli nemohou existovat instance pouze nadtřídy, pokud typ B a C dědí z typu A, pak každá taková instance musí být typu B, C, nebo jejich kombinací. Dalo by se říct, že typ A by v OOP byl abstraktní.

joint : tedy neexistuje instance, která by byla instancí více než jednoho z podtypů. Neboli nelze kombinovat podtypy, pokud typ B a C dědí z typu A, pak každá taková instance musí být typu A, B, nebo C, ale ne kombinací.

Pokud jsou použity obě vlastnosti (a typ B a C dědí z typu A) pak popisují, že každá instance dat je pouze typu B, nebo C.

Vztahy entit, neboli asociace také mohou obsahovat meta-atributy ke zvýšení popisnosti. Popisují seřazení prvků, či unikátnost prvků, to je popsané u kardinality na straně vztahu, pro kterou podmínky platí

un/ ordered : bez pořadí/ prvky ve vztahu jsou uspořádané v poli dle pořadí vkládání

non/ unique : prvky ve vztahu ne/ musí být unikátní

Pro rozdělování dat a jejich popis pohlížíme především na jejich vlastnosti a jak je vnímáme v rámci přidělené domény, proto začnu popisem sledovaných charakteristik.

Princip identity je identifikovatelnost objektu v průběhu času a libovolných změn. Dle něj popisujeme objekty jako:

- sortálního typu pokud objekt je jednoznačně identifikovatelný v čase (má princip identity) nebo
- nonsortálního typu pokud dle našeho vnímání nemá vlastní identitu (, nebo jich má více), nejsme schopni mu ji přiřadit jako například objekt typu barvy Červená. Instance nonsortální typu nikdy nezískají princip identity.

Rigidita je vlastnost zkoumaná u sortálních objektů. Pokud objekty má rigidní typ, pak je toto přiřazení neměnné a neměnní se v čase. Například objekt s typem Osoba bude i přes nedomyslitelné změny pořád považovatelná za osobu. Objekty s anti-rigidním typem pak můžou tento typ v čase získávat i ztrácet, například objekt s typem Student je anti-rigidní, protože tento objekt v čase může získávat i ztrácet status studenta.

Základním stereotypy jsou *Kind* jako druh a *SubKind* jako poddruh. Oba jsou sortální a rigidní, často se pak používá výraz *Rigidní sortály*. Sterotyp *Kind* — jako pár vybraných — označuje, že daný typ poskytuje identitu. *SubKind* na druhou stranu ne, ale tuto identitu dědí od *Kind*. Například Osoba bude typ se stereotypem *Kind* kvůli naší schopnosti identifikovat a rozlišovat osoby. A Čech bude *SubKind*, protože nejsou takto rozlišitelné, ale navíc pravděpodobně budou přebírat identitu z objektu jako je Osoba. Pokud u sortálních stereotypů není vyjádřeno, že identitu poskytují, pak ji přebírají od typu, který je schopen je poskytovat.

Dalšími klíčovými stereotypy jsou *Phase* a *Role*, oba jsou anti-rigidními sortály, tedy identifikovatelné, ale v čase měnící se. *Phase* označuje typ jako vnitřní fázi nadtypu, od kterého se specializuje. Tato fáze je s ostatními fázemi vždy ve vztahu *disjoint a complete*. U Osoby tak fází může být například zdali je zdravý, nebo nemocný. Vždy musí být jedním z toho. *Role* označuje vnější fázi, navíc ovšem podléhá závislosti na jiné entitě v modelu, který zajišťuje existenci této fáze. Například role Student potřebuje entitu jako Škola pro svou existenci.

Dalším klíčovým stereotypem pro kategorizaci sortálů je *Category*. Jedná se o rigidní nonsortál pro kategorizaci zde uvedených druhů jako *Kind*, *SubKind*, nebo další kategorie. Častokrát jsou všechny kategorizované druhy disjunktní.

Relace nebo asociace jsou rozděleny na formální a materiální.

Formální : je nejčastěji porovnávající vztah mezi instancemi stejné třídy, jsou to relace založené na vnitřních atributech objektů a nepotřebují žádnou stvrzující entitu pro platnost vztahu. Hrana tohoto vztahu se popisuje stereotypem *Formal*.

Materiální : je nejpoužívanější typ vztahu, u kterého relací vzniká nová informace. Tato informace stvrzuje tento vztah a manifestuje v novou entitu (relátor), která je symbolizovaný stereotypem *Relator*. Například pro vztah Zaměstnání mezi Zaměstnancem a Zaměstnavatelem bude existovat entita Pracovní smlouva s typem *Relator*, která stvrzuje tento vztah a obsahuje informace nabyté uskutečněním relace. Taková relace mezi Zaměstnancem a Zaměstnavatelem se popisuje stereotypem *Material*. Navíc zde vzniká vazba mezi Zaměstnancem

a Smlouvou, Smlouvou a Zaměstnavatelem pro připojení relátoru. Přidávání relátoru a dalších navazujících vazeb ovšem častokrát omezuje přehlednost modelu a protože se často jedná o odvoditelný vztah, zvolil jsem v modelech tuto relaci vypustit. Pouze pokud relátor bude mít přidanou hodnotu k vypovědění o vztahu — namísto běžného prostředníka — pak bude explicitně uveden a popsán.

Relace Celek-Část : problematika Celek-Část je často diskutovanou tématikou a řeší vztahy mezi celky a jejich částmi, v ontologii se dělí na typy jako: Kvantita, Kolektiv a Funkční Celek. V této práci je ovšem využito Funkčního Celku, který je nejběžnější, kdy celek se skládá z částí, které mají různé úlohy/ funkce v rámci celku. Tato vazba se vyznačuje propojením v UML používaným pro kompozici (linka s vyplněným kosočtvercem na straně celku), nebo agregaci (linka s prázdným kosočtvercem na straně celku). Plnost tohoto kosočtverce označuje sdílitelnost částí mezi různými celky, nebo v druhém případě jejich exkluzivnost.

Kromě objektů doposud popisovaných existuje i kategorie apektů, ty jsou samostatné entity existenčně závislé na jiné entitě pro kterou nesou informaci. Dalo by se říct, že se jedná o vytrhnutý atribut z entity. Účelem je další popisnost umožněna aspekty zde neprobíraná. Aspekty dělíme na:

Kvalita entitu označuje stereotypem Quality a reprezentuje měřitelnou veličinu.

Mód entitu označuje stereotypem Mode a reprezentuje neměřitelnou veličinu.

Aspekty se vážou na závislou entitu vazbou popsanou stereotypem *characterization* a vždy mají na obou stranách kardinalitu 1:1.

3.3 Grafová databáze

Databáze samotná je organizovaná kolekce dat. Jako tradiční databáze je i tato založena na datovém modelu a má tyto charakteristiky [11].

- „Data a databázové schéma jsou reprezentovány grafem, nebo strukturami popisující graf.“
- „Manipulace daty je vyjádřena grafovými transformacemi.“
- „Integritní omezení vynucují konzistenci dat.“

Grafová databáze je založena na teorii o grafech. Řadí se mezi *NoSQL* databáze. Je tomu protože databázový designer nedefinuje jasně dané schéma, jak jsou informace uloženy, pomocí kterého by se vynucovaly pravidla pro nová data. To je do určité míry zajištěno integritními omezeními, ale nejedná se o plnohodnotnou náhradu DDL viděného u relačních databází. To znamená a je výhodou, že např. pokud se kus dat mírně liší, není nutné hned modifikovat design databáze. Na druhou stranu to vede k sub-optimalitě dotazování. Přesto u struktur často měnících se, nebo generovaných uživatelem se jedná o výhodu [11].

V samotné formě se — jako graf — skládá z uzlů (Node) a hran (Edge/ Link). Data tedy ukládá do uzlů (ekvivalenty záznamu řádku v relační databázi) a vztahy mezi těmito daty do hran — orientovaných či neorientovaných — mezi nimi. Vztahy tedy jsou First-class objekty, což znamená, že stejně jako uzly mohou mít popisy (Labels) a přiřazené atributy (jako sloupce v relační databázi). Narozdíl od relačních databází, které zjišťují takové informace o spojeních run-time [12].

3.3.1 Porovnání s relační databází

Dle [13] jsou grafové databáze výrazně rychlejší pro propojená data. Kvůli jiné struktuře, rychlost výsledku nezávisí tak na velikosti uložených dat jako na velikosti prozkoumaného stromu. Grafové databáze ukládají spojení jako First-class objekty, dochází proto k procházení v teoreticky konstantním čase. To je podpořeno měřeními v jedné z implementací [14].

V naší doméně budou tyto hledání časté, bude se jednat o hledání na základě kontextu, historie, či preferenci uživatele.

3.3.2 Typická použití grafových databází

Použití tohoto typu databáze se uchytilo především v následujících odvětvích.

Sociální studie Jak lidé spolu interagují, jak se ovlivňují, tyto informace se přenášejí. K tomu se vybízí grafová forma. Cílem takových studií je porozumění světové demografiky, politického smýšlení, nebo náklonnost ke komerčním produktům. Hojně využívané společnostmi jako Google, Facebook, Twitter k cílení na zákazníka.

Biologické studie Interakce proteinů, molekul, či genů se dlouhodobě reprezentují grafem. Využívá se tak často pro např. nalézání léků, dle hledaných atributů, zkoumání jak tento lék reaguje na krvinky v těle atd.

Počítačová věda Běžnou praxí je reprezentace problému v grafové formě, a pak spuštění ověřeného algoritmu (např A*), který jej vyřeší. Deterministické algoritmy tak řeší skoro jakékoliv problém, často pak bývá brzdou pamětová náročnost úkolu.

Vyhledávání na Webu PageRank — grafový algoritmus — vymyšlen pro používání společností Google pro vyhledávání obsahu, jak jej dnes všichni znají, by se mohl řadit k nejpoužívanějším algoritmům na světě.

Přebráno [15].

3.3.3 Důvody pro volbu grafové databáze

Jelikož se jedná o zadání firmy Learneron, tak si i tato firma volila technologie použité k řešení. Grafová databáze byla zvolena především, protože:

1. struktura dat, se kterými portál pracuje a hodlá pracovat má grafový charakter a
2. řada pokročilých vyhledávacích funkcí využívajících metody strojového učení a umělé inteligence je rovněž založena na grafových algoritmech.

V případě Content Recommender systému se totiž doporučují obsahy jako blízcí uživatelé, zajímavé tagy, či položky kariérní cesty. Tato úloha často vypadá, že se pro jeden *node* jako např. uživatele vytváří/ hledá neexistující spojení k potřebnému vysledku. Např. hledá se další uživatel, který je zatím neznámý, ale má s námi něco společného. Tento problém — typický grafový problém — se nazývá *Predikce neexistujícího spojení*.

Pomoct může algoritmus *PageRank*, nebo vytrénovaný klasifikátor po upravení (vektorizaci) potřebných uzlů.

Implementací bylo zvoleno Neo4J. Díky komunitní verzi je cenové přístupný a zároveň neztrácí na zralosti. Podporuje spoustu knihoven a funkcí právě pro aplikaci metod strojového učení a umělé inteligence s potřebnými algoritmy jako např. *Node Embedding*. Má vlastní vyhledávací jazyk jménem Cypher, který je podobný SQL. Poskytuje taktéž kvalitní knihovnu pro jazyk Python, který je mezi lidmi pracujícím s algoritmy strojového učení a umělé inteligence velmi oblíben.

3.3.3.1 Důvody proti

Nelze diskutovat o zralosti relačních řešení, jsou zde již velice dlouho a stejně jako C kompilátory, tak mají za sebou generace optimalizací, které nelze jednoduše dohnat. Proto je samozřejmé, že existují případy, kde použití grafové databáze není nejlepší volbou.

Jedná se o například.

Rozsáhlé „mělké“ dotazy Pokud se jedná o spoustu dat bez velkých agregací, či spojování.

3.3.4 Neo4J

Neo4J je populární implementací výše popisované grafové databáze. Neo4J má hrany grafu pouze orientované, což nevytváří problém s jakýmkoliv návrhem, jelikož neorientovaná hrana se na orientovanou dá rozšířit. Tyto hrany musí mít vždy definován začátek a konec (muže se jednat o jediný uzel). Jak bylo zmíněno v části o grafových databázích, hrany mají stejné možnosti popisu jako uzly, což je rozdíl oproti ostatním databázím [15].

Databáze je plně ACID neboli garantuje pro všechny transakce následující.

A – Atomicita Transakce proběhne celá, nebo stav navrátí do původního.

C – Konzistence Transakce změní stav databáze z validního do jiného validního.

I – Izolace Paralelní transakce běží nevávisle na sobě.

D – Durabilita Změny po ukončení transakce jsou permanentní

Tato implementace je založena grafovém modelu (v původním znění *Labeled Property Graph Model*) se štítky a vlastnostmi.

Rozšířením od standartního grafu jsou:

Štítky uzlů [15] představují kategorizaci pro uzly. Uzel tak může mít žádný, či více štítků. Ty jsou sdíleny mezi podobnými uzly např. uzly nosící informaci o uživateli mají štítek *User*. Jedná se o jedinečné popisky k vytvoření podgrafů pro integritní omezení, či vyhledávání jazykem *Cypher*.

Typy vztahů Jedná se o stejný princip jako štítky ale mířen na spojení uzlů (hrany). V praxi označovány obě varianty jako štítky splývají.

Vlastnosti Každá hrana i uzel může obsahovat vlastnosti v podobě klíč-hodnota párů.

Dotazovacím jazykem je Cypher. Podobně jako SQL se jedná o deklarativní jazyk, je vyvíjen pro účely Neo4J.

Integritní omezení poskytují způsob, jak vymezit pravidla grafu.

Používaná je uživateli pro:

komplexní dotazy — ekvivalentem v relační databázi jsou dotazy spojující více tabulek pomocí klíčového slova JOIN — jsou v relačních databázích operačně drahou operací, jelikož se vytváří kartézským součinem a růst takové náročnosti je exponenciální. Například nalezení uživatelů, kteří jsou teď online, jsou nově registrovaní, nemají žádné přátele a jsou ze země X. Díky explicitním ukazatelům z uzlů na navazující uzly je možné okamžitě procházet takováto data a filtrovat ty vyhovující. To je klíčovou vlastností grafových databází, prohledávání se vyhýbá nenavazovaným datům. Složitost takového dotazu pak neroste s přibýváním nezúčastněných dat [15].

Nalézání cesty v grafu : Častým dotazem také může být dotaz jako minule, ovšem by se hledali lidé ze stejných zemí. Hledala by se tedy cesta mezi danými uživateli. Což by mohlo být extrémně složité zformátovat do nějakého plodného dotazu. V případě této databáze je potřeba akorát definovat počáteční a koncové body a databázový systém sám takovou cestu najde [15].

3.3.5 Neo4J Browser

Jedná se grafickou aplikaci přístupnou prohlížečem, která je vstupním bodem pro základní práci s databází. Nenahrazuje plně CLI tool, ale budu ji používat pro demonstraci postupu vývoje databáze a na to stačí. Poskytuje možnost:

- psát a spouštět Cypher dotazy,
- exportovat výsledky dotazů,
- vizualizovat výsledky dotazů
- a poskytuje přístup k API.

Kapitola 4
Praktická část

V této kapitole ukážu praktický postup mé bakalářské práce. Od analýzy, přes implementaci až k metodice budoucího rozvoje. První sekce popisuje prvotní představu o reprezentaci dat firmy Learneron. Vysvětluje ISCO a NACE klasifikaci, která je potřebná pro klasifikaci klíčových entit v později popisovaným modelech. Druhá sekce se věnuje návrhu ontologie dle existujících návrhů a dokumentace. Popisuje také postup získání modelu z existujícího databázového řešení. Třetí sekce pak popisuje samotnou implementaci grafové databáze včetně transformace OntoUML do grafové databáze. Čtvrtá a poslední sekce představuje metodiku pro budoucí rozvoj databáze a import nových dat.

4.1 Analýza

Společnost Learneron měla představu o struktuře dat viz. sekce 4.1.1. Tato idea vede k rozšíření stávajícího datové úložiště k umožnění pokročilých vyhledávacích funkcí jako:

- vyhledávání a doporučování optimální vzdělávací cesty (learning path),
- napojení na standardizované číselníky dovedností a pracovních pozic (ESCO Graph v prostředí EU).

Většina z těchto pokročilých funkcí využívá metod strojového učení a umělé inteligence. V úvodní fázi půjde ovšem o determinované hodnotící algoritmy. Vývoj těchto metod je plně v jejich rukou.

Data v nové databázi budou tvořit:

původní data vycházející z relační MySQL databáze , které je třeba vhodně upravit, integrita těchto dat byla hlídaná na úrovni aplikace PHP. Tedy přímo v databázovém stroji se nepoužívaly typicky samozřejmé prostředky jako například cizí klíče a

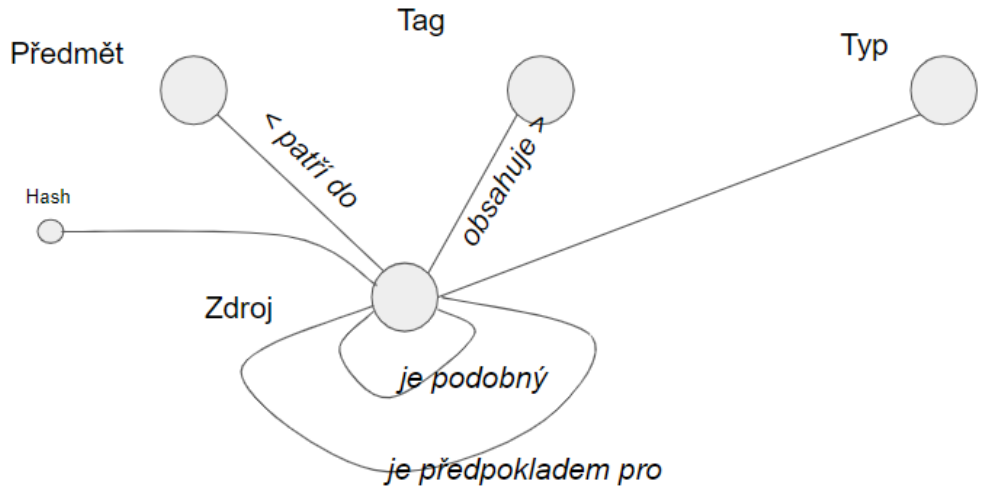
nové data potřebné pro vykonávání funkcionality klasifikátoru, které doteď nebyly používány.

4.1.1 Prvotní reprezentace zdrojů dat

Zdroj A reprezentuje — viz obrázek 4.1 — zdroje infromací pro doporučení obsahu jako součást vzdělávání, jedná se o data z původní MySQL databáze.

Takovými zdroji dat jsou například tyto typy.

- MOOC kurzy



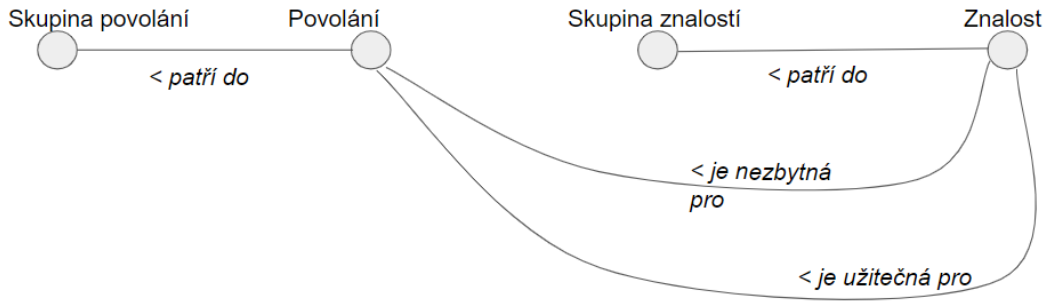
Obrázek 4.1 Koncept datového zdroje A

- články z blogů
- veřejně dostupné učebnice
- videa, podcasty
- složené zdroje:
  - vzdělávací postupy
  - vzdělávací kolekce

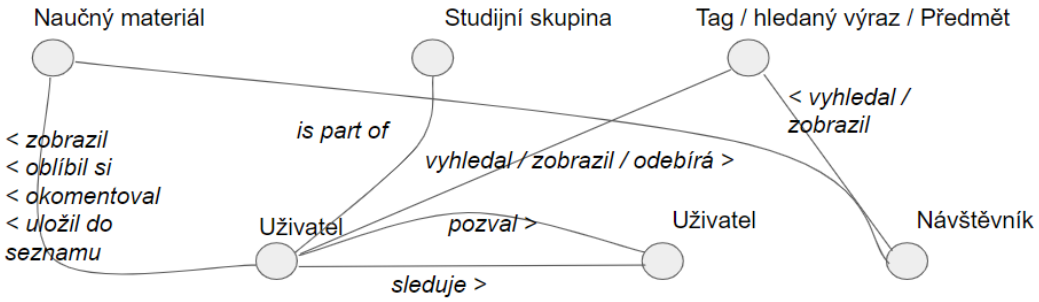
U takových zdrojů jde pak především o zachycování těchto vnitřních vlastností.

- titulek
- URL
- předmět (téma)
- vektorová reprezentace
- tagy, klíčová slova
- souhrn
- počet navštívení, počet oblíbení, počet dostupných kopií
- mateřské zdroje
- časové známky posledních změn
- typ zdroje
- náročnost pochopení (absolvování)
- délka trvání
- cena
- ověřenost zdroje

Zdroj B je — viz obrázek 4.2 — reprezentací ESCO klasifikace. Ve zkratce jde o seznam povolání a jejich potřebných dovedností, jedná se o nové data. Blíže popsáná celá doména je v sekci 4.2.5.1.



Obrázek 4.2 Koncept datového zdroje B



Obrázek 4.3 Koncept datového zdroje C

Zdroj C je — viz obrázek 4.3 — popisem uživatelů a jejich aktivit, jedná se o data z původní MySQL databáze.

Dalo by se zjednodušeně říct, že dle zdroje B (ESCO) a C (aktivity uživatele), se bude uživatelům doporučovat materiál ze zdroje A. S propojením viz obrázek 4.4

4.1.2 ISCO klasifikace

Jedná se o mezinárodní klasifikační standart pro zaměstnání. Tato kvalifikace je součástí modelovací subdomény ESCO, která se zabývá pracovními pozicemi a jejich potřebnými znalostmi. Tam je reprezentovaná jako entita s názvem *Occupation Classification ISCO*. Více pak v sekci 4.2.5.

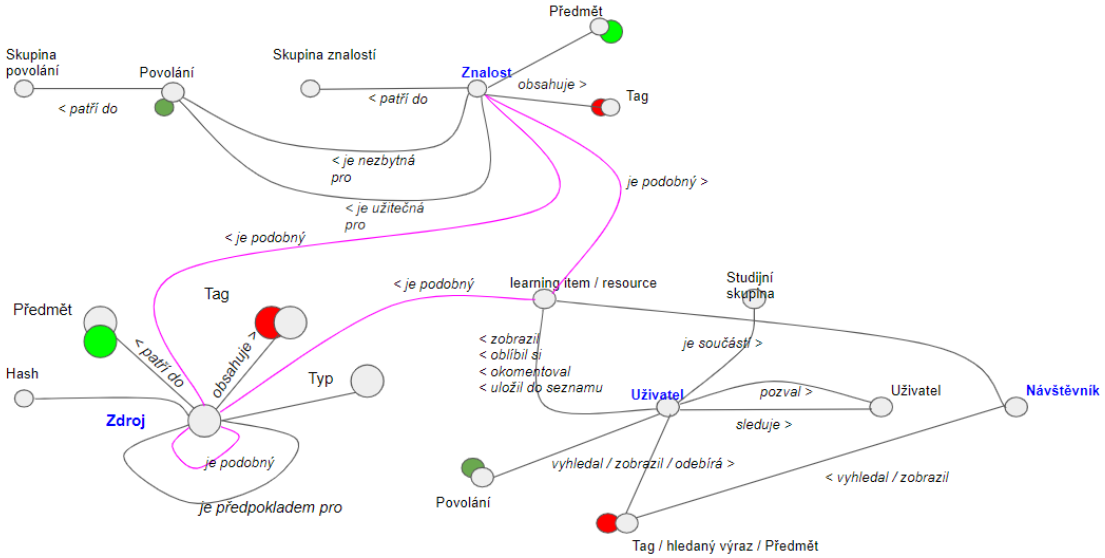
„Zaměstnání jsou v jednotlivých třídách seskupena na základě podobnosti vykonávané práce a na základě podobnosti kvalifikace (souboru znalostí a dovedností), požadované k plnění úkolů a povinností v zaměstnání.

Klasifikace se zaměřuje na kvalifikaci požadovanou pro provádění úkolů a nikoliv na to, zda pracovník vykonávající konkrétní zaměstnání je více nebo méně kvalifikovaný než jiný pracovník ve stejném zaměstnání.

Hlavní třídy jsou tedy postupně seřazeny podle úrovně a specializace znalostí a dovedností potřebných k výkonu určitého zaměstnání od nejnáročnějších (1. a 2. třída) k nejjednodušším zaměstnáním (9. třída). “[16]

4.1.3 NACE klasifikace

Jedná se o mezinárodní klasifikační standart pro ekonomické činnosti. Tato kvalifikace je součástí modelovací subdomény společností, která se zabývá zachycením struktury obchodních společ-



Obrázek 4.4 Koncept spojení datových zdrojů

ností. Tam je reprezentovaná jako entita s názvem *Economic Activity Classification NACE*. Více pak v sekci 4.2.5.6.

„NACE je akronym pro statistickou klasifikaci ekonomických činností, kterou používá Evropská unie (resp. Evropská společenství) od roku 1970. NACE vytváří rámec pro statistická data o činnostech v mnoha ekonomických oblastech (např. ve výrobě, zaměstnanosti, národních účtech).

Statistiky, které vzniknou za použití klasifikace NACE, lze srovnávat v celé Evropské unii. S nižší mírou podrobnosti (na vyšších úrovních) je možné srovnání i se světovými statistikami. Používání NACE je povinné pro všechny členské státy Evropské Unie.

Srovnatelnost dat vytvořených podle klasifikace NACE na světové úrovni je dána tím, že NACE je součástí systému statistických klasifikací, které vznikly převážně pod záštitou Statistické divize Spojených národů.

Klasifikace NACE je odvozena z klasifikace ISIC. Na vyšších úrovních jsou shodné, na nižších úrovních je NACE podrobnější, jak je na obrázku 4.5. “[17].

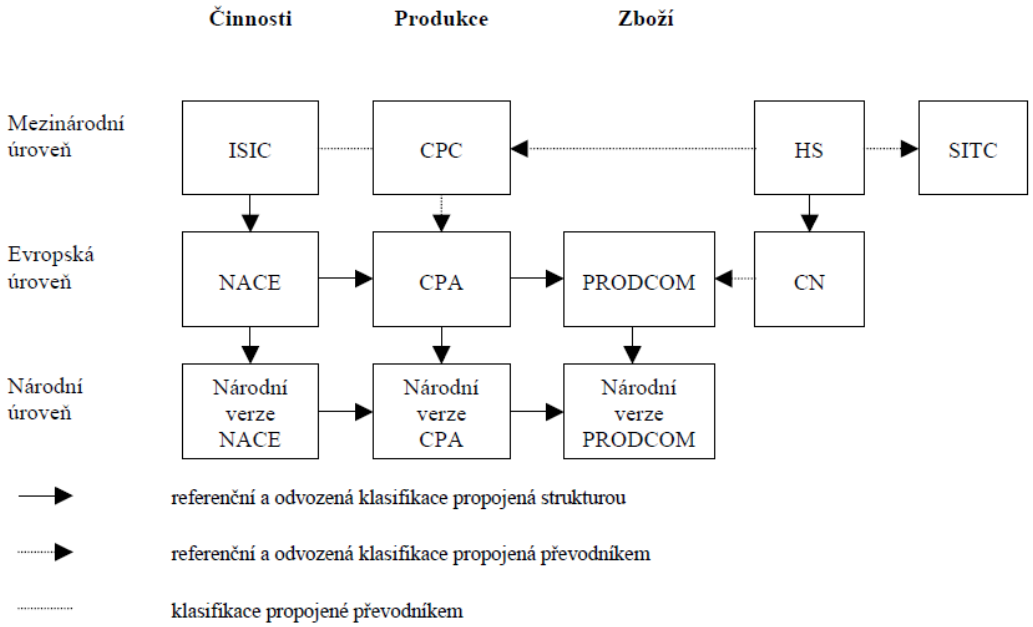
Evropa používá tzv. NACE klasifikaci, která má následující úrovně:

- hlavní sektor označená písmenem začínající od A (třeba agrokultura - A)
- a pak desetinné dělení pod sektorů (s jejich názvy) až do úrovně 3, tedy A1, A1.1, A1.11 (podobně další písmena a hlavní sektory - B, C, ...)

4.2 Návrh ontologie

Konceptuální model představuje vyjadřovací prostředek pro uživatele datové domény k jejímu pochopení. Kromě této informativnosti také najde uplatnění jako metodika při vytváření samotné databáze. Tato metodika zajistí aktuálnost informací mezi oběma zdroji informací.

V této sekci popíšu postup návrhu ontologie, nad kterou portál Learneron pracuje. V první části je popsán postup získání logického modelu reprezentující strukturu dat v původní MySQL databázi. Další abstrakcí z něj vzniká základ ontologie v notaci OntoUML, který je posléze obohacen o popis popis dovedností ESCO a o nabídky vycházející ze specifikace NACE.



Obrázek 4.5 Mezinárodní systém ekonomických klasifikací z [17]

4.2.1 Requirements Engineering

V této fázi je potřeba definovat rozsah ontologie, vytvořit, zvolit pohled na doménu a pochopit hlavní koncepty. Protože doména je příliš komplexní, použiji techniku modularizace domény, vytvořím tedy více subontologií zachycujících různé aspekty domény. Rozsah je definován požadavky spolupráce a zvolený pohled je vysvětlen při popisu subdomén modelu v následujících sekcích.

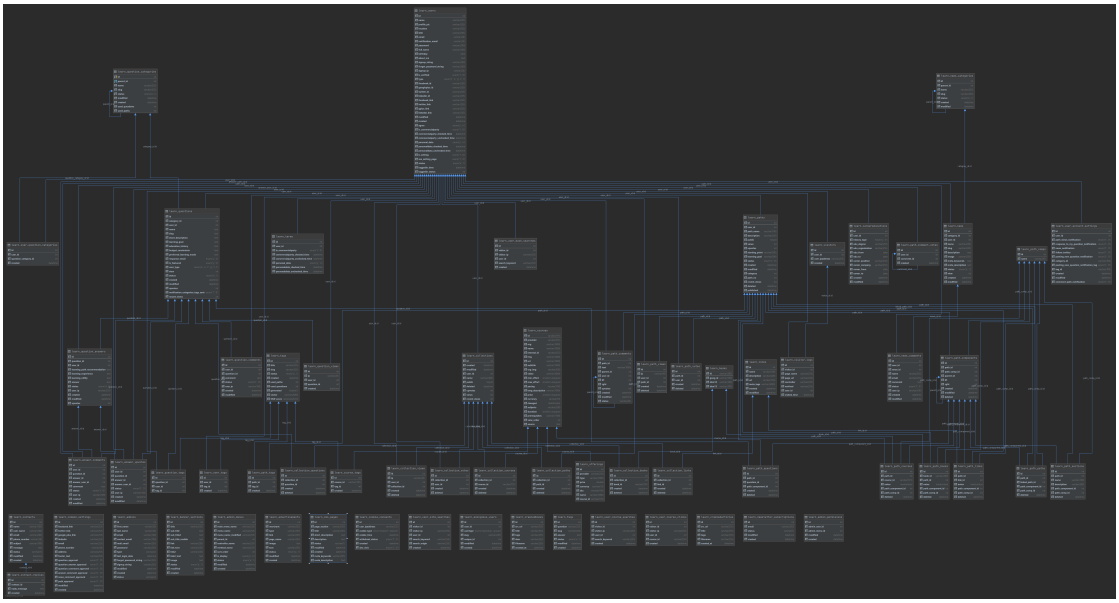
4.2.1.1 Analýza původního kódu

Pro získání schéma, které reprezentovalo původní způsob uložení dat v relacích a vazby mezi relacemi, bylo nutné vytvořit novou databázi s původními daty a vlastnostmi, ale navíc přidat potřebné vazby pomocí cizích klíčů. Důvodem byla neexistence takových vazeb v původním databázovém stroji, který se spoléhal na PHP pro zajištění integrity dat. Nová MySQL databáze pro reprezentaci původních dat pak vypadá následovně 4.6.

4.2.2 Reverse Engineering do nástroje Enterprise Architect

Připojením pomocí nástroje Enterprise Architect k upravené databázi jsem získal prvotní model, který reflektuje stav databáze. Ten slouží jako prostředek pro vytvoření základního konceptuálního schéma.

Takové schéma domény bude udržováno v nástroji Enterprise Architect, v tomto případě bude použit pro tvorbu OntoUML schémat. EA je komerční CASE nástroj pro analýzu a návrh systémů a pokrývá celý jejich life-cycle s UML diagramy.



Obrázek 4.6 Schéma původní MySQL databáze

4.2.3 Postup získání prvotního modelu v Enterprise Architect (EA)

Původně bylo zamýšleno exportovat schéma z programu DataGrip, připojeného k MySQL databázi. Zábranou byl ovšem nekompatibilní XML formát exportovaný touto cestou.

Mezi hlavní body postupu patří následující.

1. Získat přístup k MySQL databázi. Já si potřeboval nainstaloval VPN pro přístup do vnitřní sítě ČVUT, odkud jsou data přístupná.
2. Instalace ODBC driveru pro MySQL vhodné verze, v mém případě byla potřeba 32b verze, jelikož EA pracoval pouze ve 32bit módu.
3. Jelikož EA neumí natáhnout přímo SQL DDL skript. Je potřeba připojením přes ODBC driver natáhnout do EA schéma již existující databáze. Přesný postup je uveden ve veřejně dostupném návodu[a].
4. Po importu DB se ovšem vytvoří akorát tabulky. Pro vytvoření schématu je potřeba na tabulky aplikovat transformace. Přesný postup je opět uveden ve veřejně dostupném návodu[b]. Aplikace transformací vytvoří z fyzického logický model a z něj základ pro konceptuální model.

4.2.4 OntoUML model z logického

Pro dokumentaci jsem zvolil barevné označení, které popisuje výskyt entit, nebo-li na jakém uložišti leží. Toto je klíčové, aby ostatní byli schopni odlišit, které data např. zůstaly v původní relační databázi a nebudou se používat, nebo data, která existují pouze v nové Neo4J databázi.

[a] `https://sparxsystems.com/enterprise_architect_user_guide/14.0/model_domains/importdatabaseschemafromod.html`

[b] `https://community.sparxsystems.com/community-resources/512-84data-modeling-logical-and-conceptual-mda-transforms`

Toto označení je následující.

Původní oranžová pro entity, které pochází z MySQL a nachází se i v Neo4J.

Červená pro entity, které se vyskytují pouze v konceptuálním modelu a tedy jsou momentálně plánovány, jak je do budoucna implementovat.

Zelená pro entity, které se nenachází v MySQL a tedy jsou kompletně nové.

Modrá pro entity, které se nenachází v Neo4J a tedy pro ně není v tomto systému momentální využití.

OntoUML modely účelově neobsahují atributy entit a rozvedení materiálních vztahů. Rozvedení materiálních vztahů je pouhá formalita, která v těchto modelech nepřináší novou informaci, či obměnu. Takováto konvence zároveň zpřehledňuje graf, jelikož ten nemusí obsahovat přehršel doplňujících entit a vazeb. Atributy entit jsou neuvedeny z podobných důvodů. Běžně se sleduje opravdu velké množství atributů entity, to znepřehledňuje graf. Pokud se ukládá nějaká atypická vlastnost pak je zmíněná při popisu entity v odpovídající subdoméně, kde je potřeba. V případě původních dat se ukládají vlastnosti převzaté, občas filtrované z MySQL databáze viditelné na obrázku4.6. U nových dat je to určeno zdroji dat, či průběžnými konzultacemi nad řešením.

I když jsou zdroje A a C různé zdroje dat, pocházejí ze stejné DB a mají mezi sebou vazby potřebné k zachování i do výsledné grafové DB, proto je vhodné realizovat přenos dat dohromady. A to je důvodem, proč jsem zvolil od začátku ontologii těchto zdrojů spojit, protože reprezentují data původní DB, které jsou potřeba i v nové. To je výhodné pro konzultaci s původními vývojáři nad validitou modelu a zároveň mě to vede k většímu spojitému celku výsledné grafové DB, kde již zbývá především vyřešit napojení na zdroj B. Pro připomenutí části zdroje A obsahují především materiály a informace pro doporučování uživatelům, části zdroje C obsahují veškerou aktivitu sledovanou u uživatele, která byla uvedena jako smysluplná pro přiřazování doporučení.

Původní MySQL databáze obsahuje také informace, které nejsou v nové databázi potřeba. Ty v ontologickém modelu ani v budoucí implementaci nebudou. Ve většině případů se jedná o pomocné a administrativní tabulky, některé jsou zastaralé a postupným vývojem ztratily význam. Některé ovšem přinášejí důležité informace a nejsou v modelu pro jejich nepotřebnost v současném využití, nicméně v budoucnu mohou být. Proto popíšu nejdůležitější tabulky s potenciálním budoucím využitím.

News je tabulka s novými informacemi pro uživatele. Jedná se o pravidelné PSA (Public service announcement). K této tabulce pak patří:

- kategorie novinek a
- komentáře k novinkám.

Questions je tabulka s otázkami, ty mohou být pokládány k různým prvkům v aplikaci. Nejčastěji uživatelé vytvářejí otázky u vzdělávacích cest a kolekcí zdrojů vzdělávání. Jako novinky také otázky jsou rozšířeny o další pomocné tabulky. Jimi jsou:

- odpovědi na otázky samotné,
- kategorie otázek a
- komentáře k otázkám.

4.2.4.1 Uživatelská subdoména

Viz model na obrázku 4.7 popisuji následující subdoménu.

Každý návštěvník webu (IP visitor) je sledován pomocí IP adresy — proto toto jméno — a současné session. Tento návštěvník si volí nastavení pro povolení Cookies, které se taktéž

ukládají. Jako anonymní návštěvník může mimo jiné navštěvovat profily jiných uživatelů a podnikat veřejně dostupné interakce. Aby mu byla poskytnuta plná funkcionalita aplikace musí se přihlásit. Po takovém přihlášení se u návštěvníka vytvoří nová session a vede identifikátor uživatele. Uživatel má přístup ke privátním interakcím. Uživatel si mimo jiné volí jistá nastavení pro svou osobu a podmínky využívání. Nakonec pokud mu byl vybrán doporučený obsah, nebo především si zvolil nějaký obsah za oblíbený (chtěl jej sledovat), doporučuje se mu tento obsah.

Následuje popis entit samostatně, navazující na úvodní popis.

IP Visitor : návštěvníkovy informace

Cookie consent : nastavení Cookies návštěvníka

Public Interaction : sada ukládaných veřejně dostupných interakcí návštěvníka

Logged Visitor : informace přihlášeného návštěvníka

Anonymous Visitor : informace nepřihlášeného návštěvníka

User : informace běžně představitelného uživatele aplikace

Private Interaction : sada ukládaných uživateli dostupných interakcí

Settings : nastavení uživatele

Terms : souhlasy uživatele s podmínkami užití služby

Feed : přiřazený seznam informací uživateli na základě volby sledování

Followable : sledovatelná položka pro Feed

4.2.4.2 Subdoména zdrojů informací

Viz model na obrázku 4.8 popisuji následující subdoménu.

Všechny zdroje informací patří do skupiny Resource. Jedná se o kurzy, kolekce dalších zdrojů, vzdělávacích cest obsahující další zdroje, knihy, linky, či v budoucnu videa a podcasty.

Následuje popis entit samostatně, navazující na úvodní popis.

Resource : jakýkoliv zdroj informací, označení pro výukový materiál, zastřešující konkrétní položky

Course je externí výukový kurz nabízená Learneronem jako prostředníkem.

Collection je sada výukových materiálů, není setříděná a je plochá (nemá možnost vnořování pomocí vkládání dalších kolekcí).

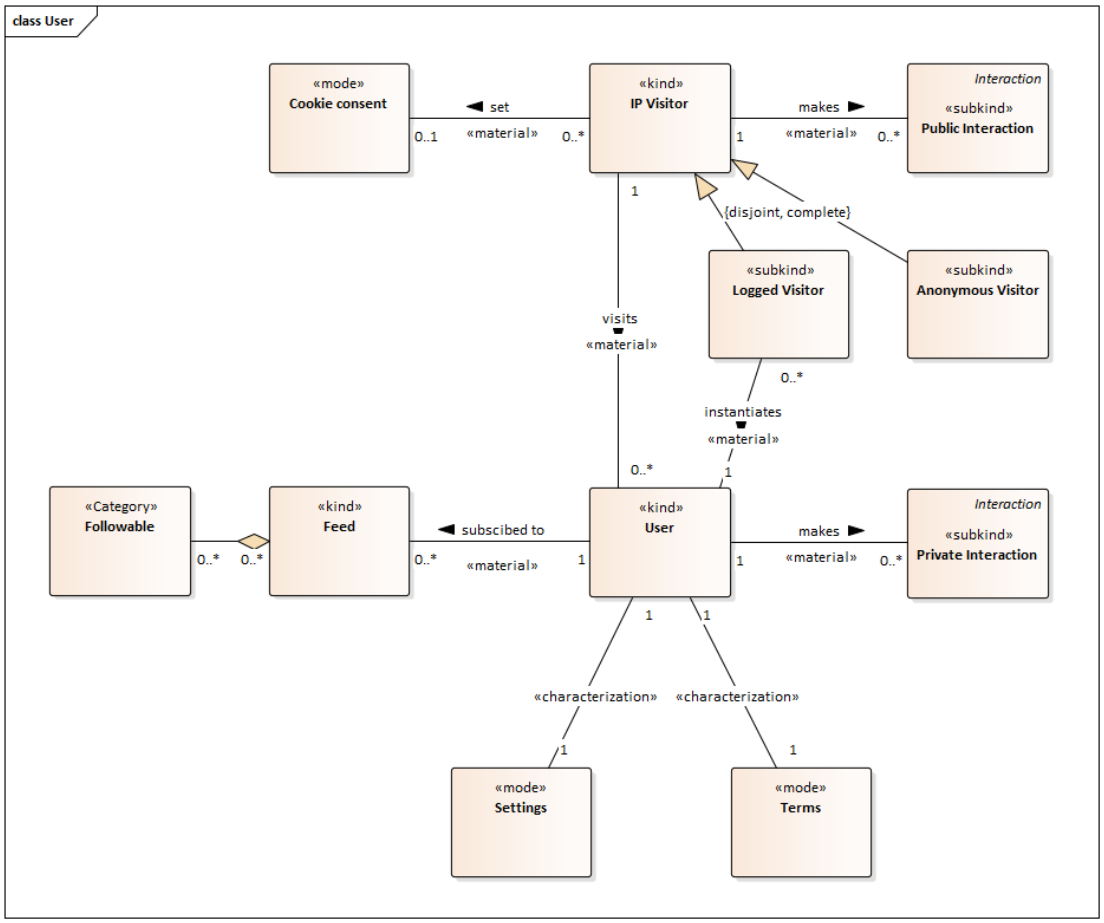
Learning Path slouží jako kurikulum, je to sada výukových materiálů, je setříděná, jedná se o strom pro využití posloupnosti při učení.

Book : entita obalující informace o knize

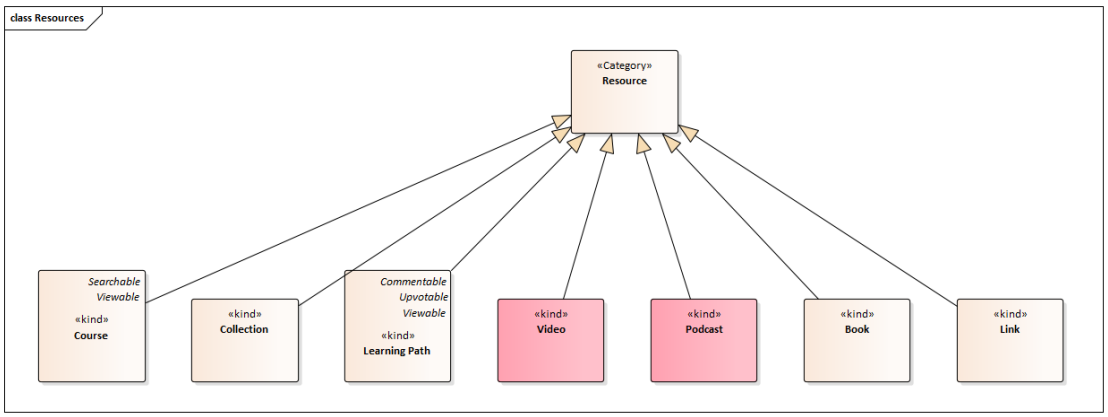
Link je zdroj informací, na jehož obsah ukazuje link.

Video : reprezentace videa, je plánovaný druh výukového materiálu, který v době psaní není ještě používán. Současně jej lze reprezentovat jako Link.

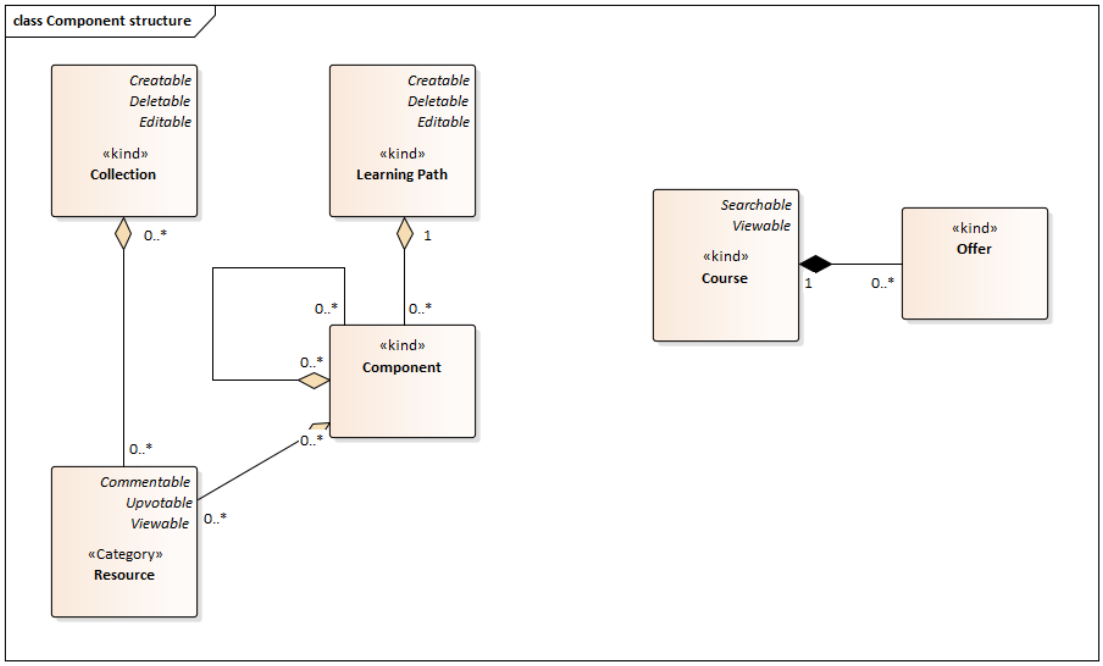
Podcast : reprezentace podcastu, je plánovaný druh výukového materiálu, který v době psaní není ještě používán. Současně jej lze reprezentovat jako Link.



Obrázek 4.7 Ontologický Model uživatelské subdomény



Obrázek 4.8 Ontologický Model pro popis zdrojů informací



Obrázek 4.9 Ontologický Model strukturalizace komponent

4.2.4.3 Struktura komponent

Viz model na obrázku 4.9 popisuji následující subdoménu.

Jak kolekce (Collection) tak vzdělávací cesta (Learning Path) se skládají z ostatních zdrojů informací. V tom je jejich hodnota, tedy slouží jako seznam zdrojů. Zatímco kolekce je jednoduchá neseřazená množina zdrojů informací, vzdělávací cesta je komplikovanější. Vzdělávací cesta je seznam komponent, tyto komponenty se mohou vnořovat a slouží jako seznam zdrojů informací. Tato reprezentace má představovat strom a je zvolena pro vhodnost provádění výukovými materiály, jak aplikace potřebuje. Kurz je definován nabídkami společnostmi, které jej nabízí a vyučují. Cílem je sledovat nabídky, doporučovat uživateli ty nejvhodnější a sbírat historické data.

Následuje popis entit samostatně, navazující na úvodní popis.

Resource : položka informací uložitelná do seznamu

Collection : definována v seznamu 4.2.4.2

Learning Path : definována v seznamu 4.2.4.2

Component je část vzdělávací cesty sloužící pro ochovávání zdrojů dat

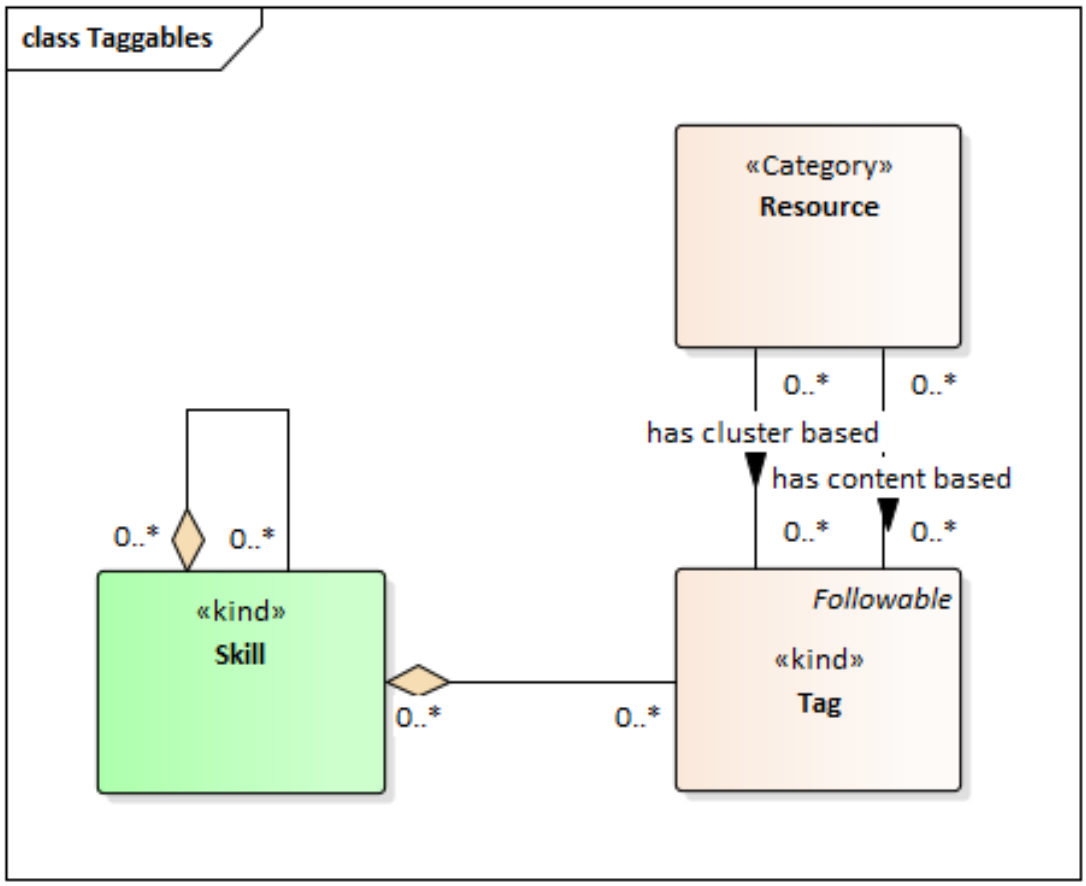
Course : definován v seznamu 4.2.4.2

Offer : cenová nabídka poskytovatele daného kurzu

4.2.4.4 Struktura štítkování

Viz model na obrázku 4.10 popisuji následující subdoménu.

Štítkování slouží jako způsob indexace podobných informací. V tomto případě se štítkují veškeré zdroje dat i znalosti pro jednu množinu štítků. Tyto tagy jsou generovány pomocí technologie TextRank na straně Learneronu a slouží pro nalézání společných dat. Při vytváření takových



Obrázek 4.10 Ontologický Model pro popis štítkování

štítků, tak se provádí kontrola na podobné a duplicitní tagy. Duplicitní jsou dobré, jelikož to znamená, že v databázi budou data propojena. Podobné ovšem jsou problematické, je potřeba nastavit dostatečnou podobnost, aby bylo možné provázat materiály, ale je potřeba dostatečné diverzity, aby data byly odpovídající, ne úplně mimo. Po takovém nastavení se materiály provážou s podobnými existujícími štítky, aby nevznikaly nové bez přidávající hodnoty.

Ze zdrojů dat jsou generovány štítky dvěmi způsoby:

- Na základě popisu — atributu *description* — jednotlivého zdroje dat (Content based).
- A na základě kolektivní podobnosti zdrojů ve skupinách (Cluster based).

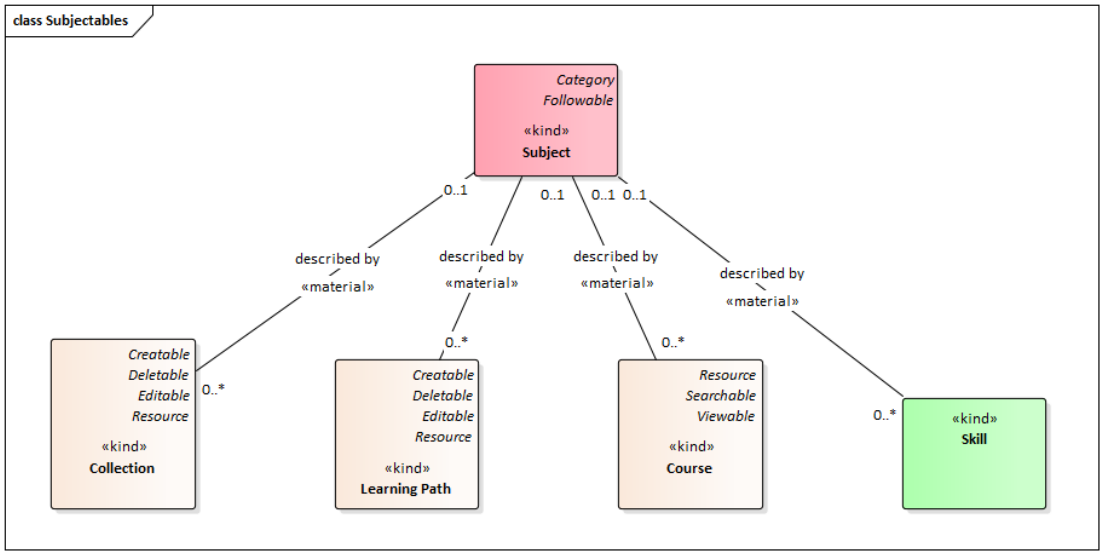
V případě znalostí se generují opět z popisu znalosti.

Následuje popis entit samostatně, navazující na úvodní popis.

Tag je štítek pro označování především výukových materiálů a znalostí. Na základě společných tagů se pak filtrují nabídky pro uživatele.

Resource : definován v seznamu 4.2.4.2

Skill : schopnost, nebo znalost uživatele určité činnosti, více v sekci 4.2.5.1



Obrázek 4.11 Ontologický Model pro popis předmětů

4.2.4.5 Struktura popisovaní předměty

Viz model na obrázku 4.11 popisuji následující subdoménu.

Podobnou funkci jako štítkování, ale obecnějšího charakteru je popisování předmětem zaměření. Tato kategorizace popisuje komplexní zdroje informací — tedy jeho podmnožinu — jako: kolekce, vzdělávací cesta, kurz a znalost. Každý obsahuje ale pouze jeden takovýto předmět narozdíl od štítků. Použití je podobné jako v případě štítkování, propojení dat podobného charakteru. Generování takových předmětů a jejich přiřazení závisí na daném materiálu. V případě kolekce a vzdělávací cesty závisí na obsahu seznamů. A v případě kurzu a dovednosti se opět přiřazuje podle vlastního popisu. Kontrolování nově vzniklých předmětů je tak stejně třeba jako v případě štítků.

Následuje popis entit samostatně, navazující na úvodní popis.

Subject je předmět označující druh materiálu. Jedná se obecnější variantu Tagu. Na základě společných předmětů se pak opět vybírá uživateli relevantní obsah.

Collection : definován v seznamu 4.2.4.2

Learning Path : definován v seznamu 4.2.4.2

Course : definován v seznamu 4.2.4.2

4.2.4.6 Struktura uživatelských interakcí

Viz model na obrázku 4.12 popisuji následující subdoménu.

U každého návštěvníka se sledují jeho interakce s aplikací, pro lepší porozumění jeho návykům a potřebám. Interakce se dělí na veřejně dostupné a privátní. Veřejné interakce jsou přístupné každému návštěvníku a také se pro každého ukládají zvlášť. Privátní interakce jsou přístupné pouze přihlášeným osobám, ty pak mohou pracovat s aplikaci, jak bylo zamýšleno. Může vytvářet nové vzdělávací cesty, interagovat s ostatními uživateli, přihlásit se a učit se dle vybraných plánu atd. Druhé dělení interakcí se zakládá na typu činnosti.

Práva pro vytvoření takovéto interakce — jako podobné přístupy — jsou řešeny na aplikační úrovni než jsou předány databázi pro spuštění.

Tyto činnosti jsou následující.

Vytváření (Creation) : interakce, která vytváří prvek, který patří mezi Vytvořitelné

Okomentování (Comment) : interakce, která vytváří komentář pro prvek, který patří mezi Komentovatelné.

Zobrazení (View) : interakce, která zobrazí prvek, který patří mezi Zobrazitelné

Vymazání (Delete) : interakce, která maže prvek, který patří mezi Mazatelné

Hlasování (Upvote) : interakce, která hlasuje o prvku, který patří mezi Hlasovatelné

Vyhledávání (Search) : interakce, která vyhledává zadané dotaz, tato interakce se oproti ostatním neváže na ostatní materiály, ale obsahuje jako atribut hledaný výraz, pro sledování zájmu uživatele. Pokud si uživatel po vyhledání zobrazí nějaký materiál, pak už se to ukládá jako interakce Zobrazení.

Editování (Edit) : interakce, která upravuje prvek, který patří mezi Editovatelné

Sledování (Follow) : interakce, která slouží ke sledování oblíbených a zvolených štítků, uživatelů, či předmětů.

Důvody proč jsou interakce takto koncipovány tedy např. Uživatel vytváří interakci komentování pro entitu vzdělávací cesty, namísto Uživatel komentuje entitu vzdělávací cesty jako na obrázku4.13, je především tento. Taková reprezentace není rozšiřitelná. Kvůli aplikace pravidel z sekce 4.3.1 by takováto vazba v grafové databázi degradovala na jednoduchou hranu. Na takovou reprezentaci komentáře by pak nešly vést další vazby, jelikož Neo4J nepodporuje něco jako vedení hrany na hranu. Takovou vazbou je například ukládání interakce Hlasování, která je aplikovatelná i na Komentář. Interakce komentování tedy musí být samostatnou entitou namísto pouhého relátoru, proto aby na ní šel vést vztah — oznamující, že tento komentář byl předmětem hlasování — ve výsledné databázi. Proto tato informace o komentáři bude vedena jako součást dané interakce samostatně.

Dalším ne tak pragmatickým důvodem je dokumentace. Protože model slouží jako dokumentace stavu domény je vhodné seskupovat interakce k sobě namísto jejich roztříštění bez očividně existující ontologické vazby.

Interaktovatelné objekty jsou definovány na obrázku 4.14. Přesněji dle následujícího textu.

Creatable : kategorizace pro vytvořitelné objekty

Commentable : kategorizace pro komentovatelné objekty

Viewable : kategorizace pro zobrazitelné objekty

Deletable : kategorizace pro mazatelné objekty

Upvotable : kategorizace pro hlasovatelné objekty

Editable : kategorizace pro upravovatelné objekty

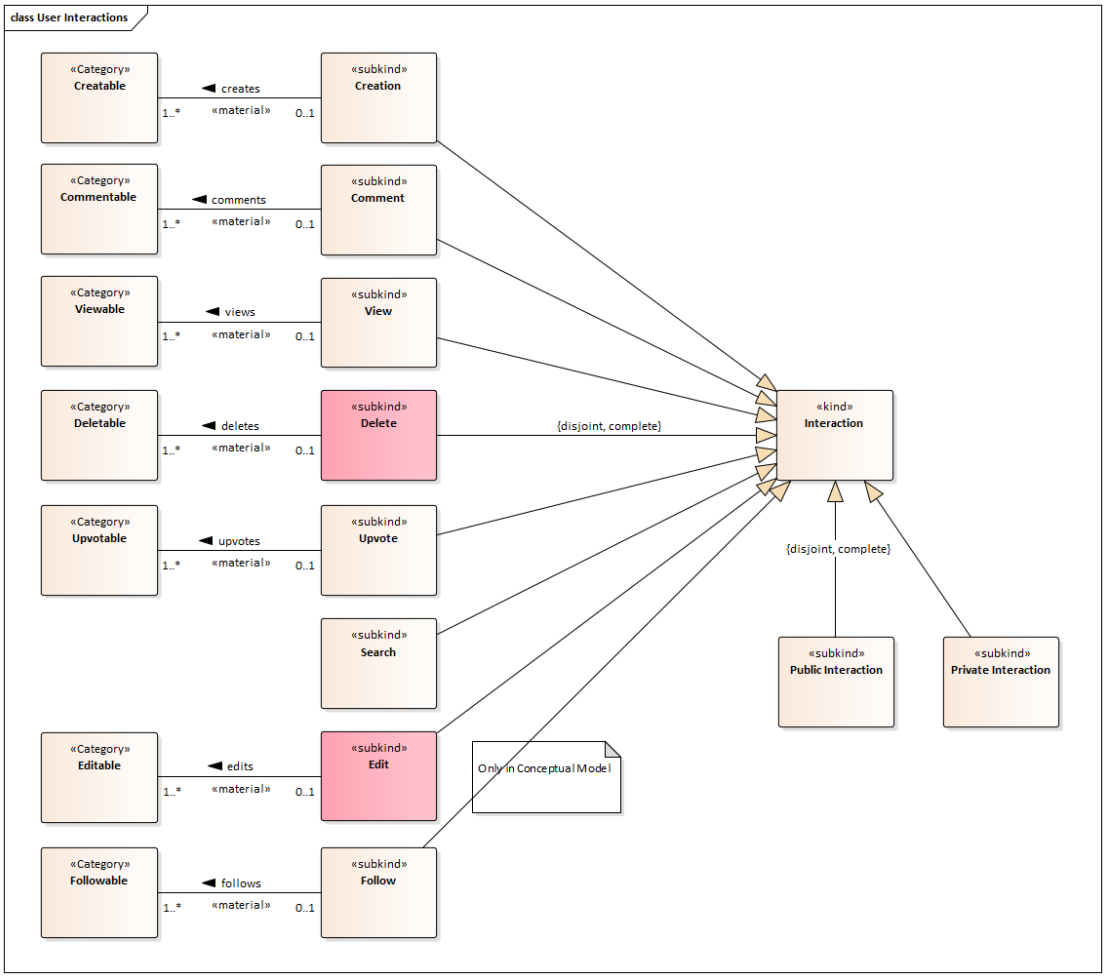
Followable : kategorizace pro sledovatelné objekty

Následuje popis zbývajících entit samostatně, navazující na úvodní popis.

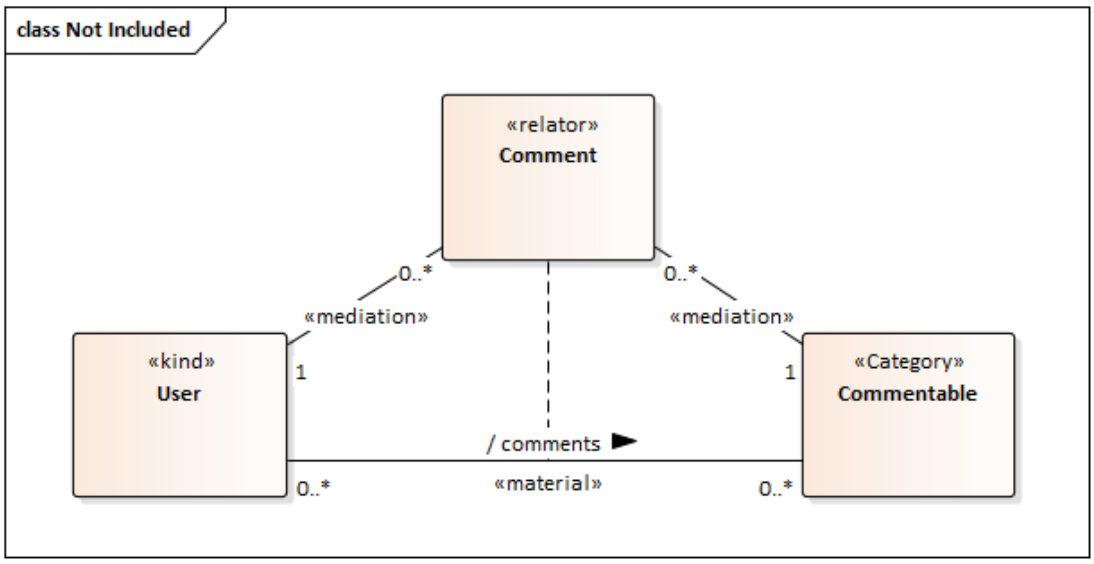
Interaction : reprezentace interakce na nejobecnější úrovni

Public Interaction : kategorie interakce přístupná libovolnému návštěvníkovi

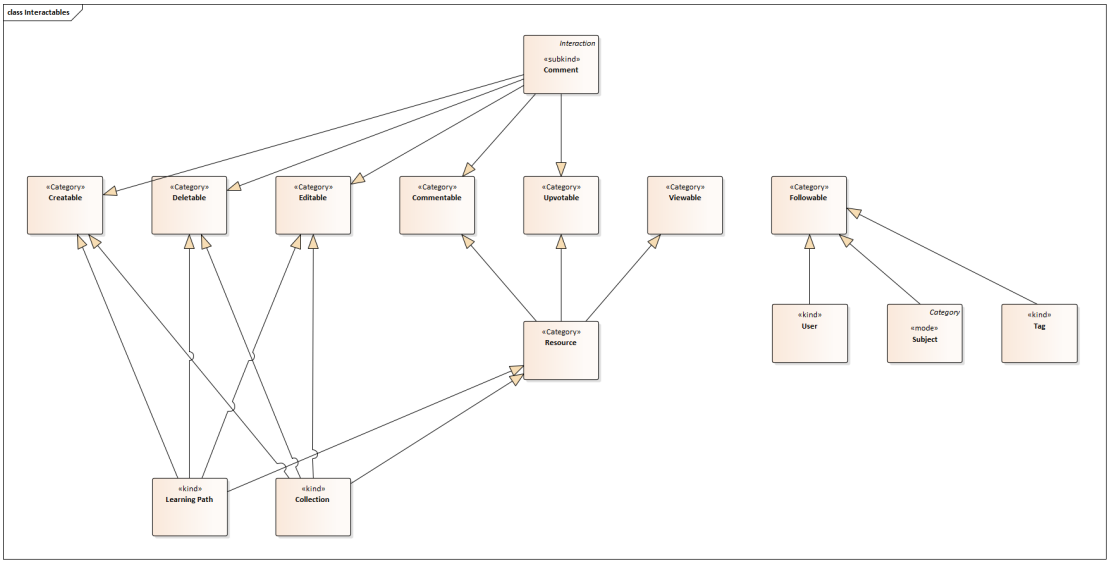
Private Interaction : kategorie interake přístupná pouze přihlášeným návštěvníkům



Obrázek 4.12 Ontologický Model pro popis uživatelských interakcí



Obrázek 4.13 Model původně zamýšlený pro reprezentaci interakce pro komentování



Obrázek 4.14 Model interaktovatelných objektů

4.2.5 Obohacení OntoUML modelu o ESCO graf a další požadavky na rozvoj databáze

4.2.5.1 ESCO model

Vyobrazen jeho úsek na obrázku 4.15. Je projektem EU jakožto systém dat pro klasifikaci dovedností, kompetencí, kvalifikací a povolání v EU. V práci potřebuji akorát část, která popisuje a třídí dovednosti a povolání. Struktura těchto entit a jejich vzájemné napojení mezi sebou je převzata z poskytovaných datových zdrojů[c] se CSV soubory.

Při pohledu na konkrétní data na obrázku 4.16 lze vidět následující. Pozice laboratorního technika (zelená) patří pod skupinu dalších pozic ISCOGroup (červená). Tato pozice sdílí s ostatními pozicemi některé znalosti (oranžová), které jsou buď vyžadovány, nebo jsou vhodné. Některé z těchto znalostí patří do skupin znalostí (modrá).

Následuje popis entit samostatně, navazující na úvodní popis.

Skill : znalost/ schopnost získatelná osobou

Skill Group : shrnutí skupiny znalostí

Occupation : pracovní pozice, v budoucnu je plánována implementace sledování růstu platu a poptávky takové pozice v dané zemi, uživateli se tak bude doporučovat podobná pozice, o kterou je v současnosti velký zájem, a firmy hledající takové lidi

Occupation Classification ISCO : klasifikace pracovních pozic dle mezinárodní metody ISCO, více v sekci 4.1.2.

Description : vlastní popisek znalosti, pro který se pomocí similarity vytváří hrana k podobným zdrojům informací, tato hrana reprezentuje spojení, jaké znalosti daný materiál poskytuje

Resource : definován v seznamu 4.2.4.2

Tag : definován v seznamu 4.2.4.4

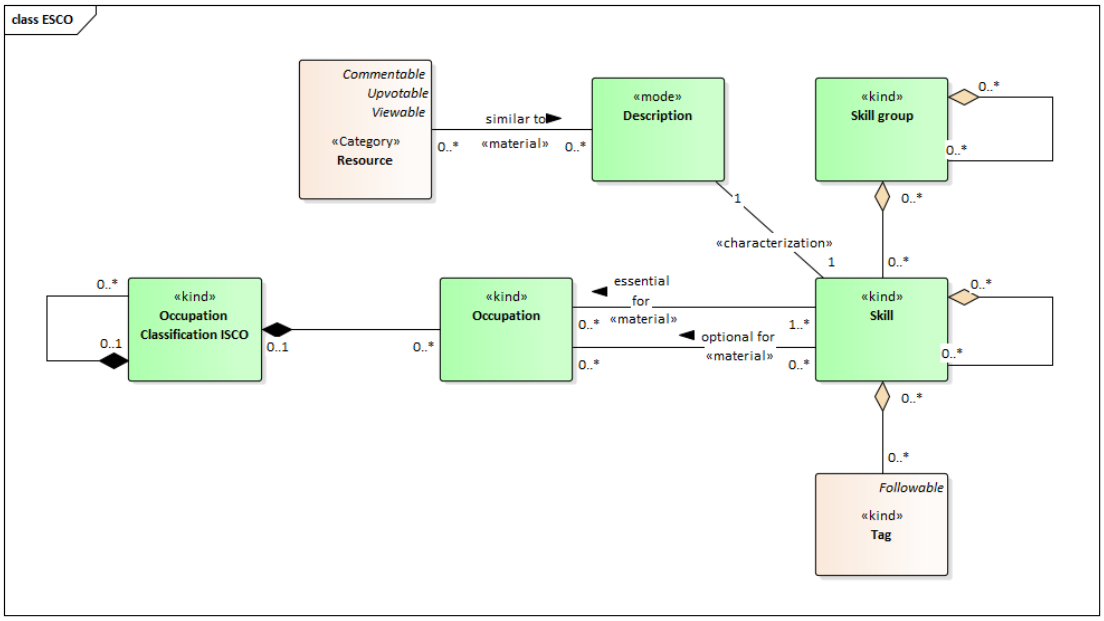
Propojení ESCO grafu ke zbytku je prostřednictvím následujících.

- Skill získaný/ podobající se výsledkům ze materiálů.
- Skill oštítkovaný tagy.
- Occupation se vztahuje k uživateli pomocí jeho pracovního záznamu. Do budoucna budou Occupation navázány také na informaci růstu/poklesu poptávky/ohodnocení v různých zemích.

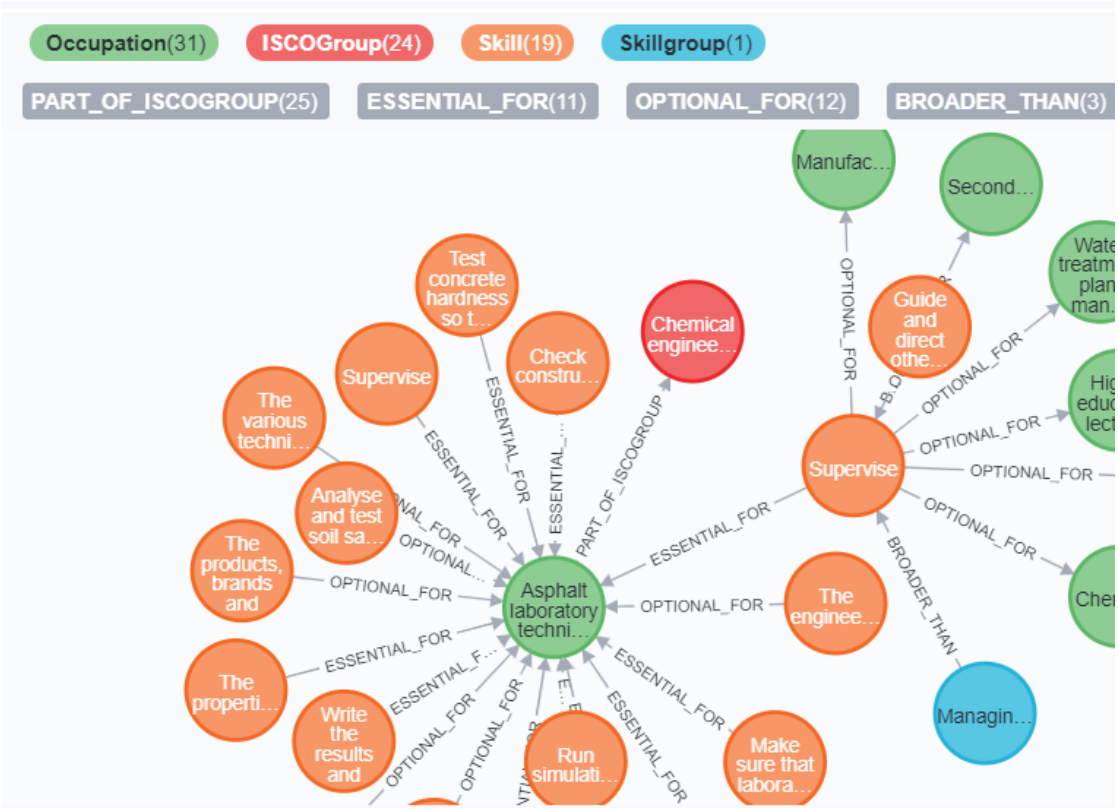
4.2.5.2 Nabídka trhu

Principem je možnost využití informace o současném stavu trhu. Jako jsou rostoucí i klesající poptávky po pracích a dovednostech. Zdrojem dat jsou měsíční výpisy[d] LinkedIn ekonomického grafu. Lze tak i sledovat poptávka po práci a po dovednosti ve vybrané zemi. Člověk se tak může zaměřit na nejpodobnější práci jeho dovednostem a firma operující v takové zemi, pak může i takové lidi oslovit.

[c] `https://ec.europa.eu/esco/portal/download`
[d] `https://graph.linkedin.com/insights/labor-market`



Obrázek 4.15 Ontologický Model pro ESCO



Obrázek 4.16 Struktura ESCO dat

4.2.5.3 Subdoména sledování poptávky

Viz model na obrázku 4.17 popisuji následující subdoménu.

Tato subdoména slouží pro ukládání historických statistik, momentálně se jedná pouze o externí historická data LinkedInu pro popularitu pracovních pozic a dovedností. Bude ale sloužit pro ukládání i interních před-vypočítaných dat, které se nevyplatí počítat real-time. Takové data budou různého charakteru a vázat se na jiné entity, nicméně budou pod druhem Výběru (Selection).

Momentálně existující výběr popisuje popularitu pracovních pozic a znalostí dle specifikované země a ekonomické aktivity NACE získané pomocí scrapingu od LinkedInu. Tato data jsou klíčová pro získání kontextu trhu minimálně v začátcích, kdy Learneron nemá sám takovou historii sbírání podobných dat. Plánem je později se od těchto externích dat odklonit, jak nejvíce to jde. Takovými důvody jsou např.

- Nezávislost na externím poskytovateli. LinkedIn poskytuje tyto data zdarma, problém ovšem nastane, pokud by přístup k ním ještě více omezil.
- Přizpůsobenost dat do velké míry nelze očekávat od nikoho jiného než sebe. Nejenže se jedná o „know-how“ LinkedInu, ale vytváří to její velkou konkurenci schopnost, která by mohla být nabourána projekty. které by získaly přístup k podobným datům.
- Nestálost dat je velký problém u těchto dat. Data set je sice poskytován každý měsíc, pro stejné země a kategorie výroby, problém ale leží v samotných předmětech příčky. Data ukazují pouze nejlépe hodnocené, což znamená např. víme, že pracovní pozice X byla minulý měsíc nejpopulárnější v naší zemi, ale tento měsíc jelikož není mezi top n pozicemi, tak o ní nemáme jednoduše žádně tušení, zdali je pouze na pozici $n+1$, nebo se propadla a není o ni žádný zájem. V tomto případě se jedná o relativně akceptovatelný problém, ale pokud se některé pozice neobjeví v tomto reportu např. pár let nebo vůbec, pak o současném stavu nemůžeme vůbec nic uvozovat.

Ekonomická aktivita, pracovní pozice a znalost je položka výběru popularity, která protože není kompatibilní se zbytkem používaných dat musí být slinkována pomocí similarity jednotlivých dat. To znamená ...

- K jednotlivým ekonomickým aktivitám výběru se musí hledat a vytvářet hrany podobnosti k ekonomické klasifikaci NACE.
- K jednotlivým pravocním pozicím výběru se musí hledat a vytvářet hrany podobnosti k pracovním pozicím ESCO.
- K jednotlivým znalostem výběru se musí hledat a vytvářet hrany podobnosti k znalostem ESCO.

Následuje popis entit samostatně, navazující na úvodní popis.

Selection : výběr předpočítaných statistických dat, jakéhokoliv charakteru

Popularity Selection : výběr dat zaměřující se na popularitu položek

Selected Activity : položka výběru obsahující informace o dané ekonomické aktivitě

Economic Classification NACE : přiřazená NACE aktivita k dané položce, která je předmětem výběru (znalost, nebo pracovní pozice)

Country přiřazená země k dané položce, která je předmětem výběru (znalost, nebo pracovní pozice)

Occupation Popularity : druh výběru popularity popisující zájem o pracovní pozici

Popular Occupation : položka výběru obsahující informace o dané pracovní pozici

Occupation : přiřazená pracovní pozice k dané položce, která je předmětem výběru (znalost, nebo pracovní pozice)

Skill Popularity : druh výběru popularity popisující zájem o znalost

Popular Skill : položka výběru obsahující informace o dané znalosti

Skill : přiřazená znalost k dané položce, která je předmětem výběru (znalost, nebo pracovní pozice)

4.2.5.4 Subdoména okolo popisu země

Viz model na obrázku 4.18 popisuji následující subdoménu.

Podobně jako štítky, tak země je ústředním bodem spojující různé domény ke spojení dat. Udávají pozici uživatele, ve které se vyskytuje, tedy má i případně zájem o práci. Udává geografickou pozici pro nabízené práce. Z čehož si lze jednoduše domyslet k čemu se bude především využívat… budou se doporučovat blízké pozice (geograficky a znalostmi). Udává taky zemi činnosti organizace, což je jednotka nabízející samotné pracovní příležitosti.

Následuje popis entit samostatně, navazující na úvodní popis.

Country : entita reprezentující označení pro zemi

User : přihlášený uživatel se zájmem pro vyhledávání geografickým způsobem

Job Description : „vylepená“ nabídka společnosti hledající v určité zemi

Organization : označení organizace pusobící na daném uzemí

Company : organizace hledající schopné lidi pomocí veřejných nabídek

4.2.5.5 Subdoména organizace

Viz model na obrázku 4.19 popisuji následující subdoménu.

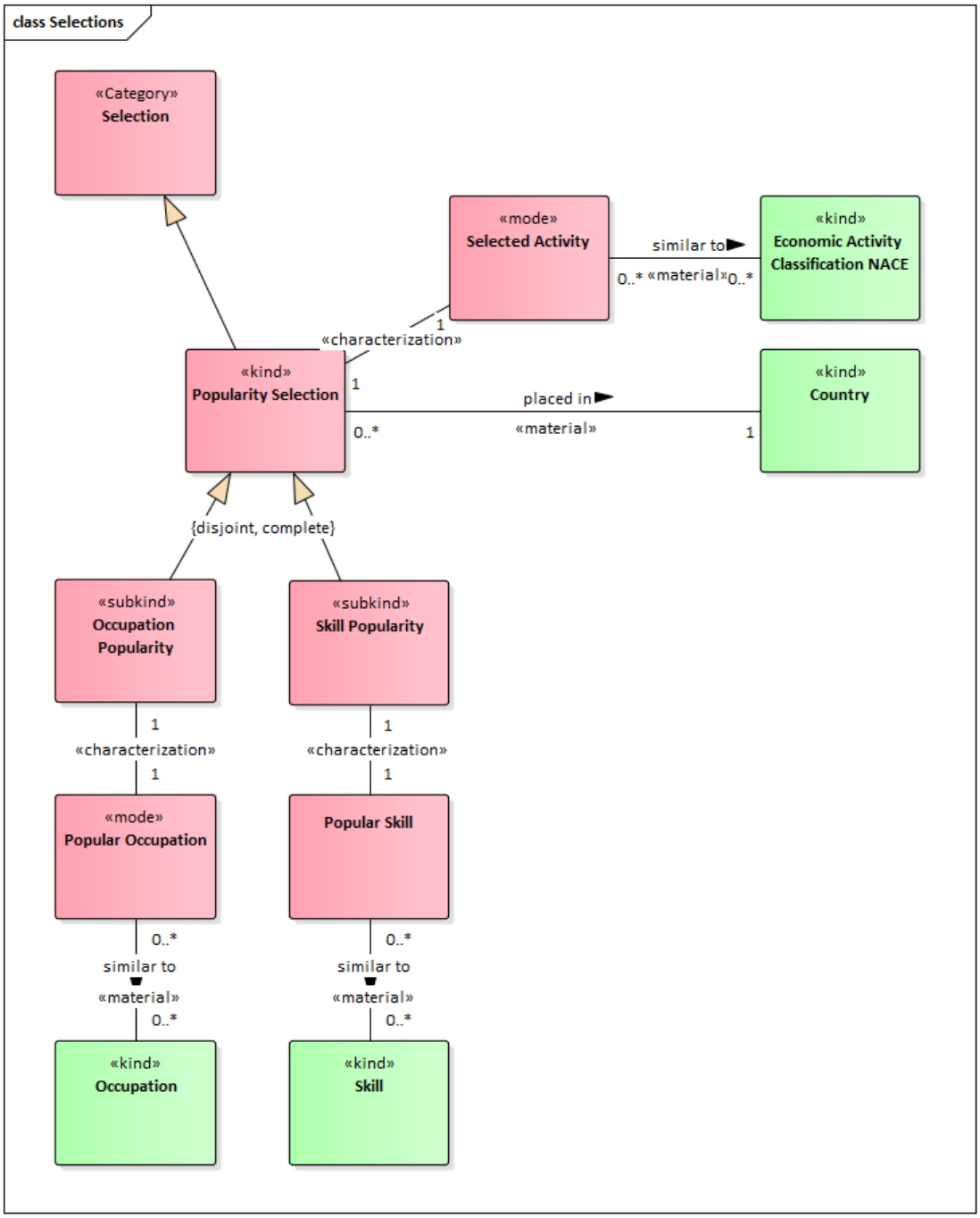
Tato subdoména popisuje strukturu organizace a její vztah s uživatelem.

U uživatele se běžně sledují:

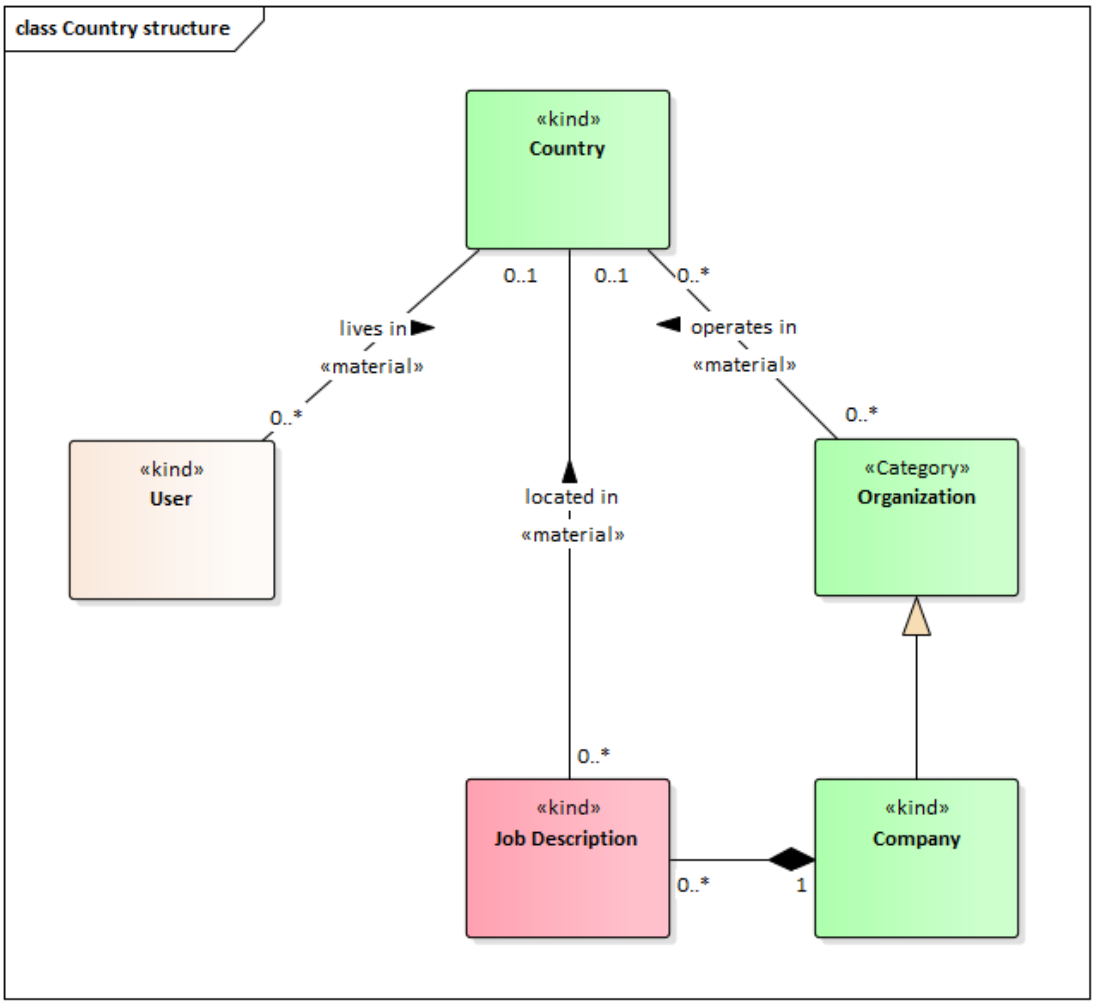
- pracovní pozice,
- vzdělání,
- škola,
- titul,
- specializace,
- dovednosti a
- obor zájmu.

A tato doména slouží k doplnění takových věcí, které ještě nebyly uvedeny v předchozích modelech.

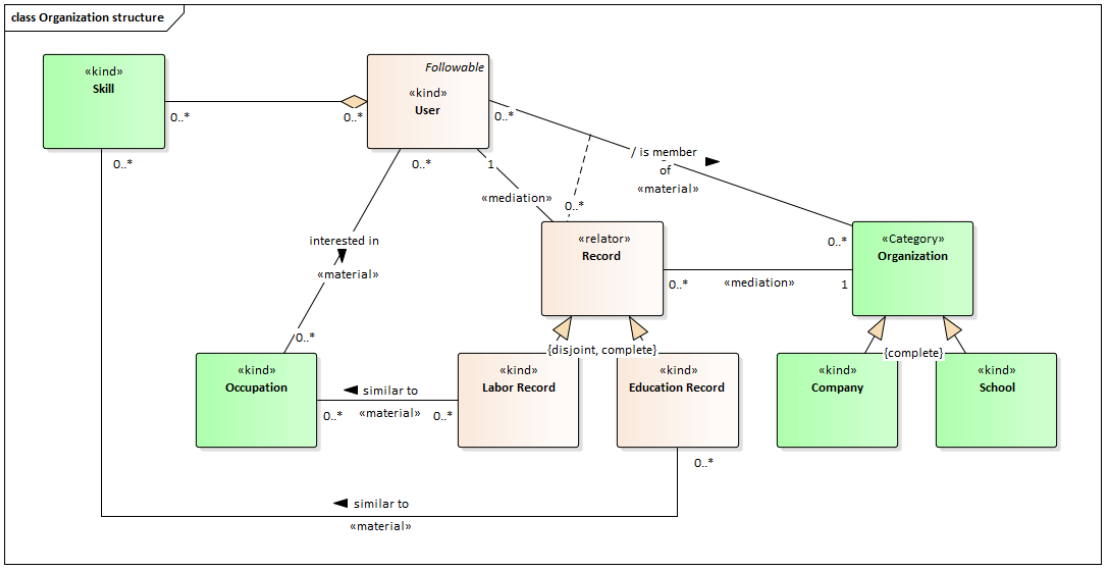
Uživatel se schopnostmi a zájmy v určitých pracovních pozicích, má možnost doplnit informace o svém spojení s organizacemi. Takové spojení je zprostředkováno Záznamy. Tyto záznamy



Obrázek 4.17 Ontologický Model pro výběry dat



Obrázek 4.18 Ontologický Model pro popis zemí



Obrázek 4.19 Ontologický Model pro popis organizace

stejně jako organizace jsou dvojího druhu. Uživatel může uvést zkušenost pracovního, nebo výučního charakteru. Neboli zdali v organizace pracoval, či byl studentem. Organizace tedy mohou být kategorizovány jako vyučovací, či typické společnosti za účelem profitu. Je důležité poznamenat, že druh organizace nezavazuje druh vztahu, ani naopak. Uživatel mohl pracovat jako učitel ve škole a stejně tak mohl být ve firmě s primárním cílem vzdělání sebe sama např. na stáži.

Zatímco u druhu záznamu je jednoduché z vlastní osoby poznat rozdíl, tak u organizace, záleží na hlavní činnosti organizace. Taková výuková organizace pak bude spíše doporučena uživateli pro zvážení jeho pozornosti o ni, přestože to může být soukromá škola a vydělávat peníze.

Pomocí similarity pracovního resp. vzdělávacího záznamu s pracovní pozicí resp. znalostí se vytvoří hrana pro lepší informaci k personalizaci obsahu uživateli. Oba záznamy se vážou na znalosti získané touto zkušeností. Pracovní záznam se váže na znalosti prostřednictvím pracovních pozic, které tyto znalosti již mají přiřazené. Vzdělávací záznamy takovou možnost nemají, proto se vážou na znalosti přímo.

Následuje popis entit samostatně, navazující na úvodní popis.

User : definován v seznamu 4.2.4.1

Organization : reprezentace libovolné zaznamenávané organizace

Company : obchodní společnost

School : organizace jako vzdělávací centrum

Record : záznam o vztahu uživatele se společností

Labor Record : pracovní typ vztahu, zaznamenává zaměstnání

Education Record : vzdělávací typ vztahu, zaznamenává studium

Occupation : definován v seznamu 4.2.5.1

Skill : definován v seznamu 4.2.5.1

4.2.5.6 Subdoména společnosti

Viz model na obrázku 4.20 popisuji následující subdoménu.

Vzhledem k tomu, že služba bude nabízena kromě soukromým osobám i právním, dává smysl se opravdu zaměřit na detailní ontologii společností. Společnostem budou nabízeny služby, které běžnému uživateli přístupné nebudou. Uživatel zaregistrovaný jako zaměstnanec dané společnosti bude vnímán jako firemní uživatel. Podmnožinou takových uživatelů budou firemní manažeři, kteří budou spravovat korporátní uživatele. Firemní uživatel má přístup k soukromým zdrojům informací např. pokud si společnost koupí hromadnou licenci na kurz. Soukromé zdroje jsou pod druhem zdrojů, jsou dostupné pouze uživatelům společnosti vlastnící je. Běžně dostupné zdroje jsou pak dostupné hromadně komukoliv. Každá společnost bude označena klasifikací NACE (více v sekci 4.1.3) pro kategorizace při potřebném vyhledávání. Také bude moct nabízet pracovní pozice, bude ji tak (s povolením uživatelů) nabízen seznam vhodných kandidátů na jejich prázdné místo.

Následuje popis entit samostatně, navazující na úvodní popis.

Company : reprezentace obchodní společnosti

User : běžný uživatel

Corporate User : typ uživatele, jehož účet byl vytvořen pro účely využívání zaměstnanci dané společnosti

Learning Manager : firemní uživatel se speciálními právy pro správu postupu přiřazené skupiny firemních uživatelů

Resource : libovolný zdroj informací

Corporate Resource : zdroj dat dostupný pouze patřící firmě

Public Resource : zdroj dat dostupný jakémukoliv uživateli

Economic Activity Classification NACE : klasisikace činností společnosti pro kategorizaci firmy

NACE Section : první úroveň ze sekcí klasifikace NACE, vyžnačuje se písmenem a to od „A“ až po „U“, poté následuje v označení až čtyřčíslí konkretizující danou aktivitu, nebo obecnější varianty vypsané jako dvojčíslí, či trojčíslí

NACE Division : druhá úroveň ze sekcí klasifikace NACE, vyžnačuje se dvojčíslím a to od „01“ až po „99“

NACE Group : třetí úroveň ze sekcí klasifikace NACE, vyžnačuje se trojčíslím a to od „01.1“ až po „99.0“

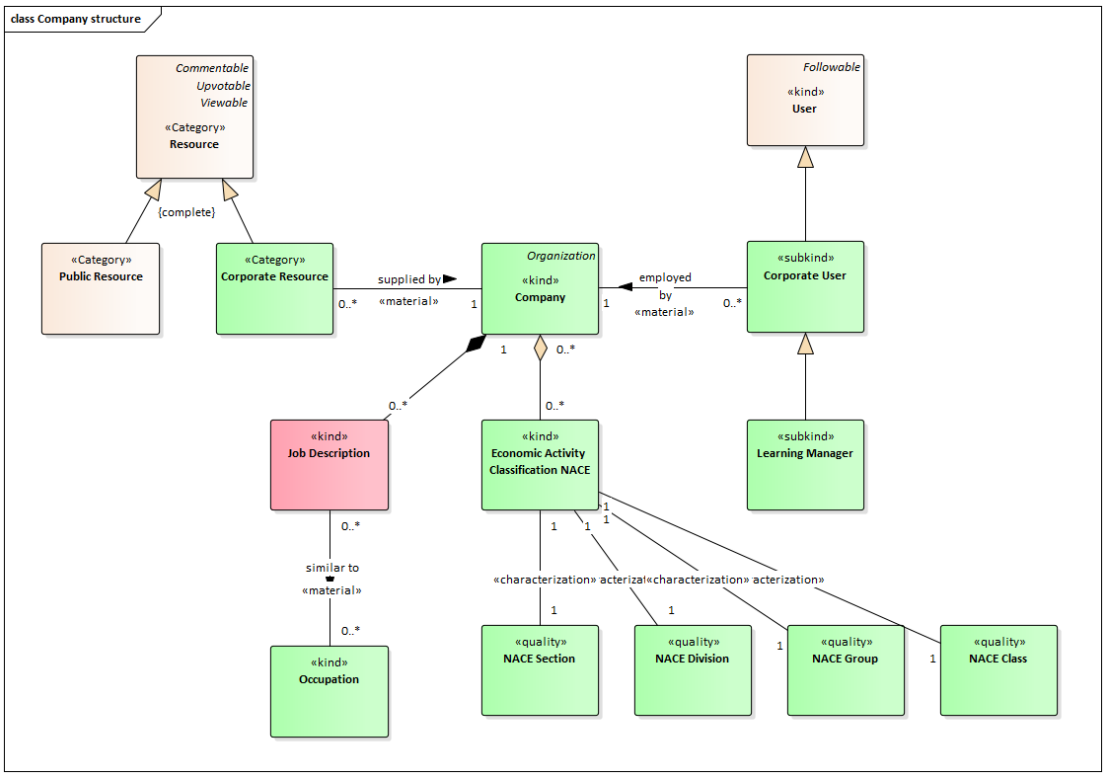
NACE Class : čtvrtá úroveň ze sekcí klasifikace NACE, vyžnačuje se čtyřčíslím a to od „01.11“ až po „99.00“

Job Description : popis pracovní pozice, jež společnost potřebuje

Occupation : definován v seznamu 4.2.5.1

4.3 Implementace grafové databáze

V této sekci se věnuji implementaci navržené ontologie v grafové databázi Neo4j. Nejprve je třeba definovat pravidla, jak ontologii zapsanou v notaci OntoUML implementovat v grafové databázi. Dále se budeme věnovat transformaci dat ze stávající MySQL databáze do Neo4j.



Obrázek 4.20 Ontologický Model pro popis společnosti

4.3.1 Transformace OntoUML do grafové databáze

V tomto odstavci shrnuji postup, sadu pravidel, která používám pro implementaci ontologie (v notaci OntoUML) v grafové databázi Neo4j. Vycházím z analogie transformace konceptuálního modelu do modelu relačního. Postup je též inspirován publikací [11], která se rovněž věnuje modelování grafové databáze, vychází však z binárního E-R modelu.

Tento přístup se obecně nazývá Model-Driven Engineering. Ontologický model je běžné transformovat na implementační modely jako relační nebo objektový. Zde se ovšem jedná o ojedinělou variantu pro model grafový.

Vzhledem k výše zmíněné analogii s transformací E-R na relační schéma je ještě vhodné podotknout, že grafové databáze (nejen Neo4j) nemají pevně definované schéma, jak je tomu u databází relačních, někdy jsou dokonce označovány jako schema-free databáze. Přesto lze pomocí implementovaných integritních omezení typu štítek (label) či unikátní hodnota atributu a dalších prostředků efektivně implementovat strukturu dat popsanou pomocí OntoUML.

Metodika se skládá z těchto pravidel.

1. Pro každý uzel existuje popisek, který vychází z entity ke které její informace patří. Uzel nesoucí informace o instanci entity Link, tak bude mít popisek Link.
2. Dědičnost je implementována kompozicí popisků, např. uzel Linku obsahuje kromě popisku Link také děděné informace z Resourse. Platí pro stereotypy: Kind, Subkind, Role, Phase, Category a Mixins.
3. Identifikatorem uzlu bude id datové instance a atributy budou taktéž přeneseny.
4. Vztahová entita se stereotypem *Relator* bude omezena na hranu přejímající popisek dle názvu a atributy dle dat entity.

5. Pro vztahy celek a část jsou hrany ve výchozím stavu směrovány od celku k části. Pojmenovány např. „IS PART OF“.
6. Meta-atributy jako Disjoint a Complete jsou vynuceny strukturou entit.
7. Aspekty jako Mode a Quality jsou atributy vlastnícího uzlu.

Uvědomuji si, že výsledky mé práce nesedí vždy s těmito pravidly. Proto bych rád zdůraznil, že se jedná o doporučení, od kterého se implementace může odklánět z různých důvodů.

4.3.2 Tvorba Cypher skriptů pro transformaci dat

Pro uplatnění sad pravidel ze sekce 4.3.1 je nutné namapovat existující MySQL data na entity konceptuální modelu. Vzhledem k tomu, že neexistovala žádná sepsaná dokumentace k původní MySQL databázi, jediný způsob z čeho bylo možné vycházet je schéma původních dat a odpovědi vývojářů, kteří s ní pracovali. Proto není divu, že takové namapování pak není velký problém, jelikož velká část struktury ontologického modelu vycházela z původní MySQL databáze. Jedná se totiž o jediný spolehlivý zdroj informací. Již od začátku bylo jasné, že tento krok bude nutný, proto jsem i při ontologickém modelování bral na zřetel tuto skutečnost. Výsledkem pak je často přímočaré namapování původních tabulek na nové entity. Potřeba transformací dat z tabulek je minimální a pokud je třeba, je v Cypher inicializačním skriptu popsána v komentáři, tento skript je popsán v sekci 4.3.2.2.

V modelech se často vyskytuje vztah popisující similaritu rozdílných prvků, takové spojení bude dynamicky vypočítáno na prvcích existujících v grafové databázi. Vyhodnocování a sestrojování takových hran je práce na straně Learneronu. K tomu bude využívána technologie FAISS — vyvíjena Facebook AI Research — je knihovna pro efektivní vyhledávání podobných entit. Projekt je open source a dostupný na Githubu[e]. Proto tyto vazby v ukazovém skriptu chybí.

4.3.2.1 Úvod

Kromě přehledné dokumentace[f] poskytnuté na oficiálním webu. Bych rád i zde podal základní úvod pro čtenáře do syntaxe Cypher dotazování. Jak jsem již zmiňoval, je podobný SQL, což na první pohled znamená, že je taktéž deklarativní, aneb je jednoduché porozumět jeho zápisu.

Vytváření indexů je následující. Tento dotaz vytvoří pro uzly *c* se štítkem Country index na atributu *name*.

```
CREATE INDEX FOR (c:Country) ON (c.name)
```

Kromě vytvoření indexu přímo, lze jej vytvořit integritním omezením na unikátnost atributu v entitě. Přestože Neo4J poskytuje možnost vytvářet integritní omezení, nebo to alespoň tak vypadá, tak ve skutečnosti vyjadřovací schopnosti takových omezení, jsou velmi omezené. V době psaní této práce lze tato omezení využít pouze k následujícím ošetřením.

- Unikátnost hodnoty atributu dle štítku daného uzlu (výše popisován).
- Existence atributu v uzlu dle štítku.
- Existence atributu ve hraně dle štítku.
- Unikátnost kombinace hodnot atributů dle štítku daného uzlu

[e] https://github.com/facebookresearch/faiss
[f] https://neo4j.com/docs/getting-started/current/cypher-intro/load-csv/, https://neo4j.com/developer/desktop-csv-import/

Zajištění unikátnosti atributu se tak zapíše následujícím způsobem. Omezení nazvané *persIdConstr* vzniká nad entitami *p* se štítkem Person k zajištění unikátnosti atributu *id*.

```
CREATE CONSTRAINT persIdConstr ON (p:Person) ASSERT p.id IS UNIQUE
```

Vytváření uzlů a hran je pro graf to nejdůležitější. Proto je nutné popsat, jak se vytváří. Při vytváření uzlů potřebujeme data, které bude entita obsahovat a její strukturu. V prvním řádku se načítají postupně cyklicky jednotlivá data řádek po řádku do proměnné *csvLine*, tyto data pocházejí ze souboru *persons.csv* a jsou načítány společně se záhlavím pro jednoduchost vybírání jednotlivých sloupců jmény přes „objekt.atribut“ syntaxi. V druhém řádku se vytváří entita *p* se štítkem Person a naplňuje se daty z proměnné *csvLine*. Protože atributy objektu *csvLine* pochází z CSV souboru, kde není popsán typ atributu, je nutné převést jej na vhodný typ. Tedy např. metoda „toInteger(x)“ převede *x* z jeho formátu — v tomto případě se jedná o řetězec — na celé číslo. Pokud převod selže, pak selže celý běh ukládání těchto dat. Entita *p* pak obsahuje atributy *id* a *name*. Toto vytvoření se pak opakuje pro všechny řádky přiděleného souboru.

```
LOAD CSV WITH HEADERS FROM "file:///persons.csv" AS csvLine
CREATE (p:Person {id: toInteger(csvLine.id), name: csvLine.name});
```

Podobný případ popíšu i pro vytváření hrany. Rozdílem je především potřeba definovat mezi kterými uzly se má hrana vytvořit. V prvním řádku se opět načítá řádek CSV souboru do proměnné. Ve druhém řádku se vyhledává odpovídající uzel *p* se štítkem Person s atributem *id* rovným záznamu CSV a uzel *m* se štítkem Movie opět s odpovídajícím id dle právě procházeného řádku. Informace odkazující na tyto entity jsou pak uloženy v proměnných *p* a *m*. Ve třetím řádku pak přichází syntaxe na vytvoření samotné hrany. Ta se vytváří směrem od *p* k *m* mající štítek PLAYED s atributem *role* ze záznamu *csvLine*.

```
LOAD CSV WITH HEADERS FROM "file:///roles.csv" AS csvLine
MATCH (p:Person {id: toInteger(csvLine.personId)}), (m:Movie {id:
    toInteger(csvLine.movieId)})
CREATE (p)-[:PLAYED {role: csvLine.role}]->(m);
```

Vhodným příkazem je také postupný commit po vytvoření *X* záznamů. To je vhodné v tomto případě, když se ze souboru načítají tisíce záznamů k vytvoření tisíce nových uzlů a hran. To snižuje overhead paměti potřebný pro zpracování transakce.

```
USING PERIODIC COMMIT X
```

4.3.2.2 Tvorba Cypher inicializačních skriptů

Vytvořil jsem dva skripty k inicializaci, první k popisu integritních omezení a druhý k načtení dat dle vytvořené struktury v souboru.

1. `src/impl/aws/scheme.txt`

   : Tento soubor slouží pro vytvoření integritních omezení, které do jisté míry udržují konzistenci modelu. Je rozdělen na tři sekce: MySQL, ESCO a Corporate. V každé z těchto sekcí se vytvářejí omezení pro entity, které zde patří dle modelů subdomén zmiňovaných v modelovací části. Neobsahuje všechny omezení dle schématu, nejenže Neo4J neumožňuje podchytit všechny možné případy, taky bylo rozhodnuto, že se budou indexovat pouze nejpotřebnější části databáze, je tomu tak protože…

   a. Režie k zachování každého indexu se projevuje při každém vložení, úpravě, či vymazání dat. Proto každá taková transakce pak trvá déle.
   b. Toto vyžaduje více místa na disku, protože každá indexace potřebuje ukládat svá mezidata.

Protože je potřeba, aby databáze byla svižná, aby real-time umožňovala doporučovat prostředky daného druhu a jelikož instance je hostovaná na AWS, které se platí kromě za licence také za použití výkonu vzdáleného stroje, se vedení týmu rozhodlo k takovému přístupu.

2. `src/impl/aws/data.txt`

: Tento soubor obsahuje postupné vytvoření všech entit v grafové databázi. Pro každou entitu bylo potřeba následující.

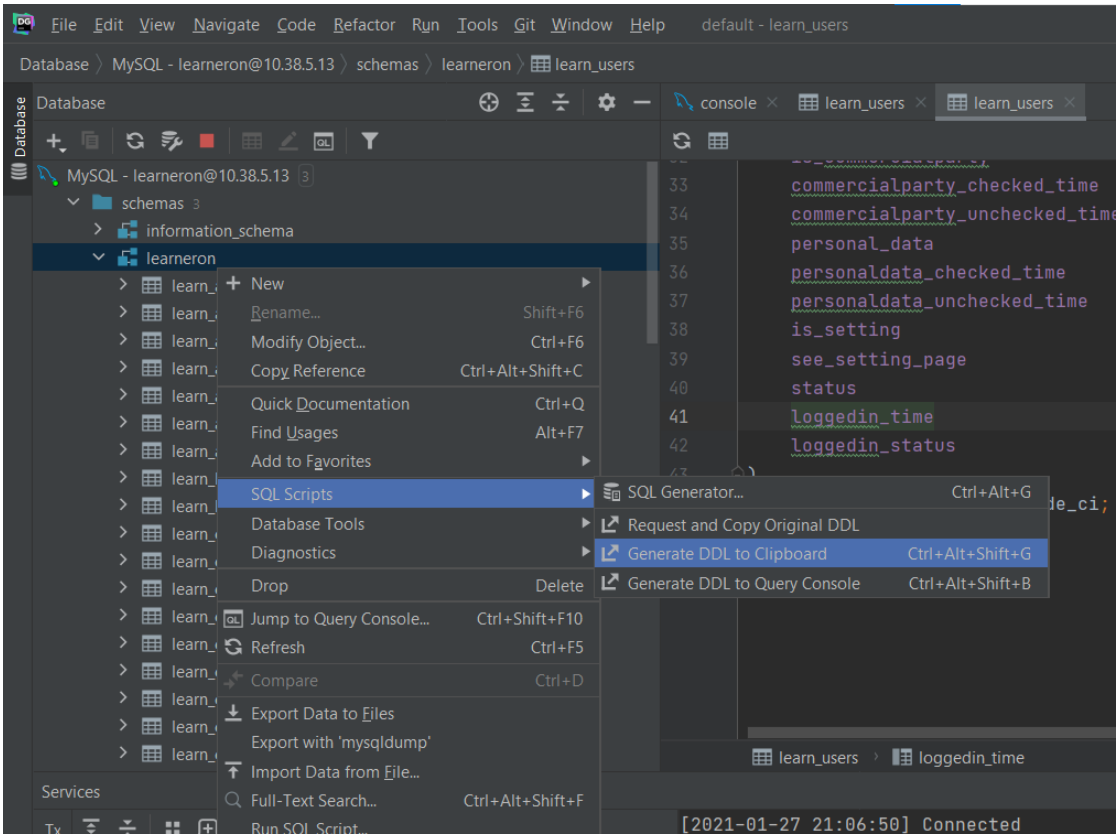
- Entitu popsat jménem odpovídající ontologickému modelu pro označení.
- Pokud entita pochází z relační databáze, pak zadat u ní SQL dotaz pro získání dat odpovídající pro danou entitu.
- Pro data pocházející z MySQL databáze jsem vygeneroval automatický DDL skript pomocí programu Data Grip — postup je dostupný na obrázku 4.21. Tato pomocná struktura slouží jako přehled o atributech a vztazích relačního řešení. Tu jsem namapoval na odpovídající část ontologického modelu. Díky relativně přímočarému návrhu popisující MySQL data, to není příliš komplexní. Díky tomuto namapování jsem byl schopen strukturu přepsat pomocí transformační metodiky v sekci4.3.1 k vytvoření uzlů s odpovídajícími atributy pocházejícími z relační databáze. Jinými slovy automaticky vygenerovaným DDL skriptem jsem získal formátovaný výstup, který jsem použil pro sestrojení dotazů na vytvoření entit, jejichž data pocházejí z MySQL databáze. Některé atributy jsou účelově vypuštěny z důvodu jejich irelevance v aplikace (s komentářem „nn“ označující atributy, které „nejsou nutné“).
- Pro ESCO data se přebírá schéma z oficiálně dodávaných zdrojů EU se všemi atributy a typy, u vkládání takových entit je možné vidět, že se záznam řádku CSV neupravuje, ale přímo vkládá databázovému stroji.
- Ostatní typy dat jsou vytvářeny manuálně a jsou často předmětem budoucího vývoje.
- Pro každý uzel se data načítají ze CSV souborů uložených přímo u databázového stroje pro jednoduchý a okamžitý přístup.
- Pokud uzel má obsahovat časový údaj přebraný z relační databáze, je potřeba jej upravit. Ve výchozím stavu údaje nejsou totiž v kompatibilním formátu a to jak typ *date* tak *datetime*. Je proto potřeba jak je uvedeno v příkladu pod tímto textem nastavit požadovaný atribut entity *u* kondicionálně. Pokud je informace ze CSV záznamu neexistující, nebo je rovna výchozímu nulovému datu „0000-00-00“ resp. „0000-00-00 00:00:00“, pak je třeba tento údaj převést metodami *date* resp. *datetime*.

```
SET u.birthday = CASE WHEN csvLine.birthday is null OR csvLine.
    birthday = '0000-00-00' THEN null ELSE date(csvLine.birthday)
    END
SET u.modified = CASE WHEN csvLine.modified is null OR csvLine.
    modified = '0000-00-00 00:00:00' THEN null ELSE datetime(
    replace(csvLine.modified, ' ', 'T')) END
```

4.3.3 Transformace dat a naplnění databáze

Nejprve je potřeba získat datové soubory, které budou přístupné databázovému stroji při spouštění skriptů. V případě již existujících dat z relační databáze je potřeba je akorát exportovat. ESCO data[g] jsou získána z veřejných dat na webu Evropské komise. Zde je zájem především o soubory:

[g] `https://ec.europa.eu/esco/portal/download`



Obrázek 4.21 Generování DDL skriptu

- `broaderRelationsOccPillar.csv`

  obsahuje vztahy označující kategorické zařazení pracovních pozicí do jednotlivých úrovní kategorizace ISCO.

- `broaderRelationsSkillPillar.csv`

  obsahuje vztahy označující kategorické zařazení znalostí do jednotlivých úrovní ESCO kategorizace skupin znalostí.

- `ISCOGroups_en.csv`

  obsahuje kategorizaci ISCO pro pracovní pozice.

- `occupationSkillRelations.csv`

  obsahuje potřebné vztahy mezi znalostmi a pracovními pozicemi. Konkrétně se jedná o informace, které znalosti jsou vyžadovány/ doporučené pro danou pracovní pozici.

- `occupations_en.csv`

  obsahuje seznam pracovních pozicí.

- `skillGroups_en.csv`

  obsahuje seznam kategorizací pro znalostí.

- `skills_en.csv`

  obsahuje seznam znalostí.

Pro ostatní data, které budou vznikat pouze nad grafovou databází, je možné vytvořit CSV soubory reprezentující jejich informaci.

4.3.3.1 Aplikace cypher skriptu

Projekt zajišťující vložení inicializačních dat je poskytnut v příloze ve složce

`src/impl/aws`

. Z pochopitelných důvodů v něm chybí informace o adrese instance i jeho přihlašovacích údajích. Data jsou pak čistě testovací a poskytnuta firmou Learneron.

Pro přístup ke vzdálené databázi a naplnění jí jsem použil oficiální Neo4J knihovnu pro Python. Python byl zvolen, protože velká část práce týmu je psaná v Pythonu. Umožňuje mi to tak sdílet práci i kód s ostaními spolupracovníky, kteří jsou na tento jazyk zvyklí. Neo4J poskytuje podporu i jiným běžně používaným programovacím jazykům jako např. Java, .NET, JavaScript a další.

Protože instance Neo4J je hostovaná službou AWS, je důležité zmínit, že pro přesun datových souborů jsem použil přístup přes SSH. Databázový systém může získávat přístup k souborům i vzdáleným, ale ty musí být veřejně dostupné. Poskytnout tyto informace i jenom na moment veřejnému internetu ovšem nepřipádá v úvahu, proto byl zvolen tento způsob.Tato instance běží na verzi 4.2.2 Enterprise edice. Připojení probíhá zabezpečeným kanálem díky SSL certifikátu po protokolu — vyvíjeným a spravovaným Neo4J — Bolt. Pokud by SSL certifikát pro AWS instanci nebyl zajištěn, ovladač neumožní zabezpečený typ komunikace přes Bolt. Neo4J tento typ komunikace označuje jako „bolt+s“.

Ve hlavním souboru projektu jménem:

`main.py`

je postup následující.

1. Připojit se k MySQL databázi a stáhnout potřebné tabulky dle informací v souboru

```
src/impl/aws/mysql-files-list.txt
```

. Ten obsahuje dotaz pro získání dat a taktéž jméno výstupního souboru. Data se ukládají ve formátu CSV.

2. Připojit se přes SSH na server s Neo4J databází pro vložení dat. Jak původních z MySQL, tak nových — uložených s projektem ve složce

```
src/impl/aws/data/additional
```

. Ty se ukládají k prvnímu vyhledávacímu bodu Neo4J databáze pro vstupní data.

3. Načít nové data do databáze. Součástí tohoto procesu je…
   - Připojit se k databázi.
   - Vymazat všechny uzly a hrany grafu.
   - Vymazat všechny indexy a integritní omezení původního grafu.
   - Spustit transakci k vytvoření nového „schématu“ databáze, neboli integritních omezení a indexů.
   - Spustit transakci k vytvoření nových uzlů a hran.

Struktura třídy *App* je inspirovaná vzorovým projektem na stránce[h] s oficiální dokumentací.
Po spuštění projektu se data nacházejí v databázi. Po spuštění Neo4J browser klienta a zadání dotazu pro zobrazení celého grafu.

```
CALL apoc.meta.graph
```

Na obrázku 4.22 je možné vidět celý graf. Pokud má uzel více štítků pak jsou vizualizovány všechny jeho štítky jako jiný uzel. Ten sdílí všechny společné atributy i vazby s jinými vizualizacemi stejného uzlu. Proto lze vidět na obrázku například uzly se štítkem *Resource*. V databázi se ovšem nevyskytují data pouze se štítkem *Resource*, ale ve skutečnosti konkrétní data jako knihy, nebo odkazy, které mají takový štítek pro označení učebního materiálu. Pro přehlednost pak databázový stroj při vizualizaci seskupuje všechny uzly se stejnými štítky.

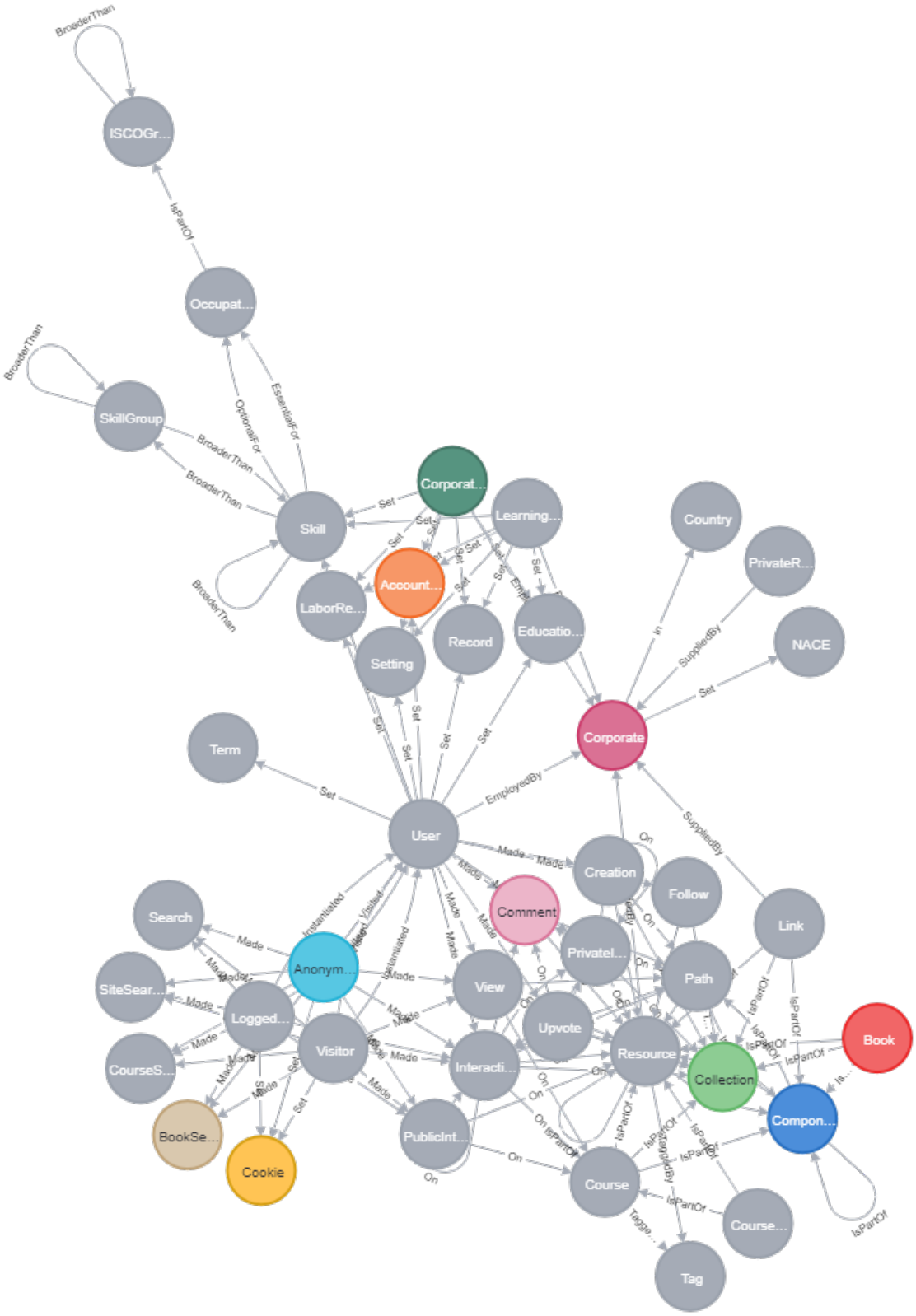
4.4 Metodika pro rozvoj databáze a import nových dat

V této sekci popíši metodický postup, díky kterému bude udržena konsistence mezi obsahem Neo4j databáze a popisem ontologie v OntoUML. Problém by byl relativně jednoduchý, kdyby při dalším rozvoji projektu portálu Learneron nepadlo rozhodnutí, že některá data budou (z bezpečnostních a jiných důvodů) udržována jak v novém úložišti - grafové databázi Neo4j, tak také v původní relační databázi.

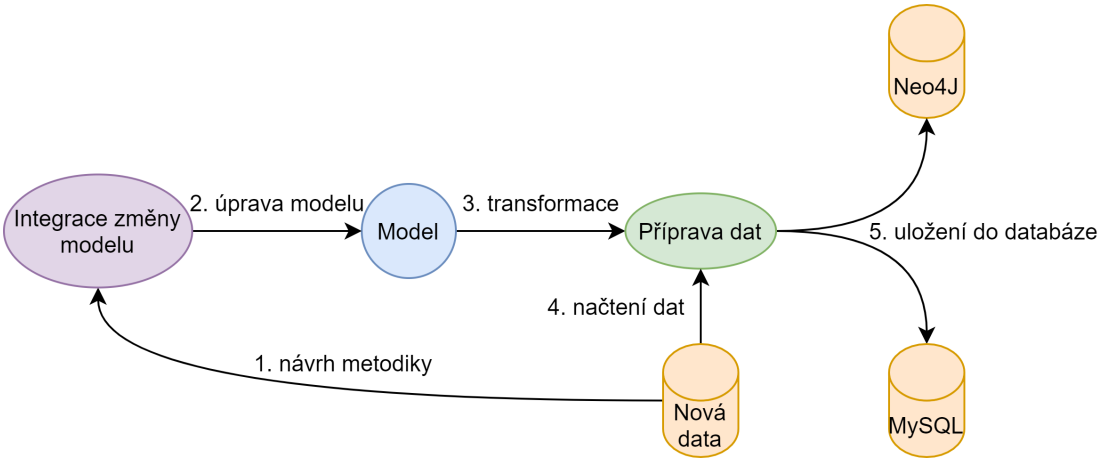
Jak je viditelné na obrázku 4.23 v případě požadavku na integraci nových dat se bude postupovat následujícím způsobem. Pro nová data je nejdříve nutné upravit model.

1. Prvně je potřeba dle charakteru dat navrhnout metodiku pro definování a zpracování dat,
2. poté aplikovat tyto metody do úpravy ontologického modelu.
3. V rámci transformace se připraví data dle skutečnosti, zda půjdou pouze do Neo4J databáze, nebo je potřeba je zálohovat i v relační MySQL databázi.

[h] `https://neo4j.com/docs/api/python-driver/current/`



Obrázek 4.22 Celý Neo4J graf po seskupení hran se stejnými štítky



Obrázek 4.23 Integrace nových dat

4. Nový zdroj dat se pak zpracuje transformací modelu

5. a data zbývá uložit.

Uložení do grafové databáze je mandatorní pro všechna data obsažená modelem. Ale rozhodnutí o ukládání do MySQL funguje především operativním způsobem. Myšlenkou je ukládat tam všechna statická data k vytvoření zálohy. Nově vygenerovaná data nad Neo4J — jako vyjádření similarity různých prvků apod. — se budou ukládat pouze do Neo4J.

Nahrávaní dat do obou databází musí být atomické pro zajištění synchronních dat. Implementace samotná není nutně složítá, nicméně bude roztříštěná a dává relativně prostor pro chyby vývojáře.

Při implementaci — v případě, že se data ukládají prvně do MySQL a pak do Neo4J — pak vznikají následující možné scénáře:

- zápisy se podaří v obou databázích, všechno proběhne v pořádku
- zápis do MySQL se podaří, ale zápis do Neo4j selže. Zápis do MySQL je potvrzený, ale je potřeba uložit neúspěch zápisu do Neo4J.
- zápis do MySQL selže a zápis do Neo4j vůbec neproběhne.

V každém případě je pak potřeba vytvořit synchronizační skript, který je nutné kontinuálně pouštět. Ten se bude muset dle nahlášených neúspěchů pokoušet znova o jednotné uložení dat v obou databázích.

Kapitola 5
Závěr

Práce se v souladu se zadáním věnuje analýze různých datových zdrojů v doméně doporučování vzdělání, které integruje do jednoho datového schématu v notaci OntoUML. Tato ontologie slouží k udržení konsistentního pohledu na data, která již portál Learneron udržuje, a také jako základ pro přidání dat z nových zdrojů a hlavně jejich integraci s daty stávajícími. Z tohoto úhlu pohledu je práce jasným příkladem užitečnosti ontologií a postupů pro jejich tvorbu, zejména konceptuálního modelování.

Dalším dílčím cílem práce je implementace modelu v grafové databázi Neo4j. Při plnění tohoto cíle byl navržen postup transformace konceptuálního schématu v notaci OntoUML do grafové databáze. Vyvinutý postup je inspirován článkem [11], zde se však vychází z jednoduché E-R notace, nikoliv z obecnějšího OntoUML. V tomto smyslu se tedy jedná o původní výsledek, který může být základem pro publikaci pro odbornou a výzkumnickou komunitu. Samotná transformace je relativně přímočará, což nahrává možnému budoucímu rozšíření notace OntoUML nejen při tvorbě ontologií, ale též při tvorbě informačních systémů.

Dále se práce zabývá transformací dat do Neo4j. Data se transformují jednak ze stávající MySQL databáze, která v projektu zůstává dále zachována, takže významná část datové základny projektu je duplikována. Další data přicházejí z jiných zdrojů (například databáze ESCO). Výsledkem této části práce jsou též skripty, které transformaci dat realizují.

Poslední část práce přináší návrh postupu jak datovou základnu projektu Learneron udržet dokumentovanou a konsistentní a přitom umožnit hladké vkládání nových dat do existujících struktur, tak také rozvoj struktury/ontologie samotné.

Dílčí výsledky popsané v této práci byly prakticky využity a v projektu Learneron se na nich dále staví. Předložená práce rovněž splňuje všechny dílčí cíle, vytyčené v jejím oficiálním zadání.

Bibliografie

1. HLAVNÍ MĚSTO PRAHA. *Smlouva o poskytnutí voucheru* [online]. 2020-09 [cit. 2021-04-22]. Dostupné z: https://smlouvy.gov.cz/smlouva/13741032.
2. SOUČEK, Jiří. *Konceptuální model informace a FRBR | Souček | ProInflow* [online]. 2021-04 [cit. 2021-04-09]. Dostupné z: https://www.phil.muni.cz/journals/index.php/proinflow/article/view/2013-3-9/864.
3. KREMEN. *Sémantický trojúhelník* [online]. 2014-10 [cit. 2021-04-09]. Dostupné z: https://wikisofia.cz/wiki/Teoretick%C3%A9_z%C3%A1klady_selek%C4%8Dn%C3%ADch_jazyk%C5%AF_v_s%C3%A9miotice#/media/File:S%C3%A9mantick%C3%BD_troj%C3%BAheln%C3%ADk.jpg.
4. VERDONCK, Michaël; GAILLY, Frederik; PERGL, Robert; GUIZZARDI, Giancarlo; MARTINS, Beatriz; PASTOR, Oscar. *Comparing traditional conceptual modeling with ontology-driven conceptual modeling: an empirical study* [online]. 2020-02 [cit. 2021-04-09]. Dostupné z: http://www.inf.ufes.br/~gguizzardi/Comparingtraditionalconceptualmodellingwithontology-drivenmodelling-anempiricalstudy.pdf.
5. GUIZZARDI, Giancarlo. *Ontological foundations for structural conceptual models*. Telematica Instituut / CTIT, 2005. ISBN 90-75176-81-3. Dis. UT.
6. FOWLER, Martin. *UML distilled: a brief guide to the standard object modeling language*. Addison-Wesley Professional, 2004.
7. RYBOLA, Zdeněk; RICHTA, Karel. *Towards OntoUML for software engineering: transformation of OntoUML into relational databases: a dissertation thesis submitted to the Faculty of Information Technology, Czech Technical University in Prague, in partial fulfilment of the requirements for the degree of Doctor*. 2017. Dis. České vysoké učení technické v Praze. Fakulta informačních technologií.
8. PERGL, Robert. *OntoUML: Základy, rigidní a anti-rigidní sortály* [online]. 2020-12 [cit. 2021-05-06]. Dostupné z: https://courses.fit.cvut.cz/BI-KOM/slides/pdf/lectures-czech/02-ontouml-basics-sortals.pdf.
9. PERGL, Robert. *OntoUML: Non-sortály, relace, relátor* [online]. 2020-12 [cit. 2021-05-06]. Dostupné z: https://courses.fit.cvut.cz/BI-KOM/slides/pdf/lectures-czech/03-ontouml-nonsortals-relations.pdf.
10. PERGL, Robert. *OntoUML: Typy agregací, Aspekty* [online]. 2020-12 [cit. 2021-05-06]. Dostupné z: https://courses.fit.cvut.cz/BI-KOM/slides/pdf/lectures-czech/04-ontouml-aggr-aspects.pdf.
11. POKORNY, Jaroslav. Modelling of Graph Databases. *Journal of Advanced Engineering and Computation*. 2017, roč. 1, č. 1, s. 04–17. ISSN 2588-123X. Dostupné z DOI: 10.25073/jaec.201711.44.

12. WIKIPEDIA COMMONS. *Graph database - Wikipedia* [online]. 2021-04 [cit. 2021-04-10]. Dostupné z: https://en.wikipedia.org/wiki/Graph_database.

13. WEBBER, Jim. *Comparison of Relational Databases and Graph Databases - Stack Overflow* [online]. 2018-06 [cit. 2021-04-10]. Dostupné z: https://stackoverflow.com/questions/13046442/comparison-of-relational-databases-and-graph-databases/17942968#17942968.

14. HAVE, Christian Theil; JENSEN, Lars Juhl. Are graph databases ready for bioinformatics? *Bioinformatics*. 2013, roč. 29, č. 24, s. 3107–3108. ISSN 1367-4803. Dostupné z DOI: 10.1093/bioinformatics/btt549.

15. BRUGGEN, Rik V. *Learning Neo4j*. 1. vyd. Olton: Packt Publishing, Limited, 2014. ISBN 978-1849517164.

16. EVROPSKÝ SOCIÁLNÍ FOND ČR. *Metodika zařazování zaměstnání do CZ-ISCO* [online]. 2017-04 [cit. 2021-04-22]. Dostupné z: https://www.esfcr.cz/documents/21802/3435234/Metodika+za%C5%99azov%C3%A1n%C3%AD+zam%C4%9Bstn%C3%A1n%C3%AD+do+CZ-ISCO+pro+%C3%BA%C4%8Dely+statistiky+trhu+pr%C3%A1ce.pdf.

17. ČESKÝ STATISTICKÝ ÚŘAD. *METODICKÁ PŘÍRUČKA K NACE Rev. 2* [online]. 2014-12 [cit. 2021-04-22]. Dostupné z: https://www.czso.cz/documents/10180/23174387/metodicka_prirucka_cz_nace_rev_2.pdf.

Obsah přiloženého média

- `readme.txt` ..... stručný popis obsahu média
- `src`
  - `impl` ..... zdrojové kódy implementace
    - `Learneron.eapx` ..... EA projekt s ontologickými modely
    - `aws` ..... projekt pro naplnění grafové databáze
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF