

Vizualizace výsledků výzkumu a vývoje

Bc. Jan Rohan

Diplomová práce
2014



Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

ZADÁNÍ DIPLOMOVÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení: **Bc. Jan Rohan**
Osobní číslo: **A12647**
Studijní program: **N3902 Inženýrská informatika**
Studijní obor: **Informační technologie**
Forma studia: **kombinovaná**

Téma práce: **Vizualizace výsledků výzkumu a vývoje**

Zásady pro vypracování:

1. Analyzujte potřeby a požadavky na informační systém pro vizualizaci výsledků výzkumu a vývoje.
2. Seznamte se strukturou datových souborů obsahující data o výzkumných výsledcích.
3. Provedte průzkum vhodných vizualizačních technologií a vybranou technologii zvolte pro implementaci.
4. Věnujte pozornost způsobu zabezpečení aplikace.
5. Vytvořte funkční prototyp aplikace.
6. Navrhněte další možný rozvoj aplikace.

Rozsah diplomové práce:

Rozsah příloh:

Forma zpracování diplomové práce: **tištěná/elektronická**

Seznam odborné literatury:

1. **ARLOW, Jim a Ila NEUSTADT. UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky. Vyd. 1. Překlad Bogdan Kiszka. Brno: Computer Press, 2007, 567 s. ISBN 978-80-251-1503-9.**
2. **ZAKHOUR, Sharon. Java 6: výukový kurz. Vyd. 1. Brno: Computer Press, 2007, 534 s. ISBN 978-80-251-1575-6.**
3. **PECINOVSKÝ, Rudolf. Návrhové vzory. Vyd. 1. Brno: Computer Press, 2007, 527 s. ISBN 978-80-251-1582-4.**
4. **GRAILS.ORG. Documentation [online]. 2014, 2014-01-19T19:30:05.000-08:00 by jonghwa [cit. 2014-02-04]. Dostupné z: <http://grails.org/Documentation>**
5. **W3schools.com: the worlds largest web development site [online]. 1999 [cit. 2014-02-04]. Dostupné z: <http://www.w3schools.com/>.**
6. **AUGUSTÝN, Michal. JavaScript očima programátora v2. In: Augiho web [online]. 29. 3. 2010. 2010 [cit. 2014-02-04]. Dostupné z: <http://www.augi.cz/programovani/javascript-ocima-programatora-v2/>.**
7. **WRÓBLEWSKI, Piotr. Algoritmy: datové struktury a programovací techniky. Vyd. 1. Překlad Marek Michalek, Bogdan Kiszka. Brno: Computer Press, 2004, 351 s. ISBN 80-251-0343-9.**

Vedoucí diplomové práce:

Ing. Radek Šilhavý, Ph.D.

Ústav počítačových a komunikačních systémů

Datum zadání diplomové práce:

21. února 2014

Termín odevzdání diplomové práce:

20. května 2014

Ve Zlíně dne 21. února 2014

prof. Ing. Vladimír Vašek, CSc.

děkan



doc. Mgr. Roman Jašek, Ph.D.

ředitel ústavu

Prohlašuji, že

- beru na vědomí, že odevzdáním diplomové/bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že diplomová/bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji diplomovou/bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen s předchozím písemným souhlasem Univerzity Tomáše Bati ve Zlíně, která je oprávněna v takovém případě ode mne požadovat přiměřený příspěvek na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše);
- beru na vědomí, že pokud bylo k vypracování diplomové/bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem diplomové/bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na diplomové práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor.
- že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

Ve Zlíně

.....
podpis diplomanta

ABSTRAKT

Cílem této diplomové práce je navrhnout a vytvořit funkční prototyp webové aplikace, která umožní snadno nahlédnout na data z informačního systému výzkumu, experimentálního vývoje a inovací, který poskytuje strohá data o výzkumných projektech hrazených z veřejných prostředků. Cílem není pouze tato data zobrazit, ale především se jedná o možnost snadno z těchto dat vytvářet statistiky a grafy. Výsledná aplikace tedy uživateli umožní snadno vytvářet nad daty různé vizualizace, ve kterých může jednoduše hledat souvislosti a vyvozovat z nich patřičné závěry. Aplikace bude implementována v programovacím jazyce Groovy a Java.

Klíčová slova: Groovy, Java, Grails, výzkum, vizualizace, MariaDB, webová aplikace

ABSTRACT

The aim of this thesis is to design and create a functional prototype of web application that will allow an easy access to data from research, experimental development and innovations information system that provides pure data regarding research projects funded by public funding. However, the main goal is not to display these data, but to allow the creation of statistics and graphs based on these data. The resulting application will allow the user to create various visualisations of the data. These visualisations will allow the user to easily search for context and to draw appropriate conclusions based on these. The application is to be implemented in programming language Groovy and Java.

Keywords: Groovy, Java, Grails, research, visualisations, MariaDB, web application

Na tomto místě bych chtěl poděkovat panu Ing. Radku Šilhavému, Ph.D. který vedl mou diplomovou práci a byl mi velkou oporou i ve zdánlivě bezvýhodné situaci. Také díky němu se rozvinulo spektrum možností a využitelnosti této práce pro budoucí pokračování. Za to mu tímto děkuji.

Rovněž moc děkuji svým rodičům a sestře Evě, kteří mě celou dobu podporovali, pomáhali mi a měli se mnou trpělivost.

Prohlašuji, že odevzdaná verze diplomové práce a verze elektronická nahraná do IS/STAG jsou totožné.

OBSAH

ÚVOD.....	9
I TEORETICKÁ ČÁST.....	10
1 INFORMAČNÍ SYSTÉM VÝZKUMU, EXPERIMENTÁLNÍHO VÝVOJE A INOVACÍ.....	11
1.1 CENTRÁLNÍ EVIDENCE AKTIVIT VÝZKUMU	11
1.2 CENTRÁLNÍ EVIDENCE VÝZKUMNÝCH ZÁMĚRŮ	12
1.3 EVIDENCE VEŘEJNÝCH SOUTĚŽÍ VE VÝZKUMU, EXPERIMENTÁLNÍM VÝVOJI A INOVACÍCH	13
1.4 CENTRÁLNÍ EVIDENCE PROJEKTŮ VÝZKUMU, EXPERIMENTÁLNÍHO VÝVOJE A INOVACÍ.....	13
1.5 REJSTRÍK INFORMACÍ O VÝSLEDČÍCH	14
2 DATA Z VÝZKUMU A VÝVOJE.....	15
2.1 POPIS DATOVÝCH SOUBORŮ Z VÝZKUMU A VÝVOJE	15
3 POUŽITÉ TECHNOLOGIE.....	17
3.1 GRAILS	17
3.2 GROOVY.....	18
3.3 MARIADB	19
3.4 JQUERY	19
3.5 FLOT	20
3.6 BOOTSTRAP.....	21
II PRAKTICKÁ ČÁST	22
4 ANALÝZA POŽADAVKŮ	23
4.1 FUNKČNÍ POŽADAVKY	23
4.2 NEFUNKČNÍ POŽADAVKY	23
5 NÁVRH APLIKACE	25
5.1 POSTUPNÉ OPERACE A VIZUALIZACE DAT	26
5.1.1 Filtr dat - filter	27
5.1.2 Množinové sjednocení dat - union	27
5.1.3 Spojení dat různého druhu - join.....	28
5.1.4 Agregace dat - aggregation	29
5.1.5 Projekce vstupních dat - projection.....	30
5.1.6 Rozdělení vstupních dat - explode	30
5.1.7 Seřazení dat – order.....	31
5.1.8 Kresba grafu – operace graph	31
6 IMPLEMENTACE	32
6.1 IMPORT DAT DO APLIKACE.....	32
6.1.1 Výběr datového formátu	32
6.1.1.1 dBase.....	32
6.1.1.2 Excelovský sešit.....	34
6.1.1.3 CSV.....	34
6.1.1.4 Struktura importovaného CSV souboru.....	35

6.2	STRUKTURA DATABÁZE.....	36
6.3	IMPLEMENTACE OPERACÍ S DATY	37
6.3.1	Filter	38
6.3.2	Union.....	38
6.3.3	Join	39
6.3.4	Aggregation.....	39
6.3.5	Projection	40
6.3.6	Explode	40
6.3.7	Order	41
6.3.8	Graph.....	42
7	UKÁZKOVÁ ANALÝZA DAT Z RIV	43
7.1	IMPORT ZKOUMANÝCH DAT.....	44
7.2	POROVNÁNÍ POČTU ODBORNÝCH ČLÁNKŮ NA FAKULTÁCH INFORMATIKY	48
7.2.1	Výběr fakult informatiky.....	48
7.2.2	Výběr odborných článků	49
7.2.3	Výsledný žebříček fakult podle počtu odborných článků	49
7.3	ZASTOUPENÍ DRUHŮ VÝZKUMNÝCH VÝSLEDKŮ NA UTB FAI.....	51
7.3.1	Vstupní data výsledků	51
7.3.2	Propojení klíče výsledku s popisem výsledku	52
7.3.3	Výzkumné výsledky UTB FAI	52
7.3.4	Výsledná četnost druhů výzkumných výsledků na UTB FAI.....	53
7.4	POČET VÝZKUMNÝCH PRACOVNÍKŮ NA UTB FAI DLE DRUHU VÝSLEDKU.....	55
7.4.1	Vstupní data výsledků	55
7.4.2	Projekce vstupních dat	56
7.4.3	Unikátní tvůrci dle druhu výsledku.....	56
7.4.4	Výsledné počty výzkumných pracovníků dle druhu výsledku	57
8	ZPŮSOB ZABEZPEČENÍ APLIKACE	61
8.1	PŘIHLAŠOVÁNÍ UŽIVATELŮ	61
8.2	SQL INJECTION	62
8.2.1	Prepared statements.....	62
8.2.2	Stavba dotazů ověřenými parametry	63
9	MOŽNÝ ROZVOJ APLIKACE.....	64
9.1	VIZUALIZACE ANALÝZY V PODOBĚ ORIENTOVANÉHO GRAFU.....	64
9.2	LEPŠÍ PRÁCE S PARAMETRY OPERACÍ.....	64
9.3	PŘIDÁNÍ DALŠÍCH TYPŮ GRAFŮ	64
	ZÁVĚR	65
	SEZNAM POUŽITÉ LITERATURY.....	66
	SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK.....	69
	SEZNAM OBRÁZKŮ	70
	SEZNAM TABULEK.....	71
	SEZNAM UKÁZKOVÝCH KÓDŮ	72

ÚVOD

Říká se, že kdo má informace, má moc. Toto platí zvláště v dnešní technické společnosti a při dnešních možnostech techniky. Ukládat obrovské soubory dat se stalo natolik finančně dostupnou možností, že se dnes data sbírají naprosto o všem kolem nás a také o nás samotných. Sama o sobě by tato data však příliš velkou cenu neměla. Dat je tolik, že obyčejný lidský pohled na ně nemůže odhalit skryté souvislosti a vzorce, které z posbíraných dat mnohdy vyplývají, ale jsou úspěšně maskovány v gigabajtech nicneříkajících čísel a údajů.

Tato diplomová práce si klade za cíl navrhnout a vytvořit prototyp webové aplikace umožňující lepší pohled na data z výzkumných projektů Informačního systému výzkumu, experimentálního vývoje a inovací. V tomto systému je vedena podrobná evidence o projektech, jejich řešitelích a výsledcích, které jsou hrazeny z veřejných prostředků. Tímto jsou archivovány a výsledky jsou takto dostupné pro širokou veřejnost. Jedná se však pouze o strohá data, ze kterých se mohou jen těžko dělat, či potvrzovat předpokládané závěry. Zde je prostor pro mou aplikaci. Ta umožní snadno provádět nad takovými daty různé vizualizace v podobě matematických operací, agregací, statistik a grafů tak, aby byla lépe pochopitelná pro člověka. Výsledkem těchto operací jsou tedy souhrnné tabulky a pro člověka lehce pochopitelné grafy. Bude tedy možné například snadno porovnávat několik výzkumných projektů mezi sebou, zobrazit si například průměrný finanční náklad na projekt, nebo vyhledat řešitele s největší spotřebou financí.

Výsledkem této práce je návrh a implementace konceptu analytického nástroje, jehož velká výhoda je, že metoda práce s daty se ukázala být všestranně použitelnou a neomezuje se pouze na výše zmíněná výzkumná data. Výsledný analytický nástroj umožňuje práci s jakýmkoliv dostupnými a strojově čitelnými daty.

V teoretické části této práce je kladen důraz na výběr vhodných technologií, ze kterých se pak celý systém bude skládat, tak aby výsledný program poskytoval kýženou funkcionalitu a splnil náročné výkonové požadavky.

V praktické části práce se zaměřuji na návrh operací, metody a algoritmus pro zpracování dat a zobrazení výsledku. Z důvodu zaměření se na vývoj software, obsahuje práce úryvky částí programového kódu.

I. TEORETICKÁ ČÁST

1 INFORMAČNÍ SYSTÉM VÝZKUMU, EXPERIMENTÁLNÍHO VÝVOJE A INOVACÍ

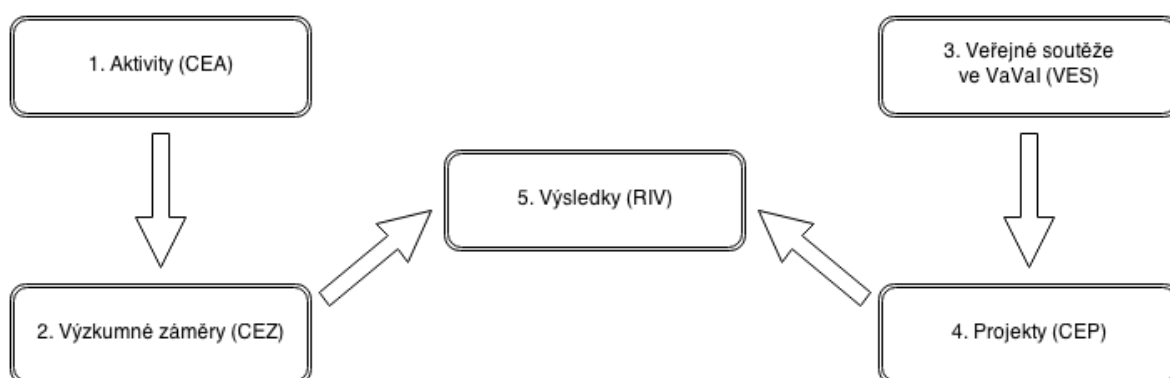
Informační systém výzkumu, experimentálního vývoje a inovací (dále jen VaVaI) je informační systém veřejné správy zajišťující shromažďování, zpracování, poskytování a využívání údajů o výzkumu, vývoji a inovacích podporovaných z veřejných prostředků.

V tomto systému je vedena podrobná evidence o projektech, jejich řešitelích a výsledcích, které jsou takto archivovány a výsledky jsou dostupné pro širokou veřejnost.

Systém VaVaI se skládá z pěti hlavních částí, neboli datových oblastí

1. CEA: Centrální evidence aktivit výzkumu a Informace o předávání údajů
2. CEZ: Centrální evidence výzkumných záměrů a Informace o předávání údajů
3. VES: Evidence veřejných soutěží ve výzkumu, experimentálním vývoji a inovacích a Informace pro předávání údajů
4. CEP: Centrální evidence projektů výzkumu, experimentálního vývoje a inovací a Informace o předávání údajů
5. RIV: Rejstřík informací o výsledcích a Informace o předávání údajů

Tyto datové oblasti jsou spolu v patřičné relaci a jejich propojení je zachyceno na Obr. 1.



Obr. 1. Schéma systému VaVaI [1]

1.1 Centrální evidence aktivit výzkumu

Centrální evidence aktivit výzkumu, experimentálního vývoje a inovací - CEA je jednou z částí (datovou oblastí) Informačního systému výzkumu, experimentálního vývoje a inovací

(IS VaVaI). Jsou v ní shromažďovány informace o příjemcích, poskytovatelích a výši podpory dle § 2 odst. 2 písm.h) zákona č. 130/2002 Sb., o podpoře výzkumu, experimentálního vývoje a inovací z veřejných prostředků a o změně některých souvisejících zákonů (zákon o podpoře výzkumu, experimentálního vývoje a inovací), ve znění pozdějších předpisů.

Údaje do CEA předávají poskytovatelé podpory z veřejných prostředků, kterými jsou správci příslušných kapitol státního rozpočtu (ústřední orgány státní správy, Grantová agentura České republiky, Technologická agentura České republiky a Akademie věd České republiky) nebo územní samosprávné celky.

Obsah CEA, postup při předání, zařazení, zpracování a poskytování údajů je stanoven zákonem č. 130/2002 Sb., nařízením vlády č. 397/2009 Sb., o informačním systému výzkumu, experimentálního vývoje a inovací, zvláštními právními předpisy a Provozním řádem IS VaVaI.

Přímé vyhledávání ve veřejně přístupných údajích CEA umožňuje vyhledávat data od roku 2009. Předání platných údajů do CEA poskytovatelem je podmínkou pro poskytnutí institucionální a účelové podpory z veřejných prostředků [2].

1.2 Centrální evidence výzkumných záměrů

Centrální evidence výzkumných záměrů - CEZ je jednou z částí (datovou oblastí) Informačního systému výzkumu, experimentálního vývoje a inovací (IS VaVaI), ve které jsou shromažďovány informace o výzkumných záměrech podporovaných z veřejných prostředků podle zákona č. 130/2002 Sb., o podpoře výzkumu, experimentálního vývoje a inovací z veřejných prostředků a o změně některých souvisejících zákonů (zákon o podpoře výzkumu, experimentálního vývoje a inovací), ve znění pozdějších předpisů.

Údaje do CEZ předávají poskytovatelé institucionální podpory z veřejných prostředků, kterými jsou správci příslušných kapitol státního rozpočtu (ústřední orgány státní správy, Akademie věd České republiky) nebo územní samosprávné celky.

Obsah CEZ, postup při předání, zařazení, zpracování a poskytování údajů je stanoven zákonem č. 130/2002 Sb., nařízením vlády č. 267/2002 Sb., o informačním systému výzkumu a vývoje (od 1. ledna 2010 nahrazeno nařízením vlády č. 397/2009 Sb., o informačním systému výzkumu, experimentálního vývoje a inovací), zvláštními právními předpisy a Provozním řádem IS VaV.

Přímé vyhledávání ve veřejně přístupných údajích CEZ umožňuje vyhledávat data od roku 1998.

Předání platných údajů do CEZ poskytovatelem a jejich zařazení do informačního systému výzkumu, experimentálního vývoje a inovací je podmínkou pro poskytnutí institucionální podpory z veřejných prostředků na výzkumné záměry [3].

1.3 Evidence veřejných soutěží ve výzkumu, experimentálním vývoji a inovacích

Evidence veřejných soutěží ve výzkumu, experimentálním vývoji a inovacích – VES je jednou z částí (datovou oblastí) Informačního systému výzkumu, experimentálního vývoje a inovací (IS VaV), ve které jsou shromažďovány informace o veřejných soutěžích ve výzkumu a vývoji vyhlášených podle zákona č. 130/2002 Sb., o podpoře výzkumu a vývoje z veřejných prostředků a o změně některých souvisejících zákonů (zákon o podpoře výzkumu a vývoje).

Údaje do VES předávají poskytovatelé účelové podpory z veřejných prostředků, kterými jsou správci příslušných kapitol státního rozpočtu (ústřední orgány státní správy, Grantová agentura České republiky, Akademie věd České republiky) nebo územní samosprávné celky.

Obsah VES, postup při předání, zařazení, zpracování a poskytování údajů je stanoven zákonem č. 130/2002 Sb., o podpoře výzkumu a vývoje, nařízením vlády č. 397/2009 Sb., o informačním systému výzkumu a vývoje, zvláštními právními předpisy a provozním řádem IS VaV [4].

1.4 Centrální evidence projektů výzkumu, experimentálního vývoje a inovací

Centrální evidence projektů výzkumu, experimentálního vývoje a inovací – CEP je jednou z částí (datovou oblastí) Informačního systému výzkumu, experimentálního vývoje a inovací (IS VaVaI), ve které jsou shromažďovány informace o projektech výzkumu, vývoje a inovací podporovaných z veřejných prostředků podle zákona č. 130/2002 Sb., o podpoře výzkumu, experimentálního vývoje a inovací z veřejných prostředků a o změně některých souvisejících zákonů (zákon o podpoře výzkumu, experimentálního vývoje a inovací), ve znění pozdějších předpisů.

Údaje do CEP předávají poskytovatelé účelové podpory z veřejných prostředků, kterými jsou správci příslušných kapitol státního rozpočtu (ústřední orgány státní správy, Grantová agentura České republiky, Akademie věd České republiky) nebo územní samosprávné celky.

Obsah CEP, postup při předání, zařazení, zpracování a poskytování údajů je stanoven zákonem č. 130/2002 Sb., nařízením vlády č. 267/2002 Sb., o informačním systému výzkumu a vývoje (od 1. ledna 2010 nahrazeno nařízením vlády č. 397/2009 Sb., o informačním systému výzkumu, experimentálního vývoje a inovací), zvláštními právními předpisy a Provozním řádem IS VaVaI.

Přímé vyhledávání ve veřejně přístupných údajích CEP umožňuje vyhledávat data od roku 1993 [5].

1.5 Rejstřík informací o výsledcích

Rejstřík informací o výsledcích – RIV je jednou z částí (datovou oblastí) Informačního systému výzkumu, experimentální vývoje a inovací (IS VaV). V této části jsou shromažďovány informace o výsledcích projektů výzkumu a vývoje a výzkumných záměrů podporovaných z veřejných prostředků podle zákona č. 130/2002 Sb., o podpoře výzkumu a vývoje z veřejných prostředků a o změně některých souvisejících zákonů (zákon o podpoře výzkumu a vývoje).

Údaje do RIV předávají poskytovatelé účelové a institucionální podpory z veřejných prostředků, kterými jsou správci příslušných kapitol státního rozpočtu (ústřední orgány státní správy, Grantová agentura České republiky, Akademie věd České republiky) nebo územní samosprávné celky.

Obsah RIV, postup při předání, zařazení, zpracování a poskytování údajů je stanoven zákonem č. 130/2002 Sb., o podpoře výzkumu a vývoje, nařízením vlády č. 397/2009 Sb., o informačním systému výzkumu a vývoje, zvláštními právními předpisy a provozním řádem IS VaV.

RIV obsahuje údaje o výsledcích výzkumu a vývoje uplatněných od roku 1993 [6].

2 DATA Z VÝZKUMU A VÝVOJE

Data v informačním systému výzkumu, experimentálního vývoje a inovací (dále jen VaVaI) jsou veřejně přístupná k online nahlížení, filtrování i exportování přímo na stránkách tohoto systému [1].

Případný uživatel, který chce nahlížet na data v systému VaVaI, vždy nejprve volí jednu z požadovaných datových oblastí, která mu bude následně zobrazena. Ve zvolené oblasti má uživatel možnost data vyhledávat, filtrovat podle zvolených kritérií a číst nalezené výsledky. K tomuto účelu slouží uživateli systému VaVaI jednoduchý filtr, kterým je možno specifikovat kritéria vyhledávání. Je tedy umožněno nalezení a zobrazení pouze takové množiny dat, která odpovídá kritériím uživatele. Tento filtr je však velice jednoduchý a neumožňuje systému VaVaI pokládat složitější dotazy na hledaná data. Slouží pouze k rychlé orientaci.

Důležitou součástí informačního systému VaVaI je export nalezených dat. Uživateli je tak umožněno snadno získat nalezená data ve strojově snadno zpracovatelné podobě a využít je tak pro své další potřeby a analýzy.

Celý proces exportu probíhá nejprve zadáním patřičných kritérií, dle kterých nalezneme v systému požadovaná data a následným výběrem druhu exportu. Na výběr jsou dva exportní formáty - zazipovaný excelovský sešit a zazipovaný soubor tabulky dBase. Oba exportní formáty obsahují stejná data.

2.1 Popis datových souborů z výzkumu a vývoje

Exportované soubory z rejstříků CEA, CEP, CEZ, VES, nebo RIV vždy obsahují tabulková data. Řádky vždy představují jednotlivé záznamy a sloupcečky vlastnosti těchto záznamů. Např. v rejstříku RIV každý řádek reprezentuje jeden výzkumný výsledek a sloupcečky nesou potřebné informace o tomto výsledku. Takovými sloupcečky mohou být název výzkumného výsledku, seznam tvůrců výsledku, nebo jazyk výsledku.

Rozsáhlý popis datových tabulek z jednotlivých rejstříků získáme vždy v samostatném HTML dokumentu, jako přílohu k exportovaným datům.

V exportovaných datech je mnoho sloupceček, jejichž hodnota v řádku je vždy pouze referenčním odkazem neboli klíčem z číselníku hodnot. Abychom získali skutečnou informaci,

kteřou hodnota takového sloupečku představuje, musíme nejprve podle daného sloupečku zvolit související číselník a z toho pak vyčíst správnou hodnotu nalezenou podle klíče.

Některé číselníky jsou společné pro všechny rejstříky. Jedná se například o číselník zemí, územně správních celků, nebo skupiny oborů. Mnoho číselníků je ale unikátních a týkají se pouze dat jednoho z rejstříků. Všechny číselníky jsou veřejně dostupné online [7].

3 POUŽITÉ TECHNOLOGIE

3.1 Grails

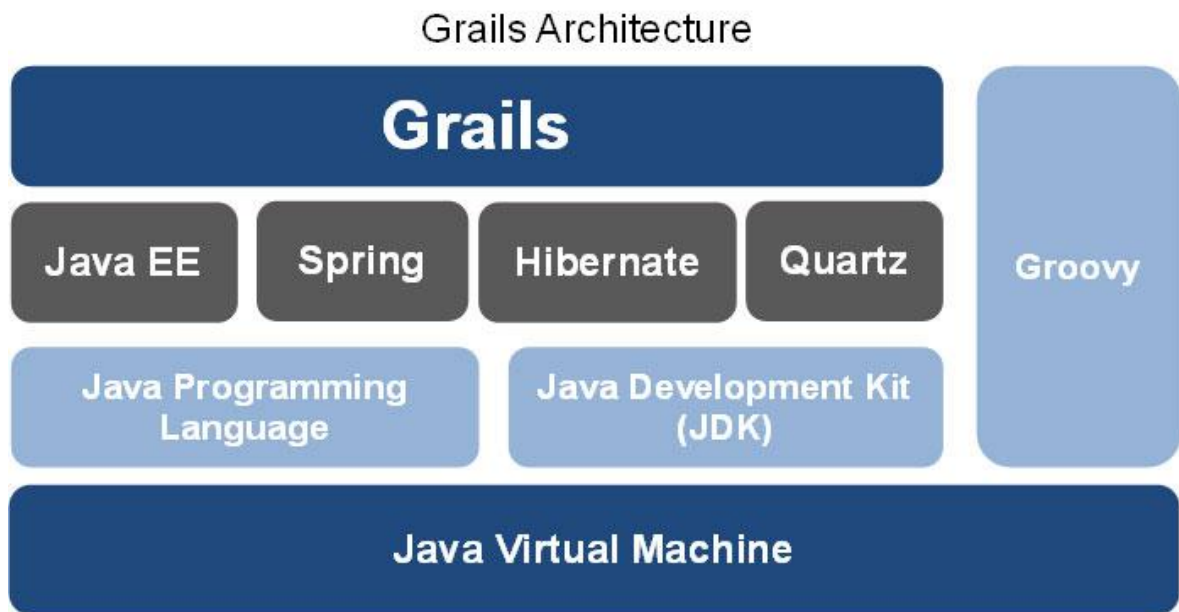
Grails je moderní open source framework pro snadný vývoj webových aplikací. Je vyvíjen a udržován komunitou a především silnou společností SpringSource. Primárním programovacím jazykem je zde jazyk Groovy, což je jazyk odvozený od jazyka Java, viz kapitola 3.2. Framework Grails se snaží inspirovat z toho nejlepšího ze současných moderních frameworků pro vývoj webových aplikací, jako je například velice známý framework Ruby on Rails. Snaží se přenést a pokud možno i vylepšit oblíbené a ověřené metody do ekosystému jazyka Java a Groovy. Z této inspirace také pramení jméno samotného frameworku, které se dříve nabývalo významu Groovy on Rails.

Jelikož aplikace využívající framework Grails se píše v programovacím jazyce Groovy, případně v jazyce Java, je k běhu aplikace zapotřebí aplikační server (např. Apache Tomcat) a virtuální stroj JVM, tedy runtime jazyka Java.

Grails se snaží uplatňovat moderní principy programování. Naprosto stěžejní a na první pohled dobře patrné principy jsou Convention over configuration (CoC) a Don't Repeat Yourself (DRY). Díky těmto principům může programátor jedním příkazem vytvořit a spustit funkční základ webové aplikace s architekturou MVC, používající relační databázi a webové služby na bázi RESTful, aniž by musel cokoli konfigurovat.

Důležitou vlastností frameworku Grails je jeho modulárnost. Každou programovanou součást aplikace lze psát jako modul a vyčlenit tak tuto součást do samostatného projektu. Takto vytvořený modul se dá velmi snadno znovu použít v jiných aplikacích. Součástí frameworku Grails je i ekosystém pro uveřejňování již hotových modulů, tzv. pluginů. Tyto veřejné moduly lze pak snadno integrovat do námi vyvíjené aplikace. Nemusíme se tedy např. vůbec zatěžovat studováním API pro přihlášení uživatele přes Facebook, stačí pouze nainstalovat již hotový modul. Grails díky své popularitě disponuje velkým množstvím těchto hotových modulů. Díky tomu se může programátor lépe soustředit na primární problém jeho práce.

Více informací a dokumentace k frameworku Grails se nachází na oficiálním webu tohoto frameworku [8].



Obr. 2. Architektura a stěžejní součásti frameworku Grails [9]

3.2 Groovy

Groovy je jeden z jazyků využívajících ke svému běhu virtuální stroj jazyka Java, tzv. JVM. Jazyk Groovy navíc přímo vychází z jazyka Java a je s ním natolik kompatibilní, že téměř všechny platný Java kód je také platný Groovy kód. Díky tomuto je přechod z jazyka Java velice snadný a intuitivní.

Jazyk Groovy přináší oproti jazyku Java řadu nových vlastností a vylepšení.

- Velmi se inspiruje principy jazyků Python, Ruby a Smalltalk.
- Podporuje jak staticky, tak také dynamicky kompilovaný kód.
- Podporuje staticky i dynamicky typované proměnné.
- Obsahuje mnoho prvků známých z funkcionálních programovacích jazyků.
- Podporuje tzv. closures, které řeší v Javě tolik chybějící ukazatele na funkce.
- Disponuje silnou syntaxí pro tvorbu vlastních doménových jazyků.
- Velice usnadňuje práci s Java reflexí.

Jazyk Groovy [10] tím velmi šetří kód a usnadňuje psaní jinak velmi složitých struktur.

3.3 MariaDB

MariaDB je open source relační databáze [11], která vznikla tzv. forkem (novou větví) relační databáze MySQL. Hlavní důvod k vytvoření této nové větve byl ten, aby se u této databáze udržela licence svobodného softwaru GNU GPL. Původní MySQL získala nadnárodní společnost Oracle díky akvizici původního Sun Microsystems. Databáze MySQL je přímý konkurent vlastního komerčního databázového řešení Oracle a proto panují obavy z možného omezování této databáze.

Díky faktu že MariaDB přímo vychází z MySQL jsou tyto dvě databáze plně kompatibilní. MariaDB má však rychlejší vývoj a přidává již řadu nových funkcionalit a především výrazné výkonové vylepšení.

Nejvýraznější změnou, kvůli které jsem se rozhodl v této diplomové práci používat MariaDB, namísto známější MySQL je ta, že MariaDB přidává nový storage engine XtraDB. XtraDB je rozšířenou verzí původního InnoDB a obsahuje všechny jeho vlastnosti, včetně jeho nejdůležitějších ACID operací. Navíc však přidává metriky pro databázové tabulky, více ladících možností a především podporu pro mnohem lepší škálovatelnost. To je pro výkonově náročné aplikace zcela zásadní.

3.4 jQuery

jQuery je javascriptová knihovna [12], která má podporu v celé řadě prohlížečů a jejím cílem je zjednodušit interakci mezi JavaScriptem a HTML. Vznikla už v roce 2006 a jedná se o svobodný a otevřený software.

Pro tuto práci jsem jQuery vybral hlavně z důvodu mnohaletých pozitivních zkušeností. Stejně jako CSS odděluje „zobrazovací“ charakteristiky od struktury HTML, jQuery odděluje „chování“ od struktury HTML. Například místo přímé specifikace on-click události přímo v HTML kódu tlačítka by stránka řízená jQuery napřed našla vhodný element tlačítka, a potom změnila jeho manipulátor události. Takovéto oddělení chování od struktury se také často nazývá jako princip nevtíravého JavaScriptu.

Stěžejní vlastnosti knihovny jQuery jsou:

- snadný výběr DOM elementů pomocí selektorů
- funkce pro procházení a změnu DOM objektů
- registrace událostí a posluchačů

- manipulace s CSS vlastnostmi
- efekty a animace HTML prvků
- usnadnění AJAXových volání
- snadná rozšiřitelnost vytvořené aplikace
- kompatibilita mezi mnoha prohlížeči

3.5 Flot

Flot je open source knihovna [13] v čistém JavaScriptu a slouží pro vytváření a kreslení grafů. Tato knihovna přímo kooperuje s knihovnou jQuery popsanou výše a dává si za cíl maximálně zjednodušit a zatraktivnit práci s grafy kreslenými v prohlížeči na straně klienta.

K vykreslení grafů používá knihovna element canvas známý z HTML5. Díky vlastnostem a vykreslovacím schopnostem canvasu nemusí být grafy pouze statické. Grafy mohou být dokonce i animované. Můžeme tak zobrazovat i realtime grafy, kdy data přicházejí průběžně ze serveru a grafika je pak každou chvíli aktualizována.

Knihovna Flot podporuje také uživatelskou interakci s vykreslenými grafy. Uživateli je tak umožněno např. zvětšit si patřičnou oblast grafu, detailně odečítat hodnoty z jakéhokoliv místa v grafu, nebo zobrazovat dodatečné informace k hodnotám zaneseným do grafu.

Flot nabízí konstrukce a metody pro vykreslení základních ale i pokročilejších typů grafů. Základní typy grafů jsou:

- spojnicové grafy
- výsečové grafy
- prstencové grafy
- plošné grafy
- sloupcové grafy
- XY bodové grafy

Další typy grafů se dají snadno odvodit od již existujících. Díky těmto vlastnostem je Flot ideální kandidát pro vizualizaci dat u klienta a provádění analýz samotným uživatelem přímo nad výslednými grafy.

3.6 Bootstrap

Bootstrap je velmi jednoduchý a volně dostupný framework [14], který usnadňuje tvorbu grafického rozhraní moderních webových aplikací. K tomuto účelu je vybaven celou řadou hotových komponent a podpůrných nástrojů v HTML, CSS i JavaScriptu. Mezi tyto prvky se řadí například tlačítka, tabulky, navigační panely, výběrové nabídky, ale i dynamické prvky jako jsou kontextová menu, modální okna a layouty. Programátor se přitom vůbec nemusí starat o design těchto prvků, protože ten je již hotov a jeho funkčnost je ověřena ve většině dnešních prohlížečů. Může se tak lépe soustředit na cíl práce, aniž by tím utrpěl design aplikace.

II. PRAKTICKÁ ČÁST

4 ANALÝZA POŽADAVKŮ

Abych mohl celý systém implementovat, musel jsem nejprve získat seznam všech základních požadavků, které musí výsledná aplikace splnit. Některé požadavky vznikly po konzultacích s panem Ing. Radkem Šilhavým Ph.D., vedoucím mé diplomové práce a další jsem získal pečlivým procházením Informačního systému výzkumu, experimentálního vývoje a inovací, kde jsem zjišťoval možnosti tohoto systému.

Všechny požadavky, které jsem posbíral, budou zohledněny při návrhu a implementaci celé aplikace. K tomuto účelu jsem také seznam požadavků rozdělil na požadavky funkční a nefunkční.

4.1 Funkční požadavky

Funkční požadavky zachycují ty funkce aplikace, které vykonává sám uživatel. Výsledný seznam funkčních požadavků je zachycen v tabulce Tab. 1.

Tab. 1. Seznam funkčních požadavků

ID	požadavek
RF1	Systém umožní import dat z Informačního systému výzkumu, experimentálního vývoje a inovací.
RF2	Systém bude mít funkcionalitu pro import obecných dat tabulkového charakteru.
RF3	Systém umožní zobrazit a procházet vytvářená nebo importovaná data.
RF4	Systém umožní provádění analýz a statistik nad daty z IS VaVal.
RF5	Systém umožní provádění obecných operací a vizualizací dat.
RF6	Systém umožní členění analýz do samostatných celků - projektů.
RF7	Systém umožní seřazení dat podle vybraného kritéria.
RF8	Systém umožní nad daty zobrazovat vybrané grafy.
RF9	Systém bude umožňovat přihlášení a odhlášení uživatele.
RF10	Systém umožní práci pouze přihlášenému uživateli.

4.2 Nefunkční požadavky

Nefunkční požadavky se netýkají funkcionality, kterou by pocítil sám uživatel. Jedná se však o důležité nároky na vzniklou aplikaci, aby zvládla správným způsobem plnit svou funkci a poskytovat např. potřebný výkon. Výsledný seznam nefunkčních požadavků představuje tabulka Tab. 2.

Tab. 2. Seznam nefunkčních požadavků

ID	požadavek
RN1	Systém bude používat relační databázi.
RN2	Systém musí být schopen pracovat s tabulkami s řádově desítkami tisíc řádků.
RN3	Systém bude implementován jako webová aplikace.

5 NÁVRH APLIKACE

Stěžejní součástí mé práce je vizualizace výsledků výzkumu a vývoje. Jedná se přitom o výzkumné výsledky z Informačního systému výzkumu, experimentálního vývoje a inovací (dále jen IS VaVaI) [1].

Při návrhu aplikace jsem měl na výběr ze dvou možností:

1. Vytvořit jednoúčelovou aplikaci, která bude obsahovat různé filtry, souhrnné agregované informace a možnosti vizualizace dat z IS VaVaI, kterému tyto možnosti chybí. Jednalo by se tak pouze o jakousi nadstavbu pro IS VaVaI.
2. Vytvořit zcela obecnou aplikaci pro tvorbu různých vizualizací obecných tabulkových dat, kterými jsou i výzkumná data z IS VaVaI. Tato data by šla filtrovat a různými způsoby např. agregovat tak, aby ve výsledku mohl uživatel provést mnoho druhů různých statistik nad zcela obecnými daty, neomezenými pouze systémem pro výzkum a vývoj.

První varianta by měla výhodu v tom, že by pro uživatele znamenala pouze parametrickou práci s již pevně danými vizualizacemi. Velmi podstatná nevýhoda tkví ale v jednoúčelovosti takovéto aplikace a nemožnosti vizualizovat data jiným způsobem, než má aplikace nastaveno z výroby.

Druhá varianta má nevýhodu v tom, že klade větší nároky na znalost uživatele, který bude muset potřebné vizualizace sám vytvořit. Tato varianta má však mnoho podstatných výhod. Tyto výhody jsou především v obecnosti výsledné aplikace, kterou tak bude možno použít pro vizualizace jakýchkoliv dat tabulkového charakteru a také v libovolných možnostech vizualizací, které nebudou omezené pouze parametrickým nastavením.

Protože jsem se rozhodl pro druhou variantu, musel jsem nejprve navrhnout, jakým způsobem budou nad daty analýzy probíhat a jak je bude uživatel vytvářet. Tento návrh bude nejdůležitějším principem této práce.

Jako velmi pokročilý uživatel relačních i nerelačních databázových systémů mám mnoho zkušeností s prováděním datových analýz přímo pomocí SQL dotazů. Z vlastní zkušenosti tedy vím, jak složité je poskládat komplexní SQL dotaz. Pro uživatele, který nemá potřebné know-how, je skladba jakéhokoliv složitějšího SQL dotazu téměř nemožná.

Protože datové analýzy v SQL provádím běžně podle zadání člověka, který nemá potřebné znalosti SQL a pouze ví, jaká data se dají z databáze získat, zjistil jsem, že i člověk, který

nemá potřebnou znalost SQL dokáže vlastními slovy poměrně dobře vysvětlit, jaká data a vizualizace potřebuje získat z dat, která se v jeho databázi skrývají. Nedokáže však komplexně spojit všechny podmínky do jediného celku, nebo SQL dotazu. Proto jsem navrhl systém postupného aplikování operací jedné po druhé, postupného získávání výsledků a postupné vizualizace dat.

5.1 Postupné operace a vizualizace dat

Navrhl jsem systém postupného aplikování velmi jednoduchých operací. Cílem je, aby se v každém kroku dala provést pouze jediná elementární operace nad analyzovanými daty a aby ihned po aplikaci dané operace vznikl pro uživatele viditelný výsledek. Tento výsledek pak uživatel bude moci vizualizovat v podobě tabulky nebo grafu a především jej použít jako vstup pro další možnou operaci.

Bylo tedy potřeba navrhnout základní množinu elementárních operací. Takových, aby byly naprosto srozumitelné pro uživatele a aby se spojením jejich výsledků daly realizovat i složité analýzy. Zkoumáním mnoha scénářů různých analýz jsem navrhl těchto šest základních operací:

1. filtr dat – dále reprezentovaný operací filter
2. množinové sjednocení dat – operace union
3. spojení dat různého druhu – operace join
4. agregace dat – operace aggregation
5. projekce vstupních dat – operace projection
6. rozdělení vstupních dat – operace explode

Výše zmíněné operace přímo manipulují s daty a upravují je. Dále jsem pro účely lepší vizualizace navrhl dvě operace, které nijak data neupravují, pouze data patřičným způsobem zobrazují. Tyto operace se dají použít ve stejném kroku, jako operace manipulační. Jsou to tyto:

1. seřazení dat – operace order
2. kresba grafu – operace graph

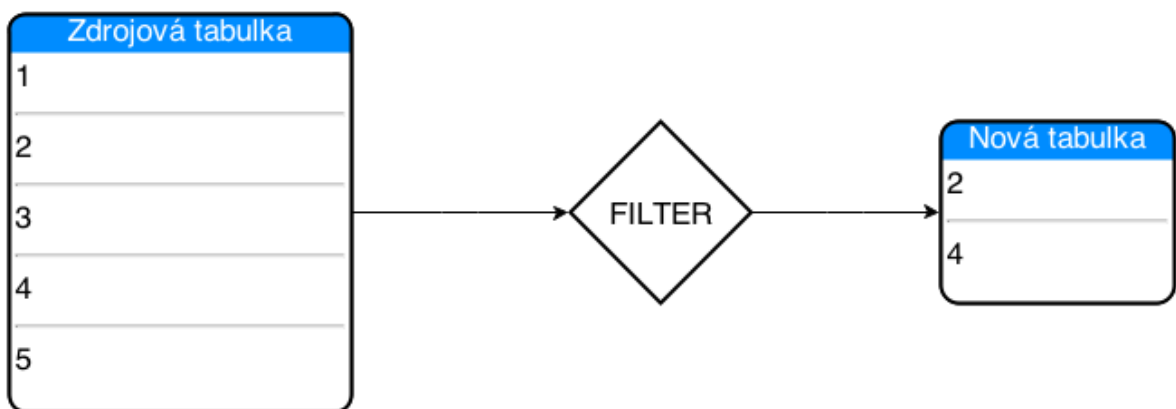
Každá manipulační operace má jednu nebo více vstupních tabulek a vždy jednu výstupní tabulku. Výstupní tabulku pak můžeme použít jako vstup pro další operace. Tímto vzniká orientovaný graf, kde uzly grafu jsou jednotlivé tabulky a orientovaná hrana grafu představuje vstup, nebo výstup operace. Celou analýzu lze tedy zachytit do orientovaného grafu.

Jak takový graf výsledné analýzy vypadá, lze vidět v kapitole 7 na Obr. 10, kde je podrobně krok za krokem vysvětlena ukázková analýza dat z Informačního systému výzkumu, experimentálního vývoje a inovací.

V podkapitolách níže uvádím, jakou funkci mají jednotlivé operace, jak manipulují s daty a jak jsem je navrhl.

5.1.1 Filtr dat - filter

Operace filter se provádí nad jedinou tabulkou a jejím výsledkem je nová tabulka s podmnožinou řádků původní tabulky. Znamená to tedy, že filtrujeme řádky z původní tabulky. Výsledná nová tabulka bude obsahovat pouze ty řádky, které splní podmínky zadaného filtru. Z toho vyplývá, že tato nová tabulka bude mít stejnou strukturu jako tabulka zdrojová a bude tedy obsahovat stejné sloupečky a jejich typy.

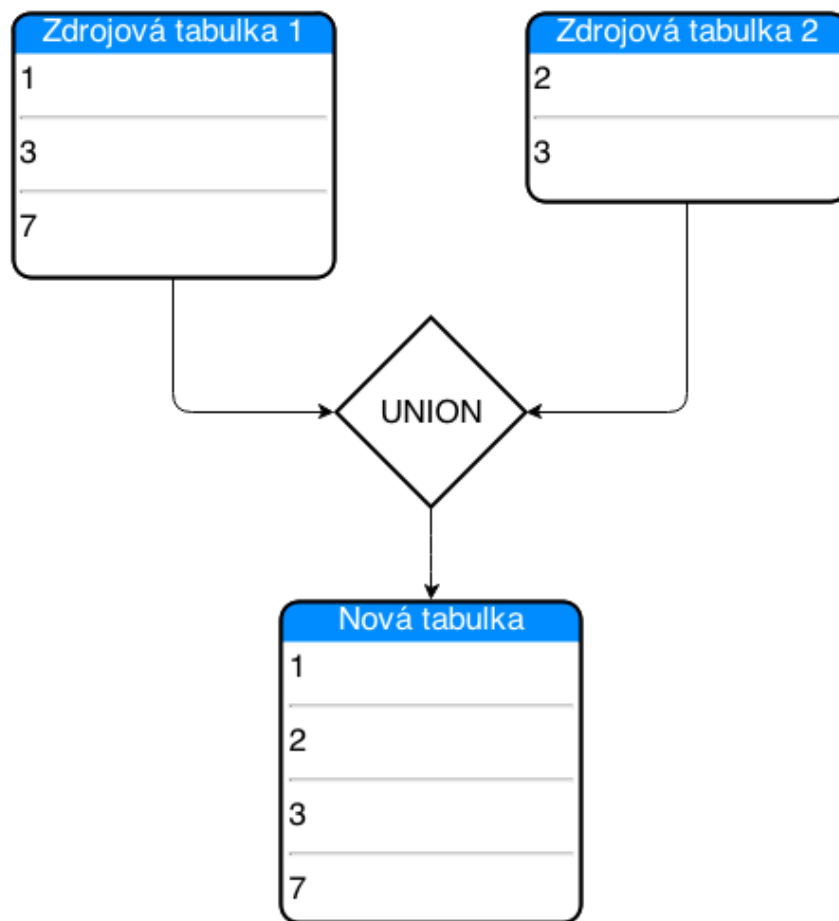


Obr. 3. Ukázka operace filter

Na ukázce, kterou můžeme vidět na Obr. 3, vidíme příklad operace filter, která z původní zdrojové tabulky vybírá pouze ty řádky, které jsou sudými čísly. Výsledkem operace je nová tabulka s vyhovujícími řádky.

5.1.2 Množinové sjednocení dat - union

Operace union se provádí nad dvěma nebo více zdrojovými tabulkami, které přichází jako vstup. Výstupem této operace je nová tabulka, obsahující sjednocení množin řádků všech vstupních tabulek. Výsledná tabulka bude tedy obsahovat všechny řádky všech vstupních tabulek a každý řádek v této nové tabulce bude unikátní, neboli nová tabulka nebude obsahovat žádné duplicitní řádky. Pro operaci union je nezbytné, aby všechny vstupní tabulky měly stejnou strukturu. Všechny musí obsahovat stejné sloupce a stejné datové typy. Výsledná tabulka bude mít tutéž strukturu, jako tabulky vstupní.



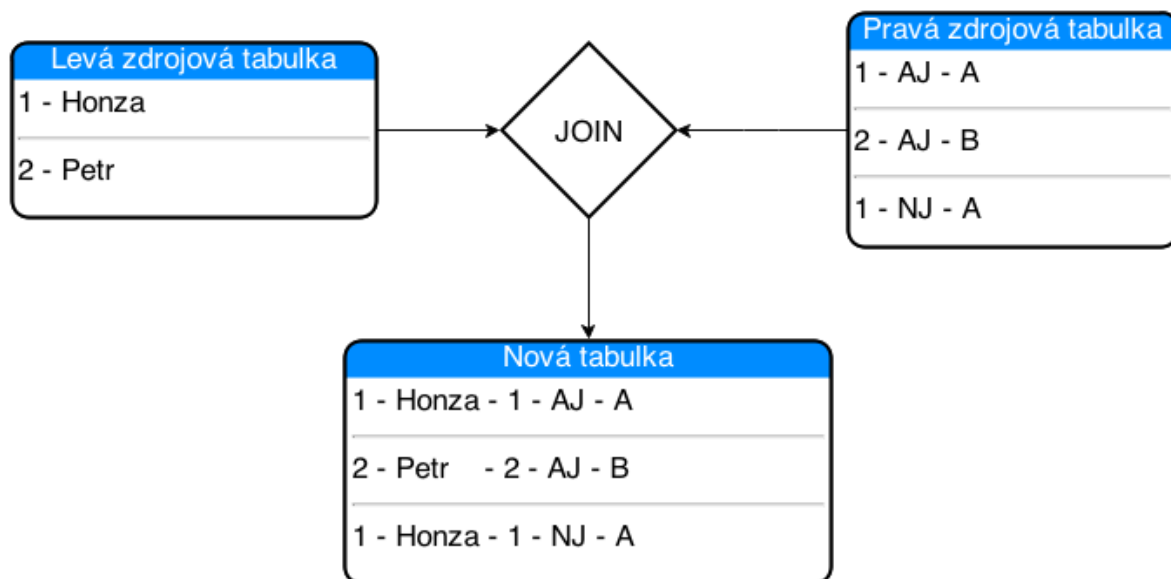
Obr. 4. Ukázka operace union

Na ukázce zachycené na Obr. 4 můžeme vidět praktický příklad použití operace union. Na vstup jsou přivedeny dvě tabulky. Výsledkem je nová tabulka, obsahující všechny hodnoty obsažené ve vstupních tabulkách.

5.1.3 Spojení dat různého druhu - join

Operace join má dvě vstupní tabulky, levou a pravou. Slouží pro spojení dvou různých tabulek s různou strukturou a datovými typy do jedné společné tabulky. Operace join musí mít specifikováno, podle jakého klíče se má vytvořit sloučení vstupních tabulek. K tomuto účelu je potřeba vybrat jeden sloupec z levé tabulky a jeden sloupec z pravé tabulky. Výsledek se pak vytváří tak, že se spojí každé dva řádky z levé a pravé tabulky, které mají stejný spojovací sloupec. Postup znázorňuje obrázek Obr. 5. V tomto příkladu se spojují tabulky podle prvního číselného sloupečku. Ty řádky, u kterých došlo ke shodě ve spojovacím sloupečku, jsou reprezentovány jedním spojeným řádkem v nové tabulce. Tento řádek má první část z levé tabulky a druhou část z pravé tabulky. Nová tabulka má také

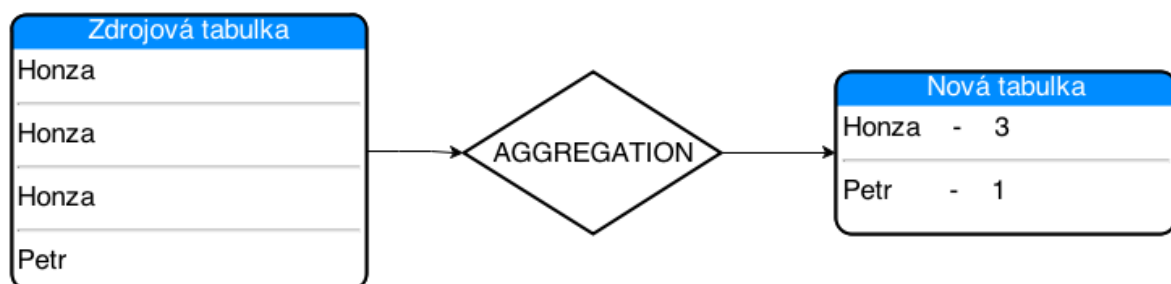
novou strukturu sloupců, která rovněž vzniká sloučením sloupců z levé tabulky a sloučením sloupců z pravé tabulky.



Obr. 5. Ukázka operace join

5.1.4 Agregace dat - aggregation

Operace aggregation slučuje více řádků z jedné vstupní tabulky do jednoho řádku. Takto vzniká tabulka, která má většinou menší počet řádků než tabulka zdrojová, ale každý jednotlivý řádek této nové tabulky může obsahovat statistické údaje o řádcích, ze kterých výsledný řádek vznikl. Nejčastěji se jedná např. o aritmetický průměr, sumu hodnot, nebo o maximální a minimální hodnoty. Většinou se tedy operace aggregation provádí za účelem statistického pohledu na data původní zdrojové tabulky.

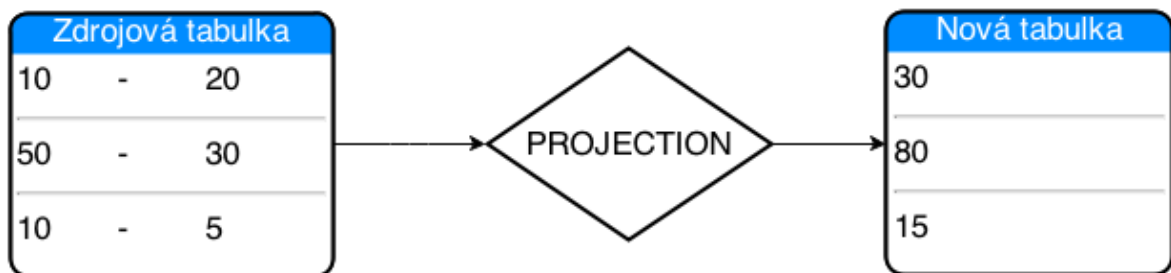


Obr. 6. Ukázka operace aggregation

Příklad operace aggregation zachycuje Obr. 6. Na vstup operace je přivedena zdrojová tabulka, která v řádcích obsahuje křestní jména. Výstupem daného příkladu je pak nová tabulka obsahující počet výskytu každého křestního jména.

5.1.5 Projekce vstupních dat - projection

Operace projection, neboli projekce, slouží pouze k jinému zobrazení dat ze zdrojové tabulky. Jiného zobrazení můžeme dosáhnout například tak, že z původní tabulky odebereme jeden či více sloupců, nebo naopak přidáním nových sloupců. Nové sloupce mohou vzniknout např. matematickými operacemi nad původními sloupci. Operace projection má jednu vstupní tabulku a umožňuje, aby výsledná nová tabulka měla jinou strukturu než tabulka vstupní. To je dáno již zmíněnou možností odebírat sloupce, případně vytvářet nové.

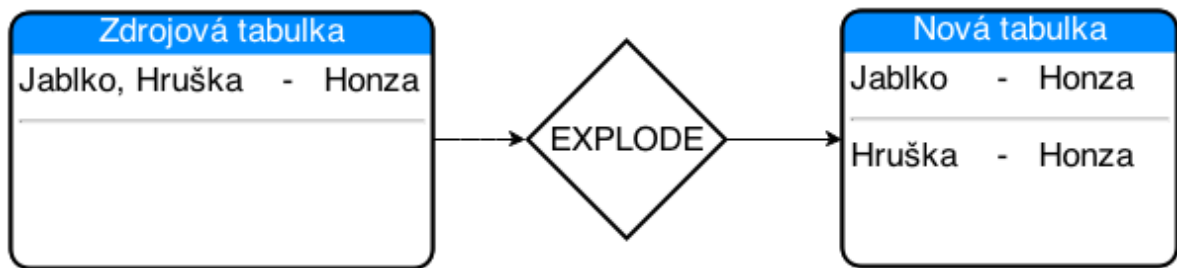


Obr. 7. Ukázka operace projection

Příklad operace projection lze vidět na Obr. 7. Jako zdrojová tabulka je na vstup přivedena tabulka se dvěma číselnými sloupci. Výsledkem je nová tabulka obsahující pouze jeden sloupec se součtem hodnot obou sloupců zdrojové tabulky.

5.1.6 Rozdělení vstupních dat - explode

Operace explode se používá tehdy, pokud máme více hodnot stejného typu v jednom datovém sloupečku. Tyto hodnoty od sebe bývají odděleny vhodným oddělovačem, např. čárkou. V daném sloupečku se tak může např. nacházet seznam ovoce, jako je tomu v příkladu na Obr. 8. Takovéto uspořádání dat však velmi ztěžuje jakékoliv analýzy, neboť nad daty uloženými v jednom textovém řetězci se dá provádět jen velmi málo operací. Důvodem je neznámá struktura takto uložených dat. Tento problém řeší právě operace explode, která podle zadaného oddělovače patřičně rozdělí data v daném sloupečku a vytvoří nové řádky v tabulce tak, aby v onom sloupečku byla vždy jen jedna z hodnot. Operace explode vytvoří tedy tolik nových řádků, kolik samostatných hodnot je uloženo v daném sloupci.



Obr. 8. Ukázka operace explode

Příklad operace explode lze vidět na Obr. 8. Jako zdrojová tabulka je na vstup přivedena tabulka se sloupečkem, který obsahuje seznam ovoce oddělený čárkou. Výsledkem je nová tabulka, kde je seznam ovoce patřičně rozdělen do odpovídajícího počtu řádků a v každém řádku je pouze jedno ovoce.

5.1.7 Seřazení dat – order

Operace order slouží pro seřazení dat v tabulce a případně i v grafech. Nijak data neupravuje, poskytuje pouze jinou vizualizaci již hotových dat. Data se mohou řadit buďto vzestupně, nebo sestupně. Protože operace order s daty nijak nemanipuluje, aplikuje se v jednom společném kroku s manipulační operací.

5.1.8 Kresba grafu – operace graph

Operace graph slouží pro vykreslení grafů z dat, která vznikla jako výsledek jiné operace. Nijak data neupravuje, poskytuje pouze vizualizaci již hotových dat. U některých typů grafu se bere v úvahu i to, jak jsou data seřazená.

6 IMPLEMENTACE

V kapitolách níže podrobně rozebírám implementaci tří stěžejních částí aplikace, kterými jsou import dat, struktura datového modelu a implementace operací a vizualizací nad daty v aplikaci.

6.1 Import dat do aplikace

Abychom mohli data analyzovat, musíme nejprve nějaká data v aplikaci mít. K tomuto účelu slouží funkce importu dat.

Importovací funkce musí být schopna načíst soubor s čitelnou strukturou dat a tato data ze souboru vhodným způsobem přenést do vlastní databáze tak, aby se nad importovanými daty daly provádět kýžené operace. Je proto nutné navrhnout vhodný postup importu dat a vhodný datový formát pro soubor s daty. Takový formát by měl být pokud možno široce známý a zároveň jednoduchý, aby se daly tyto soubory uživatelem snadno vytvářet a případně editovat. Výsledný formát musí být schopný uchovat jak samotná tabulková data z informačního systému výzkumu, experimentálního vývoje a inovací (dále IS VaVaI), tak i metadata. Ta budou sloužit k popisu povahy importovaných dat. Bude tak např. jasně patřit, jak se mají data do interní aplikace ukládat a o jaký datový typ se jedná.

6.1.1 Výběr datového formátu

Jak bylo zmíněno v předchozích kapitolách, informační systém výzkumu, experimentálního vývoje a inovací nabízí dva různé exporty dat:

1. zazipovaný excelovský sešit (.xls, formát xhtml)
2. zazipovaný soubor tabulky dBase/FoxPro/Paradox/Clipper (.dbf)

Od těchto možností jsem se odrazil a začal je zkoumat.

6.1.1.1 dBase

Formát dBase je druhou z možností exportu dat ze systému VaVaI a jedná se o jeden z nejstarších systémů pro řízení báze dat (SŘBD). Z tohoto důvodu se tento formát přímo nabízel k použití, neboť k němu existují již hotové konektory do programovacích jazyků včetně Javy. Tento formát navíc již obsahuje i metadata, která nesou potřebné informace o názvech sloupečků a jejich datových typech. Zdá se tedy, že datový formát dBase má

všechny vlastnosti, které ve své aplikaci očekávám a proto jsem se rozhodl zkusit s ním dále pracovat.

Postup s datovým formátem dBase je přímočarý. Z informačního systému VaVaI jsem si do tohoto formátu exportoval náhodně vybrané soubory dat, na kterých bych jsem chtěl celou funkčnost otestovat. Dále jsem musel najít vhodný konektor/diver pro jazyk Java. Okamžitě jsem se tak dostal k nejpoužívanějšímu řešení v podobě knihovny xBaseJ [15].

Pro testovací účely jsem vytvořil malý program v Javě, přesně podle příkladu pro čtení DBF souboru [16]. S tímto programem jsem chtěl v praxi vyzkoušet práci s DBF soubory a databází. Bohužel hned po prvním spuštění se objevil problém v podobě výjimky při načítání DBF souboru. Tato výjimka byla vyhozena na níže přiloženém řádku.

```
1 DBF classDB=new DBF("data.dbf");
```

Kód 1: Způsob načtení dat z dBase souboru

Tento řádek má za úkol vytvořit novou instanci konektoru/driveru nad daným DBF souborem. Níže je přiložena výjimka, která na tomto řádku vyskakuje.

```
Exception in thread "main" org.xBaseJ.xBaseJException: Wrong  
Version 48  
    at org.xBaseJ.DBF.openDBF(DBF.java:348)  
    at org.xBaseJ.DBF.<init>(DBF.java:206)  
    at main.Main.main(Main.java:18)
```

Kód 2: Výjimka při načítání dBase souboru

Začal jsem tedy hledat v dokumentaci a diskuzích možné řešení tohoto problému. Nakonec vyšlo najevo, že verze DBF souboru ze systému VaVaI není žádným konektorem podporována a načtení tohoto souboru je tedy velmi problematické a vyžadovalo by vlastní řešení.

Dalšími testy jsem navíc zjistil, že DBF soubor ze systému VaVaI podporuje textové řetězce pouze do maximální délky 254 znaků. Sloupečky s delšími textovými řetězci jsou pak rozděleny do více stejných sloupců s různou částí daného řetězce. Tato nepříjemná vlastnost by velice komplikovala čitelnost a práci s takovými daty.

Testováním práce s DBF soubory jsem tedy odhalil závažné nedostatky tohoto formátu a rozhodl jsem se tento formát zavrhnout a s jeho importem nepočítat. Jako další řešení pro formát dat jsem zvolil excelovský sešit.

6.1.1.2 Excelovský sešit

Excelovský sešit je první možností exportu dat z informačního systému výzkumu, experimentálního vývoje a inovací. Výhodou tohoto řešení je, že se jedná o velice snadné řešení s mnoha možnostmi. Kancelářské balíky, které mají funkce pro čtení i úpravu těchto dat, jsou dnes dostupné pro širokou veřejnost i zdarma a online. Pro uživatele je tedy velmi snadné si data v podobě excelovského sešitu prohlédnout a případně i upravit podle vlastních potřeb. Navíc uživatel může taková data snadno od začátku sám vytvářet, případně získat data v této podobě i z jiných systému než je IS VaVaI. Následně jsem se tedy začal zabývat hledáním vhodné knihovny, která by mi usnadnila čtení dokumentů z programu Excel, souborů s příponou .xls.

Jako vhodná a zároveň nejznámější a nejpoužívanější knihovna se nabízela Apache POI [17]. Tato knihovna nabízí nejen čtení, ale také tvorbu a editaci xls souborů.

Rozhodl jsem se s touto knihovnou opět vytvořit krátký program pro pouhé načtení náhodných dat z IS VaVaI. Testy malých objemů dat proběhly bez problémů, ačkoliv API této knihovny se ukázalo být dosti těžkopádným. Potíže nastaly při větších objemech dat. Konkrétně jsem narazil na problém u exportu z RIV, kde velikost xls souboru přesahovala 30MB, což představovalo 30543 záznamů z RIV. Za žádných okolností se mi nepodařilo soubor takové velikosti načíst. Aplikace vždy narazila na maximální velikost paměti a běh skončil výjimkou `java.lang.OutOfMemoryError: PermGen space`. Jak jsem poté zjistil, knihovna Apache POI tímto velice trpí. Jedná se o známý problém u velkých souborů. Nepomohlo ani jiné nastavení paměti pro JVM.

Toto byl jeden z důvodů, proč jsem nakonec také zavrhl import dat z excelovských souborů s příponou xls. Další důvod byl zcela praktický. Tím byl fakt, že se nejedná o tak snadný formát, jako mají například soubory typu CSV, přičemž ty pro mou aplikaci představují typově stejné úložiště jako soubory XLS, totiž tabulková data. Pro import dat do aplikace jsem tedy zvolil formát CSV, jehož export sice IS VaVaI přímo nenabízí, lze ho však snadno získat konverzí ze souboru XLS.

6.1.1.3 CSV

Po mnoha problémech s předešlými formáty jsem se rozhodl pro nejjednodušší možnou variantu a to formát CSV. Navíc takový formát lze velmi snadno získat prostým uložením z excelovského XLS souboru, jehož export podporuje IS VaVaI.

CSV je obyčejný textový soubor se strukturou navrženou pro data tabulkového charakteru. Zkratka CSV znamená Comma-separated values, což znamená, že jednotlivé hodnoty/sloupcečky jsou v tomto souboru od sebe odděleny čárkou. Každý řádek pomyslné tabulky je pak na vlastním řádku v textovém souboru. Podrobnější informace o možnostech CSV souborů lze najít na webu [18].

Velkou výhodou je, že framework Grails již obsahuje plugin pro velmi snadnou práci s CSV soubory [19]. Ten stačí jednoduše nainstalovat příkazem

```
grails install-plugin csv
```

Tento příkaz se sám postará o stažení potřebné knihovny a nastavení závislostí do projektu tak, aby byla aplikace připravena pracovat s tímto formátem souboru. Součástí zmíněného pluginu je API, jehož dokumentaci najdeme v pramenu [19].

Celý import dat do aplikace bude tedy prováděn skrze CSV soubory.

6.1.1.4 Struktura importovaného CSV souboru

Pro import dat do aplikace jsem zvolil formát souborů CSV. Ten však musí odpovídat jisté struktuře tak, aby aplikace poznala, jak se dané datové sloupcečky nazývají a jakého jsou datového typu. Ke každému sloupečku navíc přidávám slovní popis. Tento slovní popis pak slouží k tomu, aby měl uživatel v aplikaci lepší přehled o tom, co který sloupeček znamená, až bude s těmito daty v aplikaci pracovat a provádět analýzy. Jedná se tedy o metadata, která je potřeba importovat společně se samotnými daty.

Tab. 3. Ukázka struktury CSV souboru pro import dat

název sloupečku	VYSNAZCES	VYSAUTSZND	VYSROKUPL
datový typ sloupečku	VARCHAR(800)	VARCHAR(1024)	INT
slovní popis sloupečku - nápověda	Název výsledku česky	Seznam všech domácích tvůrců výsledku	Rok uplatnění výsledku
data	Aortální regurgitace	Gregor, Pavel; Línková, Hana	2013
data	Důsledky mobility zaměstnanců	Vnoučková, Lucie	2013
data	Jak se dělá mezinárodní reklama?	Boháček, Jiří	2013
data	Mechorosty v učivu středních škol	Novotný, Petr	2013

K účelu importu jsem navrhl strukturu zachycenou v Tab. 3. První šedý sloupec slouží pouze pro názornost a snadnější pochopení struktury dat. Ve výsledném CSV souboru pří-

tomen není. Jak je z tabulky patrné, první řádek souboru nese informace o názvech sloupečků. Druhý řádek udává datový typ hodnot v daném sloupečku. Třetí řádek slouží jako nápověda a popis ke sloupečku pro usnadnění práce uživatele, až s těmito daty bude pracovat. Z tohoto popisu by měl uživatel snadno poznat, k čemu se daný sloupec používá. Další řádky pak už nesou samotná data.

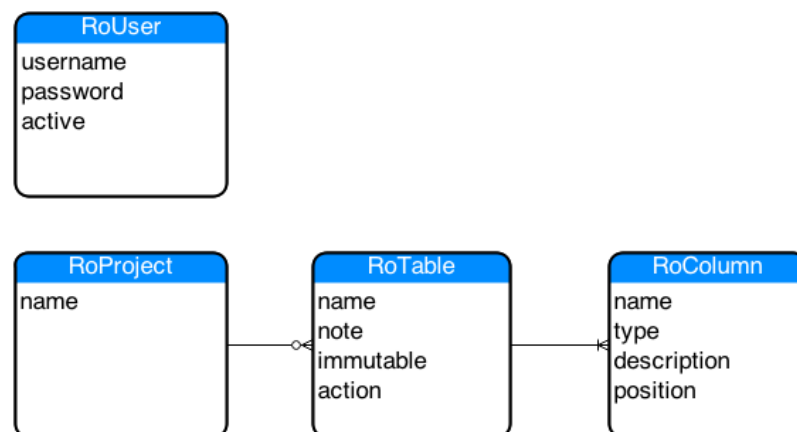
Tabulka Tab. 3 obsahuje malou podmnožinu dat přímo z informačního systému výzkumu, vývoje a inovací a to z evidence výsledků RIV. Pokud bychom tuto tabulku z IS VaVaI exportovali jako XLS soubor a následně ho převedli do importovatelného CSV souboru, bude výsledný CSV soubor při čtení vypadat takto:

1	VYSNAZCES1;VYSAUTSZND1;VYSROKUPL
2	VARCHAR(800);VARCHAR(1024);INT
3	Název výsledku česky;Seznam všech domácích tvůrců výsledku;Rok uplatnění výsledku
4	Aortální regurgitace;"Gregor, Pavel; Linková, Hana";2013
5	Důsledky mobility zaměstnanců;Vnoučková, Lucie;2013
6	Jak se dělá mezinárodní reklama?;Boháček, Jiří;2013
7	Mechorosty v učivu středních škol;Novotný, Petr;2013

Kód 3: Ukázka výsledného CSV souboru a jeho struktury

6.2 Struktura databáze

Základní schéma stěžejní části databázového modelu aplikace je velmi jednoduché a je zachyceno na Obr. 9.



Obr. 9. ER diagram stěžejní části datového modelu

Hlavní části databázového schématu jsou tabulky RoUser, RoProject, RoTable a RoColumn.

RoUser je tabulka pro uchování uživatelů, kteří se mohou do aplikace přihlásit a pracovat s ní.

RoProject je tabulka se seznamem projektů. Každý projekt může obsahovat nula nebo více tabulek (RoTable).

RoTable je rejstříkem všech tabulek, které vznikly importem nebo operacemi s tabulkami v aplikaci. V grafickém rozhraní se pak ukazují právě ty tabulky, které jsou v tomto rejstříku a odpovídají vybranému projektu. Každá z těchto tabulek musí mít vlastní signaturu, tedy definici sloupečků, jejich názvy, datové typy a slovní popis těchto sloupečků, aby byl jejich význam pro uživatele lépe pochopitelný. Pro signaturu tabulky slouží spojení s tabulkou RoColumn.

RoColumn slouží k definici signatury tabulky zmíněné výše. Jde zde o seznam všech sloupečků dané tabulky. Každý sloupeček obsahuje název, datový typ, jeho popis a informaci o jeho pozici, dle které bude sloupeček vykreslen v grafickém rozhraní.

Dalšími fyzickými tabulkami uloženými v databázi jsou všechny tabulky, které vzniknou v aplikaci importem dat, nebo jako výstup aplikované operace. O těchto tabulkách a o tom jak tyto tabulky vypadají (signatura) se aplikace dozví právě díky rejstříku tabulek RoTable. Tyto tabulky mohou libovolně vznikat a zanikat podle toho, jak s nimi pracuje uživatel v aplikaci.

6.3 Implementace operací s daty

Všechny operace pro manipulaci nebo vizualizaci dat v aplikaci, musí mít patřičné nastavení, podle něhož se daná operace může provést. Toto nastavení musí obsahovat především výběr operace, která se bude na data aplikovat a následně pak nastavení této operace. Vždy musí být jasné, jaká se volí operace a jaká je vstupní tabulka. Toto nastavení jsem implementoval jako objekt v JSON notaci, který každá operace přebírá a jenž v sobě může obsahovat všechny potřebné informace. Příklad takového objektu s nastavením lze vidět na ukázce Kód 4.

```
1  {
2    action: 'filter',
3    source: 'fakulty_informatiky',
4    filter: " VYSDRUKOD = 'J' "
5  }
```

Kód 4: Ukázka JSON objektu s nastavením operace

Na ukázce Kód 4 lze vidět nejdůležitější parametr „action“, který je pro všechny operace společný. Tímto parametrem se volí manipulační operace, jež se bude na data aplikovat. Tento konkrétní příklad znamená, že budeme na data aplikovat operaci filter. Jako zdrojo-

vá tabulka poslouží tabulka s názvem „fakulty_informatiky“ a filtr pak vybírá do nově vzniklé tabulky pouze ty řádky, které obsahují ve sloupečku s názvem „VYSDRUKOD“ písmeno J.

JSON notaci jsem vybral z toho důvodu, že se dá velmi dobře skládat i na straně klienta v prohlížeči. V budoucím plánovaném lepším grafickém rozhraní pak půjde snadno pomocí GUI vytvářet nastavení prováděných operací, přičemž se pomocí JavaScriptu bude na pozadí vytvářet právě tento snadno zpracovatelný JSON objekt.

Více o nastavení a parametrech jednotlivých operací uvádím v následujících podkapitolách.

6.3.1 Filter

Operaci filter, která slouží k filtrování vstupních dat, provádím přímo v databázi MariaDB. Jedním dotazem tak v databázi vytvářím novou výstupní tabulku s daty, která vybírám ze vstupní tabulky a omezují pomocí klauzule WHERE.

Aby aplikace mohla operaci provést, potřebuje znát zdrojovou tabulku a zadání filtru. Proto má nastavení této akce dva parametry, popsané v Tab. 4.

Tab. 4. Parametry operace filter

parametr	popis
source	Jméno zdrojové tabulky.
filter	Kritéria filtru zadána pomocí textového řetězce.

Parametr filter je použit jako kritéria pro klauzuli WHERE. Ze zadaných parametrů se postaví patřičný dotaz a v databázi se vytvoří nová tabulka. Zároveň se vytvoří záznam o této tabulce v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má stejnou signaturu jako tabulka vstupní.

Příklad se správným nastavením operace filter lze vidět v kapitole 7.2.1 v ukázce Kód 5.

6.3.2 Union

Operace union slouží pro množinové sloučení dat a provádím ji přímo v databázi pomocí konstruktů UNION a DISTINCT. Tato operace může mít více vstupních tabulek. Musí se však jednat o tabulky stejného typu, tedy tabulky se stejnými sloupečky - signaturou.

Parametry operace union uvádím v tabulce Tab. 5.

Tab. 5. Parametry operace union

parametr	popis
sources	Pole řetězců se jmény vstupních tabulek.

O nově vzniklé tabulce se vytvoří záznam v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má stejnou signaturu jako kterákoliv vstupní tabulka.

Příklad se správným nastavením operace union lze vidět v kapitole 7.4.3 v ukázce Kód 14.

6.3.3 Join

Operace join slouží pro spojení dat z tabulek různé signatury, tedy dat jiného typu, do jedné nové tabulky. Provádím ji přímo v databázi pomocí konstruktů JOIN. Tato operace má dvě vstupní tabulky a jednu výstupní.

Parametry operace join uvádím v tabulce Tab. 6.

Tab. 6. Parametry operace join

parametr	popis
left	Jméno levé zdrojové tabulky. Tato tabulka bude levou částí po výsledném spojení tabulek.
right	Jméno pravé zdrojové tabulky. Tato tabulka bude pravou částí po výsledném spojení tabulek.
leftColumn	Název sloupečku z levé tabulky, podle kterého se bude hledat spojení v pravé tabulce.
rightColumn	Název sloupečku z pravé tabulky, podle kterého se bude hledat spojení v levé tabulce.

O nově vzniklé tabulce se vytvoří záznam v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má novou signaturu, která odpovídá sloučení signatur levé a pravé tabulky.

Příklad se správným nastavením operace join lze vidět v kapitole 7.3.2 v ukázce Kód 9.

6.3.4 Aggregation

Operace aggregation slouží pro agregaci dat podle určitého sloupce, nebo sloupečků a provádím ji přímo v databázi pomocí konstruktů GROUP BY. Tato operace má jednu vstupní tabulku.

Parametry operace union uvádím v tabulce Tab. 7.

Tab. 7. Parametry operace aggregation

parametr	popis
source	Jméno zdrojové tabulky.
projection	Pole nově vzniklých sloupečků, přičemž nový sloupeček se definuje jako pole tří řetězců se signaturou [název nového sloupečku, slovní popis nového sloupečku, výraz pro výpočet nového sloupečku].
groupBy	Pole řetězců s názvy sloupečků, podle kterých se provádí agregace.

O nově vzniklé tabulce se vytvoří záznam v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má novou signaturu, která odpovídá zadanému parametru projection.

Příklad se správným nastavením operace aggregation lze vidět v kapitole 7.2.3 v ukázce Kód 7.

6.3.5 Projection

Operace projection slouží pro jiné uspořádání a zobrazení sloupečků zdrojové tabulky a provádím ji přímo v databázi pomocí specifikace sloupců v patřičném dotazu. Tato operace má jednu vstupní tabulku.

Parametry operace projection uvádím v tabulce Tab. 8.

Tab. 8. Parametry operace projection

parametr	popis
source	Jméno zdrojové tabulky.
projection	Pole nově vzniklých sloupečků, přičemž nový sloupeček se definuje jako pole tří řetězců se signaturou [název nového sloupečku, slovní popis nového sloupečku, výraz pro výpočet nového sloupečku].

O nově vzniklé tabulce se vytvoří záznam v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má novou signaturu, která odpovídá zadanému parametru projection.

Příklad se správným nastavením operace projection lze vidět v kapitole 7.3.1 v ukázce Kód 8.

6.3.6 Explode

Operace explode slouží pro rozdělení vícero hodnot z jednoho sloupečku do více řádků po jedné hodnotě. Tato operace nešla realizovat v databázi a proto je realizována přímo

v Groovy. Kód nejprve sekvenčně vybírá data ze vstupní tabulky, poté je správně rozděljuje a výsledky průběžně ukládá jako řádky do nové tabulky. Tato operace má jednu vstupní tabulku.

Parametry operace explode uvádím v tabulce Tab. 9.

Tab. 9. Parametry operace explode

parametr	popis
source	Jméno zdrojové tabulky.
column	Název sloupečku s vícero daty uloženými v řetězci jako seznam hodnot, oddělovaný vybraným oddělovačem.
delimiter	Oddělovač, podle kterého se rozdělí data ve vybraném sloupečku.

O nově vzniklé tabulce se vytvoří záznam v rejstříku tabulek tak, aby byla viditelná pro uživatele aplikace a ten s ní mohl dále pracovat. Výstupní tabulka má stejnou signaturu jako tabulka vstupní. Výstupní tabulka v zadaném sloupci bude obsahovat pouze jednu z hodnot a ostatní hodnoty z tohoto sloupce budou uloženy v duplicitních řádcích.

Příklad se správným nastavením operace explode lze vidět v kapitole 7.4.1 v ukázce Kód 12.

6.3.7 Order

Tato operace slouží pro seřazení dat podle daného sloupečku, buďto vzestupně, nebo sestupně. Provádí se ve spojení s některou z operací výše. Seřazení nijak neovlivňuje uložení dat, pouze jejich zobrazení v aplikaci. Nemá proto žádnou výstupní tabulku. Je implementováno nad výslednou tabulkou pomocí konstrukturu ORDER BY.

Parametry operace order uvádím v tabulce Tab. 10.

Tab. 10. Parametry operace order

parametr	popis
source	Jméno zdrojové tabulky.
Column	Název sloupečku podle kterého se má řadit + koncovky "desc" v případě sestupného seřazení a "asc" v případě vzestupného seřazení.

Příklad se správným nastavením operace order lze vidět v kapitole 7.2.3 v ukázce Kód 7, kde se seřazují data výstupní tabulky.

6.3.8 Graph

Operace graph slouží pro vykreslení grafu z výsledné tabulky. Provádí se ve spojení s některou z operací výše. Vykreslení grafu nijak neovlivňuje uložení dat, ovlivňuje pouze grafický výstup v aplikaci. Nemá proto žádnou výstupní tabulku.

Parametry operace graph uvádím v tabulce Tab. 11.

Tab. 11. Parametry operace graph

parametr	popis
source	Jméno zdrojové tabulky.
graph	Název typu grafu, který se má vykreslit.
xAxis	Název sloupečku, který bude použit jako zdrojová data pro osu X.
yAxis	Název sloupečku, který bude použit jako zdrojová data pro osu Y.

Příklad se správným nastavením operace graph lze vidět v kapitole 7.2.3 v ukázce Kód 7, kde je z výstupní tabulky vykreslen graf.

7 UKÁZKOVÁ ANALÝZA DAT Z RIV

Tato kapitola obsahuje praktickou ukázkou použití aplikace, která vznikla jako výsledek mé diplomové práce. Názorně vysvětluje tvorbu tří ukázkových statistik, jejichž výsledkem jsou tabulky a grafy.

Analýza obsahuje tyto tři statistiky:

1. Porovnání počtu odborných článků na fakultách informatiky.
2. Zastoupení druhů výzkumných výsledků na UTB FAI.
3. Počet výzkumných pracovníků na UTB FAI dle druhu výsledku.

Na Obr. 10 je zachyceno schéma celé této analýzy v podobě orientovaného grafu. Tento graf podrobně znázorňuje, jak celá ukázková analýza v aplikaci vypadá a jak je v aplikaci reprezentována. Graf obsahuje uzly dvojího typu, tabulky a operace.

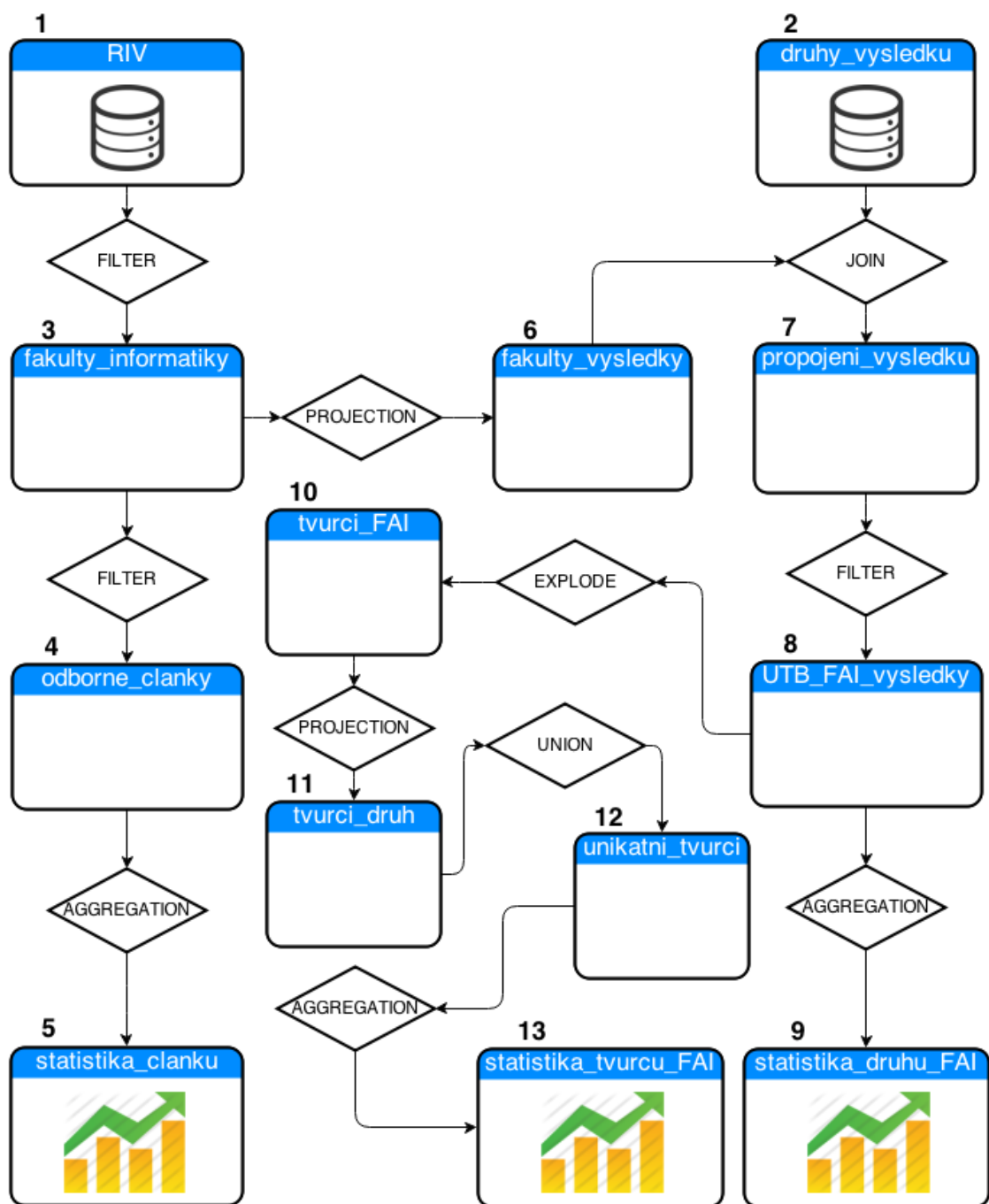
Tabulky mají podobu obdélníku se zakulacenými rohy, který zároveň obsahuje jméno tabulky v aplikaci. Každá tabulka je navíc opatřena číselným identifikátorem, pomocí kterého se v textu na danou tabulku odkazují.

Operace jsou zachyceny kosočtverci s názvem operace, kterou reprezentují.

Hrany uvedeného grafu ukazují tok tabulkových dat a jejich průchod operacemi. Každým průchodem operací vznikne nová tabulka, která může být opět použita jako vstup pro následující operace. Každá operace musí mít jednu nebo více vstupních tabulek a výsledkem je vždy jedna nová tabulka, která vzniká aplikováním dané operace na její vstupy.

Celá analýza je vytvářena nad veřejně přístupnými daty, dostupnými z informačního systému výzkumu, experimentálního vývoje a inovací (IS VaVaI) [1]. Konkrétně analyzují výzkumné výsledky z rejstříku informací o výsledcích (RIV), které lze získat na stránkách tohoto rejstříku [20]. Přípravu a nahrání těchto dat do mé aplikace popisuje samostatná podkapitola níže.

Data, která byla do aplikace importována z IS VaVaI, reprezentují tabulky číslo 1 a 2. Výsledné statistiky jsou pak v tabulkách 5, 9 a 13 ve schématu z Obr. 10.



Obr. 10. Schéma ukázkové analýzy

7.1 Import zkoumaných dat

K provedení analýzy bylo potřeba nejprve importovat zkoumaná data. Tato data jsou uložena v tabulkách 1 a 2, které lze vidět na schématu v Obr. 10.

Protože se jedná o analýzu výzkumných výsledků, vedených v rejstříku informací o výsledcích (RIV), bylo potřeba nejprve data z RIV exportovat do XLS. V analýze mne zají-

mal především rok 2012, proto jsem exportoval všechna data z RIV, která měla konsolidovaný rok uplatnění výsledku 2012 a větší. Další filtr jsem zvolil na druh předkladatele výsledku, kde jsem vybral pouze vysoké školy. Tento export jsem provedl pomocí stránek rejstříku RIV [6].

Dále bylo potřeba exportovaná data upravit do podoby vhodné pro import. Toto jsem provedl stejným způsobem, jaký uvádím v kapitole 6.1. Z exportovaného XLS souboru s daty z RIV, jsem poté vytvořil CSV soubor vhodný pro import do mé aplikace. Tento CSV soubor jsem navíc doplnil o hlavičku s metadaty nutnými k úspěšnému importu, tzn. názvy sloupečků, jejich datový typ a popis. Výsledná hlavička je zachycena v transponované tabulce Tab. 12.

Takto vytvořený soubor jsem již importoval do mé aplikace pomocí funkce „import tabulky z CSV“. Tímto vznikla tabulka číslo 1 s názvem RIV, kterou lze vidět na Obr. 10. Obsahuje seznam výsledků z RIV s rokem uplatnění 2012 a více, a kde je předkladatelem výsledku vysoká škola.

Tab. 12. Hlavička CSV souboru pro import dat z RIV

SLOUPEČEK	TYP	POPIS
VYSIDK	VARCHAR(30)	Identifikační kód výsledku (R01)
DODOZN	VARCHAR(100)	Systemové označení dodavky dat, ve které byl záznam dodan
RESDOD	CHAR(3)	Kód dodavatele (D20)
OBDSBE	INT	Období sberu dat, ve kterém byla data dodana (D03)
VYSDRUKOD	CHAR(1)	Kód druhu výsledku (R05)
VYSPDRKOD	CHAR(3)	Kód poddruhu výsledku (R68, R72, R73, R74)
STUDUVKOD	CHAR(1)	Kód stupně důvěrnosti udaju o výsledku (R12)
VYSROKUPL	INT	Rok uplatnění výsledku (R09)
CR_OBRKOD	CHAR(2)	Kód hlavního oboru výsledku (R04)
VYSJAZKOD	CHAR(3)	Kód jazyka výsledku (R07)
PKLPARNAZ	VARCHAR(254)	Název předkladatele výsledku (D07, D09, D10, D11)
PKLINSICO	INT	IC předkladatele (D06)
PKLORJKOD	VARCHAR(20)	Kód organizační jednotky - předkladatele výsledku (D08)
PKLSIDULI	VARCHAR(48)	Ulice sídla předkladatele výsledku
PKLSIDMIS	VARCHAR(48)	Místo sídla předkladatele výsledku
PKLSIDPSC	VARCHAR(20)	PSC místa sídla předkladatele výsledku
VYSNAZORI1	VARCHAR(800)	Název výsledku v původním jazyce (R06)
VYSNAZCES1	VARCHAR(800)	Název výsledku česky (RN7)
VYSNAZANG1	VARCHAR(800)	Název výsledku anglicky (R08)
AUTPOCCEL	INT	Počet tvůrců výsledku celkem (R10)
AUTPOCDOM	INT	Počet domácích tvůrců výsledku - zaměstnanců nebo studentů předkladatele (R11)

KLISLO1	VARCHAR(1024)	Klicova slova vysledku v anglickem jazyce (R13)
VYSANOORI1	VARCHAR(2048)	Popis vysledku v puvodnim jazyce (R42)
VYSANOCES1	VARCHAR(2048)	Popis vysledku v ceskem jazyce (RN8)
VYSANOANG1	VARCHAR(2048)	Popis vysledku v anglickem jazyce (R46)
VYSAUTSZN_1	VARCHAR(1024)	Seznam vseh dodanych tvurcu vysledku (A02, A03)
VYSAUTSZND1	VARCHAR(1024)	Seznam vseh domacich tvurcu vysledku (A02, A03 pro A04='A')
VYSNAVSEZ1	VARCHAR(800)	Seznam navaznosti vysledku (N01, N03)
ISIUT_KOD	VARCHAR(255)	Kod UT ISI vysledku (R67, R71)
ISS	VARCHAR(13)	ISSN publikacniho kanalu (R14, R82)
ISB	VARCHAR(20)	ISBN publikacniho kanalu (R27)
DOKNAZPRI1	VARCHAR(512)	Nazev publikacniho kanalu (R16, R30, R06)
PERROC	VARCHAR(30)	Svazek periodika (R18)
PERROCCIS	VARCHAR(30)	Cislo periodika v ramci svazku (R19)
PERZEMKOD	CHAR(2)	Kod zeme - statu vydavatele periodika (R17)
EDINAZSVA	VARCHAR(150)	Nazev edice a cislo svazku (R31)
NAKNAZ1	VARCHAR(800)	Nazev nakladatele (R34)
MISVYDKNID	VARCHAR(48)	Misto vydani (R29)
STRROZ	VARCHAR(20)	Rozsah stran vysledku (R20)
POCSTR	INT	Pocet stran vysledku (R21, R33)
KNIPOCSTR	INT	Pocet stran knihy (R69)
MISKONAKC	VARCHAR(48)	Misto konani akce (R54, R60)
ZEMKONAKC	CHAR(2)	Kod zeme - statu konani akce (R61)
DATZACAAC	CHAR(20)	Datum zahajeni akce (R35, R62)
DATKONAKC	CHAR(20)	Datum ukonzeni akce (R63)
TYPAKC	CHAR(3)	Typ akce podle narodnosti ucastniku (R55, R66)
POCUCAAKCC	INT	Celkovy pocet ucastniku akce (R64)
POCUCAAKCZ	INT	Pocet zahranicnich ucastniku akce (R65)
VZPVER	VARCHAR(50)	Verze vyzkumne zpravy (R31)
OBJVZKZPR	VARCHAR(254)	Nazev objednatele vyzkumne zpravy (R34)
DRUPRIVYS	CHAR(1)	Kod druhu pristupu k vysledku (R56)
IDCNOSIC	VARCHAR(32)	Identifikacni cislo nosice (R57)
WWWVYS	VARCHAR(128)	Uplna adresa www stranky s vysledkem (R58)
PATCIS	VARCHAR(64)	Cislo patentu nebo vzoru (R23)
VYDPATNAZD	VARCHAR(254)	Nazev vydavatele patentu nebo vzoru (R24)
MISVYDPATD	VARCHAR(48)	Misto vydani patentu nebo vzoru (R25)
ZEMVYDPAT	CHAR(2)	Kod zeme - statu vydani patentu nebo vzoru (R51)
NAZVLSPAT	VARCHAR(254)	Nazev vlastnika patentu nebo vzoru (R26)
DATREGPAT	CHAR(10)	Datum registrace prihlasky patentu nebo vzoru (R52)
DATPRIPAT	CHAR(20)	Datum udeleni patentu nebo zapisu vzoru (R53)
PATUZMOCH	CHAR(1)	Kod uzemi ochrany patentu (RX2)
PATVYUZPU	CHAR(1)	Zpusob vyuziti patentu nebo vzoru (RN9)
VYSVYUMOZ	CHAR(1)	Druh moznosti vyuziti vysledku jinym subjektem (RN1, RN3)
POZLICPOP	CHAR(1)	Pozadavek na licencni poplatek (RN2, RN4)
INTIDCPRO	VARCHAR(32)	Interni identifikacni kod vysledku prideleny tvurcem (R59)

VYSLOK	VARCHAR(254)	Lokalizace výsledku (R36)
VYSPRMTEC1	VARCHAR(1300)	Technické parametry výsledku (R37)
VYSPRMEKO1	VARCHAR(1300)	Ekonomické parametry výsledku (R38)
NAZVLSVST	VARCHAR(254)	Název vlastníka výsledku (R39)
ICOVLSVST	INT	IC vlastníka výsledku (R40)
TCHKATNKL	CHAR(1)	Kód kategorie výsledku podle nákladu na jeho dosažení (RX3)
PDPCIS	VARCHAR(254)	Číslo předpisu (R75)
ORGKPTPSL1	VARCHAR(1024)	Označení kompetenčního příslušného orgánu (R76)
VYSUZEPLA	CHAR(1)	Kód územní platnosti výsledku druhu H (R77)
ORGCER	VARCHAR(254)	Označení certifikačního orgánu (R79)
VYSCERDAT	VARCHAR(20)	Datum certifikace (R80)
ID_VYS	VARCHAR(54)	Specifikace výsledku v RIV
KONKOD	VARCHAR(14)	Kontrolní kód pro odstranění záznamu z RIV
ID_NVYS	INT	ID normalizovaného výsledku

Stejným způsobem jsem importoval tabulku číslo 2, opět viditelnou na Obr. 10. Tato tabulka reprezentuje pouze číselník výsledků a slouží k překladu klíče druhu výsledku v RIV na název druhu výsledku. Číselník je zachycen v Tab. 13.

Tab. 13. Číselník druhů výsledků v RIV [21]

Kód	Popis
J	Článek v odborném periodiku
B	Odborná kniha
C	Kapitola resp. kapitoly v odborné knize
D	Článek ve sborníku
V	Výzkumná zpráva obsahující utajované informace (takový výsledek lze do RIV vložit pouze v případě, že zpráva obsahuje utajované informace a pole R12 = U), nebo souhrnná výzkumná zpráva
P	Patent
S	Prototyp, uplatněná metodika, funkční vzorek, autorizovaný software, výsledky aplikovaného výzkumu promítnuté do právních předpisů a norem, užitný vzor
Z	Poloprovoz, ověřená technologie, odrůda, plemeno
A	Audiovizuální tvorba
M	Uspořádání (zorganizování) konference
W	Uspořádání (zorganizování) workshopu
E	Uspořádání (zorganizování) výstavy
O	Ostatní výsledky, které nelze zařadit do žádného z výše uvedených druhů výsledku
F	Výsledky s právní ochranou (užitný vzor, průmyslový vzor)
R	Software
N	Certifikované metodiky, léčebné postupy, památkové postupy, specializované mapy s odborným obsahem
H	Poskytovatelem realizované výsledky (výsledky promítnuté do právních předpisů a norem, do směrnic a předpisů nelegislativní povahy závazných v rámci kompetence příslušného poskytovatele)

G | Technicky realizované výsledky (prototyp, funkční vzorek)

7.2 Porovnání počtu odborných článků na fakultách informatiky

V této statistice porovnávám vícero fakult informatiky z různých vysokých škol, podle počtu odborných článků uplatněných za rok 2012. Tímto získáme žebříček fakult podle počtu vydaných odborných článků.

Tuto statistiku zachycuji ve schématu analýzy na Obr. 10 tabulky 1, 3, 4 a 5.

Výslednou statistikou je pak právě poslední tabulka číslo 5, která reprezentuje výsledný žebříček fakult jak tabulkou, tak sloupcovým grafem.

V podkapitolách níže, podrobně rozebírám tvorbu této statistiky krok za krokem.

7.2.1 Výběr fakult informatiky

Tento krok je ve schématu zachycen operací filter mezi tabulkami 1 a 3. Výsledná je pak tabulka číslo 3.

Provedeme výběr pouze takových výsledků z RIV, které splňují zadané požadavky. Tedy výsledky musí být z fakult informatiky a musí být uplatněny v roce 2012. To provedeme vhodně nastavenou operací filter. Nastavení operace uvádím níže.

```
1 {
2   action: 'filter',
3   source: 'RIV',
4   filter: "(PKLPARNAZ like '%informatiky%' or PKLPARNAZ like
5     '%informač%') and VYSROKUPL = 2012"
```

Kód 5: Nastavení operace filter mezi tabulkami 1 a 3

V uvedeném kódu vidíme výběr operace filter (parametr action), volbu zdrojové tabulky (parametr source) a specifikaci kritérií (parametr filter). Fakulty informatiky zde vybírám tak, že název předkladatele výsledku (sloupeček PKLPARNAZ) musí obsahovat řetězec „informatiky“, nebo „informač“. Toto se ukázalo býti dostačujícím. Další kritérium je rok uplatnění výsledku (sloupeček VYSROKUPL), který musí odpovídat hodnotě 2012.

Tímto způsobem jsme získali z tabulky 1 (RIV) novou tabulku číslo 3 (fakulty_informatiky) obsahující pouze výsledky z fakult informatiky s rokem uplatnění 2012.

7.2.2 Výběr odborných článků

Tento krok je ve schématu zachycen operací filter mezi tabulkami 3 a 4. Výsledná je pak tabulka číslo 4.

Protože nás zajímají pouze odborné články, musíme výsledky v tabulce číslo 3 (fakulty_informatiky) omezit pouze na ně. Níže uvádím nastavení operace.

```
1  {
2    action: 'filter',
3    source: 'fakulty_informatiky',
4    filter: " VYSDRUKOD = 'J' "
5  }
```

Kód 6: Nastavení operace filter mezi tabulkami 3 a 4

Tímto způsobem jsme získali z tabulky 3 (fakulty_informatiky) novou tabulku číslo 4 (odborne_clanky) obsahující pouze výsledky z fakult informatiky za rok uplatnění 2012.

7.2.3 Výsledný žebříček fakult podle počtu odborných článků

Tento krok je ve schématu zachycen operací aggregation mezi tabulkami 4 a 5. Výsledná je pak tabulka číslo 5.

Protože v tomto kroku získáme výslednou statistiku, bude tento krok obohacen o seřazení do žebříčku a vykreslení grafu s výsledky.

V tabulce číslo 4 (odborne_clanky) již máme všechna data, která potřebujeme k tvorbě žebříčku. Data je potřeba pouze vhodně agregovat, abychom dostali počty odborných článků k jednotlivým fakultám. To provedeme vhodně nastavenou operací aggregation, která sloučí data podle fakulty a sečte počty článků. Nastavení operace uvádím níže.

```
1  {
2    action: 'aggregation',
3    source: 'odborne_clanky',
4    projection: [
5      ['predkladatel', 'název předkladatele - fakulty', 'PKLPARNAZ'],
6      ['pocet_clanku', 'počet odborných článků na dané fakultě',
7      'count(VYSDRUKOD)']
8    ],
9    groupBy: ['predkladatel'],
10   graph: 'bars',
11   xAxis: 'predkladatel',
12   yAxis: 'pocet_clanku',
13   order: 'pocet_clanku desc'
14 }
```

Kód 7: Nastavení operace aggregation mezi tabulkami 4 a 5, seřazení a vykreslení grafu

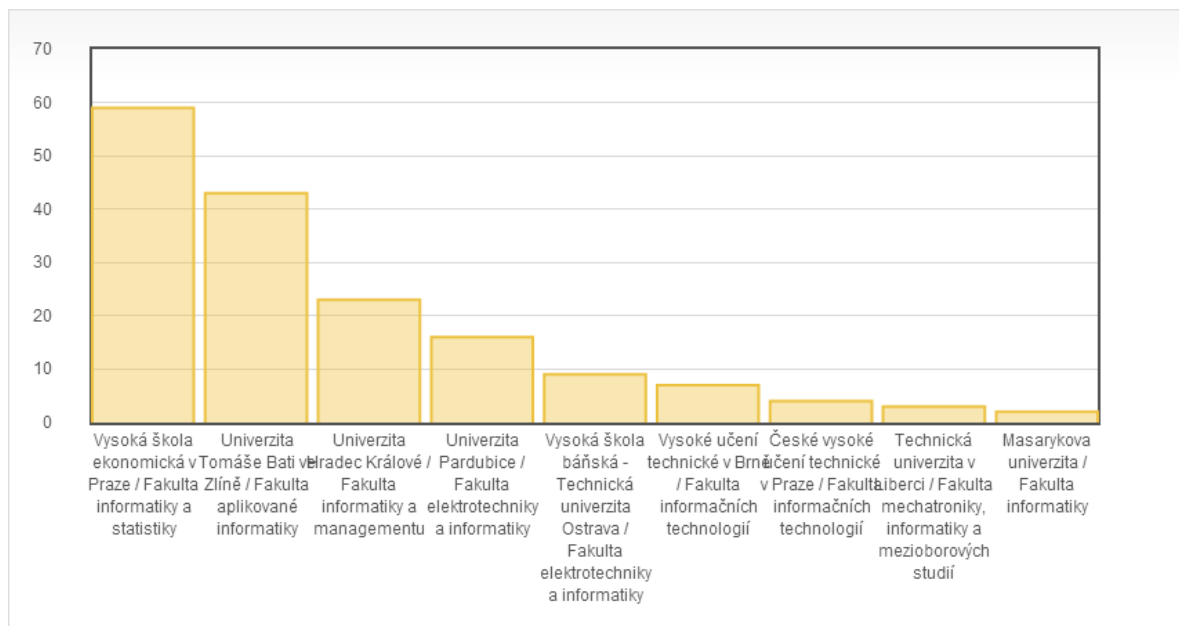
Tímto způsobem jsme získali z tabulky 4 (odborne_clanky) novou tabulku číslo 5 (statistika_clanku) obsahující údaje o počtu vydaných odborných článků ke každé fakultě.

Protože tato tabulka je zároveň výslednou statistikou, jsou do operace přidány příkazy pro sestupné seřazení podle počtu odborných článků a vykreslení sloupcového grafu. Výsledek zachycuje snímek výsledného grafu na Obr. 11 a tabulka Tab. 14.

Tab. 14. Žebříček fakult informatiky podle počtu vydaných odborných článků

#	predkladatel	pocet_clanku
#	název předkladatele - fakulty	počet odborných článků na dané fakultě
1	Vysoká škola ekonomická v Praze / Fakulta informatiky a statistiky	59
2	Univerzita Tomáše Bati ve Zlíně / Fakulta aplikované informatiky	43
3	Univerzita Hradec Králové / Fakulta informatiky a managementu	23
4	Univerzita Pardubice / Fakulta elektrotechniky a informatiky	16
5	Vysoká škola báňská - Technická univerzita Ostrava / Fakulta elektrotechniky a informatiky	9
6	Vysoké učení technické v Brně / Fakulta informačních technologií	7
7	České vysoké učení technické v Praze / Fakulta informačních technologií	4
8	Technická univerzita v Liberci / Fakulta mechatroniky, informatiky a mezioborových studií	3
9	Masarykova univerzita / Fakulta informatiky	2

Výsledek této statistiky je velice příznivý pro fakultu aplikované informatiky Univerzity Tomáše Bati ve Zlíně, která obsadila druhé místo s počtem 43 odborných článků uplatněných v roce 2012. První místo obsadila fakulta z Vysoké školy ekonomické s 59 odbornými články.



Obr. 11. Snímek výsledného grafu s porovnáním počtů odborných článků

7.3 Zastoupení druhů výzkumných výsledků na UTB FAI

V této statistice zjišťují zastoupení a počet jednotlivých druhů výzkumných výsledků uplatněných za rok 2012 na Fakultě aplikované informatiky Univerzity Tomáše Bati ve Zlíně (FAI UTB). Vznikne tak žebříček druhů výzkumných výsledků na fakultě a bude tak patrné, kterých výzkumných výsledků má FAI UTB nejvíce a kterých naopak nejméně.

Tuto statistiku zachycují ve schématu analýzy na Obr. 10 tabulky 1, 2, 3, 6, 7, 8 a 9.

Výslednou statistikou je pak právě tabulka číslo 9, která reprezentuje výsledný žebříček druhů výzkumných výsledků na fakultě.

V podkapitolách níže podrobně rozebírám tvorbu této statistiky krok za krokem.

7.3.1 Vstupní data výsledků

Protože nás zajímají pouze výsledky Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně uplatněné v roce 2012, můžeme k tvorbě této statistiky použít již hotovou tabulku číslo 3 (fakulty_informatiky) ze schématu na Obr. 10. Její tvorbu popisuje kapitola 7.2.1. Tabulka již obsahuje výsledky všech fakult informatiky uplatněné za rok 2012. Fakultu aplikované informatiky Univerzity Tomáše Bati z ní získáme snadno operací filter. To však můžeme provést až v následujících krocích. Na začátek je dobré si data v tabulce trochu omezit tak, abychom zbytečně nepracovali s daty, která nejsou pro tuto statistiku důležitá. Získáme tak nejen lepší přehlednost, ale také větší výkon díky menšímu objemu

dat. Tento krok je ve schématu zachycen operací projection mezi tabulkami 3 a 6. Výsledná je pak tabulka číslo 6. Níže uvádím nastavení operace.

```
1  {
2    action: 'projection',
3    source: 'fakulty_informatiky',
4    projection: [
5      ['predkladatel', 'název školy a případně i fakulty', 'PKLPARNAZ'],
6      ['kod_vysledku', 'kod z číselníku, označující druh výsledku',
7      'VYSDRUKOD'],
8      ['tvurci', 'seznam domácích tvurců výsledku', 'VYSAUTSZND1']
9    ]
10 }
```

Kód 8: Nastavení operace projection mezi tabulkami 3 a 6

Tímto způsobem jsme získali z tabulky 3 (fakulty_informatiky) novou tabulku číslo 6 (fakulty_vysledky) obsahující podmnožinu sloupečků původní tabulky číslo 3.

7.3.2 Propojení klíče výsledku s popisem výsledku

Protože rejstřík výsledků RIV obsahuje pouze klíč druhu výsledku v podobně jediného písmena abecedy, je pro čtenáře velmi obtížné zjistit, o jaký druh výzkumného výsledku se jedná. Abych toto čtenáři statistiky usnadnil, importoval jsem do aplikace číselník druhů výsledků v podobě tabulky číslo 2. Je tedy potřeba pouze správně spojit rejstřík RIV s tímto číselníkem. Tento krok je ve schématu zachycen operací join, která spojuje tabulky 2 a 6 přičemž vzniká výsledná tabulka číslo 7. Níže uvádím nastavení operace.

```
1  {
2    action: 'join',
3    left: 'fakulty_vysledky',
4    right: 'druhy_vysledku',
5    leftColumn: 'kod_vysledku',
6    rightColumn: 'kod'
7  }
```

Kód 9: Nastavení operace join mezi tabulkami 2, 6 a 7

Tímto způsobem jsme získali z tabulky 2 (druhy_vysledku) a tabulky 6 (fakulty_vysledky) novou tabulku číslo 7 (propojeni_vysledku) obsahující propojení výsledků s číselníkem druhu výsledku.

7.3.3 Výzkumné výsledky UTB FAI

Tento krok je ve schématu zachycen operací filter mezi tabulkami 7 a 8. Výsledná je pak tabulka číslo 8.

Protože nás zajímá pouze Fakulta aplikované informatiky Univerzity Tomáše Bati ve Zlíně, musíme výsledky v tabulce číslo 7 (propojeni_vysledku) omezit pouze na tuto fakultu. Níže uvádím nastavení operace.

```
1  {
2    action: 'filter',
3    source: 'propojeni_vysledku',
4    filter: "fakulty_vysledky_predkladatel like 'Univerzita Tomáše
5    Bati ve Zlíně%' "
```

Kód 10: Nastavení operace filter mezi tabulkami 7 a 8

Tímto způsobem jsme získali z tabulky 7 (propojeni_vysledku) novou tabulku číslo 8 (UTB_FAI_vysledky) obsahující pouze výsledky z UTB FAI.

7.3.4 Výsledná četnost druhů výzkumných výsledků na UTB FAI

Tento krok je ve schématu zachycen operací aggregation mezi tabulkami 8 a 9. Výsledná je pak tabulka číslo 9.

Protože v tomto kroku získáme výslednou statistiku, bude tento krok obohacen o seřazení do žebříčku a vykreslení grafu s výsledky.

V tabulce číslo 8 (UTB_FAI_vysledky) již máme všechna data, která potřebujeme k tvorbě žebříčku. Data je potřeba pouze vhodně agregovat, tak bychom dostali výslednou četnost druhů výsledků. To provedeme vhodně nastavenou operací aggregation, která sloučí data podle druhu výsledku a sečte jejich počty. Nastavení operace uvádím níže.

```
1  {
2    action: 'aggregation',
3    source: 'UTB_FAI_vysledky',
4    projection: [
5      ['druh_vysledku', 'druh výsledku', 'druhy_vysledku_popis'],
6      ['vyskytu', 'počet výskytů daného výsledku na fakultě',
7      'count(druhy_vysledku_popis)']
8    ],
9    groupBy: ['druhy_vysledku_popis'],
10   graph: 'bars',
11   xAxis: 'druh_vysledku',
12   yAxis: 'vyskytu',
13   order: 'vyskytu desc'
14 }
```

Kód 11: Nastavení operace aggregation mezi tabulkami 8 a 9, seřazení a vykreslení grafu

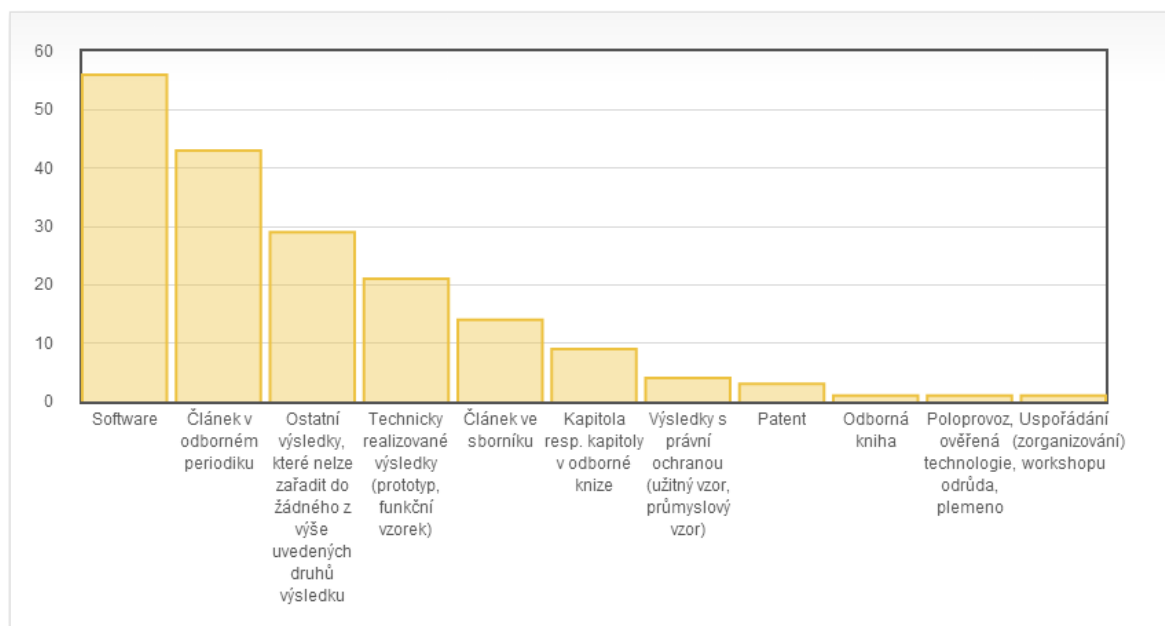
Tímto způsobem jsme získali z tabulky 8 (UTB_FAI_vysledky) novou tabulku číslo 9 (statistika_druhu_FAI) obsahující údaje o četnosti jednotlivých druhů výzkumných výsledku vytvořených na Fakultě aplikované informatiky Univerzity Tomáše Bati ve Zlíně.

Protože tato tabulka je zároveň výslednou statistikou, jsou do operace přidány příkazy pro sestupné seřazení podle počtu odborných článků a vykreslení sloupcového grafu. Výsledek zachycuje snímek výsledného grafu na Obr. 12 a tabulka Tab. 15.

Tab. 15. Žebříček druhů výzkumných výsledků na UTB FAI

#	druh_vysledku	vyskytu
#	druh výsledku	počet výskytů daného výsledku na fakultě
1	Software	56
2	Článek v odborném periodiku	43
3	Ostatní výsledky, které nelze zařadit do žádného z výše uvedených druhů výsledku	29
4	Technicky realizované výsledky (prototyp, funkční vzorek)	21
5	Článek ve sborníku	14
6	Kapitola resp. kapitoly v odborné knize	9
7	Výsledky s právní ochranou (užitný vzor, průmyslový vzor)	4
8	Patent	3
9	Odborná kniha	1
10	Poloprovoz, ověřená technologie, odrůda, plemeno	1
11	Uspořádání (zorganizování) workshopu	1

Výsledek této statistiky říká, že nejvíce výzkumných výsledků má UTB FAI v podobě softwaru, a to rovných 56 výsledků. V těsném závěsu jsou články v odborném periodiku.



Obr. 12. Snímek výsledného grafu s porovnáním počtů druhů výsledků

7.4 Počet výzkumných pracovníků na UTB FAI dle druhu výsledku

Tato statistika má za úkol zjistit, kolik různých výzkumných pracovníků z UTB FAI se podílelo na celkovém počtu výzkumných výsledků, dle druhu výsledku. Získáme tak přehled o tom, kolik výzkumných pracovníků na daných typech výsledků pracovalo. Zahrnuty budou pouze výsledky s rokem uplatnění 2012.

Při porovnání této statistiky s tabulkou Tab. 15 z předchozí kapitoly, můžeme navíc získat přehled o průměrném výkonu jednoho pracovníka, pokud jde o počet výzkumných výsledků v RIV, dle druhu výsledku.

Tuto statistiku zachycují ve schématu analýzy na Obr. 10 tabulky 1, 2, 3, 6, 7, 8, 10, 11, 12 a 13.

Výslednou statistikou je pak právě tabulka číslo 13, která obsahuje počty výzkumných pracovníků dle druhu výsledku.

V podkapitolách níže podrobně rozebírám tvorbu této statistiky krok za krokem.

7.4.1 Vstupní data výsledků

Protože nás zajímají pouze výsledky Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně uplatněné v roce 2012, můžeme k tvorbě této statistiky použít již hotovou tabulku číslo 8 (UTB_FAJ_vysledky) ze schématu na Obr. 10. Její tvorbu popisuje kapitola 7.3.3 a tabulka již obsahuje všechny výzkumné výsledky omezené na výsledky z UTB FAI.

V této statistice započítáváme každé jméno výzkumného pracovníka k danému druhu výsledku pouze jednou. Jména výzkumných pracovníků tvořících daný výsledek jsou ale v RIV uvedena v jediném textovém sloupci, a jsou oddělena středníkem. Toto velice komplikuje zpracování dat, protože se počet jmen nedá jednoduše spočítat a především se nedají odstraňovat duplicitní záznamy jmen, které by statistiku znehodnotily. Proto je nejprve potřeba tato data vhodně předzpracovat tak, aby se z každého jména pracovníka stal samostatný zpracovatelný údaj. To uděláme tak, že pro každé jméno pracovníka vytvoříme samostatný řádek. Tento krok je ve schématu zachycen operací explode mezi tabulkami 8 a 10. Výsledná je pak tabulka číslo 10. Níže uvádím nastavení operace.

```
1 {
2   action: 'explode',
3   source: 'UTB_FAJ_vysledky',
4   column: 'fakulty_vysledky_tvurci',
```

```
5     delimiter: ';'
6     }
```

Kód 12: Nastavení operace explode mezi tabulkami 8 a 10

Tímto způsobem jsme získali z tabulky 8 (UTB_FAI_vysledky) novou tabulku číslo 10 (tvurci_FAI) obsahující v každém řádku jeden výzkumný výsledek, jeho druh a jednoho z jeho tvůrců.

7.4.2 Projekce vstupních dat

Tento krok je ve schématu zachycen operací projection mezi tabulkami 10 a 11. Výsledná je pak tabulka číslo 11.

Protože z dat získaných v předchozím kroku (tabulka 10) musíme ponechat pouze ty řádky, které mají unikátní dvojici tvůrce výsledku a druh výsledku, provedeme v tomto kroku nejprve projekci dat, která ve výsledné tabulce ponechá pouze tuto dvojici sloupečků. To provedeme operací projection, jejíž nastavení uvádím níže.

```
1     {
2     action: 'projection',
3     source: 'tvurci_FAI',
4     projection: [
5     ['tvurce', 'tvurce výsledku', 'fakulty_vysledky_tvruci'],
6     ['vysledek', 'druh výsledku', 'druhý_vysledku_popis']
7     ]
8     }
```

Kód 13: Nastavení operace projection mezi tabulkami 10 a 11

Tímto způsobem jsme získali z tabulky 10 (tvurci_FAI) novou tabulku číslo 11 (tvurci_druh) obsahující v řádcích pouze dvojici druh výsledku a tvůrce výsledku.

7.4.3 Unikátní tvůrce dle druhu výsledku

Tento krok je ve schématu zachycen operací union mezi tabulkami 11 a 12. Výsledná je pak tabulka číslo 12.

Tuto operaci provádíme proto, abychom z předchozí tabulky číslo 11 získali pouze unikátní dvojice sloupečků tvůrce výsledku a druh výsledku. Toto provedeme operací union, která z řádků vstupní tabulky vytvoří množinu a odstraní tak duplicitní řádky. Nastavení operace uvádím níže.

```
1     {
2     action: 'union',
3     sources: ['tvurci_druh'],
4     order: 'tvurce'
```

```
5      }
```

Kód 14: Nastavení operace union mezi tabulkami 11 a 12

Tímto způsobem jsme získali z tabulky 11 (tvurci_druh) novou tabulku číslo 12 (unikatni_tvurci) obsahující v řádcích pouze unikátní dvojice druh výsledku a tvůrce výsledku.

7.4.4 Výsledné počty výzkumných pracovníků dle druhu výsledku

Tento krok je ve schématu zachycen operací aggregation mezi tabulkami 12 a 13. Výsledná je pak tabulka číslo 13.

Protože v tomto kroku získáme výslednou statistiku, bude tento krok obohacen o seřazení do žebříčku a vykreslení grafu s výsledky.

V tabulce číslo 12 (unikatni_tvurci) již máme všechna data, která potřebujeme k tvorbě žebříčku. Data je potřeba pouze vhodně agregovat tak, abychom dostali výsledné počty výzkumných pracovníků dle druhu výsledku. To provedeme vhodně nastavenou operací aggregation, která sloučí data podle druhu výsledku a sečte výzkumné pracovníky. Nastavení operace uvádím níže.

```
1      {
2        action: 'aggregation',
3        source: 'unikatni_tvurci',
4        projection: [
5          ['vysledek', 'druh výsledku', 'vysledek'],
6          ['pocet_tvurcu', 'počet unikátních tvůrců daného druhu výsledku',
7            'count(tvurce)']
8        ],
9        groupBy: ['vysledek'],
10       order: 'pocet_tvurcu desc',
11       graph: 'bars',
12       xAxis: 'vysledek',
13       yAxis: 'pocet_tvurcu'
14     }
```

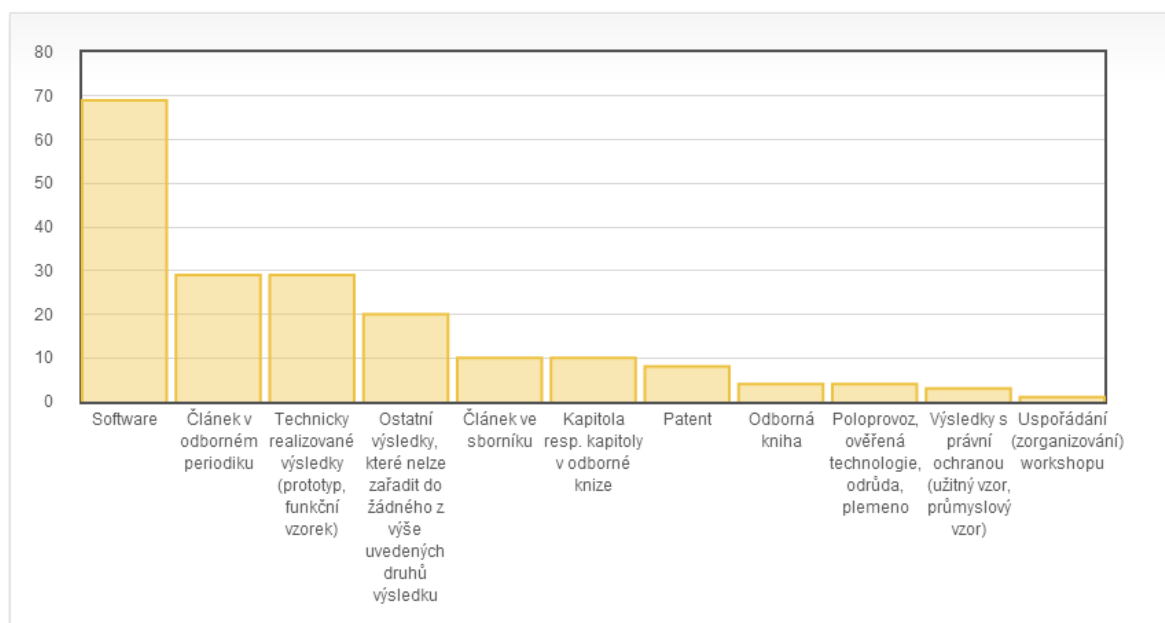
Kód 15: Nastavení operace aggregation mezi tabulkami 12 a 13, seřazení a vykreslení grafu

Tímto způsobem jsme získali z tabulky 12 (unikatni_tvurci) novou tabulku číslo 13 (statistika_tvurcu_FAI) obsahující údaje o počtu tvůrců výzkumných výsledků z UTB FAI, dle druhu výzkumného výsledku.

Protože tato tabulka je zároveň výslednou statistikou, jsou do operace přidány příkazy pro sestupné seřazení podle počtu tvůrců výzkumných výsledků a vykreslení sloupcového grafu. Výsledek zachycuje snímek výsledného grafu na Obr. 13 a tabulka Tab. 16.

Tab. 16. Výsledné počty výzkumných pracovníků dle druhu výsledku

#	vysledek	pocet_tvurcu
#	druh výsledku	počet unikátních tvůrců daného druhu výsledku
1	Software	69
2	Článek v odborném periodiku	29
3	Technicky realizované výsledky (prototyp, funkční vzorek)	29
4	Ostatní výsledky, které nelze zařadit do žádného z výše uvedených druhů výsledku	20
5	Článek ve sborníku	10
6	Kapitola resp. kapitoly v odborné knize	10
7	Patent	8
8	Odborná kniha	4
9	Poloprovoz, ověřená technologie, odrůda, plemeno	4
10	Výsledky s právní ochranou (užitný vzor, průmyslový vzor)	3
11	Uspořádání (zorganizování) workshopu	1



Obr. 13. Snímek výsledného grafu s porovnáním počtu pracovníků

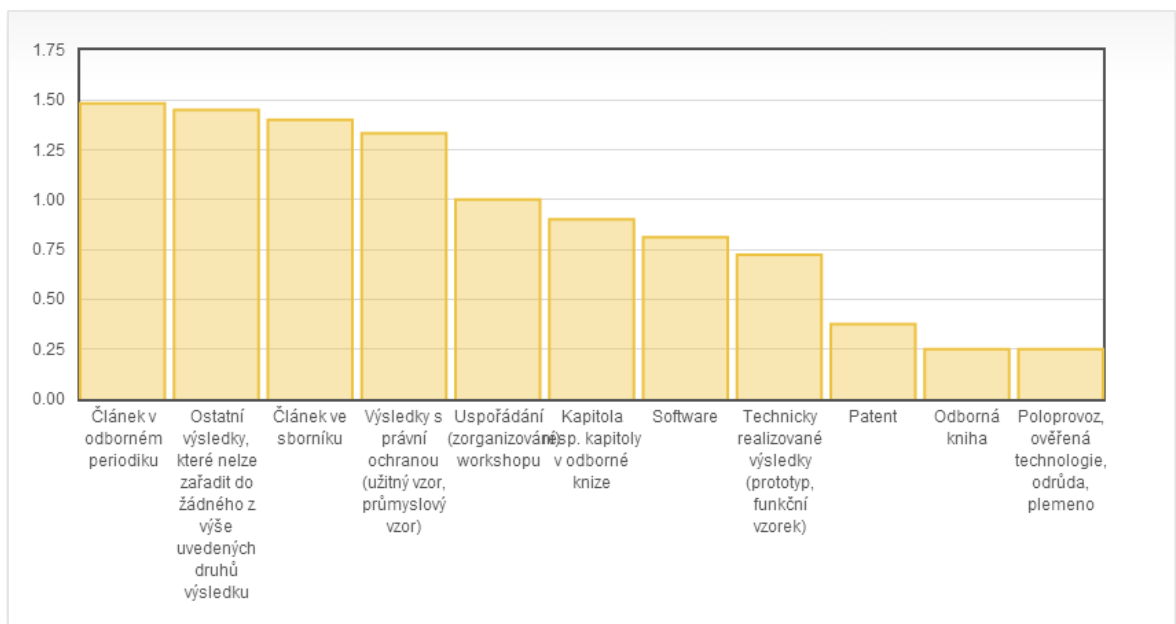
Tato statistika nám říká, že nejvíce výzkumných pracovníků z UTB FAI se podílelo na výzkumných výsledcích typu software, rovných 69 pracovníků. Druhou příčku, s počtem 29 výsledků, obsazují výsledky typu článek v odborném periodiku a technicky realizované výsledky (prototyp, funkční vzorek).

Pro porovnání průměrného výkonu v počtu výsledků výzkumného pracovníka UTB FAI dle druhu výzkumného výsledku, zde navíc uvádím spojení tabulek Tab. 15 a Tab. 16, kte-

ré je možno obdobným způsobem v aplikaci realizovat. Toto spojení zachycuje tabulka Tab. 17 a z této tabulky můžeme např. vyčíst, že každý pracovník podílející se na tvorbě článků v odborném periodiku vytvořil průměrně 1,48 těchto článků. Tabulka je rovněž zachycena v grafu na Obr. 14.

Tab. 17. Tabulka pro porovnání výkonu v počtu výzkumných výsledků

#	druh_vysledku	pocet_tvurcu	vyskytu	vykon
#	druh výsledku	počet unikátních tvůrců daného druhu výsledku	počet výskytů daného výsledku na fakultě	průměrný počet výsledků daného druhu na jednoho tvůrce
1	Článek v odborném periodiku	29	43	1.48
2	Ostatní výsledky, které nelze zařadit do žádného z výše uvedených druhů výsledku	20	29	1.45
3	Článek ve sborníku	10	14	1.40
4	Výsledky s právní ochranou (užitný vzor, průmyslový vzor)	3	4	1.33
5	Uspořádání (zorganizování) workshopu	1	1	1.00
6	Kapitola resp. kapitoly v odborné knize	10	9	0.90
7	Software	69	56	0.81
8	Technicky realizované výsledky (prototyp, funkční vzorek)	29	21	0.72
9	Patent	8	3	0.38
10	Odborná kniha	4	1	0.25
11	Poloprovoz, ověřená technologie, odrůda, plemeno	4	1	0.25



Obr. 14. Průměrný výkon v počtu výsledků jednoho pracovníka FAI UTB dle druhu výsledku

8 ZPŮSOB ZABEZPEČENÍ APLIKACE

8.1 Přihlašování uživatelů

Jeden z velmi důležitých bezpečnostních aspektů mé webové aplikace je přihlašování uživatelů.

Protože se jedná o webovou aplikaci, která tak může být ve své síti snadno dostupná, rozhodl jsem implementovat striktní přihlášení uživatelů. Bez úspěšného přihlášení nelze aplikaci jakkoliv používat, nahlížet do ní, nebo jakkoliv s ní manipulovat. Rozhodl jsem se tak z důvodu obecnosti mé aplikace, která nemusí být použita pouze k práci s otevřenými a všem dostupnými daty z Informačního systému výzkumu, experimentálního vývoje a inovací, ale snadno se také může využívat např. ve firmách pro analýzy nad vlastními uzavřenými daty.

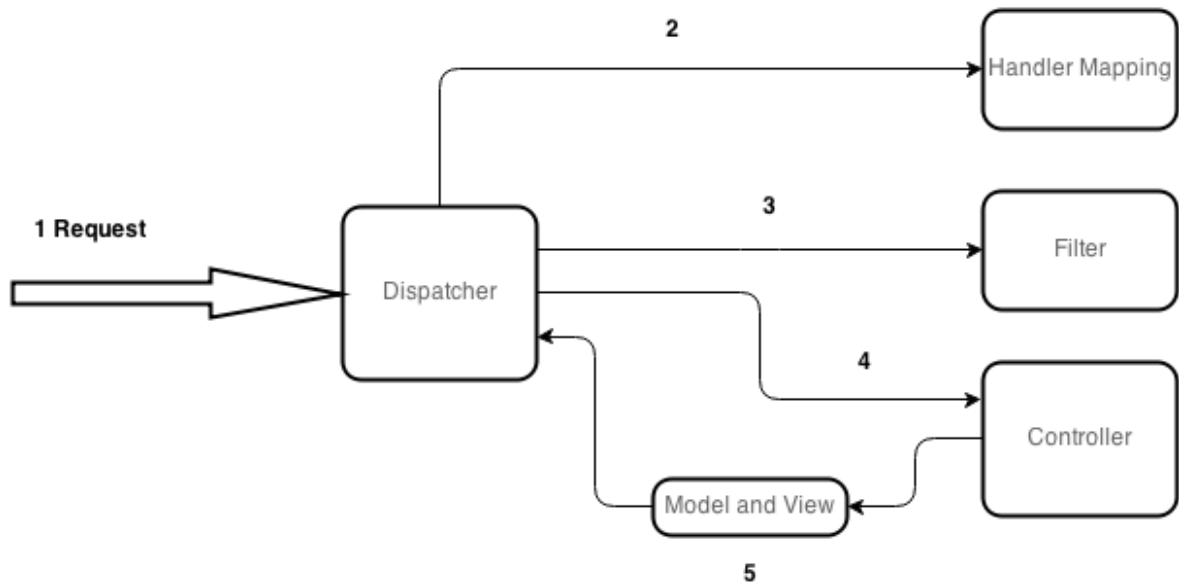
Pokud se tedy nepřihlášený uživatel dostane na jakoukoliv stránku vedoucí do mé aplikace, bude přímo v serverové části aplikace ihned přesměrován a bude mu zobrazena pouze stránka s přihlašovacím formulářem.

Abych minimalizoval chybu způsobenou lidským faktorem při programování, vymyslel jsem způsob, který ve frameworku Grails sám a plně automaticky kontroluje všechny webové operace. Ten před jakýmkoliv provedením operace nejprve ověří, zda je uživatel patřičně přihlášený a pokud ne, ihned ukončí veškerou činnost a uživatele přesměruje na přihlašovací formulář. Nemůže tak dojít k chybě, kdy by programátor zapomněl sám ošetřit dostupnou metodu a přihlášení uživatele by nezkontroloval. Této vlastnosti jsem dosáhl pomocí filtrů, kterými framework Grails disponuje.

Na obrázku Obr. 15 lze vidět posloupnost, jakou se zpracovává HTTP požadavek při příchodu do aplikace. Požadavek v prvním kroku přichází do dispatcheru. V druhém kroku se najde v tabulce vhodný handler, což je ukazatel na kontroler, kterému je požadavek adresován. Ve třetím kroku, ještě předtím než je požadavek předám správnému kontroleru, se na požadavek aplikují dostupné filtry. Až poté, co požadavek úspěšně projde filtry, může být poslán do správného kontroleru, který pak uživateli vrátí odpověď.

Právě ve třetím kroku, v jednom z filtrů, probíhá kontrola přihlášení uživatele. Ta se provádí tak, že filtr nahlédne do interní databáze session, kterou framework Grails poskytuje, a zkontroluje, zda zde existuje záznam potvrzující uživatelovo přihlášení. Pokud ano, je

požadavek předán k dalšímu zpracování. Pokud ne, jsou všechny další operace ihned přerušeny a je vyvoláno přesměrování na přihlašovací stránku.



Obr. 15. Posloupnost operací při zpracování HTTP požadavku v systému Grails

Pro větší bezpečnost jsou navíc uživatelská hesla hashována bezpečným algoritmem SHA-256.

8.2 SQL injection

Proti hrozbě SQL injection je má aplikace chráněna dvěma způsoby. U SQL dotazů u kterých to bylo možné, využívám takzvaných prepared statements. Ostatní složité a dynamické dotazy jsou v aplikaci stavěny pomocí ověřených parametrů.

8.2.1 Prepared statements

Prepared statements je metoda zabraňující útoku typu SQL injection. Jde o předpřipravené šablony SQL dotazů, zkompilované přímo v databázovém serveru. Ten pak pouze na dané místo v již zkompilované šabloně dotazu doplňuje parametry z aplikace. Protože je šablona dotazu dopředu zkompilována a jedná se pak pouze o parametrický dotaz, nelze do něj zařadit nové SQL příkazy. Takto je útočníkovi znemožněno zadat do uživatelského vstupu v aplikaci kód zahrnující SQL injection.

Tuto metodu využívám pro nedynamické SQL dotazy, kde stačí zadat pouze pevný počet a typ parametrů. Především je touto metodou ošetřen i výběr uživatele podle vstupů

z přihlašovacího formuláře. Tím znemožňují útočníkovi použít SQL injection hned při vstupu do aplikace. Bez přihlášení se pak tento útočník nemůže dostat dále.

8.2.2 Stavba dotazů ověřenými parametry

Stavbu dotazů ověřenými parametry se snažím využívat u všech dynamicky skládaných dotazů. Jedná se o ty dotazy, kde je složení celého dotazu ovlivněno uživatelským vstupem. V mé aplikaci jsou to především operace s daty, které uživatel provádí při analýze dat. Těm pomocí uživatelských vstupů nastavuje parametry, které pak mohou být ve výsledném dotazu použity třeba jako název tabulky. Toto je velmi kritické místo z hlediska SQL injection a proto v aplikaci u zadávaných parametrů ověřuji jejich správnost. K tomuto účelu slouží validátor vstupních údajů. Pokud se jedná o parametr, jehož ověření spočívá v dotazu do databáze, provádím tento dotaz právě pomocí prepared statements. Tím je zaručeno, že i samotné ověření parametrů bude odolné proti SQL injection. Dynamický dotaz se pak spouští pouze tehdy, je-li ověřena správnost parametrů.

9 MOŽNÝ ROZVOJ APLIKACE

Protože je aplikace navrhnutá tak, aby se v ní daly vytvářet vizualizace a analýzy nad libovolnými daty tabulkového charakteru a neomezuje se pouze na výzkumné výsledky z Informačního systému výzkumu, experimentálního vývoje a inovací, má aplikace uplatnění u mnohem širší skupiny uživatelů. Těmto uživatelům bude nejvíce záležet na jednoduchosti tvorby vizualizací a analýz. Proto by se dalším prioritním rozvojem aplikace měla stát úprava grafického rozhraní tak, aby bylo toto rozhraní pro uživatele intuitivní a poskytovalo patřičný komfort při práci.

Dalším krokem zlepšujícím využitelnost této aplikace by bylo přidání dalších možností pro vizualizaci dat.

Proto jsem navrhl několik stěžejních bodů pro další rozvoj aplikace, které rozepisují v podkapitolách níže.

9.1 Vizualizace analýzy v podobě orientovaného grafu

Jedná se o grafické zachycení všech tabulek, se kterými uživatel v aplikaci pracuje, v podobě orientovaného grafu, jehož podobu lze vidět např. na Obr. 10 v kapitole 7. Toto by uživateli velmi pomohlo v orientaci ve vytvářené analýze.

9.2 Lepší práce s parametry operací

Tato možnost by měla uživateli umožnit snadno vytvářet nastavení pro jednotlivé operace nad daty. Základním kamenem je našeptávání jmen tabulek a jejich sloupečků. Případně by tabulky mohl uživatel vybírat přímo z grafu analýzy. Tato možnost by velmi usnadnila práci a minimalizovala počet chyb, které může uživatel při definici operace udělat.

9.3 Přidání dalších typů grafů

Pro širší možnosti využití aplikace by se hodilo více typů grafů, které lze z analyzovaných dat vykreslovat. Aplikace by se mohla rozšířit nejen o 2D grafy, ale také o 3D grafy. Pro vykreslování 3D grafů by se daly využít velmi pokročilé možnosti dnešních prohlížečů a jejich podpory HTML5, které nabízí JavaScriptové API WebGL pro nativní zobrazování interaktivní 3D grafiky. Grafy by se tak daly libovolně zvětšovat, zmenšovat, posunovat či otáčet, což by mohlo mít pozitivní vliv na jejich přehlednost.

ZÁVĚR

Jedním z cílů práce je navrhnout a vytvořit funkční prototyp webové aplikace informačního systému pro vizualizaci výsledků výzkumu a vývoje. Od začátku je kladen velký důraz na kvalitu návrhu, aby výsledná aplikace měla široké využití a umožňovala snadno vytvářet různé analýzy a vizualizace výzkumných dat. Díky tomu vznikl v návrhu aplikace takový princip práce s daty, který nejenom že splňuje výše zmíněné požadavky, ale navíc se stal zcela obecným a neomezuje se pouze na data z výzkumu a vývoje. Vznikla tedy aplikace, jejíž pomocí lze snadno vytvářet analýzy a vizualizace jakýchkoliv dat tabulkového charakteru. Tímto se má práce stávat zajímavější a možnosti jejího jsou mnohem širší.

Dalším cílem této práce bylo provést ve výsledné aplikaci analýzu a vizualizaci veřejně přístupných výzkumných dat z Informačního systému výzkumu, experimentálního vývoje a inovací [1]. Výsledky této části práce jsou podrobně rozebrány v kapitole 7 a vyplývá z nich velmi zajímavé srovnání jednotlivých fakult informatiky v počtu vydaných odborných článků. V tomto srovnání obsazuje druhou příčku Fakulta aplikované informatiky Univerzity Tomáše Bati ve Zlíně, na jejíž půdě tato práce vznikla. Dále z této analýzy vyplývá, že nejvíce výzkumných výsledků z této fakulty je v podobě softwaru, přičemž na těchto výsledcích se podílí nejvíce výzkumných pracovníků.

Výsledná aplikace, bude nadále rozvíjena a obohacována. Vývoj bude zaměřen především na intuitivní grafické rozhraní, které poskytne uživateli lepší přehled a umožní ještě jednodušší tvorbu vizualizací.

SEZNAM POUŽITÉ LITERATURY

- [1] Výpočetní a informační centrum Českého vysokého učení technického v Praze. *Informační systém výzkumu, experimentálního vývoje a inovací* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://www.isvav.cz/>
- [2] Rada pro výzkum, vývoj a inovace. Centrální evidence aktivit výzkumu a Informace o předávání údajů. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=552270>
- [3] Rada pro výzkum, vývoj a inovace. Centrální evidence výzkumných záměrů a Informace o předávání údajů. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=979>
- [4] Rada pro výzkum, vývoj a inovace. Evidence veřejných soutěží ve výzkumu, experimentálním vývoji a inovacích a Informace pro předávání údajů. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=987>
- [5] Rada pro výzkum, vývoj a inovace. Centrální evidence projektů výzkumu, experimentálního vývoje a inovací a Informace o předávání údajů. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=975>
- [6] Rada pro výzkum, vývoj a inovace. Rejstřík informací o výsledcích a Informace o předávání údajů. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=986>
- [7] Rada pro výzkum, vývoj a inovace. Číselníky a seznamy platné pro Informační systém výzkumu, experimentálního vývoje a inovací. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-04-10]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=959>
- [8] SpringSource. *Grails* [online]. 2014 [cit. 2014-05-02]. Dostupné z: <http://www.grails.org/>
- [9] People10. Groovy and Grails application development. *People10* [online]. 2014 [cit. 2014-05-01]. Dostupné z: <http://people10.com/blog/groovy-and-grails-application-development-2/>
- [10] Pivotal. *Grroovy* [online]. 2014 [cit. 2014-03-02]. Dostupné z:
- [11] MariaDB Foundation. *MariaDB* [online]. 2014 [cit. 2014-03-12]. Dostupné z: <https://mariadb.org/>
- [12] The jQuery Foundation. *jQuery* [online]. 2014 [cit. 2014-03-12]. Dostupné z: <http://jquery.com/>
- [13] IOLA and Ole Laursen. *Flot* [online]. 2014 [cit. 2014-04-20]. Dostupné z: <http://www.flotcharts.org/>

- [14] *Bootstrap* [online]. 2014 [cit. 2014-02-22]. Dostupné z: <http://getbootstrap.com/>
- [15] SourceForge. *xBaseJ - Java Objects To Read and Write to dBase Files*. [online]. 2014 [cit. 2014-04-01]. Dostupné z: <http://xbasej.sourceforge.net/>
- [16] Reading A Database. *xBaseJ* [online]. 2014 [cit. 2014-04-01]. Dostupné z: <http://xbasej.sourceforge.net/example2.html>
- [17] The Apache Software Foundation. Apache POI - the Java API for Microsoft Documents. *The Apache POI Project* [online]. 2014 [cit. 2014-04-02]. Dostupné z: <http://poi.apache.org/>
- [18] Comma-separated values. *Wikipedia* [online]. 2014 [cit. 2013-04-04]. Dostupné z: Comma-separated values
- [19] HAZLEWOOD, Les. Grails CSV Plugin. *Grails* [online]. 2014 [cit. 2014-04-05]. Dostupné z: <http://grails.org/plugin/csv>
- [20] Výpočetní a informační centrum Českého vysokého učení technického v Praze. Výsledky VaVaI. *Informační systém výzkumu, experimentálního vývoje a inovací* [online]. 2014 [cit. 2014-05-10]. Dostupné z: <http://www.isvav.cz/>
- [21] Rada pro výzkum, vývoj a inovace. Druh výsledku. *Výzkum a vývoj v ČR* [online]. 2014 [cit. 2014-05-10]. Dostupné z: <http://vyzkum.cz/FrontClanek.aspx?idsekce=1395>
- [22] MongoDB, Inc.. SQL to MongoDB Mapping Chart. *MONGODB MANUAL* [online]. 2014 [cit. 2014-04-28]. Dostupné z: <http://docs.mongodb.org/manual/reference/sql-comparison/>
- [23] ZAKHOUR, Sharon. *Java 6: výukový kurz*. Vyd. 1. Brno: Computer Press, 2007, 534 s. ISBN 978-80-251-1575-6.
- [24] ARLOW, Jim a Ila NEUSTADT. *UML 2 a unifikovaný proces vývoje aplikací: objektově orientovaná analýza a návrh prakticky*. Vyd. 1. Brno: Computer Press, 2007, 567 s. ISBN 978-80-251-1503-9.
- [25] PECINOVSKÝ, Rudolf. *Návrhové vzory*. Vyd. 1. Brno: Computer Press, 2007, 527 s. ISBN 978-80-251-1582-4.
- [26] WRÓBLEWSKI, Piotr. *Algoritmy: datové struktury a programovací techniky*. Vyd. 1. Brno: Computer Press, 2004, 351 s. ISBN 80-251-0343-9.
- [27] GRAILS.ORG. *Documentation* [online]. 2014 [cit. 2014-02-04]. Dostupné z: <http://grails.org/documentation>
- [28] *W3schools.com: the world's largest web development site* [online]. 1999 [cit. 2014-02-04]. Dostupné z: <http://www.w3schools.com/>

[29] AUGUSTÝN, Michal. JavaScript očima programátora v2. *Augiho web* [online]. 29.3.2010 [cit. 2014-02-04]. Dostupné z: <http://www.augi.cz/programovani/javascript-ocima-programatora-v2/>

SEZNAM POUŽITÝCH SYMBOLŮ A ZKRATEK

ACID	Databázová transakce, která musí splňovat vlastnosti <ul style="list-style-type: none">• A – Atomicity• C – Consistency• I – Isolation• D - Durability
CEA	Centrální evidence aktivit výzkumu, experimentálního vývoje a inovací
CEP	Centrální evidence projektů výzkumu, experimentálního vývoje a inovací
CEZ	Centrální evidence výzkumných záměrů
CSV	Zkratka pro formát souboru Comma-separated values
FAI	Fakulta aplikované informatiky
HTTP	Hypertext Transfer Protocol
IS VaVaI	Informační systém výzkumu, experimentálního vývoje a inovací
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
MVC	Architektura Model-View-Controller
RIV	Rejstřík informací o výsledcích
SŘBD	Systém řízení báze dat
UTB	Univerzita Tomáše Bati ve Zlíně
VES	Evidence veřejných soutěží ve výzkumu, experimentálním vývoji a inovacích

SEZNAM OBRÁZKŮ

Obr. 1. Schéma systému VaVaI [1]	11
Obr. 2. Architektura a stěžejní součásti frameworku Grails [9]	18
Obr. 3. Ukázka operace filter	27
Obr. 4. Ukázka operace union	28
Obr. 5. Ukázka operace join	29
Obr. 6. Ukázka operace aggregation	29
Obr. 7. Ukázka operace projection	30
Obr. 8. Ukázka operace explode	31
Obr. 9. ER diagram stěžejní části datového modelu	36
Obr. 10. Schéma ukázkové analýzy	44
Obr. 11. Snímek výsledného grafu s porovnáním počtů odborných článků	51
Obr. 12. Snímek výsledného grafu s porovnáním počtů druhů výsledků	54
Obr. 13. Snímek výsledného grafu s porovnáním počtu pracovníků	58
Obr. 14. Průměrný výkon v počtu výsledků jednoho pracovníka FAI UTB dle druhu výsledku	60
Obr. 15. Posloupnost operací při zpracování HTTP požadavku v systému Grails	62

SEZNAM TABULEK

Tab. 1. Seznam funkčních požadavků	23
Tab. 2. Seznam nefunkčních požadavků	24
Tab. 3. Ukázka struktury CSV souboru pro import dat	35
Tab. 4. Parametry operace filter.....	38
Tab. 5. Parametry operace union	39
Tab. 6. Parametry operace join	39
Tab. 7. Parametry operace aggregation	40
Tab. 8. Parametry operace projection	40
Tab. 9. Parametry operace explode.....	41
Tab. 10. Parametry operace order	41
Tab. 11. Parametry operace graph	42
Tab. 12. Hlavička CSV souboru pro import dat z RIV.....	45
Tab. 13. Číselník druhů výsledků v RIV [21].....	47
Tab. 14. Žebříček fakult informatiky podle počtu vydaných odborných článků.....	50
Tab. 15. Žebříček druhů výzkumných výsledků na UTB FAI	54
Tab. 16. Výsledné počty výzkumných pracovníků dle druhu výsledku	58
Tab. 17. Tabulka pro porovnání výkonu v počtu výzkumných výsledků.....	59

SEZNAM UKÁZKOVÝCH KÓDŮ

Kód 1: Způsob načtení dat z dBase souboru.....	33
Kód 2: Výjimka při načítání dBase souboru.....	33
Kód 3: Ukázka výsledného CSV souboru a jeho struktury	36
Kód 4: Ukázka JSON objektu s nastavením operace	37
Kód 5: Nastavení operace filter mezi tabulkami 1 a 3.....	48
Kód 6: Nastavení operace filter mezi tabulkami 3 a 4.....	49
Kód 7: Nastavení operace aggregation mezi tabulkami 4 a 5, seřazení a vykreslení grafu.....	49
Kód 8: Nastavení operace projection mezi tabulkami 3 a 6	52
Kód 9: Nastavení operace join mezi tabulkami 2, 6 a 7	52
Kód 10: Nastavení operace filter mezi tabulkami 7 a 8.....	53
Kód 11: Nastavení operace aggregation mezi tabulkami 8 a 9, seřazení a vykreslení grafu.....	53
Kód 12: Nastavení operace explode mezi tabulkami 8 a 10.....	56
Kód 13: Nastavení operace projection mezi tabulkami 10 a 11	56
Kód 14: Nastavení operace union mezi tabulkami 11 a 12	57
Kód 15: Nastavení operace aggregation mezi tabulkami 12 a 13, seřazení a vykreslení grafu.....	57