Česká zemědělská univerzita v Praze

Provozně ekonomická fakulta

Katedra informačních technologií

Bakalářská práce

Zálohování a archivace dat

Michal Seménka

ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

Katedra informačních technologií

Provozně ekonomická fakulta

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Seménka Michal

Informatika

Název práce

Zálohování a archivace dat

Anglický název

Data backup and archivation

Cíle práce

1) Porovnání zálohovacích a verzovacích systémů se zaměřením na jejich využitelnost v praxi. V rámci srovnání bude hodnocení funkčnosti, možností využití, aspektů týkajících se bezpečnosti a výčet výhod/nevýhod oproti ostatním systémům.

2) Seznámení s problematikou zálohování, verzování a archivace dat.

Metodika

Při tvorbě práce budu převážně vycházet z literatury a odborných článků. Co nejvíce informací ale budu sám prověřovat a porovnávat z více zdrojů.

Pro možnost srovnání systémů všechny z nich důkladně otestuji a sepíši zjištěné skutečnosti.

Harmonogram zpracování

srpen - září 2014 sběr dat
září - listopad 2014 struktura, vlastní tvorba práce
listopad - prosinec 2014 vyhodnocení a závěry
prosinec 2014 - leden 2015 konečná finalizace práce

Rozsah textové části
30-40 stran

Klíčová slova
Verzování, zálohování, archivace, systém, srovnání, zabezpečení

Doporučené zdroje informací

Kastner Aleš: Záloha a archivace, 1. vydání, Praha: GComp, 1994, ISBN 80-85649-21-7
Chacon Scott: Pro Git, Praha: CZ.NIC, 2009, ISBN 978-80-904248-1-4

Vedoucí práce
Havránek Martin, Ing., Ph.D.

Termín odevzdání
březen 2015

Elektronicky schváleno dne 31.10.2014

Ing. Jiří Vaněk, Ph.D.
Vedoucí katedry

Elektronicky schváleno dne 11.11.2014

Ing. Martin Pelikán, Ph.D.
Děkan fakulty

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci "Zálohování a archivace dat" jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne 16. 3. 2015

_____

Michal Seménka

Poděkování

Rád bych touto cestou poděkoval vedoucímu Ing. Martinu Havránkovi, Ph.D za velmi přínosné rady i připomínky, vstřícný přístup a odborné vedení.

Zálohování a archivace dat

Data backup and archiving

Souhrn

Bakalářská práce se zabývá problematikou zálohování a archivace dat. Hlavním cílem je porovnat existující systémy pro správu verzí.

Teoretická část je rozdělena do tří kapitol. Nejprve je pozornost věnována problematice zálohování dat, zahrnující jak rotaci záloh, tak i analýzu možných rizik, která mohou nastat. v další části je řešena problematika archivování. Důraz je kladen především na zásady správného postupu archivace tak, aby data byla použitelná i po několika letech. Ve třetí části je uveden přehled principů fungování tzv. verzovacího systému, který je doplněn výčtem existujících druhů tohoto systému. Současně jsou vysvětleny důležité pojmy, které souvisí se správou repositáře, a současně jsou uvedeny i zásady, které je dobré při práci s ním dodržovat.

Analytická část práce je věnována porovnávání vybraných verzovacích systémů podle nejrůznějších kritérií. Na závěr jsou analyzovány průzkumy nejpoužívanějších systémů pro správu verzí a statistiky vyhledávanosti zde porovnávaných systémů.

Klíčová slova: verzování, zálohování, archivace, systém, srovnání, zabezpečení

Summary

Bachelor‘s thesis deals with data backup and archiving. The prior aim of it is to compare up-to-date systems of version administration.

The theoretical part is divided into three main chapters. At first the attention is paid to issues of the data back-up including the backup’s rotation as well as potential risks that might occur in the process of data storage. Then the principles as well as problems of data archiving are discussed and analysed. The principles of proper archiving are emphasised in order to guarantee their efficient usage in the future. In the third part there are listed principles of version control system functioning which is completed with an overview of its current types applied nowadays. In the same time the important concepts and terms related to the storage management and principles are explained in order to be followed to simplify the work with the system.

The analytical part of the thesis is devoted to comparison of selected versions of controlling systems based on a variety of different criteria. Finally there is an analysis of the most frequently used version-control systems as well as an overview of statistics of the frequency of searching for given systems.

Keywords: revision control, backup, data archiving, system, comparison, security

Obsah

1 Úvod

Pojem zálohování je starý jako samotné počítače. Existují dlouhodobě platné zásady, jak svá data uchovat v bezpečí a ochránit je co nejvíce před nepříznivými situacemi a přitom je mít stále k dispozici pro práci. Těmito zásadami jsou pravidelnost, kompletnost a dostupnost záloh.

Jednou z oblastí využití pravidel zálohování je vývoj softwaru. Pro efektivní vývoj aplikací na profesionální úrovni je již nezbytně nutné použít systém pro správu verzí, tak zvaný „verzovací systém“. Ten právě na principu zálohy funguje. Díky němu tvůrci softwaru získávají nejen možnost spolupracovat s kolegy, ale také schopnost uchovávat historii všech předchozích verzí vývoje a mohou se k nim kdykoliv vrátit.

K dispozici je celá řada nejrůznějších verzovacích systémů. Některé jsou velmi odlišné, jiné jen minimálně. Obecně platí, že neexistuje jeden ideální verzovací systém. Každý je vhodný pro jiné druhy situací. Z pohledu vývojáře, který přijde do firmy, není výběr žádný. Musí se přizpůsobit systému, který je již zaveden. Ve chvíli, kdy se ale teprve zvažuje, který verzovací systém vybrat, je nutné zohlednit mnoho aspektů. S tímto výběrem by měla pomoci právě tato práce, která klíčové informace o často používaných systémech poskytuje přehledným způsobem. Nabízí také aktuální průzkum zájmu o systémy pro správu verzí s prognózou do budoucna.

2 Cíl práce a metodika

2.1 Cíl práce

Hlavním cílem bakalářské práce je charakterizovat a analyzovat zálohování a archivaci dat. Bakalářská práce je rozdělena do dvou částí, teoretické a praktické.

Cílem teoretické části bakalářské práce je na základě studia odborné literatury představit možnosti a zásady zálohování dat. V souvislosti se zálohováním bude popsána i analýza rizik s návrhem optimálního řešení pro vybrané situace. Dále bude v práci vysvětlen princip archivace dat. Část věnovaná systémům pro správu verzí má za cíl nastínit problematiku správy verzí a jednotlivých pojmů s ní spojených. Na závěr teoretické části budou vyjmenována kritéria, podle kterých lze systémy správy verzí odlišovat.

Praktická část má za cíl analyzovat, porovnat a zhodnotit čtyři zvolené systémy správy verzí. Má přinést přehledný a ucelený souhrn, který by usnadňoval často složitý výběr jednoho z těchto systémů. Pro podpoření výběru bude k dispozici i podrobně analyzovaný průzkum nejčastěji používaných systémů pro správu verzí, shodujících se s tímto výběrem. Odhad budoucího vývoje by potom měla poskytnout statistika vyhledávání a prognóza následujícího roku.

2.2 Metodika

Metodika řešení bakalářské práce je pro teoretickou část primárně založena na shromáždění, studiu a analýze odborných informačních zdrojů. Určité informace bylo z důvodu aktuálnosti nutné čerpat z online zdrojů, ale vždy jen po ověření v odborné literatuře. Postup tvorby práce byl od obecného tématu ke konkrétnímu.

Vypracování praktické části předcházelo studium nejen odborné literatury, ale pro potřeby sestavení přehledů bylo nutné též zjistit aktuální podklady z ověřených oficiálních zdrojů jednotlivých tvůrců systémů správy verzí. Dané podklady byly následně analyzovány a uvedeny v práci.

3 Přehled řešené problematiky

Teoretická část bakalářské práce je určena pro vysvětlení pojmů a souvislostí, týkajících se zálohování a archivace dat. Je zde též nastíněn princip správy verzí a fungování systémů pro tento účel vytvořených.

3.1 Zálohování

Zálohování - *„Jedná se o ukládání aktuálních dat na médium, které má minimální vztah s primárním úložištěm.“* (Kříž 2002). Aby bylo docíleno co možná nejnižší závislosti, Kříž (2002) doporučuje zálohovat data na externí média, například CD, DVD, magnetické pásky nebo externí HDD. Díky tomu je vytvořená záloha nezávislá na „zdravotním stavu“ zálohovaného systému.

3.1.1. Pojmy

Sklad – místo určené pro uchovávání zálohovaných dat. Podle potřeby je poté možné ze skladu data obnovovat (Pecinovský 2003).

Zálohovací médium – datové úložiště, které slouží k uložení dat z provozního média. Může to být jiný pevný disk, kompaktní disk, flash disk nebo jiné typy nosičů dat (Pecinovský 2003).

3.1.2. Metody zálohování

Metod, jak zálohovat data, je více. Ty základní vycházejí z plné zálohy a jsou k ní přidávány vzniklé změny. Složitější metody mohou vyžadovat použití specializovaného softwaru a mohou pracovat až na bitové úrovni (Pecinovský 2003).

Úplná (Nestrukturovaná)

Úplná metoda zálohování funguje na principu zálohování všech vybraných souborů, aniž by se bral ohled, zda došlo k jejich změně. Nazývá se též jako „zálohování kopírováním“. Na zálohovacím médiu může být poté uloženo více verzí těchto záloh za určitou dobu, přitom ale není vyloučeno, že většina souborů bude duplicitní s ostatními verzemi. Pro běžného uživatele při malém objemu dat může dostačovat, ale pro řešení zálohování ve firmách již nedostačuje (Pecinovský 2003).

„Tato metoda je poměrně pracná a nepraktická, proto uživatelé, kteří ji používají, začínají zálohování opomíjet.“ (Pecinovský 2003)

- Výhody:
  - Jednoduchost
  - Není třeba žádný speciální SW
  - Pouze je vytvořena kopie, na to uživateli postačují nástroje operačního systému.
- Nevýhody:
  - Často zbytečná duplicita dat a s tím spojené zvýšení požadavků na kapacitu média určeného pro zálohování.
  - Větší objem dat je spojen i s delší dobou kopírování (Pecinovský 2003).

Úplná a přírůstková (Incremental)

Přírůstková záloha eviduje změny, které proběhly od poslední přírůstkové zálohy.

Při této metodě je v prvním kroku vytvořena úplná záloha všech vybraných dat. Poté je aktualizována přírůstkovým způsobem. Tedy, vůči úplné záloze se porovnají aktuální soubory a ty, které byly změněny, se uloží do přírůstkové zálohy. Při každém dalším ukládání je postup stejný, ale je srovnávána aktuální verze s poslední vytvořenou přírůstkovou zálohou.

Pro obnovení dat je nutné nahrát úplnou zálohu a poté postupně projít a přidat všechny do té doby vytvořené přírůstkové zálohy (Kříž 2002).

V každé ze záloh se zahrnuje pouze rozdíl oproti záloze předešlé.

- Výhody:
    - Vytvoření zálohy je kromě prvotní úplné zálohy velmi rychlé díky tomu, že se porovnává pouze vůči poslední přírůstkové záloze, nebo v prvním kroku vůči úplné záloze.
    - Oproti úplné záloze není třeba po každé editaci kopírovat a uchovávat celou novou verzi úložiště, ale pouze evidovat změněné soubory. Díky tomu jsou i požadavky na úložný prostor nižší.
- Nevýhody:
    - Obnovování dat je složitější než např. u rozdílové zálohy.
    - Při ztrátě jedné ze záloh jsou všechny zálohy po ní následující nepoužitelné (Pecinovský 2003).

Úplná a rozdílová (Differential)

Rozdílová záloha porovnává vždy najednou všechny změny, které proběhly od úplné zálohy, a ty eviduje.

Při této metodě je též v prvním kroku vytvořena úplná záloha všech vybraných dat. V každém dalším kroku je vytvořena nová, která vystihuje opět aktuální změny vůči úplné záloze. Proces zálohování je proto pomalejší, než u přírůstkové metody.

Naopak proces obnovení představuje především při velkém počtu různých verzí urychlení, protože všechny verze jsou na sobě nezávislé. Stačí proto obnovit pouze úplnou zálohu a k ní nahrát zvolenou rozdílovou (Kříž 2002, Storage.cz 2012).

Stejně jako přírůstková záloha je kromě prvotního vytvoření oproti úplné metodě rychlejší při postupném evidování změn. Lze ji proto použít též ve firmách během pracovního týdne.

- Výhody:
    - Rychlejší obnova dat
- Nevýhody:
    - Pomalejší ukládání rozdílové zálohy
    - Vyšší nároky na úložiště dat (Kříž 2002, Kastner 1994)

3.1.3. Rotace záloh

Rotace záloh souvisí s používáním fyzických úložných médií k záloze a archivaci dat. Obecně slouží k tomu, aby zálohy pokrývaly individuální potřeby uvažovaného subjektu. Tím může být domácnost, nebo firma. Je důležité, aby i pro případ havárie jednoho nebo více zálohovacích médií nedošlo k úplné ztrátě dat. Například pro firmu je nutné individuálně posoudit, jak moc cenná data pro ni jsou, jak často se mění a na jak dlouhou dobu je potřeba uchovávat podrobné zálohy. Podle toho je vhodné definovat režim zálohování.

Uvažuje se i to, zda jsou zálohovací média rovnoměrně opotřebovávána, zda jsou zálohy uchovávány na dostatečně různorodých médiích a odlišných místech. Obecně nejvhodnější je, když se zvolí takové nastavení rotace, aby byla uchovávána data za co nejdelší možné období, co nejobsáhlejší a na více druzích datových médií.

Jako bezpečnostní pojistka může být vhodné provést daný den dvakrát zálohy jednotlivých cyklů a jednu z kopií odvézt úplně do jiné lokace. Je dobré též jednou za čas cvičně zkusit obnovit data ze záloh, aby se ověřila čitelnost a integrita záloh a současně nacvičil postup obnovy (tzv. crash test). Tím se velmi zkrátí čas případné obnovy po havarijní situaci (Vít 2013).

Možné používané typy rotace jsou:

Druh rotace Dědeček-Otec-Syn (anglicky GFS)

Vhodný je primárně pro zálohování na magnetickou pásku, ale využít jej lze dobře i u ostatních datových médií. Jsou definovány tři sady záloh, které mohou být zvoleny,

jako například denní, týdenní a měsíční. Vždy ale záleží na analýze a konkrétních potřebách uvažovaného subjektu.

- (Syn) Denní záloha se běžně provádí jako přírůstková a je realizována v určité, předem definované dny.
- (Otec) Týdenní záloha může být úplná. Určuje se podle procentuálního množství dat, které se touto zálohou pokryjí.
- (Dědeček) Měsíční záloha se používá jako úplná. Navíc se často s tím tato záloha může duplikovat a přesunout na jiné místo jako bezpečnostní kopie.

Tabulka 1: Druh rotace - Dědeček, otec, syn

	Pondělí	**Úterý**	**Středa**	**Čtvrtek**	**Pátek**	**Sobota**	**Neděle**
1. týden	Den 1a	Den 1b	Den 1c	Den 1d	Týden 1	X	X
2. týden	Den 1a	Den 1b	Den 1c	Den 1d	Týden 2	X	X
3. týden	Den 1a	Den 1b	Den 1c	Den 1d	Týden 3	X	X
4. týden	Den 1a	Den 1b	Den 1c	Den 1d	Měsíc	X	X

Zdroj: (Junek 2013), vlastní zpracování

Druh rotace Hanojská věž

Dle Junka (2013) je tato metoda založena na stejném principu jako známá hra. Použije se několik setů médií. Po uvedení příkladu se uvažuje pět druhů.

- Médium A - Využité každý druhý den
- Médium B - Využité každý čtvrtý den
- Médium C - Využité každý osmý den
- Médium D a E - Využité každý šestnáctý den

Tabulka 2: Druh rotace - Hanojská věž

Den	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Médium	A		A		A		A		A		A		A		A	
		B				B				B				B		
				C								C				
								D								
																E

Zdroj: (Junek 2013), vlastní zpracování

Druh rotace Round-robin

Je jedním z nejstarších způsobů rotace záloh. Každý den je zálohování časově náročné, protože se provádí úplná záloha. Je dobré pro každý den použít samostatné datové médium. K dispozici jsou pouze zálohy týden nazpět.

Tabulka 3: Druh rotace - Round-robin

	Pondělí	Úterý	Středa	Čtvrtek	Pátek	Sobota	Neděle
1. disk	X					-	-
2. disk		X				-	-
3. disk			X			-	-
4. disk				X		-	-
5. disk					X	-	-

Zdroj: (Junek 2013), vlastní zpracování

3.1.4. Analýza rizik

Analýza rizik je činnost, při níž se analyzují situace, které mohou nastat, jejich dopad na informační systém a způsob jejich řešení. Data jsou při ní rozdělována na důležitá a méně důležitá a podle toho klasifikována. Data, která je nutné rychle obnovit v případě havárie, se zálohují tak, aby byla rychle dostupná. *„Podobně jako při pojištění je třeba počítat vždy s tou nejhorší alternativou, i když je absurdní si představit, že by současně řádila magnetická bouře, došlo k požáru následně uhašenému povodní a při tom si pro počítač přišli zloději.“* (Pecinovský 2003)

Převedením na model firmy rozhodující o způsobu zálohování dat by rizika a jejich eliminace mohla být například tato:

- Neúmyslné vymazání dat
    - Jako opatření proti následkům neúmyslného vymazání dat Pecinovský (2003) doporučuje data zálohovat denně, případně častěji podle potřeby. Je nutné vždy posoudit danou konkrétní situaci a podle ní přizpůsobit interval zálohování. Pro tento případ stačí zálohovat klidně na stejný disk.
- Porucha pevného disku
    - Havárie pevného disku je již problematičtější záležitost. Je typické, že k ní dojde ze dne na den, často bez předchozího varování, a disk se již nerozběhne. Poškození může být mechanického typu, vlivem počítačových virů nebo elektromagnetického impulsu. Zde již tedy nestačí zálohu ukládat na stejný fyzický disk, ale například na diskové pole. Běžně se pevné disky neopravují, pouze se vyměňují. V případě, že došlo jen k poruše elektroniky pevného disku, existuje ještě šance na záchranu dat, avšak oprava může být i dražší než nový disk. Záleží na důležitosti dat, zda se to vyplatí. Proto je výhodnější předcházet situaci, kdy by bylo jedinou možností obnovovat data z poškozeného disku (Pecinovský 2003, Leixner 1993).

- Krádež nebo porucha počítače
    - Krádež nebo porucha celého počítače znamená pro uživatele zásadní problém. Nejde jen o nepříjemnost s pořizováním nového stroje a jeho novou instalací. Znamená to i vysokou pravděpodobnost ztráty dat, pokud nebyla patřičně zálohována. Zde je již nutné zálohovat na médium, které je umístěno vně počítačové skříně a nejlépe mimo objekt. Jakákoliv opatření totiž mohou vždy pouze snížit riziko, že ke krádeži nebo poruše dojde. Možností je využití NAS serveru, externího HDD nebo cloudového úložiště (Pecinovský 2003).
- Krádež nebo ztráta archivních dat
    - Krádež nebo ztráta archivních dat je podobná předchozímu případu. Oproti počítači je přenosné médium pro zloděje ještě větším lákadlem. Způsobů ochrany proti odcizení archivních dat je více. Je nutné data uchovávat odděleně, nejlépe v trezoru v jiné budově. Proti zneužití dat je dobré chránit data šifrováním (Pecinovský 2003).
- Požár, zemětřesení, povodeň nebo jiná událost ohrožující budovu
    - Živelným pohromám nelze zabránit, ale lze se na ně připravit a zmírnit jejich dopady. Důležitá je tedy prevence. V úvahu se musí vzít i varianta, kdyby byla zničena budova, kde jsou zálohy uchovávány. Jako minimální ochranu proti ztrátě dat Pecinovský (2003) i Leixner (1993) doporučují uchovávat alespoň jednu úplnou zálohu mimo budovu, kde je standardně uložena.

Při zjišťování možných následků uvažovaných situací se pro výpočet používají údaje z minulosti nebo výsledky prováděných simulací.

Podle výsledků se poté provádí návrh zálohování jednotlivých částí systému. Způsob zálohování závisí na:

- Důležitosti jednotlivých dat a urgentnosti jejich zpřístupnění.
- Cenovém limitu jedince nebo firmy, která zálohování plánuje.

- Riziku, které bylo přiřazeno určitým možným událostem.

  Analýza rizik se provádí ve spolupráci s uživateli (Leixner 1993).

Odlišné může být plánování zálohy pro běžná soukromá data v podobě fotografií, videí, textových dokumentů a klíčových dat různých společností. Například ve společnosti vyvíjející software bude klíčové vytvářet průběžnou pravidelnou zálohu zdrojových kódů. V jiných to jsou audiovizuální data, pouhé texty nebo binární data. Někde potřebují zálohovat primárně pouze účetnictví. Toto je jeden z prvních aspektů, podle kterého se rozhoduje o frekvenci a typu zálohování pro jednotlivé kategorie dat (Leixner 1993).

3.1.5. Komprimace záloh

Komprimace je postup zakódování datových souborů do takové konzistence, kdy se sníží velikost souboru oproti původnímu stavu. Opakem k tomuto procesu je dekomprimace, při které se získá zpět původní tvar. Nesmí přitom dojít k poškození dat.

Výhodou komprimace je, že šetří místo na pevném disku. Pokud je třeba data někam přesouvat, přínos je i ve zkrácení doby. Umožňuje též rozdělení finálního balíku dat do několika souborů (Kříž 2002).

Pojmy

Adaptivní metody – Komprimační metody určené především pro použití na neurčitém druhu dat. Nemají žádné předem definované slovníky. Pro každý komprimovaný soubor jsou pomocí algoritmů budovány slovníky vlastní. Dosahuje se při nich maximálního možného kompresního poměru (Kříž 2002).

Neadaptivní metody – Komprimační metody určené především pro použití na specifickém druhu dat. Mají dopředu předdefinované řetězce znaků nebo slovníky, u kterých je pravděpodobné, že se budou v datových souborech často vyskytovat. Při použití na vhodném druhu dat lze dosáhnout dobrého kompresního poměru a času potřebného pro kompresi i dekompresi (Kříž 2002).

Archiv – *„Soubor obsahující komprimovaná data souborů a jejich strukturu adresářů.“* (Kříž 2002). Tyto archivy jsou tvořeny pomocí komprimačních nebo archivačních programů. Jejich představiteli jsou například RAR (jeho verze pro Windows

WinRAR), ZIP (jeho verze pro Windows WinZIP) a další. Slouží pro potřeby archivace, tedy dlouhodobého uskladnění, nebo též pro zmenšení objemu dat pro účel přenosu (Kříž 2002, Pecinovský 2003).

3.2 Archivace

Archivace je způsob uložení dat. Principem je, že tato data jsou v úložišti skladována na delší období, třeba i desetiletí. Oproti zálohování zde není kladen přílišný důraz na rychlost obnovení dat z archivu. Musí být ale zajištěna primárně jejich vyhledatelnost.

3.2.1. Princip archivace dat

Zálohovací aplikace mají tendenci uchovávat data v uzavřeném formátu, což může způsobovat problémy při jejich dlouhodobém ukládání. Například podnik, který vytvoří archiv a chystá se ho dlouhodobě skladovat, musí počítat s následující skutečností. Během let pravděpodobně dojde vlivem aktualizace nebo úplné obměny jím používaných programů a hardware ke znečitelnění dat. Nebude totiž k dispozici nástroj, pomocí kterého by archivovaná data mohla být přečtena. Účinným způsobem jak předejít nečitelnosti je správný výběr aplikací speciálně určených pro archivaci, které ukládají soubory do archivu v nativním formátu. Tedy v typickém otevřeném formátu pro danou aplikaci, u kterého je předpoklad, že bude dlouhodobě podporován. Příkladem mohou být soubory ve formátu .TXT nebo .XML (Adshead 2009).

Zásady pro archivaci podle Leixnera (1993):

- Archivovat pouze soubory v definitivním tvaru, ty, u nichž byla provedena všechna chystaná zpracování obsahu a jejichž data se nesmí ztratit.
- Vytvářet minimálně dvě kopie všech archivů. Tyto kopie by měly být na různých místech, případně i na odlišných datových médiích.
- Je nezbytné vést o archivech přesnou evidenci v elektronické verzi nebo i v písemné, tj. informace o tom, kdy byla která archivace provedena a na které úložné médium.

Vhodnou dobou pro archivování je čas při úplném zálohovacím cyklu. Odkládáním evidovaných kopií sad datových médií lze snadno uskutečnit archivaci při dodržení pravidel popsaných výše. Soubory takto archivované je poté možné bez obav vymazat z původního místa na provozním médiu, kde jen zabíraly prostor (Leixner 1993).

3.3 Verzovací systém (VCS)

Cílem verzovacího systému, neboli systému správy verzí, je uchování historie projektu ve vyhrazeném úložišti, tzv. repositáři. Systém umožňuje nejen tuto historii prohlížet, ale též vracet stav vývoje konkrétních souborů na jejich předchozí verze. Záznam změn je prováděn v průběhu vývoje projektu tak, že jsou uloženy zvolené editace včetně komentáře, vystihujícího danou úpravu.

Druhým podstatným cílem je umožnění spolupráce více spolupracovníků na jednom projektu. Díky verzovacímu systému je řízeno zpracování změn a synchronizace uvnitř skupiny. Při srovnání dvou paralelních změn v souboru dokáže tyto změny spojit do jednoho souboru. Pokud to podle předdefinovaných postupů možné není, nastává konflikt, který je poté dále individuálně řešen. To bude blíže specifikováno v podkapitole Sloučení.

Dále je obvykle systém správy verzí vybaven funkcí pro uchovávání více paralelních řešení jednoho projektu. Tato funkce se nazývá větvení, podrobněji bude vysvětlena v podkapitole Větvení (Dušička 2015, Kučera 2011, Reisdorph 1999).

3.3.1. Pojmy

Changeset – Seznam všech změn souborů mezi verzemi v repositáři, zahrnující, kdo a kdy změnu vytvořil a komentář. Každý prvek z této množiny změn musí být elementární, dále nedělitelný (Sink 2011).

3.3.2. Commit

Aplikování provedených změn z pracovní kopie do repositáře.

Všechny moderní nástroje pro správu verzí provádí tyto operace atomicky, neboli všechny najednou. Nezávisle na tom, kolik je v changesetu uložených změn, vždy se

zaznamenají buď všechny, nebo žádná. Není možné, aby repositář při zaznamenávání změn skončil v polovině provedených operací. Díky tomu je zajištěna integrita úložiště.

Je typické, že jsou jednotlivé commity označeny komentářem pro snadnější odlišení a zdůraznění místa nebo typu provedené editace (Sink 2011).

3.3.3. Větvení (Branch)

Větvení je operace systému správy verzí, pomocí které je duplikován určitý zdrojový kód nebo přímo adresářová struktura. Díky tomu lze provádět vývoj již rozpracovaného kódu paralelně a více způsoby. Podle potřeby je možné nové větve vytvářet nebo slučovat.

Větve dále umožňují realizovat nové postupy v kódu nebo též postupně přecházet na novější technologie. Vytvořením nové větve se zajistí nejen to, že test nové technologie poběží současně s vývojem kódu pro ty starší, ale díky nástrojům pro spojení větví je lze po čase sloučit opět dohromady. Pokud se ale ukáže, že větev vývoje by byla například neperspektivní, stačí ji pouze ukončit. Bez použití větvení by bylo nutné se vracet zpět v projektu do bodu, kdy ještě byl kód v pořádku (Sink 2011).

3.3.4. Sloučení (Merge)

Sloučení je operace, která umožňuje spojit dohromady práci provedenou na stejných datech repositáře více lidmi. Může probíhat na úrovni dvou verzí jednoho souboru nebo přímo na celém projektu. Jako první proběhne operace porovnání. Ta odhalí všechny změny oproti výchozímu stavu a podle toho se poté rozhodne o dalším postupu.

Konflikt nastává ve chvíli, pokud dva lidé provedli editace odporující si navzájem na stejném řádku stejného souboru.

Když při slučování dojde ke konfliktu, situace vyžaduje zásah člověka pro rozhodování, jak konflikt řešit.

Pokud se při porovnání neodhalí žádný konflikt, sloučení probírá relativně automaticky a výsledkem je jeden soubor obsahující všechny provedené změny obou porovnávaných verzí (Sink 2011).

3.3.5. Repositář

Při práci s repositáři Sink (2011) doporučuje:

- Spustit kontrolu změn (diff) před commitováním
- Uchovávat repositáře co nejmenší
- Třídit logicky commity
- Vysvětlit co nejpodrobněji, pomocí komentáře, prováděný commit
- Nevkládat do repositáře dynamicky měněné soubory
- Neničit strukturu složek
- Používat poznámky jak jen to jde
- Před commitem vždy zkontrolovat změny v repositáři
- Nemazat nic z historie repositáře
- Promazávat starý nepoužívaný kód
- Používat zámky editace jen střídmě
- Testovat automatizovaně kód po každém commitu

3.3.6. Způsoby rozdělení verzovacích systémů

- Podle přístupu (Lokální x Distribuovaný x Centralizovaný)
    - Lokální
        - Slouží běžně pouze jednomu uživateli na lokálním počítači.
    - Centralizovaný (CVCS)
        - Systém správy verzí je založen na centralizovaném modelu (Centralized Version Control System), který umožňuje spolupráci skupiny pracovníků na jednom nebo více projektech. K dispozici je centrální úložiště, ke kterému všichni klienti přistupují. Pouze v centrálním repositáři je možné srovnat a zaznamenat průběh práce vůči ostatním tvůrcům společných dat.

- Primární nevýhodou je riziko, že v případě kolapsu centrálního úložiště může dojít ke ztrátě dat, a to částečné nebo i úplné. Obranou proti tomu je samozřejmě zálohování repositáře na jiné místo. Tento nedostatek platí i pro systémy správy verzí založené na lokálním modelu.
- Nejen ztráta dat je obávanou situací u tohoto modelu. Tou je i případ výpadku centrálního úložiště, dočasného i dlouhodobého. Výpadek zabraňuje pokračování v práci zcela, nebo alespoň v omezené míře. Nelze totiž nijak ukládat ani přijímat změny v projektu.

o Decentralizovaný (DVCS)

- Systém správy verzí založený na decentralizovaném modelu (Distributed Version Control System), stejně jako systémy založené na centralizovaném modelu, umožňuje spolupráci skupiny pracovníků na jednom nebo více projektech. Je zde však ale absence centrálního úložiště. Místo toho probíhá vývoj a řešení jednotlivých konfliktů nejdříve v rámci lokálního úložiště. Až poté se změny poskytnou dalším spolupracovníkům do sdíleného repositáře. Sdíleným repositářem může být například jeden z počítačů nebo i server.
- Každý klient vlastní plnohodnotnou kopii repositáře, proto i v případě kolapsu sdíleného repositáře lze jednoduše celý projekt obnovit z jakékoli lokální kopie. To znamená, i když dojde k výpadku nebo pouze jednoduše klient nemá přístup k internetu, lze bez větších obtíží pokračovat v práci. Otevírá se tím celé spektrum možností, při jaké příležitosti lze pracovat, ačkoliv nemáme přístup k hlavnímu repositáři.
- Umožněna je běžně práce i s několika vzdálenými repositáři najednou, proto existuje možnost spolupráce s více skupinami lidí současně (Dušička 2015).

- Podle platforem, na kterých funguje
    - Systém pro správu verzí je závislý na platformě/platformách, pro kterou byl navržen. Při jeho výběru je tedy nezbytné zvážit i to, které operační systémy, na nichž chceme přistupovat k repositáři, používáme (Sink 2011).
- Podle licence
    - Tento aspekt je individuální pro každého, kdo verzovací systém vybírá. Výběr je prováděn podle finanční situace a ochoty investovat do softwaru. Některé ze systémů jsou plně zdarma, jiné například při použití do určitého počtu pracovníků. I to lze zvážit.

4 Porovnávané verzovací systémy

Praktická část bakalářské práce má za cíl řešit odvěký problém: který systém správy verzí je lepší. Již dopředu je nutné prozradit, že neexistuje jeden ideální. Každý je vhodný pro odlišné podmínky a ne vždy klade důraz na to samé jako ostatní. Záleží na potřebách a prostředcích osoby/osob, pro které má systém pro správu verzí sloužit. Jsou systémy, které jsou snadné pro naučení se, jiné jsou praktické a snadné pro následnou práci. Základní principy fungování jsou ale většinou podobné.

Při výběru je nutné zvážit mnoho hledisek. Nejen, zda má kladné hodnocení mezi uživateli, ale také, zda bude vyhovovat již zavedené struktuře. Společnost, která na většině vlastněných počítačů používá jeden operační systém, nemůže zvolit systém správy verzí, který by ho nepodporoval. Pokud se vedení společnosti rozhoduje o přechodu mezi verzovacími systémy, mělo by zvážit ten, který bude umožňovat snadný a bezpečný přesun dat ze stávajícího repositáře.

4.1 Git

Obrázek 1: Logo – Git

4.1.1. Obecný popis

Verzovací systém Git je vyvíjen od roku 2005. Původně měl sloužit jen pro verzování jádra Linuxu, ale postupně se vyvinul v plnohodnotný systém pro správu verzí.

Z hlediska statistik Google Trend (2015) a Eclipse Foundation (2014) pro rok 2014 je Git již nejpoužívanějším systémem pro správu verzí. Mezi lety 2013 a 2014 se dostal

před Subversion. Statistiky jsou v podkapitole Vzájemné porovnání. Trend vývoje počtu jeho uživatelů je značně rostoucí (Chacon 2014, Loeliger 2012, Chacon 2009).

Tabulka 4: Obecné údaje o systému Git

Tvůrce	Junio Hamano
	Linus Torvalds (Zakladatel)
Licence	GNU GPL v2
Webové stránky	http://www.git-scm.com/
Vyvíjeno v	C
	Bash (Bourne shell)
	Perl
Kdo systém používá	Android
	Linux kernel
	Twitter

Zdroj: vlastní zpracování

4.1.2. Technické specifikace

- Centralizovaný/Distribuovaný
    - Distribuovaný
- Kompatibilita s jinými verzovacími systémy (možnost migrovat data z)
    - Import repositářů je možný ze Subversion, Mercurial a Perforce
- Uložení dat
    - Verzovací systém Git je speciální tím, že při každém commitu ukládá všechna data souboru znovu. To přispívá k lepší odolnosti vůči poškození repositáře možnou hardwarovou chybou nebo chybou uživatele. Na druhou stranu ale tento způsob, tzv. redundance, zvyšuje nároky na diskovou kapacitu datového média, kde je všude repositář uchován. Proto jsou starší commity upravovány pomocí komprimace a odděleně archivovány.
    - Tím, že se jedná o distribuovaný verzovací systém, kromě hlavního serveru je kopie repositáře uchována na stroji každého vývojáře, který na projektu pracuje. Jednotlivé kopie úložišť jsou dle uvážení synchronizovány

s hlavním repositářem. Data jsou pro přenos zkomprimována a pro zmírnění zatížení sítě odesílána najednou. Princip distribuovaných systémů pro správu verzí popisuji v podkapitole Způsoby odlišení systémů.

- Oficiálně doporučení GUI klienti
    - Pro Windows
        - TortoiseGit (zdarma), GitHub (zdarma), Git Extensions (zdarma), SourceTree (zdarma), git-cola (zdarma), SmartGit (pro nekomerční užití).
    - Pro Linux
        - git-cola (zdarma), giggle (zdarma), SmartGit (pro nekomerční užití), GitEye (zdarma).
    - Pro Mac
        - GitHub (zdarma), Tower (30-denní trial verze), Gitbox ($14.99), GitX-dev (zdarma), SourceTree (zdarma), git-cola (zdarma).
- Síťové protokoly
    - Local, HTTP, SSH (Secure Shell) and Git
- Podporované OS
    - Linux (Ubuntu, Debian, Gentoo, Fedora, openSUSE, Arch Linux a další)
    - Microsoft Windows
    - OS X
    - Splňující standard POSIX

4.1.3. Zhodnocení

Git je nejen dobrý díky svému distribuovanému přístupu, hlavním znakem je jeho rozšířenost. Oproti například Mercurialu je nyní zatím stále udržovanější a podporovanější.

Distribuovaný přístup uživatelům zajišťuje možnost pracovat odkudkoli i bez připojení k internetu nebo přístupu k hlavnímu repositáři. Oproti Subversion a jiným centralizovaným systémům pro správu verzí bude Git pro tyto případy dobrá volba.

Systém Git je velice efektivní pro rozsáhlé projekty.

Nevýhodou je prvotní časová náročnost pro naučení se množství konceptů a příkazů. V dlouhodobém měřítku se ale tento investovaný čas brzy zhodnotí.

Revize nemají číselné označení jako například ve Subversion.

4.2 Subversion (SVN)

Obrázek 2: Logo – Subversion

4.2.1. Obecný popis

Systém byl vytvořen jako alternativa za tehdy nejpoužívanější, ale již velmi zastaralé CVS. První verze byla vydána v roce 2000.

Byl vytvořen jako open source centralizovaný systém, který se má vyznačovat svou spolehlivostí, jednoduchostí jeho modelu a použití.

Tabulka 5: Obecné údaje o systému Subversion

Tvůrce	Apache Software Foundation
Licence	Apache
Webové stránky	http://subversion.apache.org/
Vyvíjeno v	C (API)
	JavaHL
Kdo systém používá	SourceForge
	Facebook (jako primární systém)
	PuTTY

Zdroj: vlastní zpracování

4.2.2. Technické specifikace

- Centralizovaný/Distribuovaný
    - Centralizovaný
- Kompatibilita s jinými verzovacími systémy (možnost migrovat data z)
    - Git, CVS, SVN (problematické)
- Uložení dat
    - Z pohledu typického systémového souborového prohlížeče je repositář Subversion pro něj pouze jen další složka plná věcí. Kromě samotného úložiště obsahuje například podadresáře s konfiguračními soubory (Somasundaram 2013).
    - Pro organizaci repositáře jsou z pohledu administrátora na výběr dvě možnosti. První je, že na každý projekt bude vytvořen vlastní repositář. Druhou variantou potom je vytvořit jeden repositář pro všechny projekty. Výhodou společného repositáře je ušetření práce s nastavením, běžným zálohováním a je též možné přesouvat data snadno mezi projekty bez ztráty informací o historii (Collins-Sussman 2008).
    - V rámci repositáře je ukládání dat řešeno „deltami“. Soubor je uchován na dvou místech najednou, ale druhé místo obsahuje, pro zabránění duplicity dat, pouze odkaz na první místo.
    - Berkeley Database, nativní systém souborů.
- Existující GUI klienti
    - TortoiseSVN (zdarma), SmartSVN (zdarma pro nekomerční použití), WinSVN.
- Síťové protokoly
    - svn, svn+ssh, http, https

- Podporované platformy
    - Unix, Win32, BeOS, OS/2, Mac OS X
    - Pro systém Windows nejsou pro vytvoření repositáře podporovány pouze Windows95, Windows 98, Windows ME.

4.2.3. Zhodnocení

SVN je vhodné zvolit v případě menších repositářů, limit závisí individuálně na serveru, kde je repositář uchován.

Je relativně snadné se ho naučit používat a poprvé vytvořit repositář.

Nejsou rozlišovány adresáře a soubory.

Obsahuje širokou škálu všech možných pluginů pro IDE.

Pomalejší commit/checkout oproti Gitu.

Neumožňuje vyměnit si s kolegy kód, který například není stabilní bez toho, aniž by se nahrál do centrálního repositáře. Jediná, ale neefektivní možnost by byla, vytvořit záložní centrální repositář pro tyto účely.

Způsobuje veliké prodlevy při práci s repositářem u větších týmů pracovníků nebo rozsáhlejších klientských hierarchií.

Pro dosud nejnovější verzi 1.7 existuje problém pro specifickou situaci. Jedná se o moment nahrávání změn do sdíleného úložiště více lidmi těsně po sobě, pokud je editován stejný soubor. Může zde dojít až k vyrušení změn předchozích editací. Součást systému, která vývojáře upozorní na neaktuálnost jeho pracovní kopie, nezahlásí upozornění, pokud byla provedena aktualizace, ale v jiném podadresáři, než kořenovém. Pokud po poslední aktualizaci vývojář změní pracovní adresář, může nechtěně přemazat změny provedené jeho kolegou. Záznam kolegových změn je naštěstí uložen, je tedy možné jeho změny opět doplnit, ale pokud bylo editováno mnoho souborů, může toto napravování být dosti zdlouhavé. Prevencí této nepříjemnosti je provádění aktualizace (update) vždy jen v kořenovém adresáři.

4.3 Mercurial (Hg)

Obrázek 3: Logo – Mercurial

4.3.1. Obecný popis

Systém byl vytvořen v roce 2005 jako open source alternativa za verzovací systém BitKeeper.

Tabulka 6: Obecné údaje o systému Mercurial

Tvůrce	Matt Mackall, (společnost Selenic)
Licence	GNU GPL v2
Webové stránky	http://mercurial.selenic.com/
Vyvíjeno v	Python
	Binární diff je vytvořen v jazyce C
Kdo systém používá	Mozilla
	Python
	NetBeans

Zdroj: Vlastní zpracování

4.3.2. Technické specifikace

- Centralizovaný/Distribuovaný
    - Distribuovaný
- Kompatibilita s jinými verzovacími systémy (možnost migrovat data z)
    - S pluginem HgSubversion je možná práce s repositářem Subversion.

- Uložení dat
    - Klon repositáře je na stejné úrovni jako původní repositář.
    - Při zájmu sdílet vytvořený kód s kolegy ho není nutné vložit do centrálního úložiště, Mercurial umožňuje sdílení dat určeným lidem.
- Existující GUI klienti
    - TortoiseHG, gquilt, gwsmhg, HgWin, EasyMercurial, hgtui, SmartHg
    - Mac: MacHg, MacMercurial, SourceTree
- Síťové protokoly
    - http, ssh
- Podporované platformy
    - Microsoft Windows, Linux, Mac OS X, Solaris 11 Express
- Požadavky pro používání
    - Python (2.4–2.7), pro starší verze je nutné nainstalovat nejvýše verzi Mercurial 1.2.1.

4.3.3. Zhodnocení

Systém Mercurial je relativně snadný pro naučení i pro ovládání.

Oproti Gitu lépe dokáže sledovat přejmenování souborů.

Funkčnost je rozšiřitelná díky velkému množství modulů.

Obsahuje velmi přehlednou a užitečnou dokumentaci.

Je vhodný spíše pro menší týmy spolupracovníků (Borup 2011).

Příkazová řádka Mercurialu je přehledná, příkazy jednoduché pro naučení.

Je zde dobrá multiplatformní podpora.

4.4 Perforce (P4)

Obrázek 4: Logo – Perforce

4.4.1. Obecný popis

Tabulka 7: Obecné údaje o systému Perforce

Tvůrce	Perforce Software Inc.
Licence	Použití je zdarma v současné chvíli do 20 uživatelů, nebo pro účely výuky
Webové stránky	http://www.perforce.com/
Vyvíjeno v	C
	C++
Kdo systém používá	Google
	Ubisoft
	Nvidia
	Scania

Zdroj: (vlastní zpracování)

4.4.2. Technické specifikace

- Centralizovaný/Distribuovaný
    - Centralizovaný

- Kompatibilita s jinými verzovacími systémy (možnost migrovat data z)
    - Perforce obsahuje plugin pro integraci s existujícím Git repositářem.
- Uložení dat
    - Perforce umožňuje spravovat verze následujících typů souborů:
        - Textový soubor
        - Binární soubor
        - Nativní soubory apple na Macintoshi
        - Soubory s kódováním Unicode a (UTF-16)
        - Symbolické odkazy
- Existující GUI klienti
    - Perforce Visual Client (pro Windows, Mac OS X, Linux)
- Síťové protokoly
    - Vlastní
- Podporované platformy
    - Windows, Mac OS X, Linux

4.4.3. Zhodnocení

Hlavní oblastí uplatnění Perforce je ve velkých vývojových týmech nebo ještě navíc při objemných datech v repositářích. Též jako další předností se ukázalo, že je velice vhodný pro vývoj herních aplikací.

Obvykle právě kombinace velikosti úložiště, větvení a slučování modelu, případně zkušenost týmu, může přispět k rozhodnutí zvolit pro správu verzí právě Perforce.

Perforce obsahuje zpětné potvrzení vyřešení konfliktu. Pokud nastane během vývoje konflikt kódu mezi dvěma nebo více vývojáři, jeden z nich konflikt vyřeší, ale pro aktualizaci na novou verzi kódu musí ještě všichni ostatní přijmout jeho řešení tohoto konfliktu.

4.5 Vzájemné porovnání

Pro výběr verzovacího systému nemusí hrát roli jen to, zda je praktický, spolehlivý nebo zda je zdarma. Informace o tom, kolik lidí jednotlivé systémy pro správu verzí používá, případně má o ně zájem, může též alespoň z části ovlivnit výsledek výběru. Trend, jak se mění oblíbenost, může vypovídat o perspektivnosti, úrovni dlouhodobé podpory ze strany tvůrců, praktičnosti nebo i nástupu nového prvku na trh.

Tabulka, která následuje, obsahuje výčet základních informací všech hodnocených systémů.

Tabulka 8: Porovnání systémů

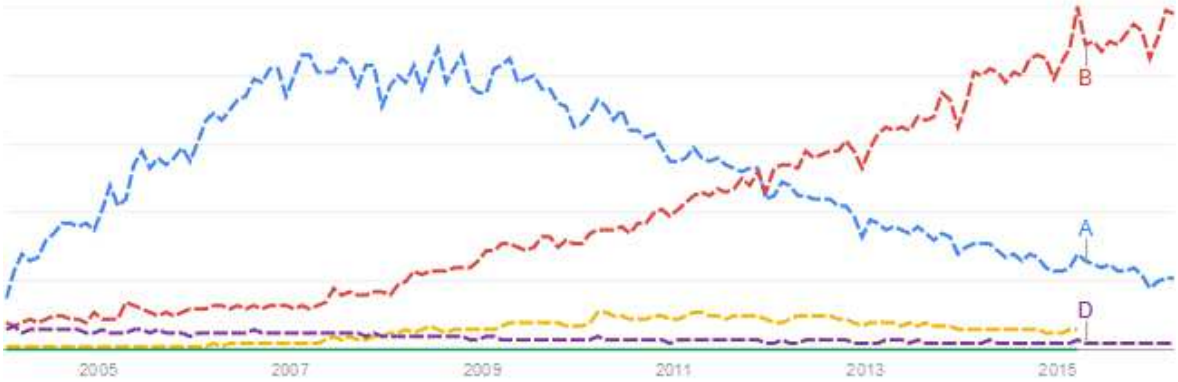
	Centralizovaný / Decentralizovaný	Rok vzniku	Licence	Snadné pro naučení	Snadné pro používání	Pro veliké projekty
Git	Distribuovaný	2005	GNU GPL v2	Ne	Ano	Ano
SVN	Centralizovaný	2000	Apache	Ano	Ano	Ano
Mercurial	Distribuovaný	2005	GNU GPL v2	Ano	Ano	Ne
Perforce	Centralizovaný	1995	Zdarma do 20 uživatelů, nebo pro potřeby výuky	Ano	Ano	Ano

Zdroj: vlastní zpracování

4.5.1. Google Trend

Následující graf z webové aplikace Google Trend (2015) ukazuje četnost hledání verzovacích systémů Subversion, Git, Mercurial, Perforce ve vyhledavači Google za období 2005 – 2015 a vypočítanou předpověď dalšího vývoje.

Graf 1: Statistika - Google Trend



Zdroj: (Google Trend 2015)

První křivka, v modré barvě, je křivka systému Subversion. Ta již od počátku měla podle této statistiky jasné prvenství a neustále stoupající trend. Systém SVN totiž byl již všeobecně známý několik let. Období zlomu jsou roky 2007 až 2009. Od něj do současnosti se míra zájmu o systém SVN dlouhodobě pouze snižovala. Kolem roku 2012 již ztratil své prvenství. I prognóza budoucího vývoje ukazuje, že se zatím neočekává výrazný obrat.

Druhá křivka, v červené barvě, reprezentuje systém Git. V roce 2005, na počátku grafu, tento systém teprve začínal. To odpovídá pomalému nárůstu zájmu. Od zlomového období, též v letech 2007 až 2009, začal zájem o tento systém výrazně růst a pokračuje, s občasnými výkyvy, až dodnes. Též odhad budoucího vývoje vykazuje spíše rostoucí změny.

Třetí křivka, ve žluté barvě, představuje systém Mercurial. Tento distribuovaný systém, velmi podobný Gitu, měl ale oproti svému konkurentovi již od svého vzniku jen zlomkový zájem. Důvodem tohoto úkazu mohlo být to, že Mercurial je vhodný spíše pro menší projekty s omezenějším počtem vývojářů. I přesto, že se zájem o něj kolem roku 2011 mírně zvýšil, dosud se k němu nikdy ani nepřiblížil a opět začal klesat. Vývoj křivky tohoto systému je pravděpodobně tak nestálý, že nelze vytvořit odhad na následující rok.

Čtvrtá křivka, ve fialové barvě, patří centralizovanému systému Perforce. Rok počátku verzovacího systému Perforce je 1995. Graf ukazuje tedy pouze část z celého působení Perforce mezi ostatními systémy. Za celých deset let, které jsou v grafu zobrazeny, se křivka systému změnila pouze minimálně. Její změna je plynulá, ale

dlouhodobě klesajícího charakteru. Při srovnání zájmu vůči ostatním byl tento systém pouze do roku 2008 vyšší oproti Mercurialu. Od tohoto roku je v rámci srovnávaných systémů nejníže. Jako důvod by se nabízelo, že je zdarma pouze do limitovaného počtu uživatelů.

4.5.2. Eclipse průzkum 2014 - verze 2

Průzkum prováděný každoročně společností Eclipse má široký záběr v oblasti IT, kterou zkoumá. Licence, pod níž je vytvořen tento průzkum, je Eclipse Public Licence v1. Ta rozšiřuje Common Public License. Pro účely této práce jsou uvedeny pouze informace týkající se verzovacích systémů a prokázání relevantnosti tohoto průzkumu. Počet respondentů je 876.

V průzkumu se objevují pouze tři z výše zmíněných systémů pro správu verzí. Ostatní neměly dostatečné výsledky pro umístění v grafu.

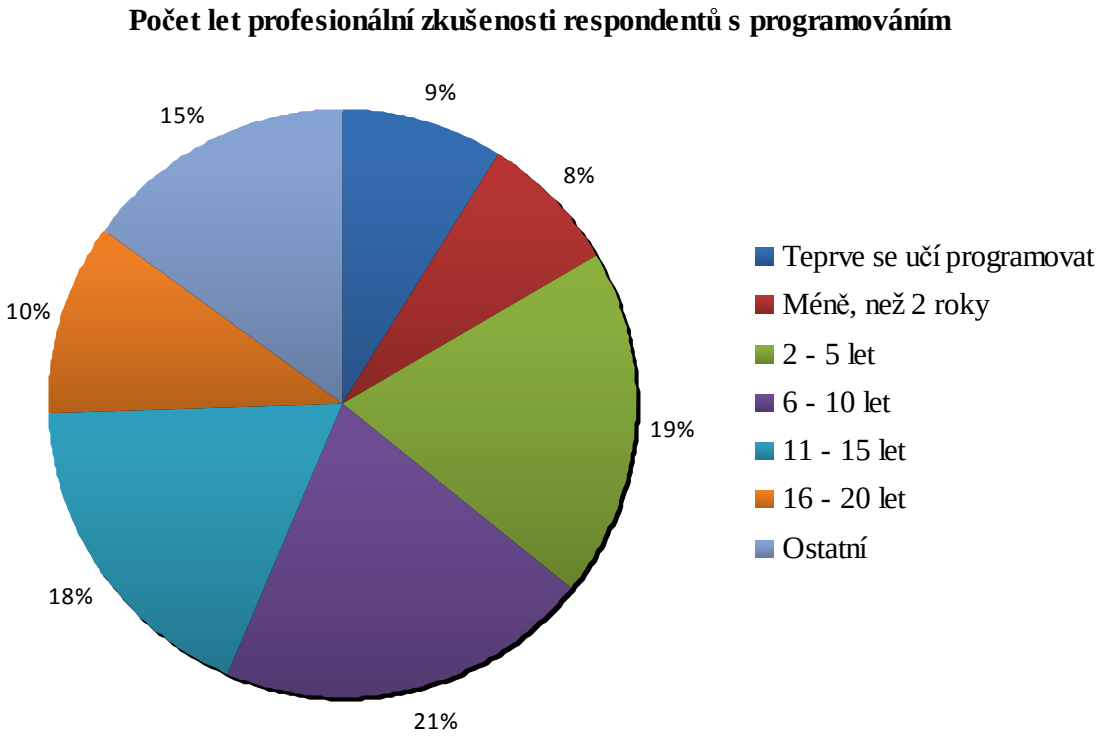
Společnost Eclipse svá zveřejněná data nejprve podkládá grafy, ve kterých uvádí zkoumaný vzorek lidí. Lidé byli nejprve dotazováni na zkušenosti s programováním na profesionální úrovni, zaměření a velikost firmy, kde pracují a jejich roli ve firmě.

Původní grafy jsou z důvodu nutnosti překladu do českého jazyka upraveny.

Grafy vzorku respondentů

V následujícím grafu jsou odpovědi respondentů na otázku, kolik let již strávili programováním na profesionální úrovni. Zastoupení bylo rovnoměrné ze všech skupin.

Graf 2: Zkušenosti respondentů

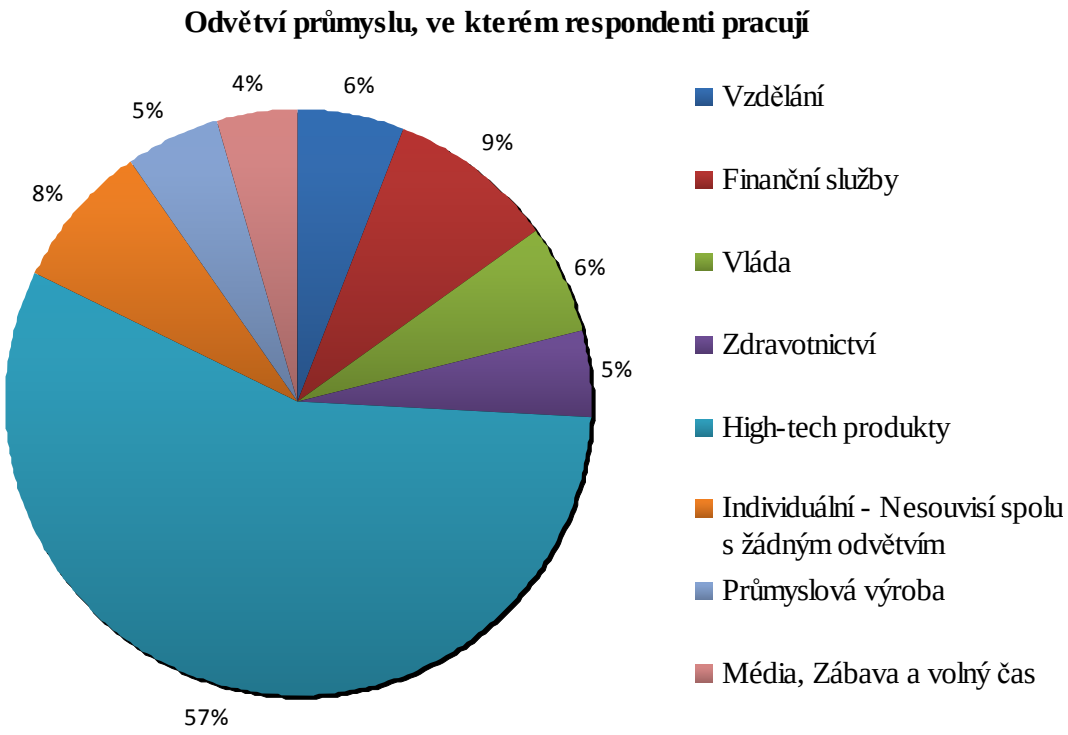


Zdroj: (Eclipse Foundation 2014), vlastní zpracování

Graf zaměření firem

Nejvíce respondentů pracovalo ve společnostech, které se zabývaly high-tech produkty, což dokazuje další graf. Bylo to 37,7%.

Graf 3: Zaměření firem respondentů

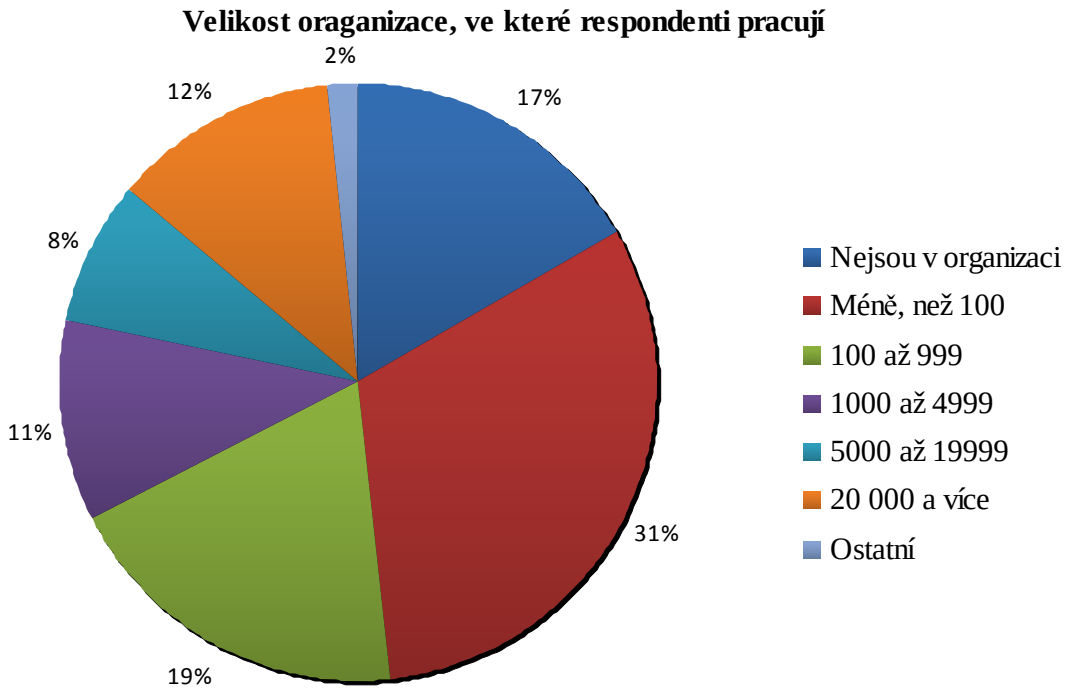


Zdroj: (Eclipse Foundation 2014), vlastní zpracování

Graf velikosti firmy

Graf, který následuje, obsahuje velikosti firem, kde respondenti pracují. Zastoupení firem bylo přibližně stejné, nejvíce však byli respondenti z firem s méně než 100 zaměstnanci.

Graf 4: Velikosti firem

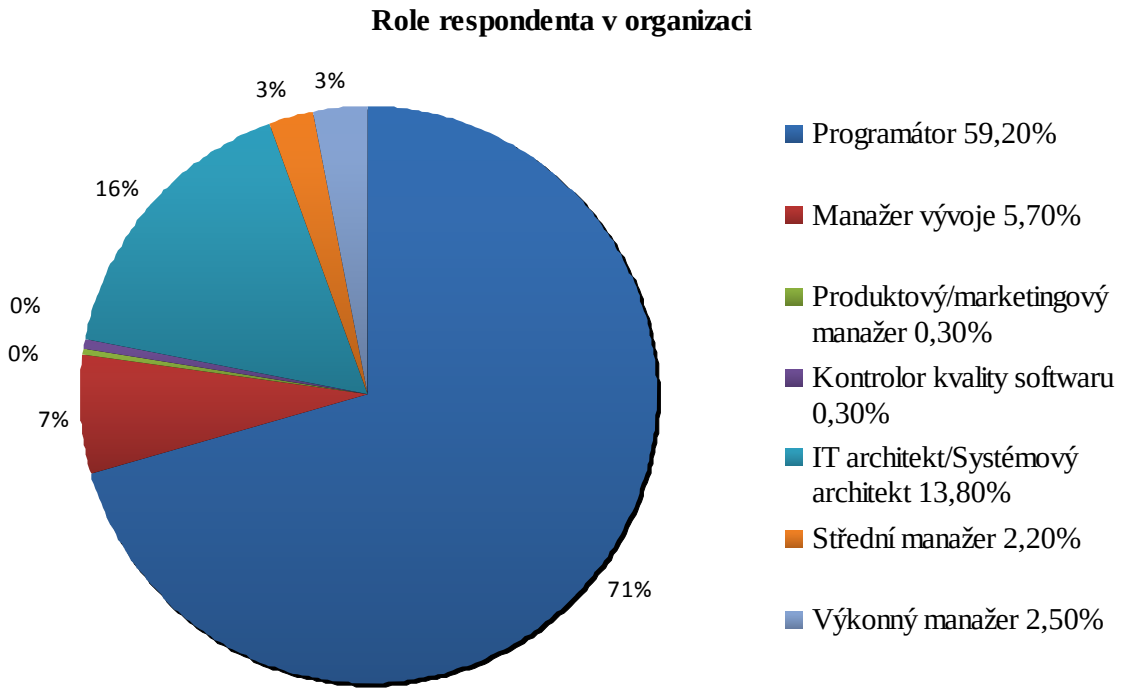


Zdroj: (Eclipse Foundation 2014), vlastní zpracování

Graf role ve firmě

Nejčastější role respondentů ve firmách byla programátor a systémový architekt. Lze tedy očekávat, že průzkum by měl být relevantní. Programátoři využívají verzovací systémy denně a často se setkají s více z nich.

Graf 5: Role ve firmě

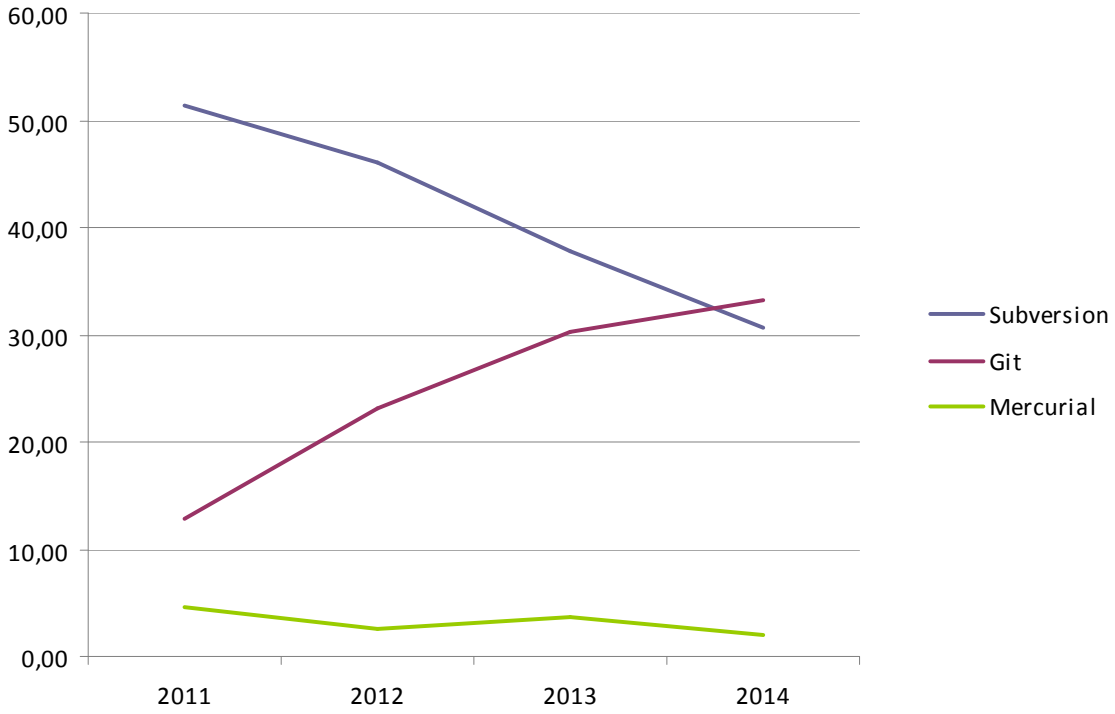


Zdroj: (Eclipse Foundation 2014), vlastní zpracování

Graf použití verzovacích systémů

V tomto hlavním grafu jsou tři systémy pro správu verzí, které byly výše rozebírány. Vývoj je vidět oproti Google Trend pouze za roky 2011 až 2014. Například při porovnání Subversion a Git je ale evidentní podobnost. Mezi rokem 2013 a 2014 došlo k výměně prvenství. Graf z Google Trend tuto výměnu ukazuje již přibližně o rok dříve, ale ten obsahuje údaje o zájmu lidí o konkrétní systémy. Avšak ke změně systému se lidé odhodlají až po určité době. Proto je zde vidět přibližně roční zpoždění.

Graf 6: Nejpoužívanější verzovací systémy



Zdroj: (Eclipse Foundation 2014), vlastní zpracování

5 Zhodnocení a návrh řešení

5.1 Zhodnocení výsledků

Při analýze vlastností systémů bylo zjištěno, že systém správy verzí Git je vhodný pro veliké projekty, včetně velikého počtu lidí, hlavní předností je jeho rychlost. I přesto, že je na začátku složitější na pochopení, než jeho konkurenti, při práci je poté neocenitelným pomocníkem.

Subversion je oproti Gitu vhodný spíše pro limitované objemy dat, ale jeho přednost je v jednoduchosti. Je snadný pro naučení se i pro následnou práci. Jednou z limitací se při výběru může stát jeho centralizovanost. Pokud ale není model server-klient na obtíž, je Subversion oblíbeným systémem správy verzí.

Systém Mercurial je distribuovaný systém, který je velmi podobný Gitu. Při použití rozšíření je poté rozdíly mezi těmito systémy těžké hledat. Otázka Git, nebo Mercurial je často pokládána. Mercurial má skvělou výhodu například proti Subversion. Při zájmu sdílet vytvořený kód s kolegy ho není nutné vložit do centrálního úložiště, Mercurial umožňuje sdílení dat určeným lidem. To je veliká výhoda, pokud je kód například nestabilní. Při rozhodování, zda zvolit Mercurial závisí velmi na individuálních nárocích a zvycích.

Perforce je využíván zvlášť ve velikých organizacích. Mezi jeho přednosti patří rychlost, schopnost práce s objemnými daty a při velikém počtu spolupracovníků. Dalším pozitivem je, že je vhodný a oblíbený pro vývoj počítačových her, a to i mezi jejich předními tvůrci jako je Ubisoft, EA nebo Nintendo.

Ze statistik uvedených v kapitole Vzájemné srovnání vyplynul očekávaný pokračující trend oblíbenosti a s ním související použití systémů pro správu verzí. Git se ukázal silně rostoucí na žebříčku popularity a použití. Naopak Subversion, původně vlastnící prvenství, postupně ztrácí na své oblibě. Perforce se ve statistikách neobjevuje na vysokých příčkách počtu uživatelů, pravděpodobně díky své komerční podstatě. Své místo v tomto porovnání si ale zajisté zaslouží.

5.2 Návrh řešení

Některé z vybraných nedostatků systémů pro správu verzí by mohly být opraveny, případně by jim mohla být přidána rozšiřující funkce, jako součást navazující práce na tuto bakalářskou práci.

O prázdninách jsem pracoval ve velké firmě na brigádě. Pokud bych byl osloven, abych navrhl systém správy verzí, zvolil bych systém Git, protože ten by pro tuto firmu vycházel mých kritérií nejlépe.

6 Závěr

Důležitost zálohování bývá přehlížena z důvodu úspory financí nebo laxního přístupu. V bakalářské práci byly proto popsány zásady, jak lze zálohování efektivně zajistit.

Verzovací systémy si v poslední době získaly své opodstatnění při nasazení ve firmách. Systém Subversion začíná postupně ztrácet na popularitě, jeho rozšířenost je ale stále podle statistiky druhá nejvyšší ze všech systémů správy verzí. Na první příčku se již dostal distribuovaný systém Git. Přestože jsou tyto dva systémy nejpoužívanější, zajisté diky své bezplatné podobě, Perforce je oblíbený více mezi velkými společnostmi jako je Google, Nvidia nebo Scania. Oproti Subversion je oblíbený především svou rychlostí. Tvrzení, že je vhodný i pro vývoj počítačových her dokládá fakt, že systém Perforce používají i mohutné společnosti Ubisoft a Electronic Arts. Je schopen pracovat i s objemnými daty v repositářích.

Práce splnila cíl představit možnosti a zásady zálohování dat. Byla popsána analýza rizik a byl vytvořen návrh optimálního řešení pro vybrané situace. Byl vysvětlen princip archivace dat. V části věnované systémům správy verzí byla nastíněna problematika správy verzí a pojmy s ní spojené. Byla též vyjmenována kritéria, podle kterých lze systémy pro správu verzí odlišovat.

V analytické části byly porovnány a zhodnoceny čtyři zvolené systémy pro správu verzí. Byl vytvořen přehledný a komplexní souhrn vlastností, který by měl usnadnit výběr jednoho z těchto systémů. Vložen byl i průzkum společnosti Eclipse nejčastěji používaných systémů spolu s aktuální statistikou vyhledávání, vygenerovanou pomocí webové aplikace Google Trend. Tato statistika obsahuje i prognózu budoucího vývoje.

Všechny předem stanovené cíle tedy byly splněny.

7 Seznam použitých zdrojů:

7.1 Seznam literatury

COLLINS-SUSSMAN, Ben. *Version Control with Subversion.* 2.vydání. O'Reilly Media, 2008. 432 s. ISBN 978-0596510336.

CHACON, Scott. *Pro Git. Praha: CZ.NIC,* Edice CZ.NIC, 2009. 263 s. ISBN 978-80-904248-1-4.

CHACON, Scott. *Pro Git.* 2. vydání. New York: Apress, 2014. 574 s. ISBN 9781484200773.

KASTNER, Aleš. *Zálohování a archivace.* 1. vydání. Praha: GComp, 1994. 100 s. ISBN 80-85649-21-7.

KŘÍŽ, Libor. *Komprimační a archivační programy.* 1. vydání. Praha: Computer Press, 2002. 115 s. ISBN 80-7226-757-4.

LEIXNER, Miroslav. *PC - zálohování a archivace dat*. Praha: Grada, 1993. 408 s. ISBN 80-85424-73-8.

LOELIGER, Jon a Matthew MCCULLOUGH. *Version control with Git*. 2. vydání, 2012. 456 s. ISBN 1449316387.

PECINOVSKÝ, Josef. *Archivace a komprimace dat*. Praha: Grada, 2003. 116 s. ISBN 80-247-0659-8.

SINK, Eric. *Version control by example*. 1. vydání. Champaign, Ill: Pyrenean Gold Press, 2011. [202] s. ISBN 978-0-9835079-1-8.

SOMASUNDARAM, Ravishankar. *Git Version Control for Everyone.* Online-Ausg. Birmingham: Packt Pub, 2013. 180 s. ISBN 9781849517539.

7.2 Seznam internetových zdrojů

ADSHEAD, Antony. Data backup vs archiving. ComputerWeekly [online]. 2009 [cit. 2015-2-15]. Dostupné z: <http://www.computerweekly.com/news/1369092/Data-backup-vs-archiving-Whats-the-difference>

BORUP, Rick. *VPF Version control with Mercurial*, Information Technology Associates [online]. 2011 [cit. 2015-2-28]. Dostupné z: <http://www.ita-software.com/papers/borup_mercurial_published.pdf>

DUŠIČKA, Martin. *Systém pro správu verzí.* ITnotes [online]. 2015 [cit. 2015-2-28]. Dostupné z: <http://itnotes.cz/repository.html>

JUNEK, Pavel. *Zálohování a archivace dat v podnikovém prostředí* [online]. 2013 [cit. 2015-3-02]. Dostupné z: <http://www.zalohovani.net/zalohovani-a-archivace-dat-v-podnikovem-prostredi-5-dil-typy-zaloh-a-jejich-rotacni-schemata/>

KUČERA, František. Distribuované verzovací systémy. Abclinuxu [online]. 2011 [cit. 2015-2-17]. Dostupné z: <http://www.abclinuxu.cz/clanky/distribuovane-verzovaci-systemy-uvod-2>

REISDORPH, Kent. *Why use version control?*. *NTK* [online]. 1999 [cit. 2015-2-26]. Dostupné z: <http://search.proquest.com.ezproxy.techlib.cz/docview/222902721/>

VÍT, Svatopluk. *Zálohování dat: 4 strategie pro firemní bezpečnost.* LinuxExpres [online]. 2013 [cit. 2015-3-4]. Dostupné z: <http://www.linuxexpres.cz/praxe/zalohovani-dat-4-strategie-pro-firemni-bezpecnost>

Definice a rotace záloh. Storage.cz [online]. 2012 [cit. 2015-2-17]. Dostupné z:<http://www.storage.cz/cs/odborna-sekce/detail/id/46-definice-a-rotace-zaloh>

Eclipse community survey 2014 v2. Eclipse Foundation [online]. 2014 [cit. 2015-3-5]. Dostupné z: <http://www.slideshare.net/IanSkerrett/eclipse-community-survey-2014>

Zájem ve službě Webové vyhledávání. Google Trend [online]. 2015 [cit. 2015-3-5]. Dostupné z: <http://www.google.cz/trends/explore#q=%2Fm%2F012ct9%2C%20%2Fm%2F05vqwg%2C%20%2Fm%2F08441_%2C%20%2Fm%2F08w6d6&cmpt=q&tz>

8 Seznamy

8.1 Seznam obrázků

8.2 Seznam grafů

8.3 Seznam tabulek