Vysoká škola ekonomická v Praze
Fakulta informatiky a statistiky

Návrh báze znalostí skautských programů

BAKALÁŘSKÁ PRÁCE

Studijní program: Aplikovaná informatika
Studijní obor: Aplikovaná informatika

Autor: Šimon Trousil
Vedoucí práce: prof. Ing. Václav Řepa, CSc.

Praha, květen 2024

Prohlášení

Prohlašuji, že jsem bakalářskou práci *Návrh báze znalostí skautských programů* vypracoval samostatně za použití v práci uvedených pramenů a literatury.

V Praze dne DD. měsíc RRRR .....

Podpis studenta

Poděkování

Děkuji panu profesoru Řepovi za podpoření mého rozhodnutí v zaměření své práce na oblast znalostí.

Abstrakt

Tato práce hledá softwarové řešení pro skautský oddíl, které by umožnilo ukládání zápisů o připravených programech prostřednictvím alespoň stejně uživatelsky nenáročného rozhraní pro zapisování jako mají dokumenty Google, které by navíc umožnilo i filtrování uloženého obsahu pomocí definovatelný parametrů. Proto jsou porovnána čtyři běžné softwarové nástroje, aby z nich byla vybrána kombinace nástrojů, které by podle stanovených kritérií byla nejlepší. Tím je dokončen výběr SW. Následně jsou analyzovány tři existující báze skautských znalostí, vybrány z nich pojmy asociované se záznamy v dané bázi a na závěr jsou vybrané pojmy interpretovány pomocí diagramu tříd podle UML. S vybranou infrastrukturou i určeným obsahem k zaznamenávání, je posledním krokem identifikace nejlepších praktik pro modelování dat na základě oficiálních publikací k vybrané databázi. S pomocí identifikovaných doporučení je pak transformován diagram tříd, na schéma vhodné k uložení do vybrané databáze. Správnost výběru infrastruktury a míra její realizovatelnosti je vyhodnocena pomocí implementace klíčových částí návrhu, které jsou identifikovány jako klíčové pro dosažení konkrétních požadavků, které na systém práce klade.

Klíčová slova

znalosti, báze znalostí, skauting, skautské programy, grafová databáze, apps scripts, neo4j

Abstract

This thesis presents a software solution for a scout group that would allow the storage of records on prepared programs through an interface at least as user-friendly as Google documents, which would also enable filtering of stored content using definable parameters. Therefore, it compares four common software tools to select a combination of tools that would be the best according to established criteria. This determines the infrastructure that will be used for the design. The thesis analyzes three existing bases of scout knowledge, selects terms associated with records in the given base, and ultimately interprets the analysis using a UML class diagram. With the selected infrastructure and determined content for recording, the final step identifies the best practices for modeling graph data based on official publications for the chosen database, which are then used to transform the class diagram into a schema suitable for storage in the selected database. The correctness of the infrastructure selection and the degree of its implementation are evaluated using the implementation of key parts of the design, which are identified in relation to the possibility of achieving specific requirements that the work specifies for the system.

Keywords

knowlage, knowlage base, scouting, scout activities, graph database, apps scripts, neo4j

Obsah

Seznam obrázků

Seznam tabulek

Seznam použitých zkratek

GDB Graph DataBase
RDB Relational DataBase
gSheets Google Sheets
gDocs Google Documents
gWorkspace Google Workspace
SW Software
SQL Structured Query Language

Úvod

Skauting i v posledních letech stále nabývá na popularitě a počet členů Junáka - českého skauta také roste (**trojakPrecteteSiVyrocni2022**). Avšak vzhledem k tomu, že se jedná o dobrovolnou činnost, ze které nemá nikdo pro sebe ani korunu, bývá pro vedoucí nezřídka obtížné si najít dost času na svědomitou a důkladnou přípravu programu na další pořádanou událost. Pro přiblížení činnosti přípravy programu by se dalo říci, že cílem vedoucího jenž program připravuje je nalezení konkrétních aktivit, které bude pak při samotné události s dětmi realizovat. Je u toho však třeba vzít v úvahu nemalou řádku věcí, aby připravený program jednak děti bavil a aby jim ideálně dal příležitost se něco naučit, něco si vyzkoušet nebo se v něčem zlepšit.

Samozřejmě zcela zásadní podíl na výsledné kvalitě programu i času potřebném na jeho přípravu, mají zkušenosti daného jedince. Není však pravdou, že by neměl člověk jinou možnost, než si zkušenosti získat vlastními pokusy a omyly. Ovšem, je to nejefektivnější, a určité (příliš komplexní) situace nejdou jednoduše někomu sdělit nebo předat. Pro všechny ostatní, jednodušší zážitky je však zcela jistě alespoň z části možné tuto zkušenost zapsat a sdílet ostatním. Tedy alespoň takový je výchozí předpoklad této práce.

Organizace znalostí skautských programů + Software pro uložení znalostí

Mimo to, práce předpokládá, že pokud nebude sdílené úložiště umožňovat dostatečně efektivní vyhledévání v uložených informacích (hledání, pokud vůbec nakonec úspěšné, bude trvat déle, než kdyby byla hledaná informace získávána alternativními způsoby) nebude nikdy možnost plně využít potenciál, který sdílení informací umožňuje.

Problém:

Bohužel v našem oddíle ještě nebyl stanoven žádný jednotný postup ani nástroj pro toto sdílení. Když už tedy se nějaké informace podaří někomu zapsat, je tak učiněno dle jeho vlastního nejlepšího posouzení a uvážení. Výsledné zápisy tak nijak neusnadňují jejich efektivní prohlížení, což značně limituje naše příležitosti k využití uložených záznamů.

Práce se proto zaměřuje na organizaci znalostí o skautských programech a na nalezení vhodného softwaru, který by umožnil využití přileżitostí vyplývajíích ze zaznamenávání a sdílení cenných informací.

Protože největším omezením z hlediska sdílení cenných informací je systém, konkrétně spíše jeho absence či fragmentovanost. Práce proto stanovuje následující cíl.

Cíl:

Navrhnout systém pro sdílení *znalostí o připravených programech.* Který by *uživatelům* poskytl možnost **efektivního** (*rychle, správně*) a zároveň **příjemného** (*S co nejnižší bariérou, která je potřeba překonat pro použití novými uživateli.*) vyhledávání v uloženém obsahu. Spolu se zachováním alespoň **stejné úrovně kvality uživatelské zkušenosti** při zapisování, jako v aktuálním řešení. Bez toho, aniž by navržený systém vyžadoval *více zdrojů na provoz* a *údržbu*, než sám ušetří při svém *využívání.*

Záměr využití systému

Konkrétně by navržená báze měla být využita na sdílení všech, v rámci oddílu připravených programů, které by umožnilo přípravu lepších programů tak, že umožní asociaci zpětné vazby k zaznamenaným programům a následné efektivní prohledávání všech historických záznamů. V takovém případě pak bude možné efektivněji využívat dříve realizované programy. Konkrétně tak, že historické záznamy realizovaných programů obohacené o zpětnou vazbu případně hodnocení, jsou následně možné využít při přípravě nových programů dvěma způsoby, za prvé je možné identifikovat která rozhodnutí opakovaně nefungují a bylo by proto pravděpodobně lepší se jim v budoucnu vyhnout a za druhé, je přesně naopak možné identifikovat rozhodnutí, která opakovaně vedla k dobrým výsledků a bylo by proto možná užitečné aplikovat konkrétní rozhodnutí i v budoucnu.

Dílčí cíle + metody:

Tabulka 1: Dílčí cíle a metody

Vybrat sw řešení umožňující dosažení cíle práce.	analýza literatury(1) komparace, multikriteriální výběr
Vybrat výseč reality relevantní pro skautské programy.	analýza literatury(1) konceptuální (2)
Navrhnout schema pro databázi.	Z modelu pojmů, podle doporučených postupů, na základě využití nejlepších identifikovaných praktik (3)(4)

Výsledné přínosy této prace jsou: - dva diagramy tříd znázorňující doménu skautských programů. - schéma pro databázi Neo4j vycházejíci z prvního z modelů

1. Metodika

V této kapitole jsou definovány metody využité prací k dosažení dílčích cílů. Pro každý z cílů je nejprve definován a vysvětlen jeho účel, popsán a zdůvodněn konkrétní postup, případně i obecný postup (pokud je takový aplikován) a na závěr jsou identifikovány části práce, ve kterých jsou výstupy konkrétních metod prezentovány.

1.1 Metodika analýzy sw kandidátů na úložiště

1.1.1 Účel postupu

Jelikož tato práce neklade na navrhovaný systém zrovna nízké nároky (viz. Cíl), nestačí pouze určit, který obsah zaznamenávat, a opomenout přitom řádný výběr softwarového nástroje nebo nástrojů pro uložení zaznamenávaného obsahu. Zároveň nelze tento krok ani vynechat, poněvadž bez volby alespoň konkrétního typu SW pro uložení dat(např. RDB, GDB, ...) není možné navrhnout ani konkrétní strukturu nové báze.
Navíc toto rozhodnutí ovlivňuje, mimo konceptuální model, jakožto prostředek nezávislý na konkrétní implementaci, všechnu budoucí práci na vývoji, údržbu nasazeného systému i šanci na to, aby byla navržená báze skutečně cílovými uživateli přijata a využívána.
Proto byl zvolen dílčí cíl: "Vybrat SW řešení umožňující dosažení cíle práce."A účelem tohoto postupu proto je určení místa (SW nástroje) pro uložení záznamů navrhovanou bází, které by splňovalo požadavky definované cílem práce (viz. Cíl) a tím tak bylo odpovědí na otázku "Kam uložit záznamy v navrhované bázi?".

Záměrem však není ani tak podrobná či kompletní analýza veškerých dostupných nástrojů, spíše jako porovnání několika softwarových zástupců (kandidátů) s odlišnými typy datových struktur, které se obvykle pro stavbu bází využívají, a zjištění, kteří kandidáti splňují požadavky stanovené v cíli (viz. Cíl).

Jako typy datových struktur k posouzení byly zvoleny dokumenty - jakožto výchozí možnost, spreadsheets - jakožto kompromis mezi databází a prostředím dokumentů, relační databáze - jakožto praktický standard při tvorbě znalostních bází a grafová databáze - jakožto modernější varianta klasické relační DB. Konkrétními kandidáty posuzovanými na funkci úložiště záznamů navrhované báze, reprezentující jednotlivé typy zvolených datových struktur, jsou gDocs za dokumenty, gSheets za spreadsheets strukturu, MySQL za RDB a nakonec databáze Neo4j, jakožto reprezentace GDB a NoSQL. První dva SW kandidáti byly zahrnuti do výběru také proto, že se jedná o nástroje aktuálně v našem oddíle využívané, což znamená, že cíloví uživatelé navrhovaného systému již mají určité standardy, na než jsou zvyklí, že systém nabízí. A jak je prací stanoveno, tyto kvality by neměly být návrhem zredukovány, ba naopak by mělo být provedeno jejich rozšíření.
Z toho důvodu jsou zahrnuti druzí dva SW kandidáti, jelikož disponují funkcemi, které v

aktuálním řešení chybí.

1.1.2 Obecný postup

Pro dosažení prvního dílčího cíle byla zvolena metoda 'analýza literatury' dodržující postup popsaný Bernedtssonem a spol. (1). Autoři knihy uvádějí, že účelem této metody, mimo získání konkrétních hledaných údajů, je přesvědčení čtenáře práce o tom, že bylo analyzováno dostatečné množství, dostatečně kvalitních a relevantních zdrojů. Aby si čtenář mohl udělat názor o tom, zda byly zdroje dostatečně relevantní, potřebuje znát konkrétní zamýšlený účel, se kterým je analýza prováděna.
Jasně definovaný účel je pak podle autorů klíčovým prvkem rovněž i pro autora analýzy, jelikož pokud si autor nebude konkrétního účelu vědom nebo ho bude přehlížet, jeho šance na přesvědčení čtenářů o validitě a přínosnosti prováděné analýzy značně klesá. Zároveň specifický účel odlišuje 'analýzu literatury' od 'rešerše literatury', kdy rešerše má primární účel seznámit autora i čtenáře s obsahem literatury z dané oblasti. (1) Autoři knihy dále identifikují následující kroky, které by jak v zájmu autora tak i čtenáře měly mít jasně definovanou strategii provedení.

- **Výběr literatury:** Zahrnuje vyhledávání relevantních zdrojů a literatury související s tématem projektu.
  Jasně definovaná strategie pomůže, aby čtenář nepochyboval, že bylo provedeno adekvátní hledání zdrojů.

- **Hodnocení zdrojů:** Kritické posouzení zdrojů na základě jejich relevance a spolehlivosti.
  ..., aby čtenář nepochyboval o dostatečném objemu a dostatečné kvalitě zpracovaných zdrojů.
  ..., aby čtenář porozuměl, proč byly některé zdroje vybrané a jiné vynechané.

- **Analýza obsahu:** Podrobné zkoumání a získávání informací z vybraných zdrojů.
  ..., aby měl čtenář šanci pochopit, jak byly výsledky získány.

- **Interpretace výsledků:** Integrace zjištění do koherentního celku a formulace závěrů.
  ..., aby čtenář rozuměl, co získané údaje reprezentují.

Pro nalezení odpovědí hledaných v rámci kroků 'hodnocení' i 'analýzy' jsou využity publikované zdroje k daným nástrojům, primárně pak dokumentace. Avšak platí, že tato analýza se nezabývá zdroji o nástrojích, ale nástroji samotnými. To znamená, že například v následující podkapitole 'Hodnocení zdrojů k analýze' jsou vnímány jako hodnocené zdroje samotné nástroje, nikoliv zdroje o nástrojích, ačkoliv právě ve zdrojích o jednotlivých nástrojích budou hledány odpovědi při prováděném hodnocení.
Není však prováděno žádné dodatečné systematické prohledávání či hodnocení dostupných zdrojů o nástrojích. Bylo tak rozhodnuto poněvadž je předpokládáno, že v rámci na webu dostupných informací o analyzovaných nástrojích, jakožto jasně definovaném softwaru, není

významná šance, že by nalezené informace obsahovaly vyloženě nepravdivá tvrzení, zejména pak pokud budou při odpovídání upřednostněny oficiální zdroje k danému nástroji.

1.1.3 Konkrétní postup

Výběr zdrojů

Co se prvotního výběru zdrojů (v tomto případě SW nástrojů využitelných jako úložiště báze znalostí) týče, ten je proveden bez rozsáhlejšího vyhledávání, protože limitovaný rozsah práce neumožňuje adekvátní zpracování většího objemu variant zároveň spolu s dosažením stanoveného cíle.
Konkrétní předvybraní zástupci posuzovaných datových struktur proto byly již, i s argumentací pro jejich výběr, představeni dříve v této kapitole. A protože i specifický účel pro analýzu byl vyjasněn, následující podkapitola se bude zabývat rovnou hodnocením předvybraných zdrojů a určením pro jakou podmnožinu ze SW kandidátů budou následně v rámci 'analýzy obsahu' zjišťovány odpovědi na stanovené analytické otázky.

Hodnocení zdrojů (sw-úložiště)

V této části budou všichni SW kandidáti vyhodnoceni ze dvou hledisek (možnosti zápisu a čtení uložených dat) odvozených z cíle práce (viz. Cíl). Každé z obou hledisek přitom bude vyhodnoceno zvlášť. To znamená, že jejich vyhodnocení vyprodukuje dvě sady výsledků, jednu pro každé hledisko.

Hodnocení nástrojů z hlediska zapisování Podmínka pro zápis říká, že navržená báze má zachovat úroveň uživatelské zkušenosti, jako poskytuje aktuální řešení. Tím jsou gDocs, případně gSheets pro data jako seznam členů. Ani k jedné z této variant, nepotřebuje uživatel téměř žádné nestandardní znalosti k tomu, aby informace zaznamenal. Pokud mu bude dán odkaz na dokument do kterého má zápis udělat, a pod jaký nadpis, mělo by to stačit každému. Jako kritérium pro vyhodnocení kandidátů z hlediska zápisu proto bude, zda vyžadují od uživatele znalost, jako například specifický jazyk k tomu, aby mohl zapisovat do úložiště. Pokud ano, nejsou pro návrh z hlediska zapisování do báze přijatelné.

Výsledkem tohoto hodnocení může být i více nástrojů než jen jeden z nich, jelikož není z pohledu cíle práce zapotřebí specifikovat jiná kritéria, než to jedno aktuálně definované.

Hodnocení nástrojů z hlediska čtení Z hlediska čtení, respektive možností prohledávání uloženého obsahu, podmínka z cíle říká, že čtení má být efektivní. Měřeno rychlostí pro získání výsledků a správností výsledků. To znamená absenci jak chyb, kdy je zobrazen výsledek neobsahující hledaný obsah (false positive), tak chyb kdy obsah hledaný uživatelem není

nalezen, i když ve skutečnosti je zaznamenán a měl by se zobrazit (false negative). Z kandidátů tedy budou vyřazeni ti, kteří toto nesplňují.

Pro zbývající pak bude vyhodnoceno stanovené kritérium příjemnosti využití. A to definované jako co nejnižší bariera pro nové uživatele, kterou potřebují překonat, aby mohli bázi prohledávat. Velikost bariéry pak bude měřena počtem znaků potřebných pro zadávání dotazů a podle toho, zda k interakci s bází je v základu k dispozici webové grafické rozhraní.

Zbývající kandidáti tedy budou porovnáni podle počtu znaků vyžadovaných k napsání dotazu a přítomnosti webového grafického rozhraní ve výchozí instalaci báze. Vyhodnoceno bude nejdříve pořadí kandidátů podle každého z těchto dvou kritérií zvlášť. Následně takto získané tři hodnoty pro každého z kandidátů budou sečteny a kandidát, který bude mít nejnižší výsledné číslo, protože se například podle každého kritéria umístil na prvním místě, bude přijat jako možná část navrhované báze.

Výsledkem tohoto hodnocení je tak jediný nástroj, který podle sečtených výsledků z vyhodnocení kritérií hlediska zapisování (rychlost, správnost, příjemnost) vychází jako ten nejlepší.

Analýza obsahu vybraných zdrojů (sw-úložiště)

Pro zdroje splňující alespoň jedno hledisko hodnocení, je určena jejich 'vnitřní datová struktura', spolu s jejich 'integrovatelností'. 'Vnitřní datovou strukturou' je myšlena struktura souborů skutečně uložených na disku. To znamená, že může být využita analytická otázka "Jak daný sw ukládá zaznamenané údaje na disk?". Pojmem 'integrovatelnost' jsou pak myšleny varianty, skrze které je možné programově přistupovat k uloženému obsahu i ho stejným způsobem modifikovat. Odpovídající analytická otázka proto může být: "Jaké možnosti programového přístupu k zaznamenaným údajům daný SW nástroj nabízí?"A co se 'programového přístupu' týče, typickými příklady jsou podpora prohledávání databáze pomocí HTTP dotazů nebo využití programovacího jazyka a s tím některé dostupné (na volbě jazyka závisející) specifické knihovny implementující metody pro komunikaci. V obou případech se každopádně jedná o způsoby externí komunikace optimalizované pro využití v kódu konkrétního programu, proto zvolené označení.

Interpretace výsledků (sw-úložiště)

Vycházeje z výsledků získaných vyhodnocením předchozích dvou analytických otázek, bude určeno, zda je možné na základě vybraných sw kandidátů možné postavit bázi, která jak specifikuje cíl práce, nebude vyžadovat víc prostředků na provoz a údržbu než sama ušetří. To konkrétně bude provedeno pomocí nalezení způsobu, jak eliminovat potřebu na ručně vykonávanou údržbu konzistence a aktuálnosti v bázi uloženého obsahu. Při tomto hledání je vycházeno z předpokladu, že báze může ušetřit nějaký čas svým využitím, jen když nebude zároveň také vyžadovat čas na údržbu pro svojí správnou funkčnost. Rovněž by také neměla

vyžadovat žádné finanční prostředky, jelikož vzhledem k neziskové povaze skauta není způsob, jak by se prostředky vydané na provoz takového systému mohly vrátit.

Základní otázkou, kterou se tedy tento krok analýzy bude snažit zodpovědět je otázka: "Je možné eliminovat či alespoň zcela minimalizovat potřebu lidských i finančních zdrojů na údržbu konzistence údajů ve vybraných softwarech zaznamenaných?"

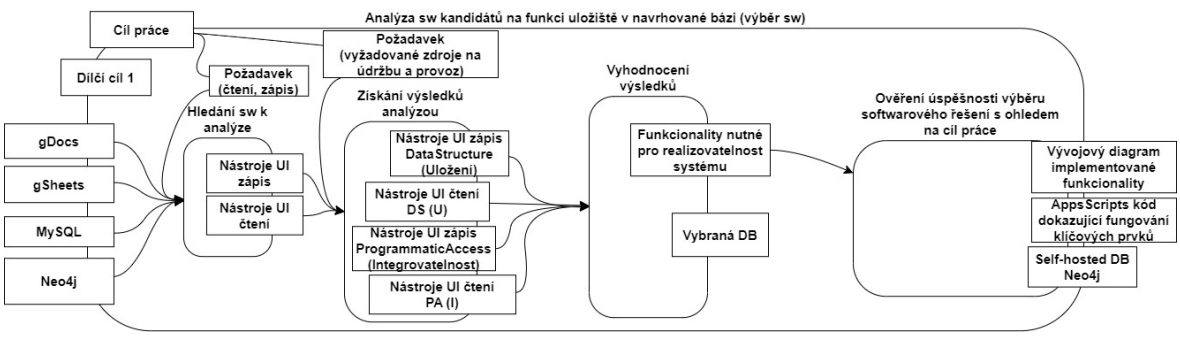
V případě nalezení takového způsobu, bude způsob popsán a následně sestaven výčet funkcionalit kritických pro funkcionalitu odpovídající požadavkům na navrhovanou bázi.

1.1.4 Ověření úspěšnosti

Pro ověření správnosti návrhu byly definované podmínky na základě zjištění z analýzy softwarových nástrojů pro uložení informací. A po

1.1.5 Výstupy

Výsledky z dosažení prvního dílčího cíle jsou prezentovány jako první v kapitole Teorie. S tím, že pořadí prezentace odpovídá pořadí jejich vypracování. Nejdříve jsou proto uvedena hodnocení SW kandidátů podle hledisek možností čtení a psaní, čímž je argumentován následný výběr jejich podmnožin podle specifikovaných hledisek.
Pro vybrané kvalifikované kandidáty jsou v další části vyhodnoceny charakteristiky uložení dat v daných softwarech a možnosti programového přistupování k jejich obsahu.
Po těchto, následuje posouzení dosažitelnosti cíle prací stanoveného a prezentování potenciálního řešení softwarové infrastruktury pro navrhovanou bázi. Samotná dosažitelnost je přitom charakterizována sadou požadavků, které musí být splněny, aby návrh bylo možné skutečně postavit.



Obrázek 1.1: Vizualizace metodiky pro dosažení 1. dílčího cíle

1.2 Metodika analýzy existujících bází

1.2.1 Účel postupu

Motivací pro tuto sekci je druhý kritický aspekt navrhované báze. Jedná se však pouze o aspekt druhý v pořadí prezentace, nikoliv druhý ve významu důležitosti pro návrh. Ve skutečnosti je tento aspekt pro návrh na podobné úrovni důležitosti jako ten řešený v předchozí sekci. Jak už bylo zmíněno, je tímto aspektem určení samotného obsahu, který bude v bázi uložený. Proto byl stanoven dílčí cíl: "Vybrat výseč reality relevantní pro skautské programy."Cílem této části proto je vybrání pojmů a jejich vztahů, kteréžto budou představovat odpověď na otázku: "Co zaznamenávat v navrhované bázi?"

Záměrem však není ani tak provedení vyčerpávající analýzy dané domény, jelikož to by vyžadovalo celou samostatnou práci, jako spíš snaha o agregování pohledů na problematiku několika existujícími bázemi.
Výsledky provedeného postupu tak umožňují tvorbu báze, která dovede odpovídat na rozsáhlejší sadu odpovědí, než báze existující. A zároveň v důsledku tohoto postupu je možné rozšiřovat navrženou bázi obsahem z těch existujících, protože možný obsah dané báze byl zohledněn při návrhu a zakomponován do výsledného schematu báze.

1.2.2 Obecný postup

Pro získání odpovědi na otázku "Co zaznamenávat v bázi skautských programů?", byla opět využita 'analýza literatury' (1), stejně jako v první provedené analýze.

Navíc bylo v rámci kroku 'interpretace výsledků' využito UML. Nástroj byl vybrán, jelikož velmi dobře standardizuje způsoby modelování struktur i skutečnosti obecně.
Ve verzi 2.5.1, ze které aktuálně vycházím, má i celou druhou polovinu zaměřenou na stavy, chování, akce, interakce,... až případy užití. Ač se jedná o velmi užitečnou část, na její využití v této práci pravděpodobně nedojde z důvodu času a rozsahu práce. Jediná odchylka od standardu UML byla provedena v případě generalizací, které místo běžné verze: byla pro generalizace využita následující notaci.1.3

Bylo tak učiněno z toho důvodu, že v používaném modelovacím nástroji (draw.io) je jednodušší s modelem manipulovat (posouvat jednotlivé části) v případě využití alternativní notace pro modelování generalizací. Je tomu proto, že v případě originální notace je zakresleno několik šipek přes sebe a v situaci, kdy je potřeba s generalizovanou třídou v rámci kreslícího plátna posouvat, je pak nutné ručně přesouvat a znovu umisťovat šipky opět přes sebe, což je nepřiměřeně časově náročné.
Alternativní notace tak využívá přidání grafického elementu (malý obdélník) mezi generalizovanou třídu a její specializace, které se všechny napojí na přidaný element a z něj pak ke generalizované třídě je vedena už jen jediná šipka, která dodržuje notaci UML. O výrazné usnadnění se jedná zejména pak v případě, kdy má generalizovaná třída několik různých

specializovaných sad.

1.2.3 Konkrétní postup

Výběr (pojmy z existujících bází)

Podobně jako při předchozí analýze nebylo opět provedeno žádné systematické prohledávání dostupných bází, nýbrž byly vybrány báze, které bývají občas, minimálně v našem oddíle, využívány jako inspirace pro přípravu realizovaných programů v rámci naší činnosti.

První z existujících bází je webová stránka 'chystamprogram.skaut.cz', jenž byla vytvořena organizací Junák - český skaut, která zároveň za celý obsah zcela zodpovídá. (5)

Druhou analyzovanou bázi pak představuje 'Velká encyklopedie her'. Ta je tvořena čtyřmi samostatnými tituly vydanými mezi lety 1987-1988, napsanými panem Milošem Zapletalem.(6)(7)(8)(9) Jak už název napovídá, její zaměření je čistě na hry, které jsou nepochybně také součástí skautských programů, těm se nicméně věnuje velmi podrobně. Slovo "Velká"v názvu totiž nijak zvlášť nepřehání. Každá z knih má v průměru přibližně 600 stran, sice menšího formátu, ale i přesto se často vejdou i dvě hry na stránku.

Za existující bázi by se daly považovat i zápisy, které byly skutečně vytvořeny a využity při přípravě konkrétních dříve realizovaným programům na sdíleném disku našeho oddílu. Systematická analýza tohoto zdroje by byla vzhledem k nesystematicky strukturovaným zápisům poměrně komplikovaná. Avšak pohled na zkoumanou problematiku touto bází je cenný, protože reprezentuje údaje, které jsou pro skautské programy považovány cílovými uživateli navrhované báze za důležité. Proto aby bylo možné při návrhu zohlednit i pohled na problematiku očima cílových uživatelů navrhovaného systému, byla tato báze ponechána ve výběru.

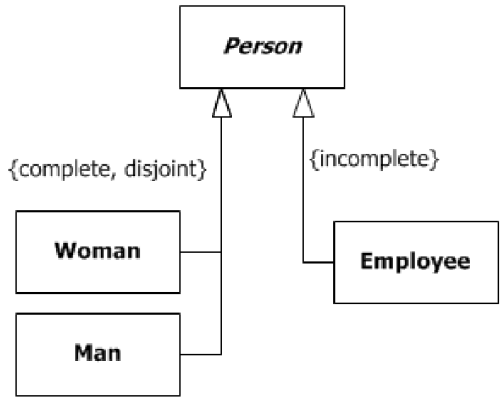
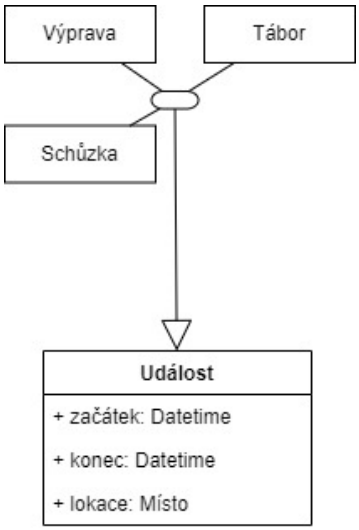


Figure 9.23 GeneralizationSets and constraints

Obrázek 1.2: Standard UML



Obrázek 1.3: Alternativní notace užívané v této práci

Hodnocení (existující báze)

Co se kvality a vhodnosti pro potřeby práce týče, představují existující báze zcela optimální zdroje k analýze. Hodnocení zdrojů proto nesloužilo v tomto případě k vybrání jejich podmnožiny pro další zpracování, jako v předchozí analýze, nicméně stále mělo smysl vybrané zdroje ohodnotit. A to konkrétně ze dvou důvodů, za prvé proto, aby byly představeny kvality bází, které zároveň argumentují jejich výběr. Za druhé pak proto, aby byly představeny jejich nedostatky, které je činí nepřijatelnými či nevhodnými pro takové využití, jaké by měla umožnit v této práci navržená báze.

Analýza obsahu (existující báze)

Cílem analýzy obsahu vybraných zdrojů pak bylo, pro každý z nich zvlášť, zodpovědět následující analytické otázky:

- Jaké pojmy jsou využity danou bází k charakterizování zapsaných aktivit?
- Podle jakých pojmů umožňuje daná báze prohledávat?
- Jaké pojmy (max. 2) nejlépe vystihují obsah dané báze a daly by se tak pro ni označit jako 'základní'?

Výjimku tvoří sdílený disku našeho oddílu (který byl do výběru zařazen z důvodu své cennosti, avšak neumožňuje tak snadné systematické vyhodnocení jako zbylé dvě báze, kvůli své "chaotičtější"struktuře), v jehož případě je vyhodnocena pouze první a poslední analytická otázka. Vzhledem k rozdílným strukturám analyzovaných bází, byly odpovědi na otázky v každém případě získány lehce odlišným způsobem.

Nejsnazší zodpovězení umožnila báze 'chystamprogam', jelikož poskytuje šablonu pro nové záznamy s pojmenovanými a popsanými políčky(prvky) k vyplnění. Odpověď na první otázku

byla proto získána vypsáním názvů, jejich popisů, případně povolených hodnot pro všechny možné prvky ze šablony. K zodpovězení druhé otázky pak byly prohledány filtry poskytnuté ve vrchní části stránky 'Hledat Aktivity' v této bázi a byly vypsány pojmy použité jako názvy parametrů umožňujících filtrování, stejně tak i jejich přípustné hodnoty.

V případě tištěné encyklopedie je trochu problém, protože zápisy v ní nejsou na první pohled nijak systematicky strukturovány, kromě symbolického značení vedle názvů a svého zařazení do kapitol. Bylo by proto potřeba vybrat vzorek ze záznamů a na základě něj se pokusit určit jaké části se napříč záznamy vyskytují opakovaně.
Naštěstí jedna z encyklopedií zahrnuje ve své Úvodní kapitole část pojmenovanou "Jak zapisovat tradiční hry". Autor se v ní sice věnuje takzvaně tradičním dětským hrám, těm, které si děti hrají, ale jejich pravidla existují pouze jako sdílené vědomosti mezi hráči, nejsou však nikde popsána. Postupem času by tyto hry tedy mohly upadnout v zapomnění. Ač se autorova motivace liší od motivace této práce. Výčet parametrů relevantních pro hry, které v této části nabízí, představuje podobně hodnotný zdroj, jako šablona pro zapisování v předchozí bázi. Pro získání odpovědi na první analytickou otázku proto budou vypsány pojmy právě z tohoto výčtu, nikoliv ze skutečně zapsaných her. Co se možností filtrování uloženého obsahu týče, to je zprostředkováno samotnými knihami, jejich kapitolami a symbolickým značením, u jednotlivých záznamů, popsaným v začátku každé z knih. Proto právě z těchto částí by bylo vhodné získat odpovědi na druhou z analytických otázek.
Kapitoly však, vzhledem ke značnému rozsahu encyklopedie, obsahují příliš mnoho pojmů na to, aby je bylo možné, v rámci limitovaného rozsahu této práce, adekvátně systematicky vyhodnotit a nezanedbat v důsledku toho další klíčové části práce. Proto v rámci analýzy encyklopedie her, k zodpovězení druhé analytické otázky, bude použito pouze symbolické značení u jednotlivých her.

Aby bylo možné vyhodnocení sdíleného disku, byla nejprve provedena stručná rešerše obsahu na něm uloženého. A na základě této rešerše jsou pak určeny odpovědi na první a poslední analytickou otázku.

Interpretace výsledků (existující báze)

Výsledky z analýzy, ve formě dlouhého seznamu pojmů, by se při návrhu schématu báze jistě zužitkovaly, nejedná se však o příliš efektivní ani elegantní řešení. Pro vytvoření koherentního celku z právě zmíněného obsáhlého seznamu, čárkou nebo novým řádkem oddělených pojmů, byl proto při návrhu využit diagram tříd. Ten byl sestaven s pomocí získaných pojmů, mých doménových znalostí (jakožto člena oddílu posledních 15 let, zároveň jako vedoucího posledních 6 let) a s pomocí následujícího postupu.

Základní definice Nejdříve byly určeny vztahy mezi obsahem z jednotlivých existujících bází.
Jelikož každá z nich se na problematiku skautských programů dívá z jiného pohledu, agregací

těchto pohledů by měla vzniknout lepší a přesnější základní definice skautských programů, než v případě využití pouze jedné báze. Pro získání této definice byly využity základní pojmy, charakterizující obsah jednotlivých bází, získané předchozím krokem.

Získané základní pojmy a jejich vztahy jsou vymodelovány pomocí jednoduchého, ale pravdivého diagramu tříd, který tak poskytne základní strukturu tvořeného diagramu.

Tento základní prvek, bude v následujícím postupu tvorby diagramu obohacován a doplňován pojmy získanými literární analýzou.

Rozšíření definice Cílem nyní bylo vhodně asociovat získané pojmy k základním třídám, zvoleným jako reprezentace domény skautských programů.

Možné asociace Prvním krokem, provedeným pro určení 'vhodného' způsobu asociace získaných pojmů na základní strukturu, bylo jejich rozřazení do skupin podle toho, k jaké základní třídě se obvykle ve skutečnosti vážou a mohou vázat. Zároveň jelikož již byly určeny konkrétní vztahy mezi jednotlivými základními pojmy, potažmo obsahem jednotlivých bází, bylo určování možné asociovatelnosti daného pojmu s konkrétní základní třídou značně zjednodušeno. Zjednodušení vyplývá ze základního modelu tak, že díky němu je zřejmé, že postačí (například pro pojmy ze sdíleného disku, reprezentujícího záznamy o událostech) vyhodnotit pouze možnou asociovatelnost s pojmem Aktivita, protože to je v generalizační struktuře jeho "sourozenec". Je tomu tak proto, že pokud možnost asociace daného pojmu, z báze primárně událostí, existuje, znamená to automaticky, že Hra, jakožto "potomek"aktivity, je také Aktivita, a není třeba dále posuzovat zda se pojem z báze událostí může asociovat i s hrami.
Zároveň pak to znamená i to, že daný pojem může být převeden do vlastnictví třídy Program, která je "rodičem"jak aktivit, tak událostí.

Z aplikování právě popsané logiky pak vyplývá následující postup pro rozřazení, nikoliv skutečné modelování asociace, ke konkrétním základním třídám.

K určení analýzou získaných pojmů asociovatelných se třídou Program, byly výsledky vybrány z jednotlivých bází podle následujících pravidel:

- z chystamprogram, asociovatelné i s událostmi
- z popisu obsahu sdíleného disku, asociovatelné i s aktivitami
- z encyklopedie her, asociovatelné s událostmi i s aktivitami

Jelikož už ze získaných pojmů byly vybrány ty, které se dají asociovat i s událostmi, nebylo třeba tento vztah dále zvažovat. K určení pojmů asociovatelných se třídou Aktivita, byly tedy výsledky vybrány podle pravidel:

- z chystamprogram, bez pojmů přiřazených programu
- z encyklopedie her, asociovatelné i obecněji s aktivitami, bez pojmů přiřazených programu

K určení pojmů asociovatelných se třídou Hra tak byly vybrány podle pravidla:

- z encyklopedie her, bez pojmů přiřazených programu, bez pojmů přiřazených k aktivitě

Obdobně k určení pojmů ke třídě Událost, byly vybrány podle pravidla:

- z popisu obsahu sdíleného disku, bez pojmů přiřazených programu

Asociovaná struktura Jelikož se mezi získanými pojmy vyskytovaly již existující struktury a naopak některé skupinky pojmů byly až příliš specializované na to, aby bylo vhodné je asociovat přímo na základní třídu, byly provedeny následující kroky (mimo jiné také pro minimalizaci počtu tříd, jenž bylo třeba napojovat).

První hledanou existující strukturou byly třídy (jako například 'oblast rozvoje' a ' podoblast rozvoje'), jenž zahrnují jedna třídu představující generalizaci tříd zbývajících, které jsou jejími specializacemi. Rovněž byly hledány skupiny tříd jako například 'věk skupiny' a 'velikost skupiny', které mezi pojmy z analýzy žádný generalizovaný protějšek nemají. V takových případech záleželo na tom, zda je pro přiřazenou základní třídu "důležité umět rozlišovat"mezi jednotlivými třídami v nalezených skupinkách či nikoliv, a je tak možné zobecnit (generalizovat) celou skupinku jedinou třídou. Příkladem situace, kdy je pro základní třídu důležité umět rozlišovat mezi jednotlivými třídami a generalizování by tak nefungovalo příliš dobře, jsou třeba právě již zmíněné pojmy 'věk skupiny' a 'velikost skupiny'. V důsledku jejich zobecnění, třeba třídou 'charakteristika skupiny' a její asociací k základní třídě (místo dvou separátních asociací k nyní specializovaným třídám), by měla základní třída totiž možnost reprezentovat hodnoty z generalizované třídy pouze jako charakteristiky. Byl by pak ztracen význam jednotlivých hodnot. Proto v situacích jako tato bylo využito místo toho kompozice. To znamená, že pro třídy z dané skupinky tříd s podobným významem byla vytvořena nová třída, která vlastní podobné třídy jako atributy.

Nakonec pak z existujících struktur byly vyhledány a vymodelovány povinné asociace mezi třídami přiřazeným k těm základním. Jelikož i tyto mohou snížit počet tříd nutných asociovat k těm základním v případě, kdy například dvě třídy jsou asociovány mezi sebou ('charakteristický okamžik', 'dokumentární fotografie'), a jedna z nich ('dokumentární fotografie') se do výběru dostala proto, že je primárně silně asociována s druhou z nich, která však už klidně může být asociována i k základní třídě.

Vhodná asociace Díky struktuře vytvořené mezi třídami, jejichž asociací k těm základním byla určována, bylo možné zjistit, které z nich je třeba explicitně asociovat k odpovídající základní třídě. A které třídy toto naopak nepotřebují, protože jsou specializací nebo částí některé jiné, která sdružuje více podobných tříd a jako jediná má tak explicitně vymodelovanou asociaci s odpovídající základní třídou. Zbývalo tedy už jen určit, jak konkrétně má být vymodelována asociace mezi základními třídami a jim přiřazenými, teď již strukturovanými, třídami.

Možnými variantami, vzhledem ke standardu UML, byly buď atribut (vlastnost) dané základní třídy nebo separátní třída, která je s tou první asociována. Toto rozhodování bylo provedeno na základě "síly"konkrétního asociačního vztahu s tím, že silné závislosti jsou modelovány jako atributy daných základních tříd a slabé závislosti jsou modelovány separátními oddělenými třídami. Podle čeho ale vyhodnotit onu sílu asociací?

Ve statistice je například míra asociace určována pro numerické proměnné tím, jak často se konkrétní hodnoty jednotlivých proměnných vyskytují spolu. Dalo by se také říci, že nejsilnější je takový vztah, který je povinný a vyskytuje se proto vždy.
Podobně v této práci je za silnou považována asociace tehdy, pokud se vyskytuje u většiny instancí konkrétní třídy. Opačně v případě, kdy bývá třída asociována jen občas, je považována za slabou. Samotné rozhodnutí pak bohužel nebylo prozatím z důvodu rozsahových možností práce založeno na jiných podkladech, než mých doménových znalostech a informacích z původně analyzovaných bází. V důsledku toho, odpovídá volba tříd k asociaci jako atributy, pravděpodobně více mému přesvědčení o tom, která třída je pro dané záznamy nejužitečnější (a tudíž by bylo nejlepší klást důraz na její zaznamenávání), než podle skutečných frekvencí výskytu ku celkovému počtu záznamů.

Obdobně byla nakonec vyhodnocena i takzvaná 'násobnost' vztahů, která udává, zda instance třídy A mohou být asociovány s více než jednou instancí z třídy B. Volitelnost (instance z třídy A musejí být asociovány s instancí z třídy B) vztahů nebyla vyhodnocena, protože je vycházeno z předpokladu, že všechno zapisování do báze je dobrovolná aktivita (jako vše ve Skautu), tudíž nebudou zápisy vynucovat zapsání žádné ze svých částí a všechny asociované třídy jsou proto volitelné. Navíc jisté zohlednění "povinnosti"k zapsání konkrétních částí bylo učiněno v předchozím kroku v rámci určování síly asociace.

1.2.4 Posouzení úspěšnosti

Úspěšnost tohoto postupu se odvíjí od rozsahu, ve kterém se podařilo identifikovat pojmy asociované ke skautským programům. Pro rozlišení dobrého výsledku od špatného pak byl využit konceptuální model, jenž jsem vytvořil, v rámci přípravy na vypracování této práce, přibližně před dvěma měsíci. Model byl tvořen jen pomocí mých doménových znalostí a pojmů které jsem uznal za vhodné, jedná se tak o velmi odlišný přístup k modelování, než byl použit v této práci.

1.2.5 Výstupy

Hodnocení existujících bází

Výsledky z hodnocení jednotlivých analyzovaných bází, argumentující jejich výběr silnými stránkami jejich obsahu a zároveň se zdůvodněním, proč není využívána daná existující báze, ale je navrhována nová, se nachází ve druhé sekci kapitoly Teorie.

Analýza obsahu bází

Jelikož všechny získané pojmy jsou jsou dále interpretovány a modelovány v navazující praktické části prezentovaných výsledků, není v tomto místě žádný výčet pojmů podle toho z jaké báze byly získány. Namísto toho jsou původní báze daného zdroje uvedeny během interpretace, pokud u daného základního pojmu vůbec nějaké pojmy z ostatních bází budou

Interpretace

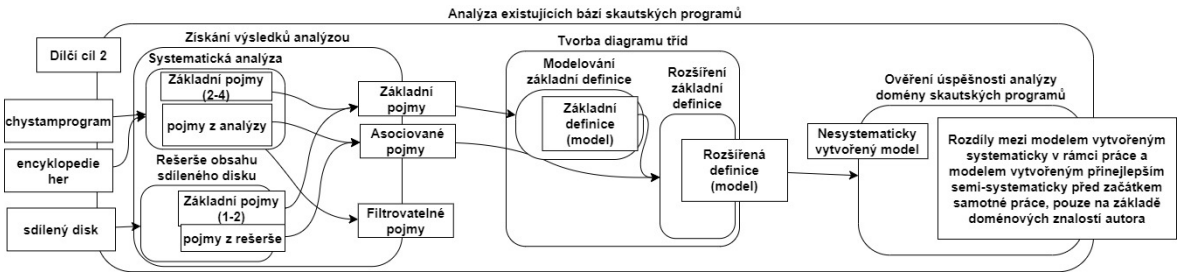
Výsledky interpretace jsou uvedené na začátku praktické části.
Není však představen celý model najednou, nýbrž jsou představeny postupně dílčí pohledy, na část vytvořeného modelu odpovídající jednotlivým základním třídám (počínaje od té nejvíce generalizované). To znamená konkrétně 4 části, kdy každá adresuje a argumentuje kroky provedené pro vytvoření prezentovaného modelu. A to takovým způsobem, aby struktura textu těmto krokům odpovídala tak, že poskytne odpovědi na následující otázky:

- Proč byly konkrétní třídy přiřazeny k dané základní třídě?
- Proč je mezi třídami v dílčím pohledu taková struktura jaká je?
- Proč byly konkrétní třídy asociovány k základní třídě takovým způsobem jak jsou?

Na závěr je uveden pohled na celý diagram.

Posouzení úspěšnosti

Úspěšnost dosažení tohoto dílčího cíle bude určována relativně k modelu, který byl sestaven nesystematicky, vycházeje pouze z doménových znalostí.



Obrázek 1.4: Vizualizace metodiky postupu pro dosažení 3. dílčího cíle

1.3 Metodika tvorby DB schematu

1.3.1 Účel postupu

Důvodem pro návrhu schématu pouze pro konkrétně jeden nástroj je to, že primární nedostatky aktuálního řešení jsou právě v možnostech prohledávání uložených záznamů, a proto práce klade důraz zejména na zdokonalení tohoto aspektu znalostních bází. V důsledku takto stanovených priorit a opět vzhledem k limitovanému rozsahu práce, je navrhována struktura pouze pro úložiště s nejlepšími výsledky z hlediska čtení, i když v rámci návrhu bude pro dosažení Cíle práce využito více než jedno úložiště.

Dílčí cíl: Navrhnout schema pro databázi.

Tento krok je důležitý z toho důvodu, že samotný výběr nástroje nezaručuje jeho správné využití a tím pádem nejvyšší šanci na splnění požadavků zohledňovaných při jeho výběru. Hlavním cílem této části bude poskytnutí odpovědi na otázku "Jak by měl obsah, uložený v navržené bázi, být strukturován, aby umožňoval požadované možnosti prohledávání?".

1.3.2 Konkrétní postup

Rešerše nejlepších praktik modelování dat pro vybranou DB

Získání schematu databáze z konceptuálního modelu by bylo možné alespoň dvěma hlavními způsoby. První z nich by byl využitím poměrně jednoduchého algoritmu, který z logického relačního modelu vytvoří schema pro databázi grafovou. (10) Jelikož i získání relačního modelu z již vytvořeného konceptuálního je velmi přímočaré, mohla by toto být snadná cesta k cíli. A pravděpodobně i je, nicméně takto vytvořený graf nebere v úvahu doporučení identifikovaná v několika oficiálních zdrojích Neo4j jako nejlepší praktiky pro modelování grafových dat, tak aby umožňovaly optimální využití. To znamená, že pravděpodobně bude následně ještě vyžadovat určité své části refaktorovat, aby využil naplno možností, které uložení v grafové struktuře nabízí. V rámci jazyka pro interakci s Neo4j existují i funkce pro snadné refaktorování uložené struktury, takže i to by bylo použitelné řešení. Vhodnější však v případě, že už by nějaká relační báze byla k dispozici, než v tomto. Postup této práce se mírně liší v tom ohledu, že nejprve v teoretické části představí ony nejlepší praktiky pro modelování grafů. Které jsou popsány v dokumentaci Neo4j (11), knize Graph Databases od vydavatelství OReilly věnující se rovněž databázi Neo4j (3) a navíc ještě na blogu jednoho z developerů Neo4j na serveru Medium (4)(12)(13). Jedná se o sadu doporučení pro jednotlivé prvky grafu label, relation, property, node jak by měly být optimálně využívány.

Transformace diagramu tříd na DB schema

Aplikování identifikovaných doporučeni na konceptuální model vytvořený v praktické části a jeho transformace do vhodného schématu.

1.3.3 Ověření kompetencí navrženého DB schematu

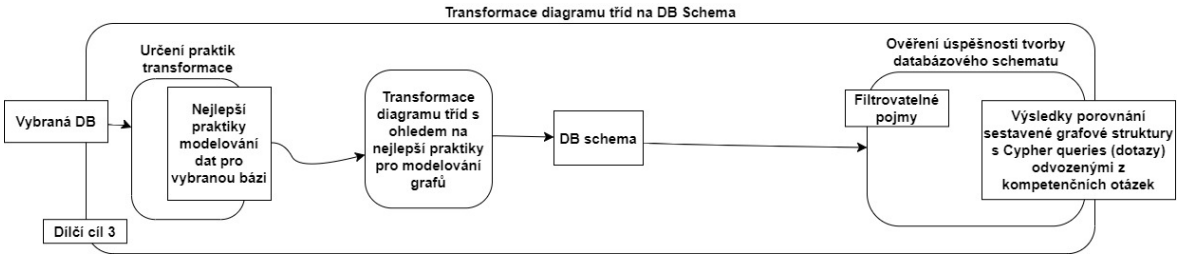
Podle Robinsona (3) je možné určit úspěšnost navrženého schematu dvěma základními způsoby.

1. pokud se graf dobře čte a slova se za sebou vyskytují v přirozeném pořadí, obvykle to signalizuje správně vytvořený model
2. druhou variantu je podle nich takzvaná "queryability", která odráží schopnosti databáze odpovídat na různé položené otázky.

Pro určení úspěšnosti dosažení třetího dílčího cíle, byly použity takzvané 'kompetenční otázky'. Jedná se o pojem využívaný například při tvorbě ontologií (14), nebo i v případě modelování grafových dat například pro Neo4j (3). V obou případech pojem 'kompetenčních otázek' znamená otázky, na které by vytvořená struktura "uměla odpovídat". Nejpochopitelnější je to na příkladu otázek pro bázi jako Neo4j, kdy je snadné si představit, že aby byla báze schopná vám vracet odpovědi na specifické dotazy, musí struktura uložených dat obsahovat všechny údaje potřebné k tomu, aby "uměla odpovědět". Proto se takto stanovené otázky následně užívají jako vodítko při modelování, jelikož pomáhají tvůrci udržet pozornost na zamýšleném účelu pro tvořenou bázi, model, či ontologii.

Otázky využité k ověření vycházely ze schopností existujících bází odpovídat. To znamená, že pokud umožňuje báze chystamprogram hledat aktivity podle toho, které oblasti Stezky odpovídají. Bude odvozená otázka: "Které aktivity jsou asociované s danou oblastí Stezky?". Pro model to tak znamená, že musí obsahovat asociaci mezi třídami OblastStezky a Aktivita. Konkrétně byly vybrány 3 otázky, které umí odpovídat encyklopedie her, 3 odlišné z chystamprogram a jedna navíc, představenou navrženým schématem jako novou možnost, která v žádné z analyzovaných existujících bází ještě není. Při výběru otázek bylo vycházeno z osobních předpokladů a odhadů pro to, které z otázek by mohly tvořit na bázi nejčastěji vznášené dotazy, v případě jejího skutečného nasazení, s tím že byl záměrem vybrat otázky jejichž frekvence výskytu je co nejvyšší. Následně byly přeloženy dotazy do jazyka využívaného databází a na základě nakresleného schématu bylo provedeno vizuální vyhodnocení jednotlivých přeložených dotazů oproti navrženému schématu.

Byla tak tedy provedena určitá kombinace obou dvou základních metod pro ověření modelu, které popisuje Robinson (3).



Obrázek 1.5: Vizualizace metodiky postupu pro dosažení 3. dílčího cíle

1.3.4 Výstupy

Výsledné schema grafové databáze, i kroky k jeho získání transformací konceptuálního modelu jsou prezentováy na závěr výsledků v sekci praktické části 'Schema databáze' spolu s šesti (3+3+1) otázkami přeloženými do Cypheru spolu s hodnocením, zda je možné, otázkou zmíněný "pattertn"(vzor) nalézt i v navrženém schématu.

2. Teoretická část

2.1 Získání infrastruktury

V této sekci jsou prezentovány výsledky z výběr softwaru, který by umožnil návrh báze odpovídající podmínkám stanovených v Cíli.

2.1.1 Hodnocení zdrojů (sw-úložiště)

Z hlediska možností zapisování ukládaného obsahu

Vyhodnocení tohoto hlediska je velmi přímočaré, vzhledem k tomu, že tento návrh klade velký důraz na minimalizaci nových nároků na uživatele. Zejména je důležité minimalizování nároků na zapisování, jelikož pro navrhovanou bázi je klíčové, aby do báze uživatelé zapisovali a sdíleli tak své zkušenosti z připravených programů, čímž budou obohacovat prohledatelný obsah. Proto z tohoto hlediska budou vyřazeni kandidáti, kteří umožňují zapisování obsahu jen pomocí specifického jazyka. Jak bylo řečeno v metodice, ani jeden z produktů společnosti Google toto kritérium nesplňuje, zůstávají tedy jako přijatelné pro návrh. Oproti tomu, ani jedna z databází přes toto kritérium neprojde. Pro interakci s databází MySQL je totiž potřeba využít SQL a v případě Neo4j se jedná pro změnu o 'Cypher', což je také jazyk, jen uzpůsobený k prohledávání grafových struktur.

Jedinými přijatelnými nástroji pro zapisování do báze jsou:

- gDocs
- gSheets

Z hlediska možností čtení uloženého obsahu

Vyhodnocení druhého hlediska však bude již komplexnější. Konkrétně tak, že pro získání výsledků využívá vícero kritérií, jejichž výsledky jsou na závěr agregovány do jednoho souhrnného vyhodnocení.

Z hlediska rychlosti získání odpovědí První kritérium se zaměřuje na rychlost získání výsledků. Při následujícím hodnocení kandidátů nebude mít podstatnou roli. Nicméně nedostatečná rychlost vyhledávání v záznamech, pokud je možné jen manuální otevírání jednotlivých dokumentů podle jejich názvu a umístnění ve složce, je primárním důvodem vzniku této práce. A proto není možnost manuálního prohledávání ani začleněna mezi kandidáty, kteří všichni toto kritérium splňují.

Z hlediska správnosti vrácených odpovědí Další kritérium, absence chyb jak prvního, tak druhého typu, je však již relevantním pro hodnocené kandidáty.

Nejsnazší vyhodnocení tohoto kritéria umožňují kandidáti databázového typu, kdy je totiž tato podmínka zahrnuta již v jejich podstatě jako databázích. Proto u nich absence chyb při vyhledávání nebude dále ověřována a bude předpokládáno, že toto kritérium splňují.

Pro vyhodnocení bezchybného vyhledávání v gSheets bude posuzována vestavěná funkce 'query', která nabízí podobné možnosti prohledávání tabulek v daném dokumentu gSheets, jako relační databáze svým SQL. Jelikož se tedy jedná opět o prohledávání přesně strukturovaných dat, pomocí exaktního algoritmu, bude předpokládáno, že tato funkce operuje bezchybně.

Pro hodnocení vyhledávání v gDocs by mohla být využita funkce 'Najít' rozšiřitelná na 'Najít a nahradit'. Ta nicméně funguje pouze v rámci jednoho dokumentu. Což vzhledem k tomu, že by navrhovaná báze měla být v rámci škálovatelnosti rozdělena do více dokumentů (případně na sebe odkazujících), nedělá z funkce 'Najít' vhodný způsob k hodnocení.
Na základě předpokladu, že by báze neměla být zapsána v jediném dokumentu, ale spíše rozdělena do více dokumentů, bude k vyhodnocení kritéria bezchybného prohledání u gDocs využita funkcionalita prohledávání služby gDrive. Prakticky se jedná o prohledávání obsahu na disku pomocí vyhledávacího řádku na vrchu webového grafického rozhraní služby gDrive. Ten nabízí, mimo možnosti vyhledávat podle názvu a typu souboru, i možnost zobrazit pouze ty dokumenty, které obsahují konkrétní slovo. Právě tato poslední možnost, vyhledávání dokumentů na základě textu v nich obsažených, bude hodnocena podle kriteria bezchybnosti vyhledávání. Toto hodnocení bude provedeno pomocí krátkého experimentu, jenž se bude skládat z vyzkoušení dvou případů.

1. Zapsání klíčového slova "test::"do nového dokumentu a následný pokus o vyhledání dokumentů podle toho, zda obsahují klíčové slovo. Aby byla zohledněna možnost, že systému trvá nějakou dobu, než provede indexování nově vytvořených dokumentů a umožní tak vyhledávání v nich, bude vyzkoušena ještě následující situace.

2. Vybrání co nejunikátnějšího klíčového textu z libovolného souboru zapsaného na mém disku déle než měsíc, následované pokusem o nalezení daného souboru podle toho, že obsahuje vybraný klíčový text.

Z výsledků těchto dvou experimentů vyplynulo, že tato varianta prohledávání jednoznačně není bezchybná. Jelikož oba pokusy o vyhledání dokumentu obsahujícího klíčový text byly neúspěšné, byly sice rychlé, avšak dokument z něhož byl získán klíčový text použitý k hledání, nebyl zobrazen ani v jednom případě. Postupujícími sw kandidáty k dalšímu hodnocení jsou tedy pouze gSheets a databáze MySQL a Neo4j. gDocs byly na základě experimentálně zjištěných výsledků vyhodnoceny jako nástroj nepřijatelný pro uživatelské rozhraní zprostředkující čtení obsahu navrhované báze.

Z hlediska příjemnosti zadávání dotazů Kritérium příjemnosti je ze všech kriterií zatím nejkomplexnější, proto i jeho vyhodnocení bude tomu odpovídat.
Jak bylo stanoveno v metodice, bude nejprve vyhodnoceno pořadí zbývajících kandidátů podle každého z dílčích kriterií zvlášť. A tak získané dílčí výsledky budou následně využity k výběru nejvhodnějšího z kandidátů z hlediska příjemnosti prohledávání obsahu navrhované báze.

Následující dílčí kriteria budou vyhodnocena pro gSheets (funkce 'query'), MySQL (SQL) a Neo4j (Cypher). Využito bude primárně odhadů a dedukce.

1. počet znaků potřebných k napsání dotazu Při hodnocení tohoto dílčího kritéria, je vycházeno z předpokladu, že funkce gSheets 'query', jakožto pouhá napodobenina funkcionality nabízené 'SQL', bude v případně jednoduchých dotazů pravděpodobně i stejně úsporná na potřebné znaky jako jako SQL pro obdobný dotaz. Pro komplexnější dotazy, vyžadující například data z více tabulek, je potom předpokládáno, že SQL bude, ve srovnání s funkcí query v gSheets, umožňovat díky své podstatně rozvinutější funkcionalitě způsob zapsání daného komplexnějšího dotazu s nižším počtem znaků, než funkce query. Pro porovnání počtu znaků vyžadovaných k napsání dotazu bázi Neo4j a MySQL je využito článku s názvem 'Use graph databases for complex hierarchies' (15). Tento článek na modelovém příkladu dat vyhodnocuje několik různých dotazů a porovnává jejich zápis a následný postup vyhodnocení v případě využití SQL oproti případu s využitím jazyka Cypher. V této práci jsou však zohledněny pouze porovnání zapsaných dotazů, nikoliv způsoby jejich vyhodnocování. Z výsledků prezentovaných v článku vyplývá, že pro zapsání SQL dotazu je téměř v každém případě potřeba více znaků, než pro získání stejných výsledků pomocí jazyka Cypher. SQL přitom v některých případech vyžaduje až několikanásobně více znaků než ekvivalent zapsaný Cypherem.
Stejný závěr vyplývá i z publikace 'The Definitive Guide to Graph Database' napsané Michaelem Hungerem a spol. (3) Proto navíc tato práce předpokládá i to, že ani pomocí funkce query v gSheets, není možné dosáhnout kratšího zápisu dotazů, než v případě Neo4j a Cypheru, jelikož odhadovaný počet znaků vyžadovaný funkcí query je v jednoduších případech podobný jako v případě SQL a ve složitějších situacích horší než SQL. Výsledné pořadí tohoto dílčího kriteria proto je:

1. místo - Cypher
2. místo - SQL
3. místo - =query()

2. Nabízí v základu webové GUI? Pro vyhodnocení druhého dílčího kriteria není třeba se uchylovat k odhadům, jelikož kandidáti buď budou nabízet možnost webového GUI v základní instalaci nebo nebudou. Určení této binární hodnoty pro kandidáty, bude provedeno prohledáním internetových zdrojů s dotazem například "MySQL web GUI". A na základě nalezených výsledků bude určeno, zda daný kandidát úspěšně splní toto kriterium. Pro gSheets

bylo vyhodnoceno, vzhledem k podstatě nástroje jako na cloudu založené služby, že vskutku nabízí ve svých základních možnostech zobrazení grafického rozhraní v prostředí prohlížeče bez nutnosti lokální instalace čehokoliv jiného než samotného prohlížeče. Pro MySQL z hledání vyplynulo, že grafická rozhraní umožňující přístup k této bázi jsou typicky povahy lokální instalace.
Rovněž však existují i možnosti jako myPhpAdmin (16), který je zdarma a nabízí webové GUI. Nicméně všechny z těchto variant jsou dodatečné nástroje, z nichž by bylo potřeba provést patřičný výběr, kdyby byly návrhem uvažovány jako možnosti. Jelikož tedy žádné, v základní instalaci zahrnuté, webové GUI není pro MySQL k dispozici, znamená to pro relační databázi podle tohoto kriteria druhé a zároveň poslední místo. Pro Neo4j naopak bylo velmi snadné najít nástroj 'Neo4j Browser', jelikož to byl první výsledek po zadání dotazu "Neo4j web gui". Jednalo se o odkaz na oficiální dokumentaci Neo4j, kde bylo řečeno, že se jedná o výchozí rozhraní pro interakci s databází a zároveň je zahrnuto ve výchozí instalaci (11). Výsledné pořadí tohoto dílčího kriteria proto je:

1. místo - gSheets, Neo4j
2. místo - MySQL

Souhrnné výsledky Finální pořadí kandidátů podle kriteria příjemnosti čtení jejich obsahu je proto následující:

1. místo - Neo4j (1+1)
2. místo - gSheets (1+3), MySQL (2+2)

A kandidátem vybraným v rámci hlediska čtení zapsaného obsahu se tak stává databáza Neo4j, jelikož z posuzovaných kandidátů umožňuje interakci pomocí dotazů s nejnižším počtem znaků a zároveň k interakci s ní není třeba žádný dodatečný software, který by nebyl zahrnut v základní instalaci.

2.1.2 Výsledky analýzy obsahu (sw-úložiště)

Vybraným softwarem pro základ navrhované báze jsou tedy gDocs a gSheets jako uživatelské rozhraní pro zapisování údajů do báze a případnou modifikaci zapsaných údajů (spolu s databází Neo4j sloužící jako uživatelské rozhraní k prohledávání zapsaných údajů v navrhované bázi skautských programů). Dále boudou představeny datové struktury využívané jednotlivými vybranými nástroji spolu s představením jejich možností vzájemné integrace.

Datové struktury (uložení)

gWorkspace Oba tyto nástroje z prostředí gWorkspace (gDocs, gSheets) mají jeden aspekt své struktury shodný. A to sice ten, že v obou případech se jedná o soubory, uložené na disku google (gDrive).

Každý ze souborů pak má přiřazené unikátní id, které je mimochodem součástí webové adresy (url) využité pro zobrazení GUI editoru daného souboru. Pokud tedy v prohlížeči bude otevřen jeden konkrétní soubor tabulek z disku s identifikátorem ID, url zobrazované ve vyhledávacím řádku prohlížeče bude ‘$https : //docs.google.com/spreadsheets/d/ID/editgid = 0$‘ (17). Adresa funguje i v případě vynechání textu za posledním lomítkem, i v případě že je text "spreadsheets"nahrazen textem "docs"a je použito ID náležící dokumentu místo tabulek.
Kromě id má také každý soubor přiřazený název, typ (gdocs,gsheets,pdf,...) a další. Navíc může být přiřazen například popis, který se zobrazuje v GUI gDrive i gDocs, ale i další volitelné atributy, se kterými je však možno interagovat jen pomocí REST API (18).

Vnitřní strukturu souborů už však mají oba nástroje specifickou. V případě dokumentů, je každý tvořen například záhlavím, zápatím a tělem dokumentu (nejedná se o kompletní výčet) (17). Tělo dokumentu je pak dále členěno na jednotlivé elementy.
V praxi je elementem každý nový řádek vytvořený stisknutím klávesy ’enter’, případně vložený objekt jako třeba tabulka, nebo obrázek. Každý element pak může mít přiřazené hodnoty reprezentující jeho formátování, ale i konkrétní text zapsaný v daném elementu. Navíc, protože pro dokumenty je důležité pořadí zapsaných elementů, je s každým elementem asociován i identifikátor vyjadřující pořadí daného elementu v rámci těla dokumentu (19). Je tak například možné získat na jakých pozicích, z hlediska pořadí v dokumentu, jsou nadpisy úrovně 1 a pomocí jednoduchých aritmetických operací získat na jakých pozicích v dokumentu začíná i končí elementy pod konkrétním nadpisem 1. úrovně.

V případě tabulek, jsou jednotlivé soubory organizovány do listů (stránek), s tím, že každý list je tvořen tabulkovou strukturou, ve které může být zapsáno i více tabulek. Konkrétní rozsahy v rámci listů mohou být také pojmenovány a reference na ně tak mohou být realizovány pomocí tohoto pojmenování (20).
Jelikož se však jedná spíše o sekundární rozhraní pro navrhovanou bázi, které je zamýšleno zejména na zapisování, dalo by se říci, konfiguračních údajů (dostupné materiály, seznam členů, výchovné cíle), popis struktury jeho souborů není rozebírán do větších podrobností.

Neo4j Neo4j, vzhledem ke své podstatě databáze, má strukturu uložených údajů značně odlišnou. Struktura označovaná jako ’property graph’ využitá Neo4j k uložení zapsaných dat, je na disku realizována pomocí několika odlišných souborů. To konkrétně znamená, že každá část uložené grafové struktury (nodes-vrcholy, relationships-vztahy, lables-popisky/štítky, properties-vlastnosti) je uložena v separátním souboru (21). Všechny tyto čtyři soubory se přitom skládají ze záznamů o fixní délce bytů. Dalo by se na ně tedy pohlížet jako na tabulky, ve kterých je možné velmi rychle přistupovat ke konkrétním záznamům, pokud známe pořadí, ve kterém byly do souboru zapsány. Právě proto je databází využita tato fixní struktura, jelikož je s její pomocí možné efektivní propojení jednotlivých částí napříč čtyřmi separátními soubory. Například pokud je k vrcholu přiřazená vlastnost, bude ve vyhrazeném místě (bytech) pro zaznamenání přiřazených vlastností v daném záznamu v souboru vrcholů uvedeno pořadí, ve kterém byla přiřazená vlastnost zapsána do souboru obsahujícího vlastnosti.
Dalo by se tak říci, že pořadí zápisu jednotlivých záznamů do souborů, představují primární

klíče pro jednotlivé "tabulky"a zachycení grafové struktury je dosaženo pomocí zápisu těchto klíčů k ostatním souvisejícím částem jako cizích klíčů.

Dále jsou v knize "Graph databases"od vydavatelství OReilly popsány i konkrétní struktury jednotlivých souborů. V rámci představení struktury databáze, jsou proto představena i tato specifika. Popsaná struktura záznamu v souboru ukládajícím vrcholy je následující (3).

- - byte 1 - (in-use flag) slouží bázi k určení, zda je daný záznam používaný nebo zda může být smazán a jeho pozice tak uvolněna
- - bytes 2-5 - reprezentují identifikátor prvního připojeného vztahu (odkaz realizovaný pořadím záznamu v souboru vtahů)
- - bytes 6-9 - reprezentují identifikátor první připojené vlastnosti (odkaz realizovaný pořadím záznamu v souboru vlastností)
- - bytes 10-14 - reprezentují odkazy na přiřazené 'lables', případně konkrétní štítky/popisky, pokud jich je přiřazeno pouze nízké množství
- - byte 15 - rezervován pro budoucí využití

Jak je řečeno v knize, jedná se tak prakticky jen o "hrstku"odkazů, odkazujících do seznamů vztahů, popisků a vlastností"(3).

Pro záznam v souboru ukládajícím vztahy je pak popsána následující struktura (3).

- - byte 1 - (in-use flag) značící, zda záznam může být smazán a nahrazen novým
- - bytes 2-5 - identifikátor prvního vrcholu v tomto vztahu (v případě směrovaného vztahu je tento vrchol počátkem)
- - 6-9 - identifikátor druhého vrcholu v tomto vztahu (v případě směřovaného vztahu je toto koncový vrchol)
- - 10-13 - identifikátor typu vztahu, který odkazuje na soubor obsahující seznam všech typů vztahů v databázi použitých
- - 14-17 - identifikátor předchozího vztahu počátečního vrcholu
- - 18-21 - indetifiátor následujícího vztahu počátečního vrcholu
- - 22-25 - identifikátor předchozího vztahu koncového vrcholu
- - 26-29 - identifikátor následujícího vztahu koncového vrcholu
- - 30-33 - identifikátor první připojené vlastnosti
- - 34 - první v řetězu vztahů?

Vlastnosti (properties) jsou potom uloženy následujícím způsobem.
Opět je využita fixní velikost pro jednotlivé záznamy. S tím, že každý záznam vlastnosti obsahuje odkaz na další související záznam. To je z důvodu, že vzhledem k fixní velikosti uložených vrcholů a vztahů, jsou k těmto částem grafu zaznamenány pouze odkazy na první přiřazenou vlastnost. Ostatní přiřazené vlastnosti jsou pak připojeny právě pomocí odkazu na následující záznam vlastnosti obsažený v záznamu první přiřazené vlastnosti k vrcholu, či vztahu.
Podobný princip je pak aplikován i při procházení všech ostatních prvků v zaznamenaném grafu.

Kromě odkazu na další záznam v tomto 'řetězu vlastností', je u každého záznamu uveden datový typ dané vlastnosti (jakýkoliv primitivní typ podporovaný Java Virtual Machine, strings, arrays JVM primitivních typů), spolu s odkazem na soubor 'indexu vlastností', který obsahuje jména vlastností použitých v bázi.
Na závěr je u každého záznamu vlastnosti buď uložena samotná hodnota vlastnosti (pokud je dostatečně malá, aby se vešla do fixně velkého záznamu) nebo v případě, že velikost hodnoty přesahuje velikost místa poskytnutého záznamem fixní velikosti, je uložen pouze odkaz na zapsanou hodnotu vlastnosti, a samotná hodnota je uložena do speciálního dynamického úložiště vlastností, které je realizováno opět samostatným souborem.
Ve skutečnosti Neo4j disponuje dvěma těmito dynamickými úložišti. První je optimalizováno pro uložení delších textů (stringů) a podporuje například full-text indexování a následné prohledávání těchto textů podle obsaženého textu. A druhé optimalizované pro uložení delších polí (arrays), typicky obsahujících čísla, v oblasti strojového učení a neuro sítí, také označovány jako vektory nebo tensory (22)(23).

Toto dynamické úložiště polí proto může být využito například pro uložení takzvaných 'embedded' hodnot, které jsou získány "zakódováním"nějakého vstupu (text, obrázek,...) pomocí AI modelu.
Takto získané hodnoty se následně využívají k 'Retrieve Augmented Genaration', což prakticky znamená proces, ve kterém jsou nejdříve vyhledány záznamy, jejichž 'embedded' hodnota je vektorově podobná 'embedded' hodnotě uživatelem zadaného dotazu. Z nalezených vektorově podobných záznamů je následně vybráno vrchních x a ty jsou poskytnuty některému z jazykových modelů spolu s uživatelovým dotazem, aby na základě takto obohacených podkladů teprve vygeneroval odpověď zobrazenou nakonec uživateli.

Integrovatelnost (programový přístup)

gWorkspace Interakci s vybranými nástroji z gWorkspace pomocí programového kódů zprostředkovává googlem provozované REST API, to umožňuje pomocí http dotazů jak získávání obsahu jednotlivých dokumentů, tak i modifikaci jejich obsahu.
Pro použití tohoto API je však potřeba každý dotaz adekvátně autorizovat (24). Existuje nicméně ještě jedna možnost interakce s dokumenty pomocí kódu, která ale nevyžaduje explicitně autorizovat každý dotaz. Jedná se o interakci se službami v rámci gWorkspace pomocí služby google Apps Scripts, která je rovněž zahrnuta v gWorkspace.

Samotné Apps Scripts představují službu, která umožňuje napsání téměř libovolného javascript kódu a jeho spouštění v rámci stanovených limitů zdarma (ve skutečnosti je to googlem přizpůsobená verze javascriptu, ale liší se pouze v drobnostech, upravených pravděpodobně proto, aby zneužívání této služby bylo složitější a vyskytovalo se tak méně často - konkrétní příklad takové drobnosti je uveden dále v textu). Scripty mohou být spouštěny buď časovačem nebo přes "zavolání"url adresy přiřazené automaticky implementaci daného kódu napsaného v Apps Scripts. Hlavní výhodou při použití Apps Scripts je to, že rozhraní ostatních služeb (gDocs, gSheets, gDrive,..) není nutné volat pomocí REST API a http do-

tazů (vyžaduje manuální autorizaci), ale stačí rozhraní dané služby přidat jako knihovnu ke psanému scriptu. Jedinkrát při prvním spuštění je pak třeba odsouhlasit, že jako majitel účtu souhlasíte s tím, aby daný script měl přístup k využité službě, tím starosti s autorizací požadavků končí (25)(26). Kromě denního limitu na počet spuštění, je bezplatné využití této služby vykoupeno ještě jedním podstatným omezením. A to sice, že není možné provádět "volání ven"ze skriptu (externí komunikaci) jinak než s využitím předdefinované funkce 'UrlFetch()'. To zároveň znamená, že i pokud se podaří dostat do skriptu knihovnu například pro komunikaci s databází nebude tato knihovna fungovat (27).

Neo4j Možnosti programové interakce s databází Neo4j závisí na tom, která z implementací je využita.
První varianta implementace Neo4j je cloud verze nabízená jako SaaS, spolu s poměrně dostatečným objemem zdrojů v rámci bezplatné úrovně účtu. Tato verze nicméně umožňuje programovou interakci, pouze pomocí knihoven, které jsou sice pro většinu nejběžnějších jazyků k dispozici (takže ve většině případů bude tato varianta nabízet dostatečnou konektivitu), avšak v případě (jako dříve zmíněné Apps Scripts, které omezují možnosti externí komunikace pouze na http dotazy skrze předdefinovanou funkci) absence podpory představuje http komunikace v cloudové verzi Neo4j poměrně problém.
Naštěstí existuje druhá varianta implementace, konkrétně takzvaná 'self-hosted' varianta, která může být například s využitím dockeru nebo pomocí klasické instalace spuštěna na libovolné výpočetní instanci (počítači). A tato 'self-hosted' varianta umožňuje jak programovou interakci pomocí http, tak pomocí knihoven pro konkrétní jazyky.

2.2 Získání pojmů

Tato sekce prezentuje první část výsledků z výběru pojmů asociovaných se skautským programem.

2.2.1 Hodnocení zdrojů (obsah existujících bází)

chystamprogram

Předností této báze je její zaměření na výchovnou a rozvojovou hodnotu programů. Zaměření je především zprostředkováno díky možnosti zapisovat k jednotlivým programům jejich výchovný cíl. Ale také možností zapisovat, na který bod ve Stezce je program napojen. Což je dobré a užitečné proto, že Stezka nepředstavuje jen rámec pro děti, podle kterého by se mohly samy všestranně rozvíjet. Rovně ale jako pomůcka pro vedoucí, když připravují vhodný program pro nadcházející schůzku například. (28)

Drobná nevýhoda však vyplývá z toho, že se jedná o veřejnou bázi za kterou zodpovídá samotná organizace Junák. A to sice, že pro zapsání nového programu, je potřeba být přihlášen

skautským účtem ze skautIS a výsledný zápis musí být nejdříve ověřen jejich metodickým týmem. Což sice bude mít pravděpodobně pozitivní vliv na úroveň kvality zaznamenaného obsahu. Nicméně pro sdílení například programu, který je teprve připravován, to tak není vhodné řešení.

Jako větší nedostatek však vnímám spíše omezenou možnost poskytování zpětné vazby, a komentářu k zapsaným programům. Jediná možnost hodnocení, je totiž zaslání svého hodnocení pouze soukromě autorovi daného programu. Což opět pro interní využití, které by mělo umožňovat sdílení obsahu a na něm následnou spolupráci, není vyloženě příhodné.

encyklopedie her

Hlavní a nespornou předností této báze je její úctyhodný rozsah. Ikdyž vzhledem k době jejího vydání, existuje šance, že některé zapsané hry, nebudou již dnes pro děti zábavné. Avšak vzhledem ke zmíněnému objemu, by se musela od té doby změnit kompletně celé podstata dětských her, aby tento zdroj již nebyl relevantní, což tato práce nepředpokládá. S rokem vydání encyklopedie souvisí však i další nevýhody, které přehlédnout nelze. Klíčovým nedostatkem jou značně limitované možnosti efektivního prohledávání, které jsou implicitním důsledkem tištěné formy báze. Hry jsou sice jednotlivými knihami rozděleny podle prostředí do kterého jsou vhodné a dále pak pomocí kapitol. Navíc každá hra má vedle názvu uvedeno specifické značení, popsané na začátků každé z knih, které poskytuje informace například o tom, pro jaký počet, nebo věk hráčů je hra vhodná. Nicméně to pořád znamená, že je třeba záznamy procházet manuálně a vizuálně kontrolovat, zda chci o dané hře číst více. Nemluvě o tom, že doplňování zpětné vazby či komentářů rovněž není možné.

sdílený disk našeho oddílu

Jak už bylo řečeno, z hlediska této práce je největší předností této báze její původ, tedy to, že byla vytvořeny členy oddílu. V Souvislosti s konkrétními připravovanými programy během činnosti oddílu. Navíc společnost Google Podporuje Skauting a poskytuje nejen o 40 GB sdíleného disku jednotlivým oddílům ale i pak dalších 40 GB každému jednomu členovi. (29) Poskytuje tím tak sjednocenou platformu pro zaznamenávání jak pracovních podkladů využitelných jen jejich autorem, ale i následné sdílení předatelných informací a znalostí z jednotlivých zápisů časem syntetizovaných.

Zároveň se také jedná "nativně"(původně) cloudovou službu, s čímž je spojená například perfektní real-time kolaborace při úpravách dokumentů, a možná nepřímo i příjemně ovladatelné mobilní aplikace pro jednotlivé služby.

Na druhu stranu však ve spojitosti s cloudovou podstatou této služby se vyskytují i určité nedostatky. Příkladem mohou být limitované možnosti prohledávání uloženého obsahu, identifikované během hodnocení spolehlivosti prohledávání obsahu analyzovaných sw kandidátů na uložiště.

Např. OneNone Desktop umí také vyhledávat "jen"pomocí jednoho pole pro zadání hledané části textu a následného prohledání full-text indexu z uloženého obsahu. A při obdobném experimentu, jako pro vyhledávání mezi soubory na gDisku, podle jejich obsahu (Zapsání někam na běžné využívané místo v dané struktuře, unikátní string a následný - pokus o jeho vyhledání pomocí testované funkcionality dané struktury.), provedeném v desktopové aplikaci OneNote, však nebyla nalezena jediná nepravdivá odpověď a navíc bylo vyhledávání místo cca 2s téměř instantní.

2.2.2 Výsledky analýzy obsahu (obsah existujících bází)

rešerše obsahu sdíleného disku

Většina dokumentů na sdíleném disku jsou dokumenty využívané pro administrativní účely, nicméně několik složek je dedikováno záznamům o programech. První složka pojmenovaná 'Výpravy' obsahuje jeden dokument na jednu výpravu, s tím že zapsaných výprav je přibliženě 40. Výpravy představují události organizované typicky na jeden, či více dní, kdy vedoucí připravují pro děti nějaký program. Jelikož však i samotná výprava potřebuje přípravu, dá se říci, že samotná událost výpravy je připravovaným programem. Mezi typicky organizované události patří kromě výprav ještě schůzky a tábory. Všechny události přitom mají tu společnou vlastnost, že se skládají z určitých bloků, které mají nějaké naplánované pořadí, to se však může lišit od reálného průběhu události. Jednotlivé bloky pak představují konkrétní aktivity a hry, které jsou při události realizovány. Druhá složka nese název 'Programy' a obsahuje několik neroztříděných aktivit, které mohou být využity při libovolné připravované události. Navíc jsou zde však i podsložky pojmenované podle jednotlivých věkových skupin (vlčata, skauti), které obsahují popsané aktivity (programy) zamýšlené buď pro mladší, nebo pro starší.

2.3 Získání nejlepších praktik pro modelování dat ve vybrané DB

V této sekci jsou prezentovány výsledky z výběru nejlepších praktik identifikovaných pro vybranou DB.

2.3.1 Nodes (vrcholy)

Vrcholy jsou podle oficiální dokumentace definovány jako první entitu, kterou je vhodné celé modelování domény začínat. Rovněž se jedná o jeden ze dvou základních stavebních prvků z nichž se graf skládá (tou druhou jsou vztahy mezi jednotlivými vrcholy).(30) Všechny zdroje se zároveň shodují na tom že vrcholí se používají k modelování věcí v doméně našeho zájmu. Tyto věci pak bývají v přirozené řeči typicky reprezentovány podstatnými jmény, ale je tak možné modelovat i konkrétní fakta jako například konkrétní rok. Pokud by pak gra-

fové schéma obsahovalo kromě vrcholů a roků i vrcholy konkrétních uskutečněných událostí, bylo by možné vztahy mezi vrcholem události a vrcholem roku reprezentovat v jakém roce se událost uskutečnila. (3) To by samo o sobě nebylo nic úžasného, avšak v kombinaci s grafovou strukturou jejíž specifika jsou popsána v teoretické části, volba této varianty znázornění umožní vyhledat velmi snadno všechny záznamy událostí, které se uskutečnily v daném roce. Je to díky tomu, že v grafové struktuře jsou vztahy tím prvkem, který propojuje jednotlivé záznamy, znamená to prakticky to, že záznamy událostí jsou dostupné hned vedle záznamu odpovídajícímu roku jejich uskutečnění Vrcholy tak mohou být vhodnou volbou pro kategorické proměnné, které mají velmi vysokou kardinalitu (4) je tak pomocí těchto prvků možné podobně reprezentovat i komplexní (složené) hodnoty. (3) Je však potřeba dát si pozor na takzvané 'super vrcholy'. To značí vrcholy, které mají příliš mnoho vztahů. Vzhledem totiž k tomu jak Neo4j ukládá dat (vztahy daného vrcholu odkazují na další i předchozí vztah daného vrcholu stejného typu), je při každé prohledávání do kterého je zahrnut onen 'super vrchol' procházet všechny tyto jeho vztahy, což zásadně ovlivňuje rychlost na získání odpovědí.(12) Jako přirovnání k běžné relační databázi by se pak dalo říci, že jednotlivé vrcholy odpovídají jednotlivým řádkům v tabulkách.

2.3.2 Labels

Než bude prezentována definice pojmu 'label' ve spojitosti s Neo4j, je pro tento pojem stanoveno významově odpovídající české slovo, které když bude použito v textu této práce a nebude řečeno jinak, odkazuje právě na pojem 'label'. Za český ekvivalent je dále využíváno označení 'štítek'. Toto slovo nebylo zvoleno jen pro svoji významovou blízkost se slovem label, ale zároveň protože se relativně málo používá, oproti například pojmu 'označení', kteréžto slovo mi vrátil slovník.

Podle oficiální dokumentace "Štítek je pojmenovaný prvek struktury grafu, který je užíván k seskupování vrcholů do skupin, respektive jejich různých setů (sad)."(30) Bylo by tak možné tyto štítky přirovnat k tabulkám v relační databázi, na rozdíl však od relační databáze a tabulek, vrchol může zároveň mít i více štítků a být tak členem několika různých sad. Efektivně to tak umožňuje přistupovat přímo k podmnožinám z celého uloženého grafu. Proto je vhodné, modelovat štítky vrcholu podle toho, které údaje budou zahrnuty v dotazech jenž budou databázi zadávány.

Na rozdíl však od vrcholů, nejsou štítky vhodné pro proměnné s vysokou kardinalitou, nýbrž se hodí pro reprezentaci proměnných s nízkou kardinalitou a v případě, kdy se jednotlivé kategorie stanovené proměnnou mohou překrývat (vyskytovat zároveň). Je také doporučováno používat maximálně 4 štítky na jeden vrchol, což by dle tvrzení autorů mělo být více než dostačující prostou většinu případů, i těch když se ukládána opravdu rozsáhlá data. V běžných případech by pak na dostatečné rozlišení jednotlivých podmnožin v grafu mělo bohatě stačit 1 až 2 štítky ne 1 vrchol. (13)

Autoři zároveň odrazují čtenáře od toho, aby s pomocí štítků modelovali hierarchické vztahy

typu "pojem A"JE "pojem b". Důvodem pro to je jednak to, že databáze nepodporuje nastavování omezujících podmínek pro štítky přiřazené k vrcholu a rovněž proto, že s prohlubující se hierarchií velmi rychle roste i počet vyžadovaných štítků. Což opět vzhledem ke struktuře ve které databáze ukládá data, má negativní dopady na výkon.
Argumentují také tím že modelování generalizovaných tříd není nutné, protože pro získání odpovědí se stejným obsahem jako při využití dané generalizované třídy stačí pouze v dotazu explicitně vyjmenovat jednotlivé podtřídy a spojit je pomocí logických výrazu, a databáze tak vrátí odpovídající například sjednocené množiny záznamů.(13) Navíc podobně jako u vrcholů, mužem dojít k vytvoření takzvané super třídy v případě že je 1 štítek přiřazený k příliš mnoho záznamů. Kromě hierarchií by také štítky neměly být využity pro reprezentaci vztahů, které jsou typu "MÁ", takový vztah by měl být vyjádřen pomocí struktury k tomu určené, což jsou vztahy.

Na rozdíl od hierarchie a vlastností však autoři důrazně doporučují zakládat výběr použitých štítků na konkrétních otázkách které jsou požadovány, aby tvořená báze byla schopná zodpovídat(13)(3)

2.3.3 Relationships

Vztahy představují druhý ze základních prvků grafové struktury. Díky nim je v grafu uložená veškerá struktura a propojení mezi jednotlivými záznamy. (3) Hodí se říci že při vytváření vztahu mezi vrcholy databáze vyžaduje databáze aby byl specifikován směr tohoto vztahu, nicméně databáze umožňuje procházet i tyto vztahy jako nesměrované.(30) Klíčovým principem grafové databáze je takzvaný "žádné rozbité odkazy"princip, který zajišťuje, že vztah nikdy nebude odkazovat k neexistujícímu vrcholu, výsledně to tak znamená, že není možné smazat vrcholu pokud k němu vede nějaký vztah.(30) Jakoby pro identifikaci vrcholů měly posloužit podstatná jména, pro identifikaci vztahů by měl fungovat slovesa.

Užitečné je i vědět, že využitím vztahu dochází k normalizaci dat.(4) Účelem tohoto procesu je odstranění duplicit nich hodnot vyskytujících se mezi množstvím záznamů v bázi a jejich nahrazením za odkaz na danou hodnotu, která byla z duplicitních výskytů extrahována a umístěna do vlastní tabulky v případě relační databáze. Ve zvolené grafové databázi se tento postup příliš neliší, rozdíl je zejména v tom, že oproti tabulkám je grafová struktura podstatně flexibilnější a není tak problémem refaktorovat uloženou strukturu podle potřeby. Není k tomu třeba podstupovat migraci databáze či jiný složitý proces, nýbrž je možné využít nativně podporovaných funkcí určených pro transformaci daných prvků uložený struktury na strukturálně odlišný typ prvku. Je tak možné převést třeba hodnotu vybraného štítku, na reprezentaci odděleným vrcholem, jenž nebude mít změněný význam.

2.3.4 Properties

V textu dále označované jako vlastnosti představuji nejuniverzálnější prvek grafové struktury, je možné ho přiřadit totiž jak k vrcholům tak ke vztahům. Avšak ke vztahům se doporučuje přiřazovat pouze v nejnutnějších případech, jelikož typicky bývá nejvýhodnější vymodelovat místo 1 vztahu který bude mít větší počet vlastností, radši několik různých vztahů které povedou mezi stejnými vrcholy a každý z nich bude reprezentovat jinou z těch vlastností. Vhodnější je to z pohledu výkonnosti databáze jelikož v případě vlastností je potřeba vždy procházet skrz všechny asociované vlastnosti (dokud není nalezena odpověď), zatímco vztahy můžeme identifikovat jejich typem a na základě toho přistupovat přímo k nim.

Vhodným využitím vlastností, je naopak podle autorů pro frekventovaně měněné hodnoty, zaznamenávání metadat jako časové známky nebo čísla verze mezi vlastností vrcholů nebo pro vztahy vyjadřování jejich síly, váhy, nebo kvality, zároveň s metadaty stejně jako v případě s vrcholy. (4)

Nehodícím řešením jsou vlastnosti oproti tomu v případě, kdy se vlastností reprezentované hodnoty mohou překrývat, nebo jich může být pro jednu instancí více než jedna. Stejně tak nešikovným je přitom využití vlastnosti pro hodnoty, které budou často využívány v rámci dotazů s účelem získat podmnožinu záznamů sdílejících konkrétní hodnotu dané vlastnosti. V takových případech se hodí modelovat pojem spíše jako separátní vrchol.

3. Praktická část

3.1 Infrastruktura pro bázi

3.1.1 Předpoklad

Jak bylo stanoveno v metodice, tato část popisuje infrastrukturu (softwarové nástroje) do základu navrhované báze, která by v souladu s cílem práce (viz. Cíl), nevyžadovala víc prostředků na údržbu, než sama ušetří a zároveň přitom odpovídala i ostatním požadavkům Cíle.

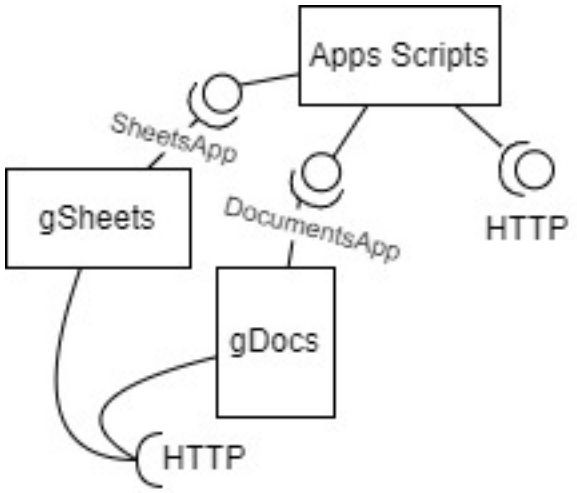
Konkrétně, vzhledem k tomu, že aktuálně není k dispozici způsob, jak změřit ušetřený čas při využívání báze, je vycházeno z předpokladu, že pokud budou vyžadovány lidské či finanční prostředky na to, aby byl obsah v gDocs udržován konzistentní s obsahem v efektivně prohledatelné databázi, nebude ušetřený čas větší, než ten vyžadovaný na údržbu.

3.1.2 Infrastruktura pro navrhovanou bázi

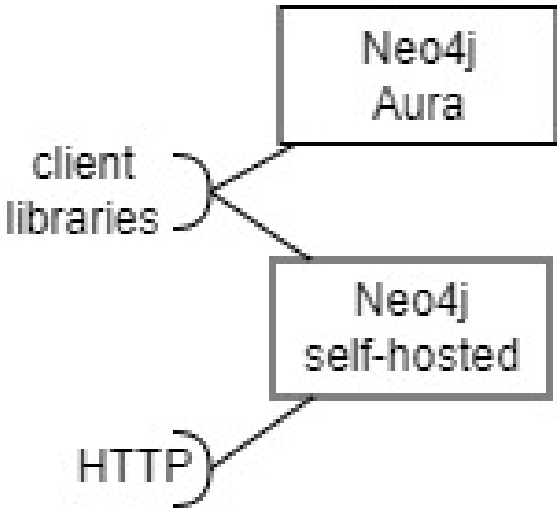
Vizualizace zjištění integrovatelnosti

Oba uvedené diagramy jsou založené opět na UML, i když i v tomto případě je učiněna drobná odchylka od standardu. Aby totiž uvedené diagramy odpovídaly normě, musely by obdélníky představující třídy, ještě mít menší čtverečky po stranách, kterými by byly reprezentovány konkrétní porty, které daná třída má a spojení s ostatními by pak měla být realizována pouze přes ony porty.
Jelikož se jedná jen o vizualizaci výsledků z předchozího kroku, nezahrnují diagramy žádnou informaci, která už by nebyla řečená. Nicméně pro vysvětlení použité notace, následuje stručný popis získané mapy možný integrací (propojení mezi softwary).

Jednodušší z obou diagramů zobrazuje dvě různé varianty databáze Neo4j, verze Aura je cloudová služba, která nabízí i celkem dostatečný objem možný k využití a bylo proto původním plánem vystačit při návrhu s ní, aby nebylo nutné nasazovat svoje vlastní řešení.
Bohužel, jak je patrné z vizualizace a jak bylo rovněž už i řečeno, cloudová verze databáze podporuje pouze komunikaci skrze knihovny, které si uživatel musí nejdříve nainstalovat (uložit spustitelný soubor do 'cesty' konkrétnímu programu). Což by ani nepředstavovalo problém, pokud tedy zrovna není potřeba s databází komunikovat z prostředí, kde si takové knihovny nainstalovat nelze, jako je například Apps Scripts. (31) Posuzovaná množina softwarů tak nechává pouze jedinou možnou cestu (HTTP), která může být využita k programovému zrcadlení údajů z dokumentů do databáze.



Obrázek 3.1: gWorkspace- mapa integrovatelnosti



Obrázek 3.2: Neo4j - mapa integrovatelnosti

Vyplývá ze zjištění

Aby bylo možné

1. zapisovat do báze alespoň tak snadno, jako je to možné nyní
2. v krátkém čase zobrazit (číst) pouze ty uložené záznamy, jenž skutečně vlastní hodnotu atributu zadaného v dotazu
3. provozovat navrženou bázi s nulovými náklady na infrastrukturu s minimálním časem potřebným na údržbu báze (konzistence dat mezi rozhraním pro zápis a rozhraním pro čtení),

níže vypsaná tvrzení musejí pro jednotlivé požadavky být pravdivá, přitom je vycházeno z předpokladu, že uživatelská rozhraní budou fungovat a uživatelé je budou umět používat

Klíčové prvky z hlediska realizovatelnosti návrhu splňujícího stanovené požadavky:

1. úroveň uživatelské zkušenosti při zapisování do báze - byl ponechán aktuálně používaný nástroj pro záznam informací v digitální podobě nebo bylo zvoleno nějaké, z hlediska nových uživatelů, přívětivější řešení
2. správnost a spolehlivost vyhledávání v uloženém obsahu - rozhraní pro čtení uloženého obsahu, dokáže odpovídat na otázky o svém obsahu bez chyb prvního i druhého typu
3. minimální lidské i finanční zdroje, vyžadované na provoz
    - obsah z gDocs je možné číst s pomocí AppsScripts
    - je možné s pomocí AppsScripts přistupovat pouze ke specifickým částem dokumentů na základě jejich absolutního i relativního umístnění ve struktuře dokumentu
    - je s pomocí AppsScripts možné komunikovat s Neo4j
    - free self-hosting

3.1.3 Ověření realizovatelnosti navržené infrastruktury

Demo proces automatizované údržby konzistence uložených dat mezi rozhraními pro čtení a zápis.
Je však důležité mít na paměti, že se jedná pouze o prostředek pro ověření předpokládané funkcionality, je tak opravdu jen v nejnutnějším rozsahu.

Struktury

Aby bylo možné vyzkoušet, že nástroje fungují, je nutné je testovat na datech. Protože konkrétní struktura pro navrženou bázi ještě nebyla vypracována, je pro ověřovací implementaci definována modelová struktura za prvé dokumentu (nadpisy, záhlaví, ...) a za druhé báze. Pro ověření využitá struktura dokumentů, vychází z předpokladu, že pro jednu aktivitu bude vyhrazen běžně jeden dokument. Ale počítá i s variantou, kdy by v jednom dokumentu bylo zahrnuto více než jedna aktivita, například pokud by se jednalo o lehce odlišné variace jedné hry. Zároveň je ale možné ukládat i dokumenty, které jsou vyhrazeny pro přípravu jedné konkrétní události, proto je potřeba umět mezi těmito dvěma druhy dokumentů rozlišovat. Aby toho byl script schopen, byl do dokumentové struktury nejprve zvolen prvek záhlaví, který obsahuje text 'Aktivita', pokud tomu obsah dokumentu odpovídá. Zároveň je tak provedena i příprava na situaci, kdyby by mělo být načítáno více různých typů dokumentů, je pak i přesto možno v kódu scriptu využívat tu stejnou funkci.

Kromě záhlaví byl testovací dokument tvořen dvěma nadpisy úrovně jedna a pod každým z nich, přesně do elementu s id o jedna větší, než id daného nadpisu, byla umístěna tabulka se dvěma sloupci jež měla sloužit pro jasné rozlišení parametrů reprezentovaných klíčem a přiřazenou hodnotou.

Proces

Jelikož už byly identifikovány konkrétní způsoby přesunu dat mezi zvolenými rozhraními, i konkrétní datové struktura, kterou bude přesouvána, zbývá tak pouze vyhodnotit už jen samotná realizovatelnost. Následující diagram reprezentuje implementovanou funkcionalitu. A bude budou-li ta fungovat, poskytne tím důkaz, že identifikovaná automatizovaná cesta mezi oběma zvolenými rozhraními je skutečně možná.

V diagramu jsou využity čtverce jako reprezentace aktivit, alespoň podle UML, jelikož se však jedná o reprezentaci, v prakticky javascriptu napsaného kódu, bude na čtverce, potažmo obdélníky dále odkazováno jako na funkce. Každá funkce přitom může mít nějaké vstupy a nějaké výstupy, může se přitom jednat buď pouze o konkrétní hodnotu uloženou do proměnné, nebo se může jednat o volání další funkce, případně i spolu s předáním konkrétních hodnot uložených v proměnných. Tyto hodnoty jsou v diagramu znázorněny pomocí pojmenovaných kosodélníků, a jejich čtení potažmo ukládání konkrétními funkcemi je diagramem znázorněn pomocí šipek s přerušovanou linkou.

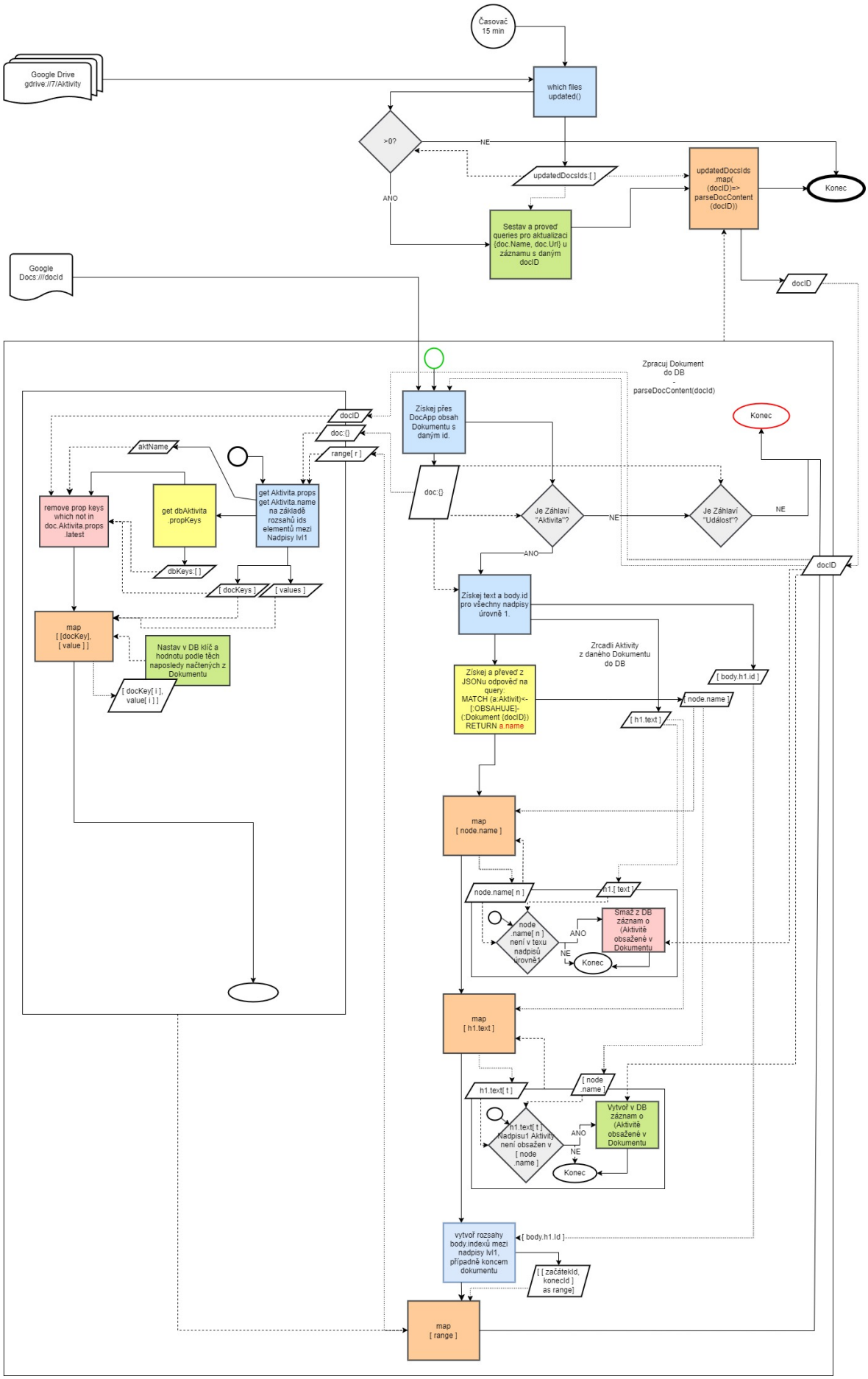
Šipky s nepřerušovanými linkami pak signalizují, že se jedná o hlavní sekvenci, která spojuje vstupní a výstupní hodnoty dané funkce. Pro značení začátku funkce je využito kolečka pro ukončení pak oválu a kosočtverce znázorňují prováděná rozhodnutí a větvení tak postupu v závislosti na situaci.

Barevné značení pak rozlišuje funkce podle toho, zda čtou obsah z dokumentů (modrá), čtou obsah z databáze (žlutá), nebo provádějí zápis do databáze (zelená), anebo mažou data z databáze (červená). Speciálním případem jsou pak čtverce oranžové, které reprezentují funkci map. Tato bere jako vstup rozsah hodnot a funkci, kterou následně pro každou hodnotu z daného rozsahu aplikuje.

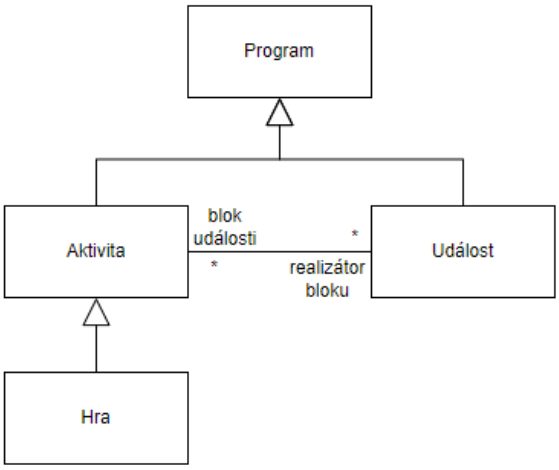
Čistě z hlediska funkcionality se jedná o časovačem spouštěný script, který vždy vybere od poslední kontroly aktualizované soubory z disku získá seznam jejich identifikátorů. Seznam je následně předám první funkci map (nejvrchnější oranžová), která prvky z daného seznamu zpracovává tak, že nejdříve na základě obdrženého identifikátoru dokumentu jsi od dokumentové služby vyžádá obsah odpovídající identifikátoru , pro ten vyhodnotí jakou hodnotu má zapsanou v záhlaví jakým způsobem má být tedy interpretován ta extrahován v něm zapsaný obsah.

Jelikož je implementace využita pouze jako prostředek k dokázání nutné funkcionality pro návrh, je realizován postup pouze pro 1 možnost, tou je konkrétně aktivita. Pokud je tedy v záhlaví dokumentu uvedeno 'Aktivita', funkce si přečte jejich obsah, přečte si obsah z databáze a upraví stav údajů v databázi, aby byl konzistentní se stavem v dokumentech. Jedná se však pouze o část, která umí pracovat pouze se strukturou určenou k ověření implementace dříve v textu.

Zdrojový kód víceméně odpovídající uvedenému vývojovému diagramu je možné si prohlédnout zde - důkaz možného automatického zrcadlení obsahu gDocs do databáze



Obrázek 3.3: Vývojový diagram znázorňující demo proces synchronizace obsahu z dokumentů Google do databáze Neo4j



Obrázek 3.4: Zvolené 'základní' pojmy představující základní definici modelované domény

3.2 Model pojmů

3.2.1 Základní definice

Program

Jedná se o nejobecnější z pojmů. Prakticky je programem cokoliv, co připravujeme v rámci činnosti oddílu. Kdyby Činnost oddílu byla mezi modelovanými pojmy, nacházela by se ta, na vršku generalizační struktury. Jejími dalšími specializacemi, kromě programu, by pak ještě pravděpodobně byl například pojem Administrativa.

Událost

Instancemi této třídy jsou například konkrétní schůzky a výpravy, pořádané během roku, ale i tábory a další akce přes léto. To implikuje, že každá z instancí této třídy má stanovený nějaký specifický termín svého pořádání. Ve skutečnosti to typicky bývá konkrétní čas začátku a alespoň přibližný čas návratu (ukončení akce). Pojem "Akce"je zde použit jako rovnocenný k Události, nicméně je možné, že Událost je specializací Akce, jakožto obecnějšího pojmu. V každém případě by mělo být možné označovat Událost za Akci. Klíčovou charakteristikou Událostí je, že průběh každé z nich, i "jen"dvouhodinové schůzky, se skládá z jednotlivých Programů. Tedy alespoň se to tak říká hovorovým označením. Jelikož byl Program určen jako nejobecnější pojem v modelované doméně, samo toto tvrzení o složení (částech) Události přílišnou výpovědní hodnotu sice nemá, avšak je tím implikováno, že Události by měly být asociovány s nějakou třídou, která je zahrnuta mezi specializacemi pojmu Program.

Aktivita

Třída Aktivita - reprezentuje entity, které samy, na rozdíl od událostí, nemají specifikován žádný konkrétní termín jejich pořádání. Instance třídy Aktivita představují totiž soubory znalostí určené ke znovuvyužívání, postupnému obohacování a zkvalitňování pořádaných Událostí. Jestli je možné pomocí zdokonalení známých aktivit dosáhnout kvalitnějších událostí, naznačuje to že by mezi Aktivitou a Událostí měl být modelovaný vztah. Tento vztah pak skutečně existuje. Pro pojmenování rolí v tomto vztahu bylo využito pojmu, používaného primárně v rámci pořádání táborů, kdy každý den je rozdělen na jednotlivé "bloky", neboli bloky programu, které představují hlavní stavební prvky táborového dne. Jelikož pak tábor představuje Událost a jednotlivé takzvané "bloky", jsou prakticky jen instancí třídy Aktivita na jejichž přípravu bylo vynaloženo typicky více úsilí než na běžné programy, je možné využít označené role Aktivity "programový blok"a role Události pak 'realizátor bloku'. S tím, že obě strany vztahu mohou být zastoupeny žádným, ale i několika instancemi z koncových tříd.

Hra

Třída Hra-je speciálním případem aktivity, to znamená, že Události mohou jako bloky využívat i instance her, protože jsou zároveň také Aktivitami. Jak se hra liší od aktivity, či jaké může mít hra další vztahy, je popsáno v následujících částech práce. Není zde tedy rozváděna do větších podrobnosti.

3.2.2 Rozšíření základních tříd

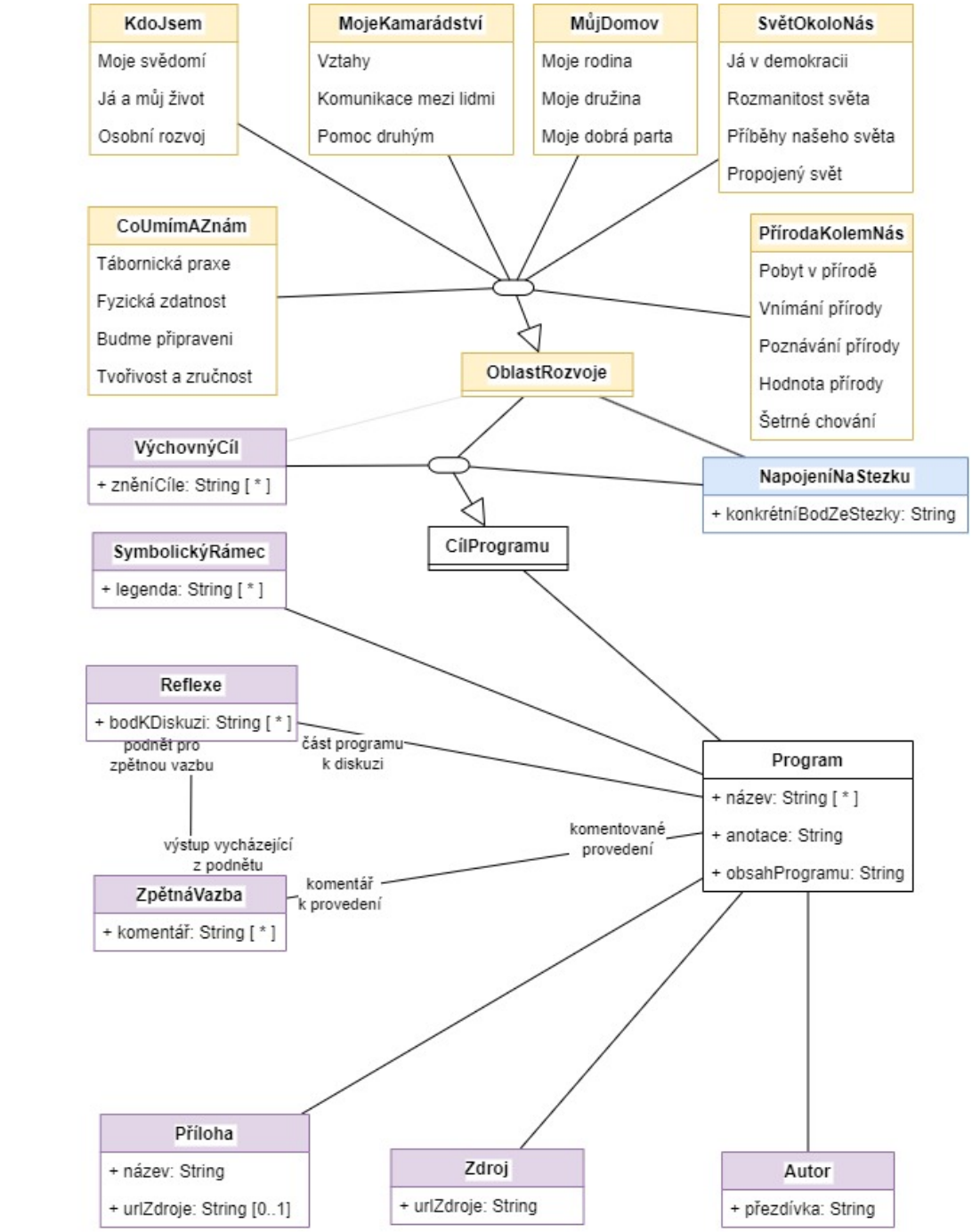
Program

Ke třídě Program byly pojmy vybrány, ve všech případech, kromě zpětné vazby, z báze 'chystamprogram'.

Nalezení první skupinky podobných pojmů nebylo obtížné, jelikož 'chystamprogram' dělí aktivity podle 'OblastRozvoje' a dále podle podoblastí každé z oblastí. V modelu je proto každá z oblastí modelována jako samostatná třída obsahující její přípustné hodnoty a připojená ke třídě 'OblastRozvoje' jako její specializace. Tato notace vychází z takzvaných číselníků neboli enumerates v UML, pomocí kterých jsou reprezentovány atributy, které mohou nabývat pouze hodnot z předdefinované sady.

Následně byl identifikován i 'VýchovnýCíl' a 'NapojeníNaStezku' jako specializace ze stejné sady jako 'OblastRozvoje'. Bylo tak učiněno proto, že všechny ze tříd představují z hlediska programu jeho zaměření, něco, čeho se autor programu snažil jeho prostřednictvím dosáhnout. Byla proto vymodelována třída 'CílProgramu', která tato různá zaměření sdružuje.

Alternativně by se daly vztahy znázornit ke každé ze specializací zvlášť, ale použitá varianta



Obrázek 3.5: Dílčí pohled - Program

je vhodnější pro případy, kdy je šance, že specializací v této sadě může potenciálně být i více než je v aktuálním modelu zachyceno. Program může být například zacílen i na dosažení určitého milníku v dlouhodobé hře nebo na specifickou dovednost, kterou by děti při jeho pořádání měly zapojit a tím si ji tak procvičit. Třída reprezentující tyto pojmy by pak mohla být pojmenována jako OddílovéCíle a sdružovala by rozličné specifičtější zaměření, která oddíl společně vyhodnotí jako relevantní, ale nejsou konkrétně adresovány oficiální literaturou.

Mezi zbývajícími třídami se dále nepodařila identifikovat žádná další skupinka tříd s podobnými názvy, u kterých by dávalo smysl je generalizovat. Sice si blízké, ale nezaměnitelné jsou dvojice tříd zdroj-Příloha, reflexe-Zpětná vazba, v obou případech jsou však obě třídy, k té základní, asociovány z jiných důvodů.

Zdroj značí materiály využité při tvorbě daného programu a přílohy jsou zamýšlené na využití při pořádání. Reflexe znázorňuje představu autora o tom, které části programu by po provedení měly být diskutovány a tvořit tak podněty pro sbíranou zpětnou vazbu. Asociativní vztah je proto modelován jak mezi oběma těmito třídami, tak i pak od každé zvlášť ke třídě Program. Symbolický rámec pak představuje témata a tématické celky, jenž dětem ozvláštňují běžné úkony a usnadňují si zapamatovat lépe informace, které se během programu dozvěděly. (32)

Jako jediné třídy, které jsou asociované, většinou instancí třídy Program nakonec byly vyhodnoceny pouze Název, Anotace (Popis) a Obsah hry, jenž tvoří dále nespecifikovaný prostor, do kterého mají být zapsány všechny jinde nezařazené informace potřebné či hodící se při provedení daného programu.

Aktivita

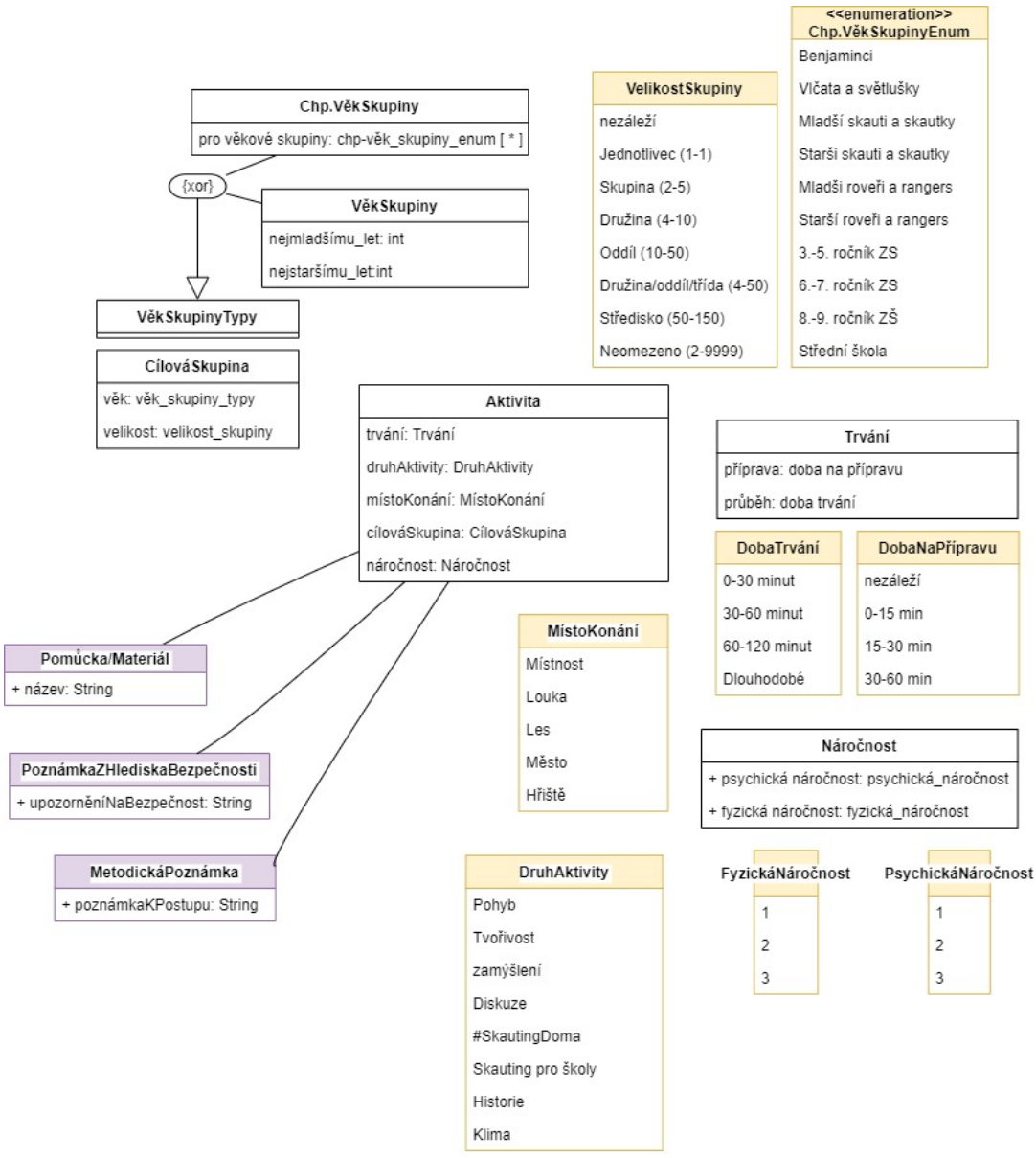
K základní třídě aktivita byly, kromě zbývajících pojmů z báze 'chystamprogram', přidány ještě pojmy 'VěkNejmladšího' a 'VěkNejstaršího', které se původně nacházely v encyklopedii.

Kromě těchto dvou zmiňovala encyklopedie také pomůcky a materiál, velikost skupiny, místo konání, věk skupiny a dobu trvání.

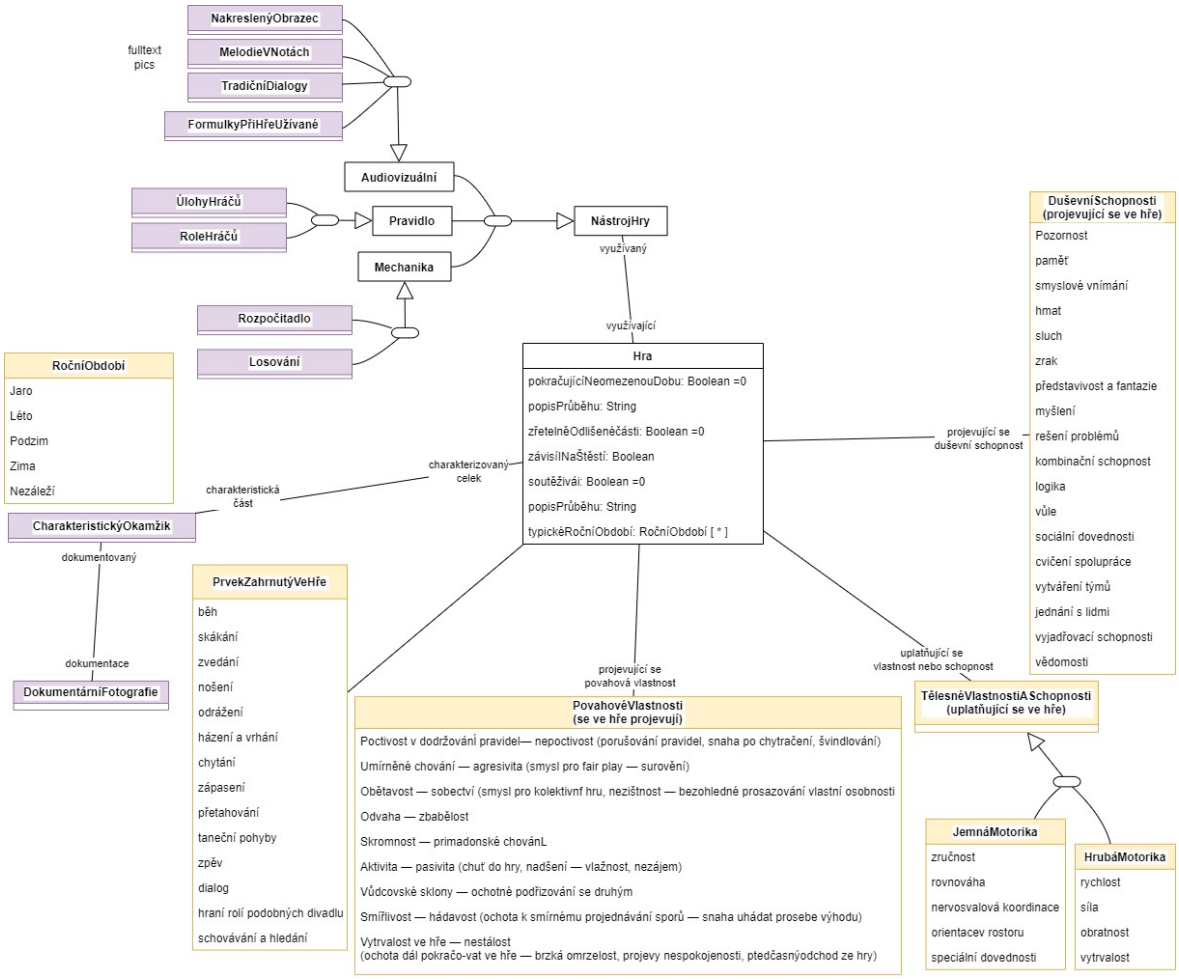
Generalizovaná třída byla v tomto případě vytvořena pouze jedna. Je využita ke sjednocení dvou možných variant zaznamenání věku skupiny (buď pomocí definování spodní a vrchní hranice pro věk hráče nebo pomocí opět číselníků s předdefinovanými hodnotami, pro jednotlivé věkové kategorie).

Spolu s generalizovanou třídou věku skupiny je k aktivitám přiřazena ještě třída velikost skupiny, která je rovněž reprezentována výčtem přípustných hodnot.

Jelikož obě třídy představují charakteristiku skupiny, pro níž je program zamýšlen, jsou obě třídy sloučeny pomocí nové třídy s názvem CílováSkupina. A opět, stejně jako v případě věku skupiny, je zde využito kompozice namísto generalizace, jelikož věk skupiny i velikost mají pro instance třídy Aktivita podstatně odlišné význam.



Obrázek 3.6: Dílčí pohled - Aktivita



Obrázek 3.7: Dílčí pohled - Hra

Obdobně bylo provedeno pro třídy 'DobaTrvání' a 'DobaNaPřípravu' stejně jako psychická a fyzická náročnost.

Asociačním vztahem byly připojeny pouze třídy poznámek. Jelikož oproti například třídám Trvání nebo Náročnost, jsou poznámky uvedeny pouze u menší částí z nich.

Ostatní třídy mají jako datový typ (přípustné hodnoty) buď samotné číslo nebo sadu přípustných hodnot, takže je možné jich relativně bezúsilné zaznamenávání, na rozdíl od třeba metodických poznámek. Navíc tyto numerické a kategorické proměnné představují jeden z nejrelevantnějších atributů, při přípravě nových programů a hledání inspirace, podle kterých je možné si snadno a celkem přesně zúžit prohledávané varianty. V důsledku těchto faktorů bývají tyto atributy uvedeny u většiny zaznamenaných aktivit, proto byly ke třídě 'Aktivita' připojeny jako jí vlastní atributy.

Hra

Mezi zbylými pojmy z encyklopedie her jsou identifikovány 3 hlavní skupiny podobných termínů.

První skupinou jsou jednoduché pojmy prezentovatelné binární hodnotou (ano, ne), které zároveň přímo charakterizují přiřazené instance her. Tyto pojmy byly proto modelovány jako atributy třídy 'Hra'.

Druhou skupinou představovaly pojmy, jenž jsou instancemi her využívány, proto byly sjednoceny generalizovanou třídou 'NástrojHry'. Nicméně i tak se mezi všemi specializovanými nástroji her stále nacházely tři podstatně odlišné celky, těm byla proto přidána další úroveň generalizace, která je tvořena třídami 'AudiovizuálníNástrojHry', 'PravidloHry' a 'MechanikaHry'.

Pravidla přitom představují omezení, jenž jsou hráčům stanovena a v jejichž vymezení by se měli pohybovat. Zatímco mechaniky jsou prostředky, které jsou hráčům poskytnuty, aby je v rámci hry mohli využívat.

Poslední skupinou jsou pak vlastnosti a schopnosti, které se při hře projevují nebo uplatňují, v této skupině je však obtížné najít smysluplnou generalizaci. Na první pohled se totiž všechny jeví jako lidské rysy (PovahovéVlastnosti, DuševníSchopnosti, TělesnéVlastnostiASchopnosti, bylo by proto teoreticky možné sloučit je pomocí atributů v nově vytvořené třídě. Jedná se však jen o tři podstatně odlišné celky a není pravděpodobné přidání dalšího podobného výčtu, jelikož už tento je poměrně dost obsáhlý a na první pohled dost kompletní, co se zahrnutých hodnot týče. Bylo proto pro zachování sémanticky přesného modelu rozhodnuto, připojit každou z těchto tříd separátně.

A protože většina her by měla nějakým způsobem rozvíjet alespoň nějaké z těchto vlastností a schopností, jsou tyto připojeny opět jako atributy vlastněné třídou 'Hra'.

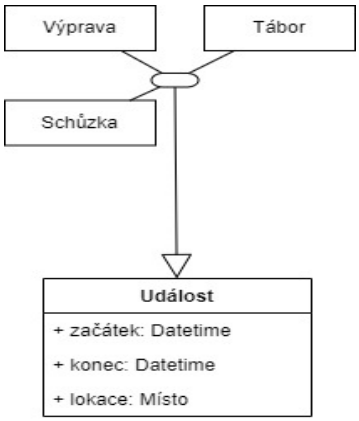
Třída 'DokumentárníFotografie' je v původní literatuře zmiňována v souvislosti s pojmem 'CharakteristickýOkamžik', proto i v tomto modelu je asociována stejně.

Co se třídy 'PrvekZahrnutýVeHře' týče, její přípustné hodnoty se sice zdají být velmi významově blízko třídám, které byly generalizovány jako mechaniky, avšak třída hrou zahrnutých prvků byla ponechána odděleně. Rozhodnutí bylo učiněno proto, že pro její přípustné hodnoty, které jsou, dalo by se říct primitivní, ve srovnání s mechanikou (jako rozpočítadlo, které může mít i relativně komplexní význam, alespoň vzhledem k pojmům jako běh, zpěv nebo dialog), se zdá použití výrazu 'Prvek' býti výstižnějším.

Událost

Pojem Událostí je v důsledku nesjednoceného způsobu pro jejich zápis nejobtížnější na přínosné vyhodnocení, proto u něj byly znázorněny pouze ty nejvýraznější charakteristiky.

Těmi jsou atributy stanoveného začátku a konce průběhu, což je tak podstatně odlišuje od aktivit, které jsou ze své podstaty spíše na konkrétním čase nezávislé. Nakonec je pak znázorněno základní rozdělení událostí, to přitom dalo by se říci, závisí na době trvání dané události. s tím že schůzky bývají tak na dvě hodiny, výpravy na jeden či dva dny a tábor



Obrázek 3.8: Dílčí pohled - Událost

přibližně dva týdny.

3.2.3 Porovnání s nesystematicky nakresleným modelem

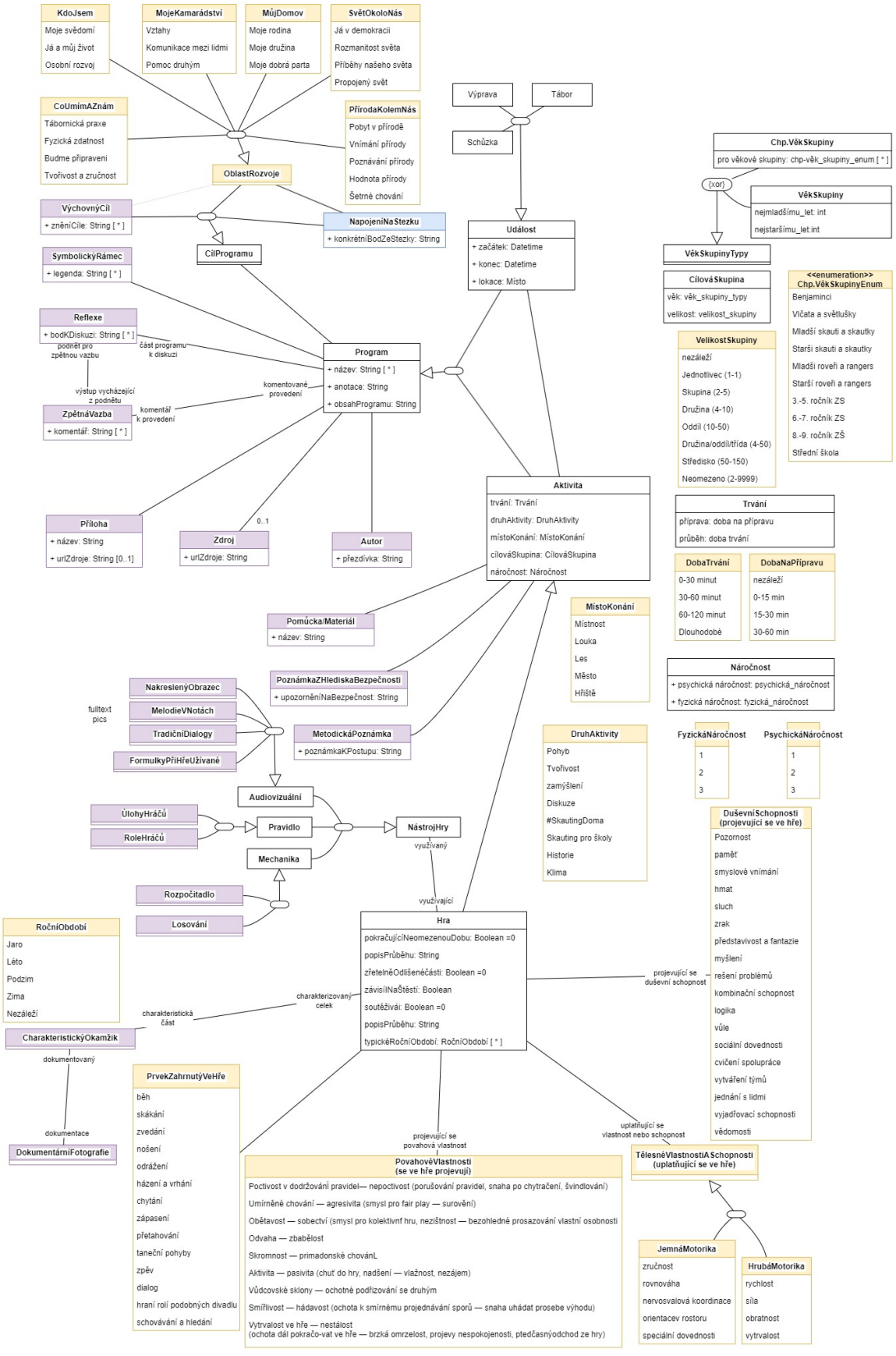
Podle prvního dojmu působí nepochybně systematicky vytvořený model obsáhleji než ten tvořený nesystematicky primárně na základě vlastních doménových znalostí. Avšak i ten jen podle znalostí vytvářený má pár předností. První z nich představuje symbolický rámec, který je více rozvedený v případě nesystematicky vytvořeného modelu, zatímco existující báze mezi žádnými drobnějšími částmi symbolického rámce nerozlišují.

Zároveň také rozlišuje i mezi víc různými druhy cílů, ke kterým navíc umožňuje získávat měření podle stanovených metrik. Zatímco například báze "chystamprogram"se na cíle dívá převážně jen jako na oblasti a úkoly ze stezky.

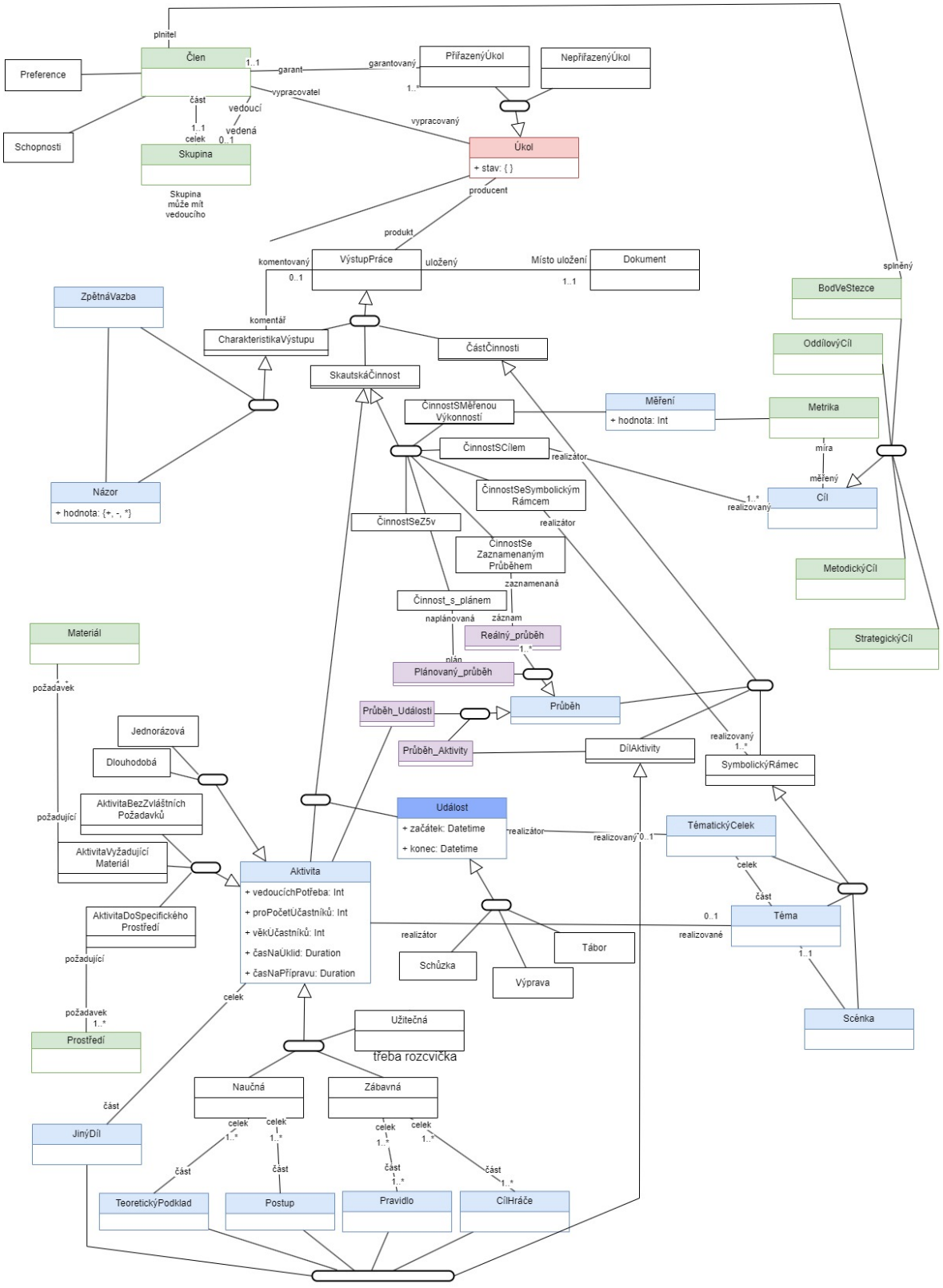
Nesystematický model se obecně na problematiku dívá z širší perspektivy a jeho cílem bylo spíše zmapovat různé odlišné oblasti, které na připravované programy nějakým způsobem navazují.

Oproti systematicky tvořenému modelu tento nabízí navíc zejména pohled na situaci z úhlu řízení práce. Z tohoto úhlu jsou připravované programy viděny jako úkoly, které potřebují dokončit a všechno připravené či dokončené je viděno jako výstup naší práce, u kterého na základě sbírané zpětné vazby je možné monitorovat dosahovanou kvalitu právě díky měřeným cílům.

Širší pohled na situaci nabízí nesystematický model jen díky tomu, že během jeho tvorby jsem chtěl zaměřovat svoji práci lehce odlišným způsobem, než jsem ji ve finále zaměřil. Pokud tedy bude řešena pouze problematika skautských programů a nikoliv řízení úkolů přitom vznikajících, nesystematický model by pro navrhovanou bázi mohl být také použitelný, kdyby bylo potřeba. Nicméně vzhledem k rozsahu obsahu, který je nabízen systematicky sestaveným modelem, neposkytuje ten nesystematicky vytvořený jiné podstatné výhody, než podrobněji zmapovaný symbolický rámec a možné cíle na které může program být zaměřen.



Obrázek 3.9: Diagram tříd sestavený pomocí popsané metodiky

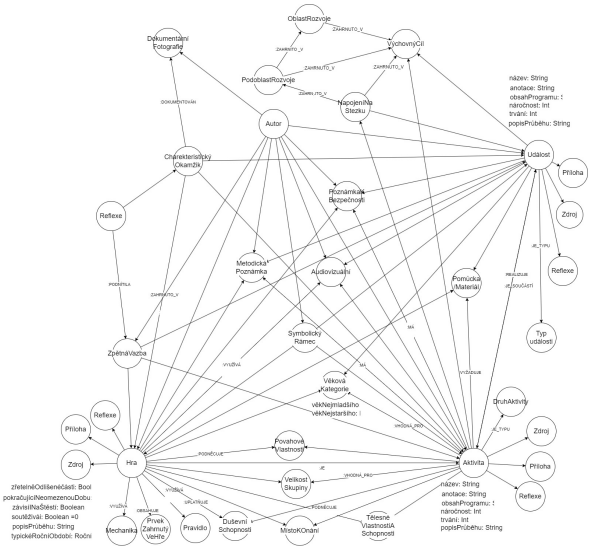


Obrázek 3.10: Diagram tříd sestavovaný nesystematicky, primárně podle doménových znalostí

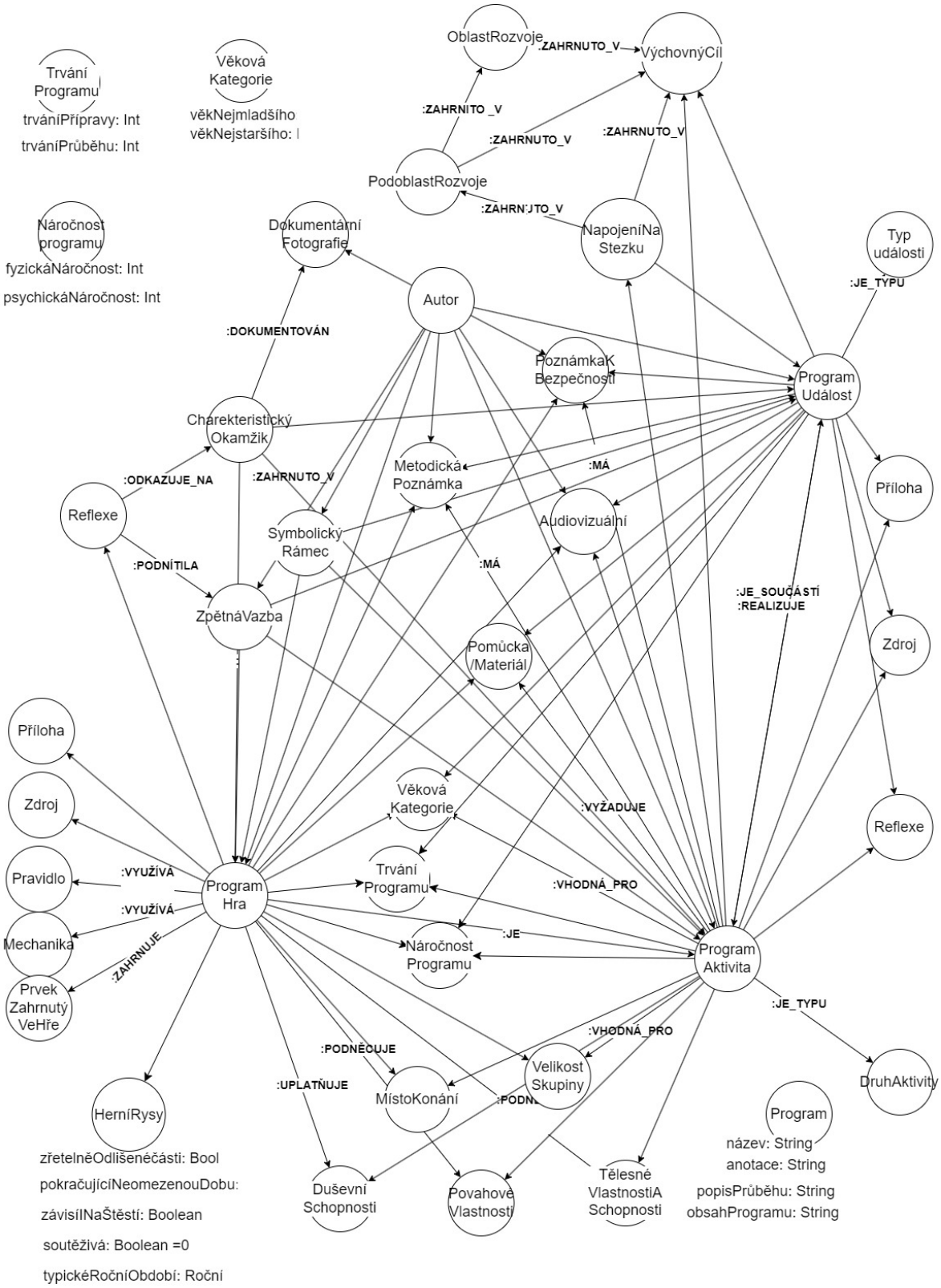
3.3 Schema databáze

Obrázek 3.11: Schema pro grafovou databázi - 1.pokus

Názvy v kulatých objektech reprezentují vždy název použitý pro pojmenování štítku (v rámci uložené struktury Neo4j).



Obrázek 3.12: Schema pro grafovou databázi - 2.pokus



Obrázek 3.13: Schema pro grafovou databázi - 4.pokus

3.3.1 Cypher dotazy

chystamprogram

1. místo konání
   Které 'Aktivity' jsou vhodné na {$MístoKonání$}?
   $MATCH(a : Aktivita) - [: NA\_MÍSTO] - > (m : MístoKonání)$
   $RETURN*$
2. doba trvání
   Které 'Aktivity' trvají {$DobaTrvání$}?
   $MATCH(b : Aktivita) - -(t : TrváníProgramu)WHERE t.průběh = x$
   $RETURN*$
3. napojení na stezku (stupeň, úkol)
   Které 'Aktivity' mají {$NapojeníNaStezku$}?
   $MATCH(a : Aktivita) - -(n : NapojeníNaStezku)$
   $RETURN*$

encyklopedie her

1. pomůcky vyžadovány
   Které 'Hry' vyžadují {$Pomůcka/Materiál$}?
   $MATCH(h : Hra) - [: VYŽADUJE] - > (p : Pomucka/Materiál)$
   $RETURN*$
2. věkové kategorie Pro kterou je hra vhodná
   Které hry jsou vhodné pro {$VěkováKatgorie$}?
   $MATCH(h : Hra) - - > (k : VěkováKategorie)$
   $RETURN$
3. soutěživá hra
   Které hry jsou soutěživé?
   $MATCH(h : Hra)WHERE h.soutěživá = true$
   $RETURN h$

Přidáno

1. reflexe
   Které 'Aktivity' mají {$Reflexe$}, která podnítila {$ZpětnáVazba$}?
   $MATCH(a : Aktivita) - - > (r : Reflexe) - [: PODNÍTILA] - > (z : ZpětnáVazba)$
   $RETURN a, r, z$

Díky nepřílišně vysokým kompetencím ostatních bází, byl tento cíl dosažen poměrně snadno.

4. Diskuze

Snahou této práce bylo navržení systému pro uložení sdílených informací. Bylo tedy potřeba identifikovat odpovědi Na tři otázky:

- Kam informace ukládat?
- Co za informace ukládat?
- Jak vybrané informace ukládat?

Proto bylo nejprve potřeba rozhodnout, který software bude zvolen jako "základní kámen"pro tvořený systémem. Vzhledem k tomu, že centralizovaná úložiště nejsou něco, co by bylo měněno častěji nežli je opravdu nutné, je k tomuto kroku přikládána patřičná důležitost. Proto byl nejprve proveden multikriteriální výběr a hodnocení z několika úhlů pohledu na analyzované softwarové nástroje.

Tomuto hodnocení a analýze však nepředcházel žádný komplexnější či širší výběr na úvod, ale byli předvybrány pouze čtyry softwarové nástroje, každý reprezentující odlišnou datovou strukturu než ty ostatní. Byl tak přijat kompromis pro zjednodušení výběrového procesu, který avšak alespoň bral v potaz různé struktury a nespokojil se pouze s nejběžnější variantou.

Správnost návrhu byla v případě prvního dílčího cíle úspěšně ověřena pomocí experimentální implementace, která dovedla pracovat pouze s nadpisy první úrovně a tabulkami umístěnými hned na řádek pod něj. Nezní to sice jako příliš zásadní úspěch, nicméně pro dokázání dostatečné funkcionality by měl postačovat. Dokazuje totiž, že Apps Scripts dovedou přistupovat k dokumenty obsaženým elementům podle jejich jak absolutního umístnění ve struktuře dokumentu, stejně tak jako procházet obsah pomocí relativních odkazů jednotlivými elementy na jejich sousedy.

Pro zmapování cílové domény a zodpovězení tak druhé otázky, byly prohledány existující báze, které se jeví jako vhodný základní zdroj. Nicméně při porovnávání s nesystematicky vytvořeným modelem vyšlo najevo, že pro samotné programy může být relevantních značně více různých asociací, než standardně existující báze umožňují.

Pro následnou tvorbu schematu pro databázi, byl jako základ využit konceptuální model, což každopádně není špatně řešení. Avšak během identifikace nejlepších praktik pro modelování grafových struktur v Neo4j, že doporučovanějším postupem je využití 'user stories' takovým způsobem, že je z nich vycházeno po celou dobu návrhu, již od úplného začátku. Tento postup každopádně dává smysl, nicméně bude též pravděpodobné komplexnější a náročnější na provedení, než touto prací provedený postup.

Nejzásadnějším omezením této práce je rozložení úsilí mezi několik celků, ty však byly v této fázi vývoje systému všechny klíčovými částmi, umožňujícími dosáhnout výsledků jenž byly dosaženy. Pro budoucí vývoj by však bylo vhodné se soustředit detailněji na jednu oblast. A pravděpodobně také začlenění konkrétních způsobů využití tvořeného systému, již od počátku

navrhováni.

Závěr

Tato práce se zaměřuje na návrh znalostní báze pro skautské programy, která by umožňovala efektivní a příjemné vyhledávání informací s minimální bariérou pro nové uživatele. Cílem je zachovat kvalitu uživatelské zkušenosti při zapisování a zároveň nevyžadovat více zdrojů na provoz a údržbu než je nutné. Práce zahrnuje výběr softwarového řešení, analýzu literatury, konceptuální modelování a návrh databázového schématu. Důraz je kladen na multikriteriální výběr softwaru a na modelování dat podle nejlepších praktik. Výsledky práce ukazují, že bylo možné navrhnout funkční systém, který splňuje stanovené cíle a umožňuje efektivní správu a sdílení informací.

Seznam použité literatury

1. BERNDTSSON, Mikael; HANSSON, Jörgen; OLSSON, Björn; LUNDELL, Björn. *Thesis Projects: A Guide for Students in Computer Science and Information Systems*. London: Springer, 2008. ISBN 978-1-84800-008-7.
2. Unified Modeling Language, v2.5.1. *Unified Modeling Language*. [N.d.].
3. ROBINSON, Ian; WEBBER, James; EIFREM, Emil. *Graph Databases*. Second edition. Beijing: O'Reilly, 2015. ISBN 978-1-4919-3089-2.
4. ALLEN, David. *Graph Data Modeling: Categorical Variables* [online]. Neo4j Developer Blog, 2020-10-23. [cit. 2024-05-06]. Dostupné z: `https://medium.com/neo4j/graph-data-modeling-categorical-variables-dd8a2845d5e0`.
5. *Chystam Program.O Projektu* [online]. [cit. 2024-04-17]. Dostupné z: `https://chystamprogram.skaut.cz/o-projektu`.
6. ZAPLETAL, Miloš. *Zapletal - Velká Encyklopedie Her (1. Svazek - Hry v Přírodě) (1987) [Vojak_svejk].Pdf*. [B.r.].
7. ZAPLETAL, Miloš. *Zapletal - Velká Encyklopedie Her (2. Svazek - Hry v Klubovně) (1996) [Vojak_svejk].Pdf*. [B.r.].
8. ZAPLETAL, Miloš. *Zapletal - Velká Encyklopedie Her (3. Svazek - Hry Na Hřišti a v Tělocvičně) (1987) [Vojak_svejk].Pdf*. [B.r.].
9. ZAPLETAL, Miloš. *Zapletal - Velká Encyklopedie Her (4. Svazek - Hry ve Městě a Na vsi) (1988) [Vojak_svejk].Pdf*. [B.r.].
10. ABDELHEDI, Fatma; AIT BRAHIM, Amal; ATIGUI, Faten; ZURFLUH, Gilles. MDA-Based Approach for NoSQL Databases Modelling. In: BELLATRECHE, Ladjel; CHAKRAVARTHY, Sharma (eds.). *Big Data Analytics and Knowledge Discovery* [online]. Cham: Springer International Publishing, 2017, vol. 10440, s. 88–102 [cit. 2024-04-10]. ISBN 978-3-319-64282-6 978-3-319-64283-3. Dostupné z DOI: `10.1007/978-3-319-64283-3_7`.
11. *Neo4j Browser - Neo4j Browser* [online]. [cit. 2024-04-23]. Dostupné z: `https://neo4j.com/docs/browser-manual/current/`.
12. ALLEN, David. *Graph Modeling: All About Super Nodes* [online]. Neo4j Developer Blog, 2020-10-23. [cit. 2024-05-06]. Dostupné z: `https://medium.com/neo4j/graph-modeling-all-about-super-nodes-d6ad7e11015b`.
13. ALLEN, David. *Graph Modeling: Labels* [online]. Neo4j Developer Blog, 2020-10-23. [cit. 2024-05-06]. Dostupné z: `https://medium.com/neo4j/graph-modeling-labels-71775ff7d121`.
14. NOY, Natalya; MCGUINNESS, Deborah. Ontology Development 101: A Guide to Creating Your First Ontology. In: 1992.
15. MCHUGH, Jim. *Use Graph Databases For Complex Hierarchies* [online]. Medium, 2022-05-31. [cit. 2024-05-06]. Dostupné z: `https://medium.com/@jim.mchugh/use-graph-databases-for-complex-hierarchies-c5b7cfb41623`.

16. CONTRIBUTORS, phpMyAdmin. *phpMyAdmin* [online]. phpMyAdmin. [cit. 2024-05-06]. Dostupné z: https://www.phpmyadmin.net/.

17. *Google Docs API Overview* [online]. Google for Developers. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/docs/api/how-tos/overview.

18. *Add Custom File Properties | Google Drive* [online]. Google for Developers. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/drive/api/guides/properties.

19. *Structure of a Google Docs Document* [online]. Google for Developers. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/docs/api/concepts/structure.

20. *Class NamedRange | Apps Script* [online]. Google for Developers. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/apps-script/reference/spreadsheet/named-range.

21. SEHGAL, Siddhartha. *Neo4j Data Modelling 101* [online]. Neo4j Developer Blog, 2020-08-14. [cit. 2024-05-06]. Dostupné z: https://medium.com/neo4j/neo4j-data-modelling-2982bd90aa0c.

22. *Full-Text Indexes - Cypher Manual* [online]. Neo4j Graph Data Platform. [cit. 2024-05-06]. Dostupné z: https://neo4j.com/docs/cypher-manual/5/indexes/semantic-indexes/full-text-indexes/.

23. *Vector Search Indexes - Cypher Manual* [online]. Neo4j Graph Data Platform. [cit. 2024-05-06]. Dostupné z: https://neo4j.com/docs/cypher-manual/5/indexes/semantic-indexes/vector-indexes/.

24. *Create Access Credentials | Google Workspace | Google for Developers* [online]. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/workspace/guides/create-credentials.

25. *Advanced Google Services | Apps Script* [online]. Google for Developers. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/apps-script/guides/services/advanced.

26. *Built-in Google Services | Apps Script | Google for Developers* [online]. [cit. 2024-05-06]. Dostupné z: https://developers.google.com/apps-script/guides/services.

27. *Load External JavaScript Libraries in Google Scripts with Eval()* [online]. Digital Inspiration. [cit. 2024-05-06]. Dostupné z: https://www.labnol.org/code/20380-load-external-javascript-with-eval.

28. *Stezky a Cestičky Vlčat, Světlušek a Žabiček – Skautská Křižovatka* [online]. [cit. 2024-04-17]. Dostupné z: https://krizovatka.skaut.cz/svetlusky-zabicky-vlcata/stezky-a-cesticky-vlcat-svetlusek-a-zabicek.

29. *Změny ve skautských Google službách - nově max. 40 GB na uživatele* [online]. Skautské zpravodajství, 2022-05-02. [cit. 2024-05-06]. Dostupné z: https://zpravodajstvi.skaut.cz/clanek/zmeny-ve-skautskych-google-sluzbach-nove-max-40-gb-na-uzivatele.

30. *Graph Modeling Guidelines - Getting Started* [online]. Neo4j Graph Data Platform. [cit. 2024-05-06]. Dostupné z: `https://neo4j.com/docs/getting-started/data-modeling/guide-data-modeling/`.

31. KOSTYUK, Dmitry. *Mastering NPM Modules in Google Apps Script: Your Ultimate Roadmap* [online]. Geek Culture, 2023-05-18. [cit. 2024-05-06]. Dostupné z: `https://medium.com/geekculture/the-ultimate-guide-to-npm-modules-in-google-apps-script-a84545c3f57c`.

32. HOŘAVOVÁ, Kateřina; KLÁPŠTĚ, Petr; *Symbolický Rámec Podle 3.B (Metodika)*. Junák - svaz skautů a skautek ČR, [b.r.].