

Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Bakalářská práce

Statistiky času z kalendářových dat

Prohlášení

Prohlašuji, že jsem bakalářskou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni, dne 6. května 2014

Tomáš Krásný

Poděkování

Děkuji doc. Ing. Přemyslu Bradovi, MSc. Ph.D., za odborné vedení mé bakalářské práce a za cenné rady, které mi pomohly tuto práci vytvořit.

V Plzni, dne 6. května 2014

Tomáš Krásný

Abstract

This work is based on calendar data which almost everyone uses every day. People can keep time records in many ways, more and more in online calendars. We can use calendars for saving events, tasks, alarms etc. After filling up calendar with events or data, various time statistics can be calculated. We can obtain total time of events from previous month or specific events representing for instance business meetings.

The aim of this bachelor thesis is processing calendar data. The final goal is to create a tool for generating specific time statistics from events entered in online calendars and then display the results to user. This tool has various options the user can set, e.g. granularity (per days, per weeks, ...), an output unit (minutes, hours, ...) or keywords which can separate important events from others only by a keyword in its subject. The final data are recorded in table and consequently saved in HTML file or Excel workbook.

This tool is implemented in Java language using various libraries. The functionality is verified on real-world Google calendars.

Abstract

Tato práce je založena na zpracování kalendářových dat, která jsou dnes velmi hojně využívána. Lidé si mohou vést časové záznamy více způsoby, z nichž jedním jsou online kalendáře. Po naplnění kalendáře událostmi je možné vypočítávat určité časové statistiky; lze například získat celkový čas trvání událostí v minulém měsíci nebo trvání konkrétních událostí, jakými mohou být například firemní schůzky.

Hlavním cílem této bakalářské práce je vytvoření nástroje pro generování časových statistik z událostí v online kalendářích a následné zobrazení těchto statistik uživateli. Uživatel má k dispozici mnoho možností nastavení aplikace. Příkladem může být granularita (po dnech, po týdnech apod.), jednotky výstupu (minuty, hodiny, ...) nebo klíčová slova, která mohou oddělit důležité události od ostatních. Výsledky jsou poté zobrazeny v tabulce a uloženy ve formátu HTML nebo v sešitu aplikace Microsoft Excel.

Aplikace je implementována v programovacím jazyce Java, pro samotný vývoj byly použity další externí knihovny. Funkčnost je ověřena na skutečných kalendářích.

Obsah

1	Úvod	1
1.1	Cíle práce	1
1.2	Využití v praxi	2
1.3	Struktura práce	3
2	Výkazy času a time management	5
2.1	Proč vykazovat čas	5
2.2	Existující nástroje	5
2.3	Time management	6
2.3.1	Pravidla	6
2.3.2	Kategorizace	7
2.3.3	Priority	7
3	Online kalendáře a jejich data	10
3.1	Kalendář	10
3.1.1	Využití	10
3.1.2	Výhody a nevýhody	11
3.2	Formát iCalendar	13
3.2.1	Historie	14
3.2.2	Popis souboru	14
3.2.3	Struktura	15
3.2.4	Komponenty	15
3.2.5	Využití a použití	21
3.2.6	Knihovny	22
4	Zadání	23
4.1	Účel	23
4.2	Funkčnosti aplikace	23
5	Implementace	26
5.1	Architektura aplikace	26

5.2	Popis architektury	26
5.2.1	Technologie	26
5.2.2	Průchod architekturou	28
5.3	Zpracování dat	29
5.3.1	Události	29
5.3.2	Klíčová slova	33
5.4	Generování výstupu	35
5.4.1	Počítání součtů	36
5.4.2	Výstupní formáty	37
5.5	Uživatelské rozhraní	38
5.5.1	Modely a renderery	38
5.5.2	Integrace s aplikační vrstvou	39
5.5.3	Využití událostí	40
5.6	Výsledná podoba aplikace	41
6	Ověření aplikace	43
6.1	Ověření hlavních modulů	43
6.1.1	Ověření načítání dat	43
6.1.2	Ověření statistik a exportu	44
6.2	Testy dob běhu	44
6.3	Ukázka výstupu	46
7	Závěr	49
7.1	Zhodnocení	49
7.2	Možná rozšíření	50
7.3	Uživatelská dokumentace	51

1 Úvod

Každý už určitě byl někdy ve stavu, kdy musel v hlavě nosit spoustu důležitých informací, úkolů, připomínek, většinou ve spojení s nějakým datem nebo s nějakou akcí, např. schůzkou, poradou nebo úkolem. Díky tomu se velmi snadno může stát, že se na nějakou takovou událost prostě zapomeneme. A aby se takovéto situace omezily co možná nejvíce, vznikly způsoby, jak si tyto informace, schůzky, srazy, úkoly nebo čistě jen poznámky zaznamenávat. A jedním z těchto způsobů je použití kalendářových dat.

Kalendářová data nám umožňují plánovat si svůj vlastní čas tak, abychom měli vždy přehled o tom, co nás dnes, za týden nebo za měsíc čeká. Takovým událostem lze nastavit připomenutí v určitý čas, abychom na nějakou z nich nezapomněli. Tyto informace jsou uloženy v kalendáři, který je reprezentován nějakým standardem. Nejpoužívanější standard je formát *iCalendar* (zkráceně *iCal*). Do tohoto kalendáře lze ukládat určitá data související s časem. Těmi může být např. událost, úkol, alarm atd. Není tedy divu, že jsou v dnešní době kalendářová data velmi rozšířena a využívána.

1.1 Cíle práce

V úvodu byla zmíněna kalendářová data a jejich využití k usnadnění práce s časem a časovými údaji. Prvním úkolem bude prozkoumat formáty uložení těchto dat, jaké možnosti tato kalendářová data nabízejí a jaký se skrývá potenciál v jejich zpracování. S tím souvisí výzkum, jak vůbec tato data zpracovávat a jaké (jestli nějaké) nástroje již existují. Pokud nějaké existují a poskytují nějaké výsledky a výstupy, bylo by vhodné provést jejich analýzu za účelem získání představy například o lepším využití vynaloženého času. Aby tento čas byl využit co nejefektivněji, budou prozkoumány možnosti, jak toho docílit.

Tato práce je dále orientovaná na časové údaje z pohledu vykazování stráveného času. Budou prozkoumány existující nástroje, jak čas evidovat a vykazovat, jaké jsou jejich výstupy a k čemu je vykazování spotřebovaného času vůbec vhodné.

Praktickým cílem je navrhnout a vytvořit nástroj, který bude umět zmi-

ňovaná kalendářová data zpracovat. Aplikace bude umět navrženým způsobem vypočítávat souhrnné statistické informace sloužící jako podklad pro časové výkazy práce. Jelikož zdrojem těchto informací jsou kalendářová data obecně, je nutné prozkoumat možnosti, jak tyto údaje týkající se výkazů oddělit od ostatních údajů, které se mohou v kalendáři vyskytovat, ale s výkazy nesouvisí.

Výsledná aplikace bude otestována na netriviálních datech a bude ověřena správnost vypočtených údajů.

1.2 Využití v praxi

Aplikace může mít v reálném používání hned několik způsobů využití, mezi ty hlavní lze zařadit například

- vykazování práce,
- sumarizace trvání (určitých) událostí,
- sledování spotřebovaného času na projektu,
- přehled o osobním časovém vytížení.

Vykazování času

Vykazování času neboli evidence odpracované doby, ať už pro zaměstnavatele nebo za jiným účelem, je velmi podstatná záležitost. Zaměstnavatel si na základě těchto výkazů může kontrolovat odpracovanou dobu svých zaměstnanců nebo provádět například měsíční zpracování výplat. Jelikož si uživatel může vést jeden kalendář pro zaznamenávání všech událostí, je nutné nějak oddělit události, které spolu mají něco společného - tento úkol se nazývá kategorizace. Následně lze ze statistik zjistit, jak dlouho se uživatel věnoval určitým událostem.

K tomuto vykazování času lze ještě podotknout, že není potřeba žádného speciálního softwaru pro vykazování spotřebovaného času. Práce s kalendářem je velmi snadná a dostupná a vložení nové události je velmi rychlou záležitostí.

Sumarizace trvání událostí

Při pohledu na nějaké časové období není trvání jednotlivé události příliš hodnotná informace. Mnohem hodnotnější informací je součet trvání všech událostí či nějaké podmnožiny. Jelikož z prvního pohledu na událost v nějaké kalendářové aplikaci není vždy zřejmé, jak dlouho událost trvala (natož když je potřeba události spojovat do nějakého určitého časového období), je nutné trvání těchto událostí nějak sumarizovat. Díky této sumarizaci lze získat nad zvoleným obdobím přehled o celkovém trvání.

Sledování spotřebovaného času na projektu

Toto využití lze spíše chápat jako konkrétní případ předchozího odstavce. Spotřebovaný čas na nějakém projektu či zakázce je velmi důležitou informací pro projektový management ve firmě. Díky tomu má vedení přehled o vývoji projektu z pohledu spotřebovaného času, ať už za jednotlivce či za celý vývojářský tým, a může podniknout určité kroky či opatření k jeho redukci (samozřejmě s ohledem na kvalitu produktu).

Osobní časové vytížení

Výstup aplikace nemusí být použit konkrétně k vykazování času nebo jiným pracovním účelům. Se statistikami lze sledovat i osobní časové vytížení. Například po vygenerování jednotlivých výstupů za poslední tři měsíce může uživatel jejich srovnáním zjistit, že v minulém měsíci se jeho volný čas oproti předešlým dvěma zmenšil o 20%. Dalším příkladem může být celkový čas strávený na rodinných oslavách a večírcích. To vše je samozřejmě podmíněno tím, že kalendářové události, reprezentující spotřebovaný čas na těchto událostech, jsou nějak odlišeny od ostatních, a to buď samostatným kalendářem, nebo klíčovým slovem v předmětu události.

1.3 Struktura práce

Celou práci lze rozdělit na dvě hlavní části. První část (kapitola 2 a 3) je teoretická a jejím hlavním tématem je práce s časem. Jsou v ní popsány způsoby plánování času tak, aby byl maximálně a efektivně využit (time management, sekce 2.3). K tomu jsou popsány některé existující nástroje a jejich možnosti, které nabízí. Dále jsou v této části (sekce 2.1) vysvětleny způsoby vykazování a evidence spotřebovaného času a v neposlední řadě hlavně popis online kalendáře jako hlavního zdroje dat, jeho využití a formáty či způsoby jeho uložení, to vše v kapitole 3.

Druhá část je spíše praktická a je zaměřena na konkrétní vytvořenou aplikaci zpracovávající tato kalendářová data. V kapitole 4 je vysvětlen účel vytvoření této aplikace a její hlavní funkčnosti. Dále je v kapitole 5 popsána architektura a implementace včetně logické a prezentační vrstvy. Po této části následuje ověření aplikace (kapitola 6), ukázka výstupu a jeho ověřování a testování.

Za druhou částí je v kapitole 7 uveden závěr a zhodnocení práce. V poslední části jsou popsána možná rozšíření vytvořené aplikace a uživatelská dokumentace v sekci 7.3.

2 Výkazy času a time management

2.1 Proč vykazovat čas

Výkazy času slouží k získání informací o odpracovaném času na jednotlivých úkolech či celých projektech. Dalším důvodem je například získání představy o pracnosti projektů či podklady pro výplaty nebo odměny zaměstnancům. Výkazy mohou sloužit jako důležité vodítko při sledování odpracované doby na projektu. Před spuštěním vývoje produktu se odhadne doba potřebná k jeho realizaci a následně lze v průběhu vývoje sledovat a srovnávat, jak odhadnutá doba odpovídá skutečnému času strávenému na projektu.

2.2 Existující nástroje

Existuje mnoho nástrojů, které nějakým způsobem slouží k vykazování či evidenci času a práce. Některé jsou orientovány spíše k evidenci stráveného času než k popisu provedené práce, např. **kalendáře**. Jedněmi z největších zástupců jsou webová aplikace Google Calendar či známý Microsoft Outlook. V nástrojích tohoto typu slouží k evidenci času události. Událost sama o sobě nemá příliš charakter výkazu vykonané práce. K události lze uvést její předmět, stručný popis a hlavně časové údaje: čas a datum od, do, opakování, místo události, účastníci atd.

Další nástroje obsahující informace o stráveném času jsou nástroje pro správu projektů: Redmine¹, Jira² či TargetProcess³ nebo jiné **bug tracking systémy** (bug tracking systém je speciálně navržená aplikace tak, aby pomáhala programátorům při vývoji projektů a následných oprav chyb - *bugů*): Bugzilla⁴, FlySpray⁵, Obě tyto kategorie spolu souvisí a prolínají se. Tyto systémy umožňují vést záznamy o spotřebovaném čase na jednotlivých úkolech (viz obrázek 2.1) a na jejich základě lze získávat informace o časovém vývoji celého projektu. Při vytváření úkolu lze specifikovat časový odhad tr-

¹<http://www.redmine.org/>

²<https://www.atlassian.com/software/jira>

³<http://www.targetprocess.com/>

⁴<http://www.bugzilla.cz/>

⁵<http://flyspray.org/>

vání jeho splnění a tento odhad následně srovnávat se skutečně stráveným časem.

Třetí kategorií mohou být systémy nasazené ve firmách a v institucích, kde je současně důležitý nejen popis odvedené práce, ale také časové údaje, kdy byla práce prováděna. Např. z důvodu výpočtu mzdy.

Změnit vlastnosti (Více)	
Stav *	Ve vývoji - In Entwicklung
Priorita *	Normální - Normal
Přířazeno	Tomáš Krásný
Přířazeno k verzi	
Začátek	2014-04-14
Uzavřít do	2014-04-30
Odhadovaná doba	35 Hodiny
% Hotovo	90 %

Přidat čas	
Strávený čas	4 Hodiny
Aktivita	Vývoj - Entwicklung
Komentář	

Obrázek 2.1: Ukázka možné aktualizace úkolu v systému Redmine

2.3 Time management

Time management [5, 6] je soubor dovedností, postupů a nástrojů za účelem lepšího využití svého času a dosažení větší produktivity [3]. Hlavním úkolem je plánování, realizace a komunikace. Cílem je správné rozvržení aktivit po dobu celého dne, aby výsledky práce byly efektivní. Mezi hlavní aktivity patří plánování, stanovení cílů a priorit, delegování, ...

2.3.1 Pravidla

Jak již bylo řečeno, time management je složen z několika pravidel či postupů vedoucích k efektivnímu řízení svého času. Tato pravidla nejsou složitá. Důležitější je se jimi řídit a soustředit se na ně. Mezi jedny z důležitých pravidel patří:

- **stanovení priorit:** naléhavá aktivita nesmí předběhnout důležitou,
- **delegování aktivit:** nevykonávat aktivity, na které jsou tu jiní,
- **odmítnutí:** být schopen říci „ne“ (odmítá se nabídka, ne člověk),
- **informovanost:** včas znát veškeré (negativní) aspekty týkající se aktivity, neinformovanost může způsobit špatný odhad situace,
- **rušivé vlivy:** odolnost proti nečekaným událostem je velmi důležitá,
- **časové rezervy:** uvolnit zhruba 40% času na spontánní a nečekané aktivity.

2.3.2 Kategorizace

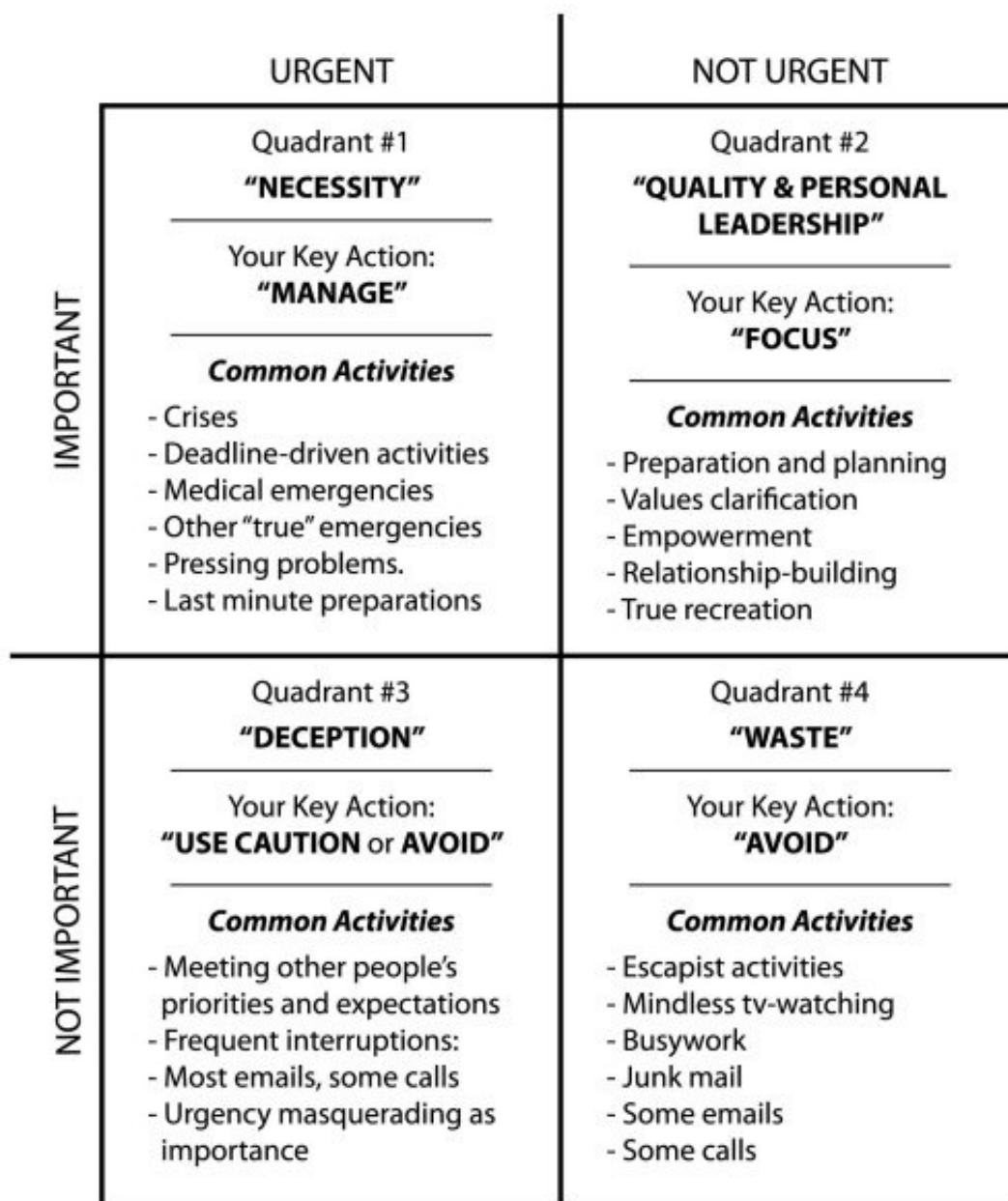
Důležitým faktorem zvyšování vlastní produktivity je vyloučení nebo alespoň omezení aktivit, které nepřináší zisk, nejsou důležité a jsou jinak pouze ztrátou času. K rozdělení aktivit do kategorií podle důležitosti a naléhavosti může pomoci metoda navržená Stephenem Coveyem z jeho knihy *First Things First (1994)* [1]. Metoda obsahuje definice 4 typů aktivit (viz obrázek 2.2): důležité a naléhavé, důležité a nenaléhavé, nedůležité a naléhavé a nedůležité a nenaléhavé. Každou aktivitu lze zařadit alespoň do jedné z těchto skupin. Úkoly se poté vyřizují podle čísla kvadrantu, v kterém jsou.

2.3.3 Priority

Při rozdělování aktivit do skupin podle důležitosti a naléhavosti je nutné přidělit každé aktivitě nějakou prioritu. K nastavení priorit slouží několik technik, například Paretova analýza nebo ABC analýza.

Paretova analýza

Myšlenkou Paretovy analýzy je, že 80% aktivit může být splněno ve 20% dostupného času. Zbylých 20% aktivit bude trvat 80% času. Aplikací této analýzy se množina aktivit rozdělí na 2 skupiny, z nichž první skupině se přiřadí vyšší priorita a bude vykonávána přednostně.



Obrázek 2.2: Stephen Covey - diagram typů aktivit (převzato z [1])

ABC analýza

Tato technika je běžně kombinována s Paretovou analýzou a využívá se také pro kategorizaci do skupin. Skupiny jsou standardně označovány písmeny A, B, C - z toho vyplývá, že je jejich počet neomezený. Aktivita se rozdělují podle kritérií:

- **A** - důležité a naléhavé aktivity (*velmi důležité*),
- **B** - důležité, ale nenaléhavé aktivity (*středně důležité*),
- **C** - nedůležité ani naléhavé (*málo důležité*).

Po zařazení jsou skupiny následně seřazeny podle priority.

3 Online kalendáře a jejich data

3.1 Kalendář

Kalendář je systém pro dělení času a orientaci v čase, jehož základní jednotkou je den. Kalendáře slouží k efektivní organizaci času, ke komunikaci ve firmě, k organizování schůzek atd. Nejznámější aplikace umožňující práci s kalendáři jsou například: Google Calendar [7], Apple iCal, Mozilla Thunderbird, Microsoft Outlook a další. Všechny tyto aplikace používají stejný formát pro reprezentaci kalendářových dat zvaný *iCalendar* definovaný standardem RFC 5545 [2]. *iCalendar* je standard pro výměnu kalendářových dat a je v současné době nejpoužívanější.

3.1.1 Využití

Kalendáře pokrývají široké spektrum akcí, které se nějakým způsobem týkají času a souvisí s časovými údaji, ať už přímo, nebo jen úzce. V mnoha případech kalendáře usnadňují práci v organizování schůzek, plánování soukromého i firemního času, připomínají důležité události a data, poskytují přehledy o časovém vytížení a další jiné výhody.

Plánování událostí je jedním z nejčastějších úkonů prováděných nad kalendářem. Událost je časové období v kalendáři mající daný začátek a konec. Mohou existovat výjimky, kdy událost nemá přesně definovaný čas, například výročí. Události je možné nastavit opakování, které má široké možnosti nastavení, např.: po dnech/týdnech/měsících, každé úterý, každé třetí úterý, každou druhou středu a pátek a tak dále. Události mohou mít nastavené upozornění, kdy uživateli přijde připomínka o blížícím se začátku dané události. Nastavení tohoto upozornění je opět velice variabilní. Díky těmto událostem a upozorněním je málo pravděpodobné, že se na nějakou důležitou událost nebo akci zapomene.

Na události je možné zvát účastníky a díky tomu plánovat schůzky a různá školení. Jelikož je možné kalendáře mezi sebou sdílet, organizátor může jednoduše zjistit, kdy mají všichni účastníci v kalendáři volno, vytvořit událost a poslat účastníkům pozvánku. Ti mají na výběr, zda pozvánku přijmou nebo odmítnou. K tomuto rozhodnutí lze připojit i nějaký popis, proč např.

uživatel schůzku nepřijal. Organizátor současně může celou událost zrušit či změnit její datum nebo čas. V obou případech dostávají účastníci informaci o provedené změně události.

Kalendáře také umožňují vytvářet úkoly, které musí být splněny v určitý den. I tomuto úkolu předchází upozornění pro uživatele. Příkladem může být přiřazování úkolů zaměstnavatele do kalendáře zaměstnance, kdy zaměstnanec ví, jaké povinnosti ho čekají.

Správa zdrojů ve firmě je další důležitou záležitostí, kterou je třeba efektivně řešit. Zdrojem může být například projektor, tabule, firemní automobil atd. Ke správě těchto zdrojů lze rovněž použít kalendáře. Ty obsahují nástroje ke zjištění dostupnosti zdroje a umožňují jeho rezervaci na konkrétní dobu. Příkladem může být rezervace zasedací místnosti. Před naplánováním schůzky potřebuje organizátor vědět, zda je místnost volná nebo obsazená. Místo složité komunikace mezi zájemci postačí sdílený kalendář, ve kterém má každý přehled o obsazenosti daného zdroje a může podle toho podnikat příslušné kroky.

3.1.2 Výhody a nevýhody

Výhody online kalendářů

- **Dostupnost** - nezáleží, kde se člověk zrovna nachází. Stačí být připojený k internetu a mít nějaké zařízení s aplikací podporující správu kalendáře (mobilní telefon, tablet, ...). Poté jsou k dispozici téměř všechny možnosti, které kalendáře nabízí (pozor, ne všechny aplikace nabízí stejné možnosti, některé konkrétní příklady jsou uvedeny později).
- **Sdílení** - sdílení kalendářů například ve firemním prostředí je velice užitečné. Díky tomu lze získat přehled o časových aktivitách a možnostech jiné osoby. Sdílení je také hojně využíváno ve správě zdrojů. Současně je možné specifikovat režimy sdílení (ekvivalence přístupových práv): provádět změny a spravovat sdílení, provádět změny událostí, zobrazit všechny podrobnosti událostí nebo je skrýt (zobrazit pouze informace o dostupnosti).
- **Synchronizace** - velmi mocný nástroj, bez kterého by bylo používání kalendářů složitější. Díky synchronizaci lze využívat více zařízení

ke správě (svých) kalendářů. Veškeré změny provedené na jednom zařízení jsou následně viditelné ve všech jiných zařízeních pouze za předpokladu existujícího připojení k internetu. Pokud je připojení nedostupné, jsou změny ukládány lokálně. K jejich promítnutí do online kalendáře dojde až bude připojení dostupné.

- **Trvanlivost informací** - při využívání online kalendářů jsou informace velmi bezpečně uloženy na serverech poskytovatele kalendáře a je velmi nepravděpodobné, že by se tato data ztratila. Dříve používané kalendáře (v papírové podobě) měly řadu nevýhod, hrozilo například poškození vodou či ohněm.
- **Elektronická podoba** - internetové kalendáře jsou ukládány v elektronické podobě. Díky tomu odpadá nutnost používání psacích potřeb a velkých fyzických kalendářů, které měly oproti online kalendářům částečně omezenou kapacitu. Současně umožňuje elektronická podoba velmi jednoduchou správu událostí: přidávání, mazání a editace.
- **Vyhledávání** - pokud je znám název či předmět události (nebo jen část), je velice jednoduché událost ve zlomku času vyhledat. U fyzických kalendářů je nutné ruční prohledávání.
- **Podpora aplikací** - aplikací pro správu kalendářů je mnoho. Standardní akce prováděné nad kalendářem (tj. správa událostí, sdílení, synchronizace atd.) podporují téměř všechny. Každá aplikace poté navíc podporuje určité kalendářové akce a nastavení, které naopak jiná nepodporuje.

Nevýhody online kalendářů

- **Připojení k internetu** - aby mohly být internetové kalendáře plně využívány, je nutné být co nejčastěji (téměř stále) připojen k internetu. To samozřejmě není vždy možné, internetové připojení například nemusí být zrovna k dispozici.
- **Elektronické zařízení** - správa kalendáře přes počítač je samozřejmá. Určitě ale někdy nastane situace, kdy u něj člověk zrovna není a potřeboval by uložit důležité datum či událost. V tu chvíli se velmi hodí nějaké zařízení s podporou kalendářů. Tímto zařízením může být například mobilní telefon či tablet připojený k internetu.

- **První setkání** - první setkání s online kalendářem může být matoucí a ne příliš pozitivní. Jelikož jde o mírně odlišný způsob plánování času, může člověk nabývat pocitu, že se kalendář nikdy nenaučí používat. Jeho výhody ovšem převažují nad výhodami fyzických kalendářů a investice času k jeho poznání se později určitě vyplatí.

Konkrétní příklady

Jak již bylo řečeno, každá kalendářová aplikace podporuje mírně odlišné služby a možnosti. Typickou prací s kalendáři ale zvládají všechny. Zde je uvedeno pár rozdílů konkrétních aplikací, jak se mohou mezi sebou lišit.

Aplikace Google Calendar neumí vyhledávat podle části slova, který obsahuje předmět události. Příklad: v kalendáři je událost s předmětem *Petrovy narozeniny*. Při hledání části „Petr“ aplikace Google Calendar událost nenajde. Při hledání slova „Petrovo“ už ano. Oproti tomu aplikace Microsoft Outlook 2013 a Yahoo! Calendar událost naleznou. Zajímavě se chová aplikace iCal od Applu, která událost vyhledá, když hledaným slovem je „Petr“. Pokud hledaným slovem bude „etrov“, už událost nenalezne.

Kategorizaci událostí aplikace Google Calendar standardně nepodporuje. Jedinými možnostmi, jak události nějakým způsobem od sebe rozlišit, je vkládání do nových kalendářů, kde by každý kalendář reprezentoval jednu kategorii. V přehledu poté budou mít události ze svých kalendářů barvu právě toho kalendáře. Dalším způsobem rozlišení může být nastavení barvy události. Aplikace iCal kategorizaci také neumožňuje a taktéž ji řeší použitím více kalendářů, z nichž každý má jinou barvu. Lépe již kategorizaci řeší aplikace Microsoft Outlook 2013 a Yahoo! Calendar, ve kterých se při vytváření nové události může specifikovat, do jaké kategorie patří. V aplikaci Microsoft Outlook 2013 lze navíc vytvořit kategorii novou a událost do ní přiřadit. Události v Microsoft Outlook 2013 mohou mít oproti Yahoo! Calendar kategorií více.

3.2 Formát iCalendar

Aby bylo možné s kalendářovými daty nějak pracovat, je nutné definovat strukturu (formát), jakým způsobem se budou zapisovat a reprezentovat a jakým způsobem bude probíhat jejich výměna. Vytvoření jednotného formátu usnadňuje komunikaci vývojářů kalendářových aplikací, neboť každý ví, jak jsou kalendářová data definována. Výhodou je taktéž přenositelnost mezi

těmito aplikacemi, pokud všechny podporují stejný standard. Tímto standardem je *iCalendar*, též známý pod zkratkou *iCal* (iCal je současně jméno kalendářové aplikace firmy Apple).

iCal není prvním formátem, který se používá k výměně kalendářních a plánovacích informací v rámci internetu. iCalu předcházela formát vCalendar sloužící taktéž k výměně kalendářních informací. Oproti vCalendaru je formát iCal nezávislý na přenášeném protokolu.

3.2.1 Historie

První verze standardu, označená RFC 2245, byla vydána v listopadu roku 1998 organizací IETF¹ zabývající se internetovými standardy a byla založena na starším formátu vCalendar. vCalendar vydala společnost IMC², která dnes již neexistuje. Prvním updatem prošel iCal v roce 2009. Důvodem bylo rozšíření standardu internetové pošty MIME³ právě o iCal (typ obsahu *text/calendar*) a současně nezávislost na přenášeném protokolu. Nový standard nese označení RFC 5545, je nejpoužívanějším a nejrozšířenějším formátem pro výměnu kalendářových dat a jeho autorem je Bernard Desruisseaux.

3.2.2 Popis souboru

Soubor typu iCalendar má příponu .ics a obsahuje prostý ASCII text. Standardní kódování iCalu je UTF-8. Použití jiného kódování je podmíněno nastavením MIME parametru „charset“ při přenosu. Maximální délka řádku v souboru by neměla přesáhnout 75 bajtů (pozor, ne znaků). Pokud délka řádky přesahuje 75 bajtů, měla by být rozdělena do více řádek. Po rozdělení se před každý nový řádek vloží bílý znak (mezera či tabulátor), a díky tomu lze rozpoznat, že tato řádka je pokračováním předchozí. Řádka je ukončena znaky CR+LF.

¹Internet Engineering Task Force

²Internet Mail Consortium

³Multipurpose Internet Mail Extensions

3.2.3 Struktura

Každý .ics soubor představuje právě jeden kalendář, který je definován první a poslední řádkou v souboru. Mezi těmito řádkami je definované tělo kalendáře. To se skládá z vlastností daného kalendáře a alespoň jedné komponenty (viz 3.2.4).

První řádka souboru⁴, BEGIN:VCALENDAR, představuje začátek kalendáře a poslední řádka, END:VCALENDAR, představuje konec kalendáře. Tyto řádky mají pevně dané místo a nelze je měnit. Další řádkou v souboru je informace o verzi použitého formátu. VERSION:2.0 indikuje, že data jsou ve formátu iCalendar, naopak VERSION:1.0 značí, že se jedná o formát vCalendar. Další vlastností objektu kalendáře musí být identifikátor označující autora či produkt, který kalendář vytvořil. Například kalendář aplikace Google Calendar má identifikátor PRODID:-//Google Inc//Google Calendar 70.9054//EN.

Ukázka kalendáře:

```
BEGIN:VCALENDAR
VERSION:2.0
PRODID:-//Google Inc//Google Calendar 70.9054//EN
BEGIN:VEVENT
UID:19970610T172345Z-AF23B2@example.com
DTSTAMP:19970610T172345Z
DTSTART:19970714T170000Z
DTEND:19970715T040000Z
SUMMARY:Bastille Day Party
END:VEVENT
END:VCALENDAR
```

Dále se uvnitř kalendáře definují komponenty, viz 3.2.4.

3.2.4 Komponenty

Kalendáře standardně obsahují nějaká data, například události, úkoly atd. Tato data se nazývají komponenty. Komponenta kalendáře je datová oblast v souboru definovaná svým prvním a posledním řádkem představující kon-

⁴Většina ukázek v této kapitole je převzata z RFC 5545 [2].

krétní prvek kalendáře. Všechny komponenty začínají písmenem V a jsou následující:

- **VEVENT** - skupina vlastností popisující událost,
- **VTODO** - skupina vlastností popisující úkol,
- **VJOURNAL** - skupina vlastností popisující záznam v deníku,
- **VFREEBUSY** - skupina vlastností popisující požadavek na informaci o volném/obsazeném čase, odpověď na tento požadavek či pouze oznámení o obsazeném čase,
- **VALARM** - skupina vlastností popisující alarm,
- **VTIMEZONE** - skupina vlastností definující časovou zónu.

Komponenta VEVENT

Komponenta VEVENT popisuje událost zabírající v kalendáři nějaký časový úsek. Příkladem může být například návštěva doktora zítra od 10:00 do 11:00. Události jsou standardně zahrnuty jako „obsazený“ čas pro komponentu VFREEBUSY. Čas zabraný událostí lze uvolnit nastavením vlastnosti TRANSPARENT dané události.

Příklady vlastností komponenty VEVENT:

- **DTSTAMP** - datum a čas vytvoření objektu iCalendar, datum a čas je uváděn ve formátu UTC (*povinná vlastnost*),
- **UID** - jednoznačná globální identifikace komponenty (*povinná vlastnost*),
- **SUMMARY** - definice předmětu kalendářové komponenty (ne nutně události) (*volitelná vlastnost*),
- **CREATED** - datum a čas vytvoření komponenty (*volitelná vlastnost*),
- **RRULE** - vlastnost definující pravidlo či opakovací vzor pro komponenty, které se mohou opakovat (události, úkoly, záznamy v deníku, definice časových zón) (*volitelná vlastnost*),

- **CATEGORIES** - definice kategorií, do kterých událost patří. Tyto kategorie by se daly chápat jako klíčová slova, kdyby je všechny kalendářové aplikace podporovaly (více v 3.1.2) (*volitelná vlastnost*),
- *a další.*

Vlastnost RRULE je velmi často používána. Vyskytuje se u komponent, které se nějakým způsobem opakují. Příklady opakování mohou být:

- RRULE:FREQ=DAILY;COUNT=10
(10x, opakování denně),
- RRULE:FREQ=MONTHLY;COUNT=10;BYDAY=1FR
(10x, první pátek v měsíci),
- RRULE:FREQ=MONTHLY;INTERVAL=2;BYDAY=TU
(každé úterý každý druhý měsíc),
- RRULE:FREQ=WEEKLY;UNTIL=20141224T000000Z
(týdenní opakování až do 24. prosince 2014).

Příklady událostí

Událost v kategorii *School*, která začíná 03.09.2013 v 16:30:00 UTC, končí 03.09.2013 v 19:00:00 UTC a má předmět *Sraz ze školy*, může mít následující iCal reprezentaci:

```
BEGIN:VEVENT
UID:20130901T130000Z-123401@example.com
DTSTAMP:20130901T130000Z
DTSTART:20130903T163000Z
DTEND:20130903T190000Z
SUMMARY:Sraz ze školy
CATEGORIES:SCHOOL
END:VEVENT
```

Událost, která má trvání jeden den a opakuje se vždy od 02.10.2013 každý rok, může vypadat v iCal formátu následovně:


```
BEGIN:VEVENT
UID:20130901T130000Z-123403@example.com
DTSTAMP:20130901T130000Z
DTSTART;VALUE=DATE:20131002
SUMMARY:Our Blissful Anniversary
TRANSP:TRANSPARENT
CATEGORIES:ANNIVERSARY,PERSONAL,SPECIAL OCCASION
RRULE:FREQ=YEARLY
END:VEVENT
```

Komponenta VTODO

Komponenta VTODO poskytuje sadu vlastností reprezentující úkol. Úkol může být jednorázový nebo může mít nastavené nějaké opakování (vlastnost RRULE). Současně tato komponenta nemůže být sama o sobě vnořena v jiné komponentě, ale může být k jiné komponentě vztažena pomocí vlastnosti RELATED-TO.

Příklad úkolu, který musí být splněn do 1. května 2014:

```
BEGIN:VTODO
UID:20140313T123432Z-456553@example.com
DTSTAMP:20140313T123432Z
DUE;VALUE=DATE:20140501
SUMMARY:Submit Quebec Income Tax Return for 2006
CLASS:CONFIDENTIAL
CATEGORIES:FAMILY,FINANCE
STATUS:NEEDS-ACTION
END:VTODO
```

Komponenta VJOURNAL

Tato komponenta reprezentuje záznam do deníku. Záznamem do deníku se myslí nějaký text spojený s určitým datem. Jelikož tato komponenta nezabírá v kalendáři žádný čas, má automaticky nastavenou vlastnost TRANSP na hodnotu TRANSPARENT. Díky tomu neblokuje žádná vyhledávání volného či obsazeného času komponentou VFREEBUSY. Komponenta VJOURNAL může být taktéž vztažena k jiné komponentě vlastností RELATED-TO.

Příklad záznamu do deníku:

```
BEGIN:VJOURNAL
UID:19970901T130000Z-123405@example.com
DTSTAMP:19970901T130000Z
DTSTART;VALUE=DATE:19970317
SUMMARY:Staff meeting minutes
DESCRIPTION:1. Staff meeting: Participants include Joe\,
  Lisa\, and Bob. Aurora project plans were reviewed.
  There is currently no budget reserves for this project.
  Lisa will escalate to management. Next meeting on Tuesday.\n
2. Telephone Conference: ABC Corp. sales representative
  called to discuss new printer. Promised to get us a demo by
  Friday.\n3. Henry Miller (Handsoff Insurance): Car was
  totaled by tree. Is looking into a loaner car. 555-2323
  (tel).
END:VJOURNAL
```

Komponenta VFREEBUSY

K rezervaci či zjišťování dostupnosti zdrojů slouží komponenta VFREEBUSY. Díky ní lze vytvářet požadavky na zjištění dostupnosti zdroje, odpovídat na tyto požadavky a nebo lze napevno nastavit obsazenost zdroje nebo časového intervalu.

Příklad požadavku na zjištění volného času (ORGANIZER je tazatel, ATTENDEE je tázaný):

```
BEGIN:VFREEBUSY
UID:19970901T082949Z-FA43EF@example.com
ORGANIZER:mailto:jane\_doe@example.com
ATTENDEE:mailto:john\_public@example.com
DTSTART:19971015T050000Z
DTEND:19971016T050000Z
DTSTAMP:19970901T083000Z
END:VFREEBUSY
```

Komponenta VALARM

Tato komponenta slouží k reprezentaci alarmu nebo upozornění k nějaké události nebo úkolu a musí obsahovat vlastnosti ACTION a TRIGGER. Vlastnost ACTION určuje, co se má stát po spuštění alarmu (například přehrání zvuku, zobrazení hlášky či odeslání emailu). Vlastnost TRIGGER určuje, kdy bude alarm spuštěn.

Příklad alarmu přehrávajícího zvuk 4x v 15ti minutových intervalech:

```
BEGIN:VALARM
TRIGGER;VALUE=DATE-TIME:19970317T133000Z
REPEAT:4
DURATION:PT15M
ACTION:AUDIO
ATTACH;FMPTYPE=audio/basic:ftp://example.com/pub/
    sounds/bell-01.aud
END:VALARM
```

Komponenta VTIMEZONE

Komponenta VTIMEZONE slouží k definici časové zóny. Časových zón může kalendář obsahovat více, pokud mají jedinečné TZID. Mezi hlavní vlastnosti časové zóny patří zmíněné TZID. TZID je textová hodnota, která je pro každou časovou zónu unikátní. Každá komponenta VTIMEZONE musí dále definovat alespoň jednu ze subkomponent STANDARD (ekvivalence zimního času) nebo DAYLIGHT (ekvivalence letního času). Obě tyto subkomponenty musí dále definovat vlastnosti TZOFFSETFROM a TZOFFSETTO, které definují časový posun (offset) v dané časové zóně oproti UTC (Coordinated Universal Time). Časové zóny mohou také obsahovat opakování.

Příkladem může být definice časové zóny kalendáře pro Českou republiku.

```
BEGIN:VTIMEZONE
TZID:Europe/Prague
X-LIC-LOCATION:Europe/Prague
BEGIN:DAYLIGHT
TZOFFSETFROM:+0100
```

```
TZOFFSETTO:+0200
TZNAME:CEST
DTSTART:19700329T020000
RRULE:FREQ=YEARLY;BYMONTH=3;BYDAY=-1SU
END:DAYLIGHT
BEGIN:STANDARD
TZOFFSETFROM:+0200
TZOFFSETTO:+0100
TZNAME:CET
DTSTART:19701025T030000
RRULE:FREQ=YEARLY;BYMONTH=10;BYDAY=-1SU
END:STANDARD
END:VTIMEZONE
```

Z příkladu je zřejmé opakování změn vždy poslední neděli v březnu a říjnu z důvodu střídání letního a zimního času. Jsou také vidět časové posuny (TZOFFSETFROM a TZOFFSETTO) v obou subkomponentách (DAYLIGHT a STANDARD).

3.2.5 Využití a použití

Standard iCalendar je určen k popisu kalendářových dat a je to současně nej-používanější standard v této oblasti. Tento formát je podporován mnoha kalendářovými aplikacemi, například Google Calendar, iCal (Apple), Microsoft Outlook, Yahoo! Calendar, Mozilla Thunderbird a další. Standard iCalendar je navržen tak, aby byl nezávislý na přenášeném protokolu. Sám o sobě ani nedefinuje, jak kalendářová data přenášet.

Pro přenos těchto dat se používají jiné protokoly. Obecně nejznámějším prokem pro transport kalendářních informací je protokol CalDAV (RFC 4791⁵). CalDAV současně umožňuje přístup k těmto datům na straně serveru a jejich správu. Dalším protokolem může být iTIP (RFC 5546⁶).

⁵<http://tools.ietf.org/html/rfc4791>

⁶<http://tools.ietf.org/html/rfc5546>

3.2.6 Knihovny

Existuje mnoho knihoven pro různé programovací jazyky, které poskytují nástroje pro správu souborů ve formátu iCalendar. Mezi nejznámější knihovny pro programovací jazyk Java patří:

- **iCal4j** - <http://wiki.modularity.net.au/ical4j/>,
- **biweekly** - <http://sourceforge.net/projects/biweekly/>,
- **Free Association (libical)** - <http://sourceforge.net/projects/freeassociation/>,
- **Mulberry iCalendar library** - <http://trac.mulberrymail.com/repos/wiki/icalendar>.

4 Zadání

Cílem práce je vytvoření aplikace sloužící ke generování časových statistik z kalendářových dat. Aplikace bude umět zpracovat soubory ve formátu iCalendar, vypočítat z nich příslušné statistické informace (dle nastavení uživatele) a následně statistiky uživateli zobrazit. Dále bude možné parsovat¹ klíčová slova (viz 5.3.2) z předmětů událostí zvoleného kalendáře. Klíčová slova slouží jako popisy skupin událostí nebo kategorií, do kterých lze události zařazovat a budou hlavním podkladem pro počítání statistik.

4.1 Účel

Aplikace byla vytvořena za účelem počítání statistik z kalendářových dat. Výstupy aplikace obsahují sumarizace trvání událostí za určitá časová období. Tyto sumarizace mohou poskytovat přehled o časové náročnosti daného úkolu či mohou posloužit jako podklady při vykazování stráveného času. Více o využití aplikace lze najít v sekci 1.2.

4.2 Funkčnosti aplikace

Postup generování statistik se dá považovat za krátké workflow² se 4 hlavními kroky:

1. načtení kalendáře / kalendářů,
2. nastavení parsování klíčových slov,
3. nastavení statistik,
4. export.

Načtení kalendáře / kalendářů

Aplikace umožňuje vložení více kalendářů (neboli více souborů ve formátu

¹parsování = proces rozdělování textu na menší části vyhovující nějakému vzoru

²workflow - pracovní postup, průběh pracovní operace

iCalendar). Soubory je možné do aplikace vkládat buď z lokálního umístění v počítači nebo přes adresu URL v internetu. Reference na tyto kalendáře je možné uložit do souboru ve formátu XML. Soubor má název *data_ocr.xml*, kde „ocr“ v názvu znamená „ical representation“. Po spuštění se aplikace pokusí tento soubor najít a načíst z něho dříve uložené reference. Je důležité mít na paměti, že aktualizace tohoto souboru se provádí manuálně stisknutím tlačítka *Save*. Pokud uživatel přidá nějaký kalendář a neuloží, po restartu aplikace kalendář v seznamu nebude.

Nastavení parsování klíčových slov

Klíčová slova je možné do aplikace nahrát buď ze souboru (formát CSV³) nebo je vyparsovat. Nastavení parsování klíčových slov:

- **zdrojový kalendář:** kalendář, z jehož událostí se klíčová slova parsují,
- **minimální výskyt:** minimální výskyt klíčového slova,
- **regulární výraz:** hledání klíčových slov odpovídajících regulárnímu výrazu,
- **časové období:** nastavení časového období, ze kterého se budou klíčová slova parsovat,
- **odstranění diakritiky:** ignorace diakritiky,
- **rozlišení velkých a malých písmen:** ignorace velkých a malých písmen.

Do seznamu vyparsovaných klíčových slov lze dále přidat i nějaké výjimečné klíčové slovo, které nevyhovovalo nastavení, ale i přesto je považováno za klíčové. Současně je možné i nějaká odebrat.

Nastavení statistik

Nastavení statistik má být variabilní a obsahovat následující možnosti:

- **granularita:** spojování událostí do skupin podle dnů, týdnů, měsíců a roků,
- **jednotka výstupu:** specifikace jednotky výstupu (minuty, hodiny nebo dny),

³Coma-separated values

- **zahrnutí jednotlivých událostí:** vypsání jednotlivých událostí v časové skupině,
 - **zahrnout datum:** zobrazení data události,
 - **zahrnout den:** zobrazení dne v týdnu události,
 - **zahrnout od-do:** zobrazení času začátku a konce události,
- **zahrnout prázdné řádky:** zobrazení skupin spadajících do časového intervalu neobsahujících žádnou událost,
- **časové období:** nastavení časového období, z kterého se budou statistiky generovat,
- **výstupní formát:** nastavení výstupního formátu statistik.

Export

Veškeré nastavení zvolené v předchozích krocích je nyní aplikováno do generování samotných statistik. Jelikož statistické informace jsou strukturovány do tabulky, je nejvhodnějším formátem uložení formát aplikace Microsoft Excel XLS nebo XLSX. Současně bude umožněn export do formátu HTML.

Po úspěšném vygenerování do vybraného formátu se statistiky buď uloží na zvolené místo, nebo pouze otevřou. Pokud by při generování nastala chyba, je o ní uživatel informován a chyba je logována.

5 Implementace

5.1 Architektura aplikace

Architektura aplikace je navržena tak, aby byla aplikace jednoduše přenositelná na jiné platformy. Jádro se při přenosu na jinou platformu či jiný typ aplikace (např. webová aplikace) nemění. Aplikace se ovládá grafickým uživatelským rozhraním, které je oddělené od aplikační a datové vrstvy. Aplikační vrstva představuje jakési jádro aplikace.

5.2 Popis architektury

Architektura je standardně rozdělena na 3 části (viz obrázek 5.1): jádro, datová vrstva a vrstva uživatelského rozhraní (prezentační vrstva). Aplikace se řídí příkazy z prezentační vrstvy, ve které jsou prováděny hlavní úkony uživatele.

Datová třída provádí základní operace s iCal soubory a soubory s vygenerovanými statistikami. Současně tato vrstva poskytuje standardní nástroje pro práci se soubory (ukládání, otevírání, vytváření streamů apod.).

Jádro aplikace tvoří třídy zajišťující responzivnost uživatelského rozhraní při déletrvajících operacích a třídy pro generování samotných statistik.

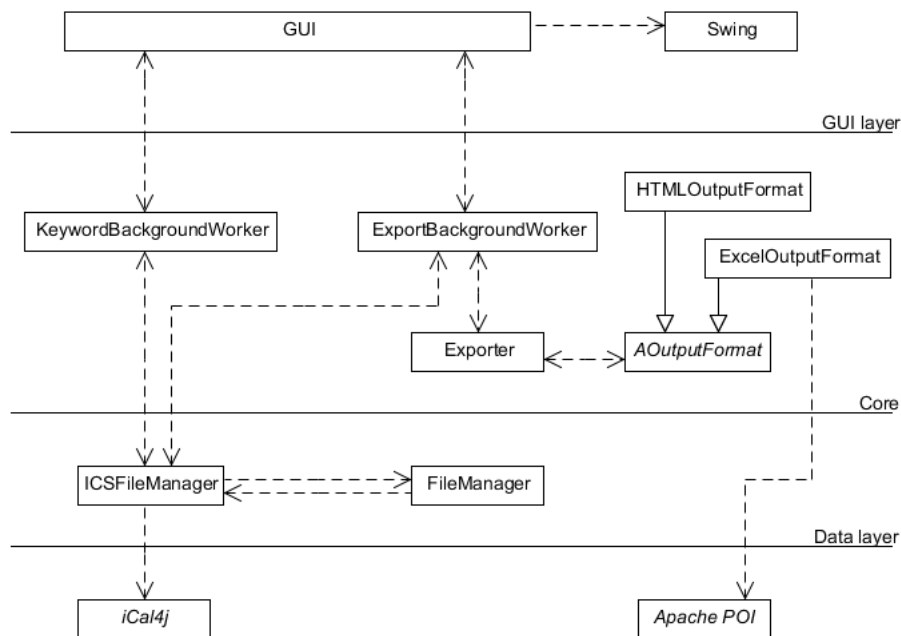
5.2.1 Technologie

Aplikace byla vyvíjena v programovacím jazyce Java verze 7 a používá další známé související technologie, například Swing pro tvorbu uživatelských rozhraní a různé knihovny umožňující integraci jiných komponent.

Použité knihovny

V aplikaci vytvořené v rámci této bakalářské práce byla pro práci s kalendářovými daty zvolena knihovna iCal4j¹ verze 1.0.5. V oblasti zpracování

¹iCal4j: http://wiki.modularity.net.au/ical4j/index.php?title=Main_Page



Obrázek 5.1: Základní design architektury aplikace

iCal souborů je jednou z nejpoužívanějších. Její dokumentace, byť není příliš rozsáhlá, obsahuje ukázkové příklady a návody, jak knihovnu používat. Ve srovnání například s knihovnou biweekly je iCal4j mírně složitější, zato podporuje širší spektrum možností, které lze nad kalendářovými daty provádět. Licence knihovny je dostupná zde: <http://m2.modularity.net.au/projects/ical4j/license.html>.

Ke generování výstupu do formátů XLS a XLSX byla použita knihovna Apache POI² verze 3.10. Tato Java knihovna se skládá z komponent poskytujících API pro čtení a zápis souborů ve formátu aplikací společnosti Microsoft z balíku Microsoft Office. Použité komponenty v této aplikaci jsou HSSF³ (pro souborový formát XLS) a XSSF⁴ (pro souborový formát XLSX). Obě komponenty pracují se sešity aplikace Microsoft Excel. Knihovna dále podporuje OOXML⁵, což je souborový formát navržený společností Microsoft pro ukládání kancelářských balíčků. Licence této knihovny je dostupná

²**Apache POI:** <http://poi.apache.org/>

³**HSSF** = Horrible SpreadSheet Format

⁴**XSSF** = XML SpreadSheet Format

⁵**OOXML** = Office Open XML

zde: <http://www.apache.org/licenses/>.

Aplikace dále ke svému bezchybnému běhu potřebuje další knihovny (knihovny jsou již přibalené v souboru aplikace *.jar*). Včetně iCal4j a Apache POI se jedná o tyto knihovny:

- *backport-util-concurrent-3.1.jar*,
- *commons-codec-1.8.jar*,
- *commons-io-1.4.jar*,
- *commons-lang-2.6.jar*,
- *commons-logging-1.1.3.jar*,
- *dom4j-1.6.1.jar*,
- *ical4j-1.0.5.jar*,
- *jcalendar-1.4.jar*,
- *joda-time-2.3.jar*,
- *poi-3.10-FINAL-20140208.jar*,
- *poi-ooxml-3.10-FINAL-20140208.jar*,
- *poi-ooxml-schemas-3.10-FINAL-20140208.jar*,
- *xmlbeans-2.3.0.jar*.

5.2.2 Průchod architekturou

Nejprve je nutné vysvětlit význam tříd `KeywordBackgroundWorker` a `ExportBackgroundWorker`. Obě třídy jsou potomky třídy `SwingWorker` a umožňují provádět déletrvající operace ve vedlejších vláknech. To zaručí responzivnost uživatelského rozhraní, které díky tomu v pořádku reaguje na požadavky uživatele, zatímco ve vedlejší vláknech se v případě třídy `KeywordBackgroundWorker` parsují klíčová slova a v případě třídy `ExportbackgroundWorker` se generují statistiky. Aplikace není na těchto třídách závislá. Jsou přítomny jen proto, aby byla prezentační vrstva responzivní, jelikož generování některých statistik může trvat dlouhou dobu.

Průchod architekturou pro získání statistik je následující. Nejprve je nutné přidat kalendáře, ze kterých se budou statistiky generovat. Po přidání těchto referencí se pomocí třídy `KeywordBackgroundWorker` vyparsují klíčová slova. To probíhá načtením kalendáře, který byl zvolen jako zdroj dat. Instanci třídy `ICalFile` reprezentující kalendář vytvoří třída `ICSFileManager` buď ze zadané URL nebo z lokální cesty. Po načtení kalendáře zavolá `backgroundworker` metodu kalendáře, která podle uživatelského filtru vyparsuje klíčová slova a vrátí je prezentační vrstvě k zobrazení. (*Poznámka: klíčová slova není nutné parsovat, lze je vložit z již existujícího souboru*).

Po nastavení výstupních statistik uživatel vybere zdrojové kalendáře. Ty jsou dle architektury předány třídě `ExportBackgroundWorker`, jejíž význam je vysvětlen výše. Tato třída komunikuje s třídou `Exporter` zařizující export statistik. Třída `Exporter` (dle nastavení uživatele) zvolí výstupní formát a předá řízení třídě pro příslušný formát. Těmito třídami jsou `HTMLOutputFormat` a `ExcelOutputFormat` mající společného předka, abstraktní třídu `AOutputFormat` s abstraktní metodou `generateStatistics()`. Vygenerované statistiky v podobě instance třídy `File` jsou vráceny třídou `Exporter` zpět do `ExportBackgroundWorkeru`, který soubor nakonec vrátí až do prezentační vrstvy.

Tímto je celý proces generování statistik dokončený.

5.3 Zpracování dat

5.3.1 Události

Pro práci s událostmi z iCal souboru je potřeba vytvořit jejich programovou reprezentaci. Touto reprezentací je třída `MyEvent` fungující jako wrapper⁶ třídy `VEvent` (z knihovny `iCal4j`) představující samotnou událost. Tento wrapper byl vytvořen z důvodu implementace rozhraní `Comparable` umožňující řazení objektů definovaným způsobem. V případě událostí se objekty řadí podle data a času začátku události. Dalším úkolem wrapperu je přidat třídě `VEvent` v podobě metod funkcionalitu, která bude velmi často používána a která usnadní práci s tímto objektem. Příkladem může být například metoda `getDuration()`, která vrátí trvání události v zadaných jednotkách.

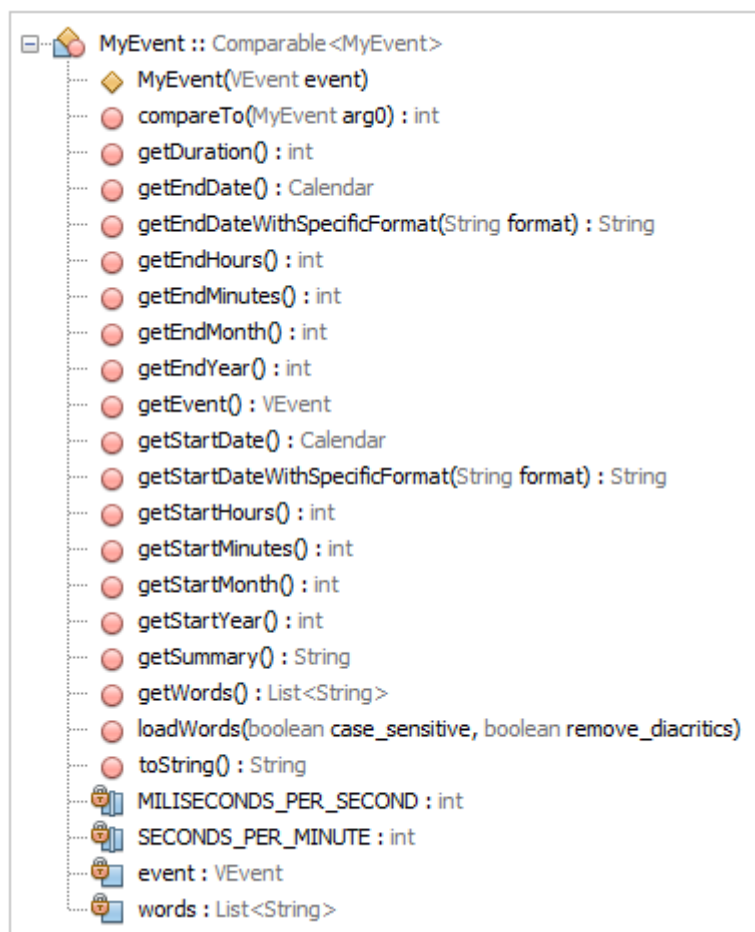
⁶**wrapper** = obalení či zapouzdření funkčnosti jiné třídy

```
public double getDuration(Units unit)
    throws IllegalArgumentException{
    Calendar from = null, to = null;
    double duration = 0;
    from = Calendar.getInstance();
    to = Calendar.getInstance();
    from.setTime(event.getStartDate().getDate());
    to.setTime(event.getEndDate().getDate());
    duration = to.getTime().getTime() /
                MILLISECONDS_PER_MINUTE;
    duration -= from.getTime().getTime() /
                MILLISECONDS_PER_MINUTE;
    switch(unit){
        case Minutes: ; break;
        case Hours: duration = duration / MINUTES_PER_HOUR;
                    break;
        case Days: duration = duration / MINUTES_PER_HOUR /
                    HOURS_PER_DAY;
                 break;
        default: throw
            new IllegalArgumentException("Unknow unit: "+unit);
    }
    return duration;
}
```

Prvním zpracováním tedy projdou všechny události ihned po vytvoření objektu kalendáře (instance třídy `ICalFile`). Tato třída má privátní atribut `events` typu `TreeSet<MyEvent>`. Po vložení všech událostí do této kolekce jsou všechny události seřazeny podle data začátku a snáze se s nimi poté pracuje. Na obrázku 5.2 je vidět struktura třídy `MyEvent`.

Zvláštní pozornosti se dostává událostem, které mají nastavené nějaké pravidlo opakování. Po načtení všech událostí dochází k filtraci těch, které nemají nastavený konec opakování nebo počet opakování. Tyto události jsou ze seznamu vyjmuty a nejsou do statistik zahrnuty.

Naopak události, které mají nastavený počet opakování nebo datum, kdy opakování končí, do statistik zahrnuty jsou. Takové události musí projít tzv. rozbalovacím procesem. Jelikož opakovaná událost je v iCal souboru reprezentována pouze jedním záznamem, je nutné takovou událost rozbalit do tolika událostí, kolik jich definuje její pravidlo opakování. Následuje ukázka metody



Obrázek 5.2: Struktura třídy MyEvent

`unwrapRepeatedEventsCOUNT()` třídy `ICalFile`, která rozbaluje opakované události s definovaným počtem opakování.

```
private TreeSet<MyEvent> unwrapRepeatedEventsCOUNT
    (MyEvent repEvent, int count)
    throws ParseException, IOException, URISyntaxException{
    Property propRRULE = null;
    Recur recurrence = null;
    VEvent copy = null;
    VEvent repeatedEvent = repEvent.getEvent();
    TreeSet<MyEvent> events = new TreeSet<>();
    net.fortuna.ical4j.model.Date seedSTART =
        repeatedEvent.getStartDate().getDate();
```

```
net.fortuna.ical4j.model.Date seedEND =
    repeatedEvent.getEndDate().getDate();
net.fortuna.ical4j.model.Date start =
    repeatedEvent.getStartDate().getDate();
net.fortuna.ical4j.model.Date end =
    repeatedEvent.getEndDate().getDate();

propRRULE = repeatedEvent.getProperty(Property.RRULE);
recurrence = new Recur(propRRULE.getValue());

for(int i = 0 ; i < count-1 ; i++){
    start = recurrence.getNextDate(seedSTART, start);
    end = recurrence.getNextDate(seedEND, end);
    copy = (VEvent) repeatedEvent.copy();
    copy.getStartDate().setDate(start);
    copy.getEndDate().setDate(end);
    events.add(new MyEvent(copy));
}

return events;
}
```

Nyní dochází ke zpracování předmětu události. Předmět události je rozdělen na slova a dle volby uživatele je aplikováno nastavení diakritiky nebo velkých a malých písmen. Pokud si uživatel přeje velká a malá písmena ignorovat, je celý předmět události převeden na malá písmena.

Nejprve se odstraní závorky, tečky, čárky a jiné znaky, které nesouvisí se slovem, a poté se zbylý text rozdělí podle mezery. Vzniklá slova se uloží ke každé události a jsou později využita při parsování klíčových slov nebo při hledání, zda událost nějaké klíčové slovo obsahuje.

Posledním místem, kde se s událostmi pracuje, je generování statistik. Události vyhovující uživatelskému filtru (např. obsahující hledané klíčové slovo) jsou zahrnuty do generování a z dob jejich trvání se počítají výsledné součty.

5.3.2 Klíčová slova

Klíčové slovo je takové slovo, které se opakuje v předmětech událostí a z pohledu uživatele hraje nějakým způsobem důležitou roli. Označování předmětů událostí klíčovými slovy umožňuje události pomyslně třídit do určitých skupin (či kategorií), kdy každá skupina je definována právě nějakým klíčovým slovem. To se hodí například u aplikace Google Calendar, která sama o sobě kategorizaci událostí neumožňuje. Klíčové slovo dále neobsahuje bílé znaky a je omezeno zleva i zprava mezerou (nebo začátkem a koncem věty apod.).

Zpracování klíčových slov

Parsování klíčových slov z předmětů událostí neprobíhá ihned, ale až po zavolání metody `getKeywordsList()` třídy `ICalFile`, které je současně předán filtr klíčových slov. Filtr obsahuje nastavení ohledně diakritiky, datového intervalu či ignoraci velkých a malých písmen. Poté se prochází všechny události a zjišťuje se, jestli událost patří do časového intervalu. Pokud ne, přeskočí se, pokud ano, slova vygenerovaná při načtení události se postupně uloží do mapy, jejímž klíčem je slovo a hodnotou je výskyt tohoto slova.

Jakmile je výskyt roven minimálnímu výskytu, je slovo vloženo do množiny klíčových slov. Před vrácením seznamu s klíčovými slovy se každé klíčové slovo otestuje proti regulárnímu výrazu (pokud nějaký uživatel zadal). Pokud slovo výrazu odpovídá, zůstává v seznamu. Neodpovídající slova jsou odebrána. Nakonec je seznam se zbylými klíčovými slovy vrácen jako návratová hodnota metody.

Ukázka metody `getKeywordsList()` třídy `ICalFile` vracející seznam klíčových slov vyhovujících filtru. Po začátku metody je volána metoda `parseWordsInEvents()` s 2 parametry z filtru. Tato metoda zajistí zpracování předmětu každé události ve smyslu vytvoření pole slov s odstraněnou diakritikou (v případě ignorace diakritiky) a převedením na malá písmena (v případě ignorace velkých a malých písmen).

```
public List<Keyword> getKeywordsList(KeywordsFilter filter){
    HashMap<String, Keyword> map = new HashMap<>();
    Set<Keyword> keywords_set = new HashSet<>();
    Pattern pattern = null;
    Keyword keyword;
```



```
List<String> words = null;
parseWordsInEvents(filter.case_sensitive,
    filter.ignore_diacritics);
for(MyEvent event: events){
    words = event.getWords();
    if(filter.isWholeCalendar() ||
        (event.getStartDate().after(filter.getStartDate()) &&
        event.getStartDate().before(filter.getEndDate()))){
        //date is OK
        for (String word : words) {
            if(word.equals(""))
                continue;
            keyword = map.get(word);
            if (keyword != null)
                keyword.setOccurrence
                    (keyword.getOccurrence() + 1);
            else
                keyword = new Keyword(word, 1);
            map.put(word, keyword);
            if(keyword.getOccurrence() >=
                filter.getOccurrence())
                keywords_set.add(keyword);
        }
    }
}
//check if keywords matches to regex
if((pattern = filter.getPattern()) != null){
    Iterator it = keywords_set.iterator();
    while(it.hasNext())
        if(!pattern.matcher(((Keyword) it.next())
            .getKeyword()).matches())
            it.remove();
}
return new ArrayList<>(keywords_set);
}
```

5.4 Generování výstupu

Jelikož obsah a strukturu statistik nejvíce ovlivňuje zvolená granularita, je pro každý její typ vytvořena speciální metoda. Každý typ granularity současně potřebuje data v mírně odlišném formátu. K tomu je připravena třída `DataTransformer` zajišťující předání dat statistické třídy ve vhodné struktuře. `DataTransformer` funguje tedy jako jakási datová nadstavba statistické třídy poskytující požadovaná data. Při vytváření její instance se jí předá seznam všech událostí a poté je podle volané metody seznam zpracován a data jsou vrácena v příslušné struktuře.

Metody třídy `DataTransformer` jsou následující (podle názvů je zřejmé, jaká data metoda vrací):

- `getDataPerIndividualEvents(DateRange date_range)` - návratovou hodnotou metody je seznam událostí, které vyhovují časovému intervalu. Jelikož jsou události odebírány z kolekce `TreeSet` (kde jsou seřazeny) a zpracovávány sekvenčně, jsou i ve výsledném seznamu seřazeny.
- `getDataPerDays(DateRange date_range)` - návratovou hodnotou metody je mapa s událostmi po dnech. Jedná se o instanci `TreeMap<String, List<MyEvent> >`, kde klíčem je datum v podobě řetězce ve formátu `dd.MM.yyyy`. Mapa má překrytý komparátor pro správné řazení podle klíče. Při průchodu všech událostí se zjistí datum začátku události a dle něho je událost vložena do příslušného seznamu.
- `getDataPerWeeks(DateRange date_range)` - návratovou hodnotou metody je opět mapa s událostmi po týdnech. Opět se jedná o `TreeMap`, kde klíčem je vždy pondělní datum daného týdne. Při průchodu všech událostí se zjistí pondělní datum týdne, kdy událost začíná, a podle toho se událost vloží do příslušného seznamu.
- `getDataPerMonths(DateRange date_range)` - opět se jedná o `TreeMap`, kde klíčem je měsíc a rok, například „June 2014“. Události vyhovující datovému intervalu se opět dle měsíce začátku zařadí do příslušného seznamu.
- `getDataPerYears(DateRange date_range)` - klíčem mapy tentokrát není řetězec ale `Integer` (proto u této mapy není nutné překrývat komparátor), představující rok. Události se zařazují do seznamů podle roku jejich začátku.

Téměř ve všech metodách, kde jsou použity mapy, jsou překryty speciální komparátory, aby bylo řazení dle klíče mapy vždy správné. Současně každá metoda filtruje události nevyhovující časovému intervalu. Při výstupu z metod jsou granularitní skupiny seřazeny od nejstarší po nejnovější.

Ve statistikách jsou také obsaženy součty trvání událostí, které vybraná klíčová slova neobsahují. Jelikož je událostí obecně mnoho, není výhodné, aby se nejdříve zjišťovaly události, které klíčové slovo obsahují, a někdy poté události, které klíčové slovo neobsahují. Toto je v aplikaci vyřešeno způsobem, kdy se události vrácené třídou `DataTransformer` procházejí znovu (`DataTransformer` totiž vrátí pouze požadovanou strukturu a události vně časového intervalu, o klíčová slova se nezajímá) a rozdělují se do dvou seznamů; na události, které nějaké klíčové slovo obsahují a které žádné takové slovo neobsahují. Potom už jsou všechny potřebné skupiny událostí vytvořeny a jsou použity v metodách počítajících součty.

Statistické třídy současně řeší zahrnutí prázdných řádků, které může uživatel požadovat. Jedná se o zobrazení granularitní skupiny událostí, která je prázdná (tj. nemá žádné události). Ve statistikách budou poté na příslušném řádku zobrazeny 0. Na to je ve třídě pro každý typ granularity připravena metoda, která zjišťuje, jestli je následující klíč přímým následovníkem předchozího. Pokud není, metoda vytvoří prázdný list a vloží ho do mapy. Metoda také zohledňuje časový interval statistik a za poslední událostí již prázdný list nevytvoří, i když je koncové datum intervalu ještě pozdější. Příklad: poslední událost má začátek 13.5.2014 a uživatel zvolí jako koncové datum deních statistik 15.5.2014. Skupina událostí odpovídající datům 14.5. a 15.5. již v mapě nebude.

Díky těmto prázdným listům se v metodách generujících výstup nemusí řešit, zda jsou pro následující skupinu data či nikoli a kód metod tak zůstává stejný.

5.4.1 Počítání součtů

Po získání příslušných dat si už statistická třída počítá souhrny sama. Jelikož vždy dostane pro danou granularitu už připravené skupiny událostí, je velmi jednoduché součty spočítat. Ke zjištění trvání události je použita metoda `getDuration()` třídy `ExcelOutputFormat` (interně volající metodu `getDuration()` třídy `MyEvent`). Ta má dva parametry, z nichž jedním je událost (`MyEvent`) a druhým je klíčové slovo, pro které se trvání aktuálně počítá

(událost může obsahovat více klíčových slov). Pokud událost klíčové slovo neobsahuje, vrátí metoda 0, jinak vrátí trvání události ve specifikovaných jednotkách. Ke zjištění celkového trvání skupiny událostí dané granularity se tato metoda volá pro každou událost opakovaně a součty se sčítají.

Událost trvající přes půlnoc je započítána do dne, kdy začala. To může být matoucí, jelikož například událost začínající v 23:55 a trvající 5 hodin trvá většinu svého času následující den. Její trvání je ovšem započítáno do dne, kdy začala. To už je pouze věcí názoru, jak chce uživatel tento problém řešit. Aplikace ho řeší popsaným způsobem. Rozdělení události podle půlnoci může být dalším tématem možného rozšíření aplikace.

Původně bylo zamýšleno nepočítat hlavní součty v aplikaci, ale pouze vytvořit vzorec do buňky v Excelu, který by součet spočítal sám. Zde bohužel nastal problém kvůli internacionalizaci. Jelikož ne všichni uživatelé používají stejnou lokalizaci Excelu, není možné například vzorec pro součet použít. V české lokalizaci se vzorec nazývá *SUMA*, v anglické *SUM*.

5.4.2 Výstupní formáty

Generování výstupu probíhá ve třídě reprezentující výstupní formát statistik. Na začátek je důležité říci, že aplikace podporuje pouze formáty XLS, XLSX a HTML. Formáty XLS a XLSX jsou zahrnuty v jedné třídě a jediné rozlišení mezi nimi je volání jiného konstruktora. Knihovna Apache POI definuje rozhraní *Workbook*, které implementují třídy *HSSFWorkbook* (pro formát XLS) a *XSSFWorkbook* (pro formát XLSX). Dále se při práci se sešitem téměř nerozlišuje, zda se jedná o sešit typu XLS nebo XLSX. Formát si knihovna řeší interně. Jediné rozlišení, na které se musel brát ohled, bylo ve stylování sešitu, konkrétně při nastavování pozadí buňky.

Ke generování formátu HTML jsou použity také třídy z knihovny Apache POI. Za pomoci těchto tříd lze konvertovat existující Excel soubor do formátu HTML, a to včetně kaskádových stylů. Proto při zvoleném výstupním formátu HTML nejdříve dojde k vygenerování statistik do formátu XLS a z toho následně do HTML.

Knihoven pro práci s těmito formáty je mnoho. Apache POI bylo použito na základě mnoha příkladů, návodů a zejména uživatelských recenzí, například: <http://stackoverflow.com/questions/14980717/>.

5.5 Uživatelské rozhraní

Grafické uživatelské rozhraní využívá pouze Swingové technologie a komponenty. Jedinou výjimkou, která standardně nepatří do Swingu, ale je v aplikaci využita, je komponenta pro výběr data. Jelikož Swing takovou komponentu nenabízí, bylo potřeba použít nějakou alternativu. Touto alternativou je knihovna *jCalendar*⁷, jejímž autorem je Kai Tödter. Knihovna poskytuje komponenty pro výběr celého data nebo jen roku apod. a doplňuje zmiňovaný nedostatek technologie Swing.

Ke tvorbě uživatelského rozhraní byl použit editor ve vývojovém prostředí NetBeans 8.0⁸, ve kterém je tvorba uživatelských rozhraní snadná. Všechny komponenty v uživatelském rozhraní, které spolu nějak souvisí, jsou vkládány do skupin v podobě instancí třídy `JPanel`. Každý tento panel v aplikaci má rozvržení *GridBagLayout*, se kterým se velmi snadno pracuje. Celá aplikace je umístěna do komponenty `JTabbedPane` umožňující používat karty (záložky). V případě této aplikace má tato komponenta 4 karty: pro kalendáře, klíčová slova, statistiky a export.

5.5.1 Modely a renderery

Modely a renderery [4] neboli *vykreslovače* hrají při tvorbě prezentační vrstvy velmi důležitou roli. Modely definují „co“ se má zobrazit a renderery definují „jak“ se to má zobrazit. Modelem je třída implementující rozhraní, přes které grafická komponenta přistupuje k datům. Výhodou těchto modelů je oddělení prezentační vrstvy od aplikační vrstvy.

Renderery se používají ve chvíli, kdy je požadována nějaká změna v zobrazování dat oproti standardnímu zobrazení. Vykreslovačů je více typů. Existují vykreslovače pro tabulky (`JTable`), pro seznamy (`JList`), pro rozjížděcí pole (`JComboBox`) a další. Podobně jako u modelů má každá komponenta své rozhraní definující „jak“ se budou data vykreslovat. Pokud chce uživatel něco zobrazit jinak než standardně, stačí implementovat rozhraní dané komponenty. Poté stačí už jen instanci této třídy nastavit komponentě jako její renderer.

⁷`jCalendar`: <http://toedter.com/jcalendar/>

⁸`NetBeans`: <https://netbeans.org/>

Modely v aplikaci

Aplikace obsahuje 3 modely k zobrazování dat. Prvním modelem je `CalendarsComboBoxModel`, který je zdrojem dat pro rozjížděcí menu s výběrem kalendáře. Druhým modelem je `ICalRepresentationTableModel`, který obsahuje data pro tabulku s vloženými kalendáři v aplikaci. Posledním modelem je `KeywordsTableModel`, který je zdrojem dat pro dvě tabulky zobrazující klíčová slova. Jelikož obě tabulky zobrazují stejná data, je instance tohoto modelu předána oběma tabulkám a není nutné vytvářet modely dva.

Renderery v aplikaci

V aplikaci je pouze 1 renderer, a to pro rozjížděcí menu pro výběr kalendáře při parsování klíčových slov. V ostatních případech nebyla implementace jiného vykreslovače nutná, neboť k vykreslování dat v jiných komponentách stačilo základní nastavení. Tím jediným vykreslovačem je `CalendarsComboBoxCellRenderer` pro výběr kalendáře z komponenty `JComboBox`. Ten definuje, jaký text bude uveden v rozjížděcím poli při výběru kalendáře.

5.5.2 Integrace s aplikační vrstvou

Dvě hlavní místa, kde prezentační vrstva komunikuje s vrstvou aplikační, jsou při parsování klíčových slov a při generování statistik.

Parsování klíčových slov se spouští stisknutím tlačítka *PARSE*. Po jeho stisknutí aplikace vytvoří instanci třídy `KeywordsFilter` a nastaví její atributy podle uživatelského nastavení v aplikaci. Současně se provádí testy správnosti tohoto uživatelského nastavení. Pokud něco není nastaveno nebo vyplněno správně, zobrazí se chybová hláška. Následuje vytvoření instance třídy `KeywordBackgroundWorker`, které se předá tento filtr, dále model klíčových slov, do kterého budou vyparsovaná klíčová slova vložena a samozřejmě instance třídy `ICalRepresentation` jako zdrojový kalendář pro parsování. Nyní se jen zavolá metoda `execute()` třídy `KeywordBackgroundWorker`, která ve vedlejším vlákne provede parsování.

Podobně funguje generování statistik. Po stisknutí tlačítka *Export* se vytvoří instance třídy `ExportConfiguration` reprezentující nastavení statistik a nastaví se podle voleb uživatele v aplikaci. Nyní se vytvoří instance třídy `ExportBackgroundWorker`, které se předá seznam zvolených kalendářů, ze kterých se bude generovat, dále seznam vyparsovaných klíčových slov a samozřejmě zmiňované nastavení statistik. Zavoláním metody `execute()` třídy

`ExportBackgroundWorker` dojde ke spuštění samotného generování. Tomuto workeru je také přidán posluchač události `PropertyChange`; tato událost je vyvolána uvnitř workeru po dokončení generování. V obsluze události se soubor s výslednými statistikami dle nastavení uživatele buď otevře nebo uloží.

5.5.3 Využití událostí

Každá komponenta definuje svůj seznam událostí, které mohou při jejím používání nastat. Tyto události mohou mít nějakou obsluhu v podobě posluchačů (listenerů), jejichž kód se provede při nastání dané události. Jelikož je toto téma velmi rozsáhlé, jsou zde popsány jen nějaká konkrétní využití událostí v této aplikaci.

Události jsou využity například při různém povolování a zakazování tlačítek či jiných komponent. Například tlačítko *PARSE* při parsování klíčových slov je aktivní pouze tehdy, je-li zvoleným zdrojem kalendář a rovnou je nějaký vybraný. Událostí je tedy volba (nebo změna) kalendáře v rozjízďecím poli a obsluhou je aktivace či deaktivace tlačítka.

Nastavení aktivity polí *Include dates*, *Include Day of Week names* a *Include from-to* je opět podmíněno zaškrtnutím pole *Include events*, kde událostí je změna zaškrtnutí a obsluhou je aktivace či deaktivace zmíněných 3 polí. Ostatně celé nastavení statistik velmi hojně využívá události, kde změna každé hodnoty provede aktualizaci této hodnoty na kartě *Export*, aby zobrazovala vždy aktuální nastavení generování.

Zde je ukázka vyvolání události po dokončení generování statistik v aplikační vrstvě a její následná obsluha v prezentační vrstvě.

Vyvolání události po dokončení

```
@Override
protected void done(){
    if(ex == null){
        try {
            if(isCancelled())
                firePropertyChange
                    ("generation_finished", null, null);
        }
        else
```

```
        firePropertyChange
            ("generation_finished", null, get());
    } catch (Exception _ex) {
        ex = _ex;
    }
}
.
.
.
}
```

Obsluha události po dokončení

Celý kód obsluhy je obsažen v metodě `generationFinished()` třídy hlavního okna aplikace. V tomto posluchači je tato metoda z důvodu přehlednějšího kódu pouze zavolána.

```
ExportBackgroundWorker ebw = new ExportBackgroundWorker(this);
ebw.addPropertyChangeListener(new PropertyChangeListener() {
    @Override
    public void propertyChange(PropertyChangeEvent evt) {
        if(evt.getPropertyName().equals("file_changed")){
            generationFinished((File) evt.getNewValue());
        }
    }
});
```

5.6 Výsledná podoba aplikace

Výslednou aplikaci tvoří jediný `.jar`⁹ soubor nazvaný `ical_statistics.jar`. K jeho běhu není potřeba dalších knihoven, protože veškeré potřebné knihovny jsou již zabaleny v archivu. Překlad zdrojových kódů a vytvoření spustitelného souboru je možné nástrojem Ant¹⁰. Aplikace se spouští příkazem `java -jar ical_statistics.jar` a její ukázka je na obrázku 5.3.

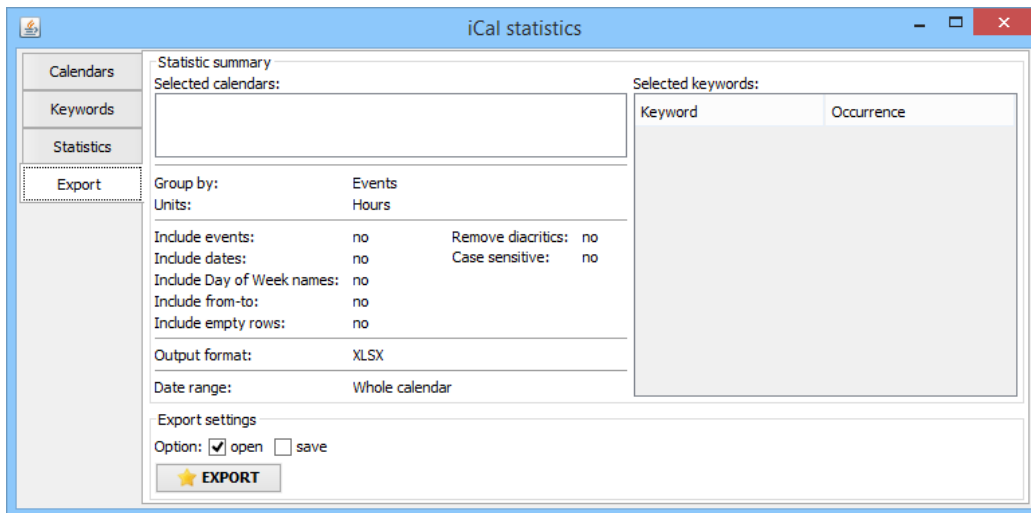
Celý projekt byl veden v nástroji Assembla¹¹ za pomoci verzovacího sys-

⁹JAR: Java Archive

¹⁰<http://ant.apache.org/>

¹¹<https://www.assembla.com/home>

tému Subversion¹².



Obrázek 5.3: Ukázka grafického uživatelského rozhraní aplikace

¹²<http://subversion.apache.org/>

6 Ověření aplikace

Tato kapitola se týká ověřování funkčnosti aplikace. Byla provedena ověření nejen správné funkčnosti modulů pro zpracování událostí a generování statistik, ale také správnost vypočtených statistik. Testování rychlosti aplikace je součástí tohoto ověřování.

6.1 Ověření hlavních modulů

Ukázkové výstupy a ověřování bylo prováděno na testovacích netriviálních datech. Tato data nebyla příliš rozsáhlá a byla vytvořena pouze pro tyto účely. Testy dob běhu byly prováděny již na reálných datech, které poskytl vedoucí této bakalářské práce. Tato data již byla velice rozsáhlá a bylo nutné zvolit období, na kterém se bude testování provádět. Období bylo zvoleno v rozsahu 4 měsíců a obsahovalo necelých 500 událostí. Testování nad celým kalendářem by trvalo příliš dlouho. (*Pozn.: bylo by nutné zpracovat 3750 událostí*).

6.1.1 Ověření načítání dat

První důležitou částí podléhající testování bylo správné načítání událostí a jejich filtrace. Testováním prošly metody filtrující události jdoucí do nekonečna a také metody, které prováděly rozbalování opakovaných událostí. Na toto testování nemohlo být použito žádného nástroje, neboť žádný podobné akce nenabízí. Testování bylo nutné provést ručně s vizuální kontrolou, že události, které mají být načteny, se načetly a události, které mají být rozbaleny, se povedlo korektně rozbalit a jejich časy začátku a konce odpovídají časům v kalendáři.

Testování dále podstoupila klíčová slova a správné výpočty jejich výskytů. Opět bylo nutné vizuálně kontrolovat a počítat klíčová slova v kalendáři a srovnávat je s výsledky aplikace.

6.1.2 Ověření statistik a exportu

Ověření správnosti výsledných statistik nebylo příliš jednoduché. Je jasné, že formát HTML není nutné testovat, protože je konverzí z formátu XLS. Formáty XLS a XLSX jsou na druhou stranu generovány téměř stejným kódem, takže pozornosti se dostává pouze třídě generující statistiky aplikace Microsoft Excel.

Největší testování bylo prováděno nad statistickými výpočty. Bylo kontrolováno, zda se události správně třídí do svých skupin dané granularity, zda se správně vypočítává doba jejich trvání pro dané klíčové slovo a zda součty za skupiny odpovídají. Ty už bylo možné otestovat například vzorcem v Excelu funkcí *SUMA*. Oproti tomu jednotlivá trvání za klíčová slova bylo nutné kontrolovat s údaji v kalendáři ručně a srovnávat, zda trvání souhlasí.

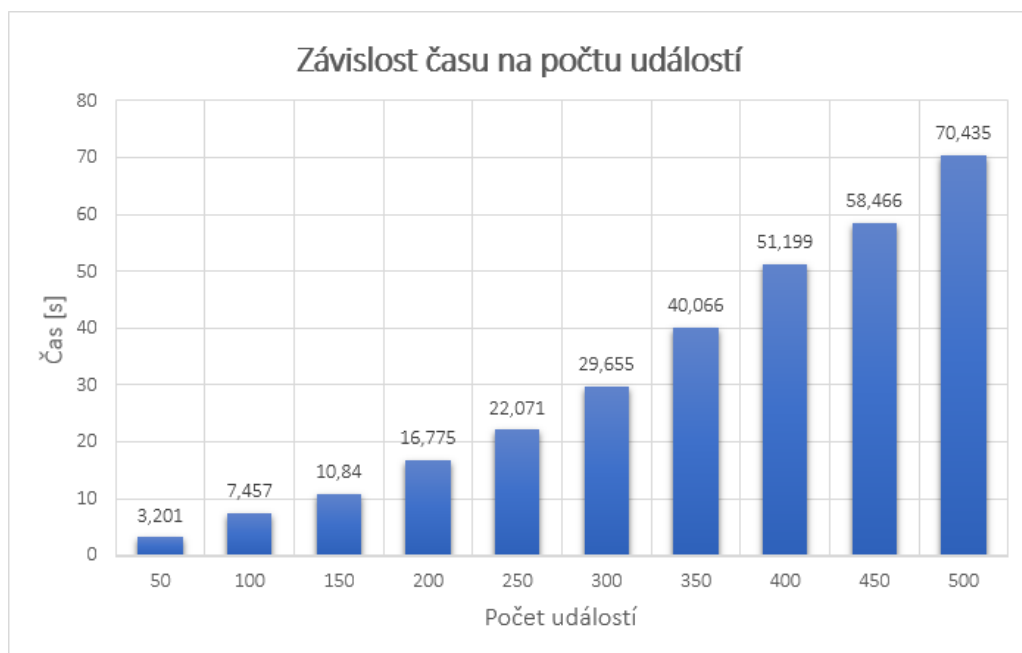
Poslední kontrolou prošla samotná struktura statistik a vizualizace, jestli je vše zobrazeno správně na svém místě, jestli žádná informace nechybí a zda je veškeré nastavení uživatele aplikováno správně. Například zahrnutí dat událostí, dnů a časy začátku a konce by mělo být ignorováno, pokud není nastaveno zahrnutí samotných událostí. Bylo kontrolováno také vkládání prázdných řádků, pokud si tak uživatel zvolil. Granularita, jednotky výstupu a časový interval statistik byl nepochybně testován také.

6.2 Testy dob běhu

Následující grafy zobrazují orientační doby běhů generování statistik při různém nastavení. Testování probíhalo na následujícím hardware: AMD A6-5400K APU 3600MHz, 4GB RAM 1600MHz. Každé testování bylo provedeno 3x a výsledné časy byly zprůměrovány. (*Pozn.: lišily se průměrně o 1s*).

Graf na obrázku 6.1 zobrazuje dobu generování statistik při následujícím nastavení: 7 klíčových slov s výskytem větším než 50. Granularita statistik nastavena na denní, zahrnuto vložení data, dne, začátku a konce události. Výstupním formátem byl formát XLSX. Při tomto testu se již počítá s vyparsovanými klíčovými slovy. Časovač byl spuštěn před zavoláním metody `export()` třídy `Exporter` a zastaven po jejím vykonání.

Graf na obrázku 6.2 zobrazuje dobu generování statistik při změně počtu klíčových slov. Počet událostí je konstantní - jejich počet je 250. Klíčová



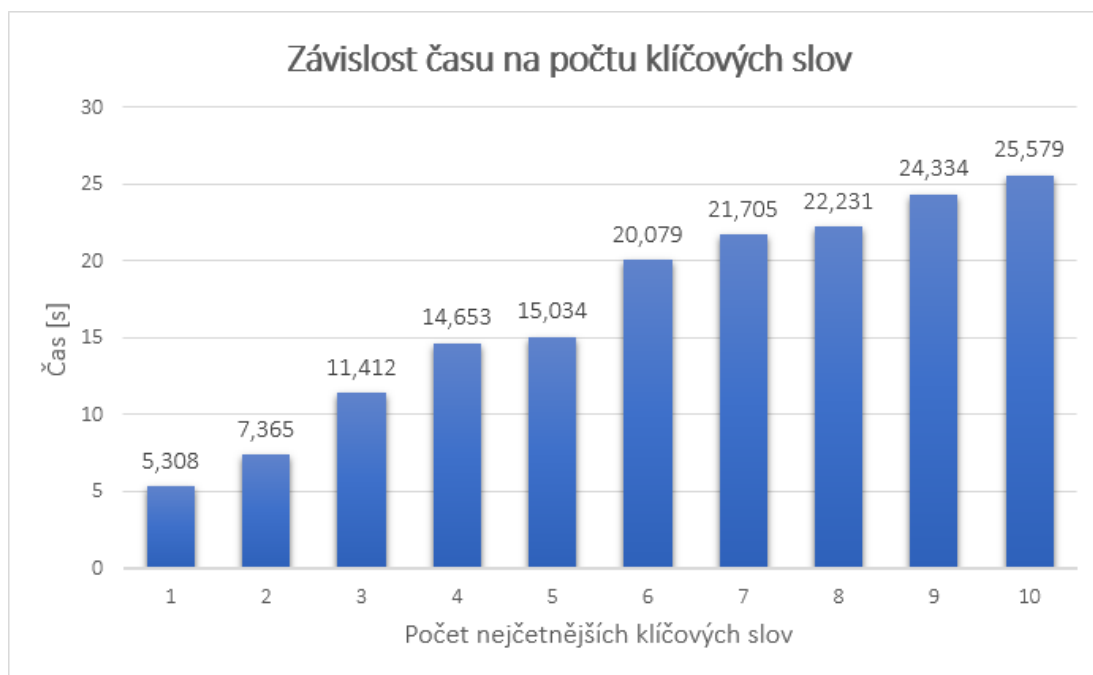
Obrázek 6.1: Závislost doby běhu na počtu událostí

slova jsou parsována ze stejného období, ze kterého je použito zmiňovaných 250 událostí. Nejprve se provedl výpočet doby pro nejčtenější klíčové slovo (jeho výskyt byl 97), dále se v každém výpočtu jedno klíčové slovo přidávalo. Nastavení výstupu bylo stejné jako u předchozího testování.

Shrnutí výsledků

V případě prvního testu trvalo zpracovat 50 událostí 3,201s. Nejvyšší počet událostí, 500, trvalo zpracovat 70,435s. Dá se tedy říci, že při 10x větším počtu událostí se výpočetní čas zvýšil přibližně 22x. V tom případě lze lineární závislost na počtu událostí vyloučit a pro přesnější vyjádření závislosti by bylo potřeba provést další měření při zvyšování počtu událostí. Z provedených experimentů se zdá, že při vyšším počtu měření se může závislost blížit závislosti exponenciální.

Ve druhém testu byl počet událostí konstantní (250). Generování statistik s 1 nejčtenějším klíčovým slovem trvalo 5,308s. 10 nejčtenějších klíčových slov zabralo při generování 25,579s. Výpočetní doba se při zdesetinásobení počtu klíčových slov zvýšila pouze 5x. Z výsledných časů je tedy zřejmé, že se nejedná o žádnou exponenciální či kvadratickou závislost. Není zjevná ani lineární závislost (např. mezi body 4, 5 a 6). Příčin může být hned několik.



Obrázek 6.2: Závislost doby běhu na počtu klíčových slov, při počtu událostí 250

Důležitou roli mohl hrát například velký rozdíl v počtu výskytů mezi 4., 5. a 6. nejčtetnějším klíčovým slovem nebo jejich odlišná délka. Negativním vlivem mohlo být také dočasné vytížení procesoru jiným procesem.

Ke zpřesnění výsledků a přesnějšímu odečtení závislosti (alespoň přibližně) by bylo taktéž potřeba provést více měření včetně minimalizace okolních vlivů (výkon počítače, výskyt klíčových slov lineárně rostoucí apod.).

Je vidět, že aplikaci lze použít i na velké kalendáře.

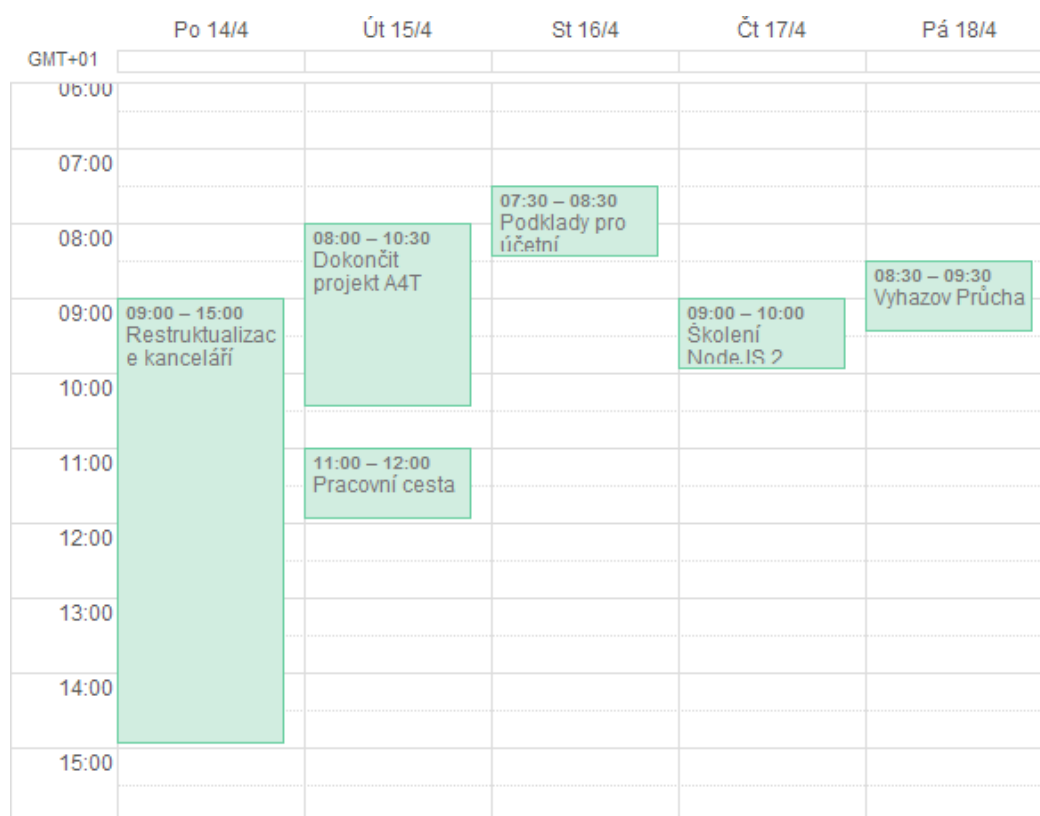
6.3 Ukázka výstupu

Na obrázcích 6.4 a 6.5 je vidět možný ukázkový výstup aplikace. Jelikož výstup je příliš široký, je obrázek rozdělený na dvě části. V ukázkových statistikách je zvolena týdenní granularita s jednotkou v minutách. Dále je zahrnuto zobrazení událostí, jejich datum a den. Zahrnuty jsou i prázdné řádky.

Na obrázku 6.4 jsou zobrazené týdenní skupiny. Sloupec *sum* popisuje buď trvání konkrétní události nebo součet trvání událostí v dané skupině (v tomto případě týdně).

První 4 sloupce tabulky na obrázku 6.5 symbolizují vybraná klíčová slova a trvání událostí s obsaženým klíčovým slovem. Sloupec *others* popisuje dobu trvání událostí ve skupině, které neobsahují žádné klíčové slovo. Sloupec *total* je součtem sloupců *sum* a *other* a popisuje trvání všech událostí ve skupině.

Z obrázků je zřejmé, že týden od 24.3.2014 - 30.3.2014 obsahuje samé 0 a je jedním z prázdných řádků (nastaveno před generováním). V týdnu od 7.4.2014 do 13.4.2014 jsou pouze události, které neobsahují žádné klíčové slovo a v součtu trvají 300 minut. V týdnu od 14.4.2014 do 20.4.2014 byla událost s předmětem „Pracovní cesta“ trvající 60 minut, dále událost „Školení NodeJS 2“ trvající taktéž 60 minut a současně nějaká jiná událost (nebo události) trvající 630 minut. Celkový součet trvání událostí v tomto týdnu je tedy 750 minut a lze je vidět v reálném kalendáři na obrázku 6.3.



Obrázek 6.3: Ukázka reálného kalendáře v aplikaci Google Calendar

week from	week to	event subject	date	day	sum
10.03.2014	16.03.2014				300
		Porada vedení noví zaměstnanci	11.03.2014	Tue	120
		Školení NodeJS	12.03.2014	Wed	60
		Nové oddělení - porada	14.03.2014	Fri	120
17.03.2014	23.03.2014				300
		Cesta Praha	17.03.2014	Mon	90
		Školení XML technologie	18.03.2014	Tue	60
		Školení AJAX	18.03.2014	Tue	60
24.03.2014	30.03.2014				0
31.03.2014	06.04.2014				390
		Školení iCal 2	31.03.2014	Mon	60
		Pracovní cesta	01.04.2014	Tue	60
		Schůzka v zasedačce	02.04.2014	Wed	60
		Porada se zaměstnanci	03.04.2014	Thu	150
		Vedení schůzka	04.04.2014	Fri	60
07.04.2014	13.04.2014				0
14.04.2014	20.04.2014				120
		Pracovní cesta	15.04.2014	Tue	60
		Školení NodeJS 2	17.04.2014	Thu	60

Obrázek 6.4: Ukázka souboru se statistikami, část 1

školení	cesta	pracovní	porada	schůzka	others	total
60	0	0	240	0	300	600
0	0	0	120	0		
60	0	0	0	0		
0	0	0	120	0		
120	180	90	0	0	420	720
0	90	0	0	0		
60	0	0	0	0		
60	0	0	0	0		
0	0	0	0	0	0	0
60	60	60	150	120	120	510
60	0	0	0	0		
0	60	60	0	0		
0	0	0	0	60		
0	0	0	150	0		
0	0	0	0	60		
0	0	0	0	0	300	300
60	60	60	0	0	630	750
0	60	60	0	0		
60	0	0	0	0		

Obrázek 6.5: Ukázka souboru se statistikami, část 2

7 Závěr

Zadáním bakalářské práce bylo vytvořit nástroj generující statistické informace z kalendářových dat. Výsledkem je aplikace s názvem *iCal statistics*, která umí zpracovávat kalendářová data ve formátu iCalendar a následně z nich vypočítávat požadované statistické součty. Aplikace umožňuje parsovat klíčová slova a současně podporuje široké možnosti nastavení výstupních statistik včetně nastavení zmiňovaného parsování klíčových slov. Pokud v aplikaci nastane chyba, je tato chyba uložena do logu s časem a místem, kde k chybě došlo. Veškeré operace prošly testováním úspěšně a aplikace funguje správně na operačních systémech Linux a Windows.

Výstupy z aplikace mohou být využity jako podklady pro výkazy času, jako prostředky pro sledování spotřebovaného času (například na nějakém projektu) či jako přehledy o osobním časovém vytížení.

7.1 Zhodnocení

Zadání bakalářské práce bylo v plném rozsahu splněno. Celá práce se vztahuje k času, časovým informacím a jejich zpracování. Vysvětleny jsou principy řízení svého času, jakým způsobem lze čas využívat, aby jeho využití bylo efektivní a ztráty byly minimální. Dále jsou popsány kalendářové aplikace sloužící k plánování a rozvrhování času. Tyto aplikace poskytují další nástroje, které mohou být v praxi velice užitečné a mohou v mnoha případech usnadnit práci a ušetřit čas. Tato teoretická část je obsažena v první polovině práce. Druhá polovina obsahuje popis aplikace, která již s kalendářovými daty pracuje a zpracovává je.

Po celý vývoji aplikace byl využíván verzovací systém *Subversion*, který velmi pomohl v pochopení verzování projektů a využívání potenciálu verzovacích systémů. Mnoho nástrojů, které *Subversion* nabízí, bylo použito.

Bakalářská práce přinesla mnoho nových informací, zkušeností a vytvořila přehled o efektivním využívání času, které je zajisté stále aktuálnějším tématem.

7.2 Možná rozšíření

Často se stává, že se při vývoji aplikace nebo po jejím dokončení objeví nápady či návrhy na zlepšení. Uvedené návrhy nejsou v aplikaci implementovány, ale jsou označeny jako nice-to-have¹ rozšíření.

- **Ukazatel postupu zpracování** - při generování statistik nebo parsování klíčových slov je zobrazen uživateli dialog představující informaci, že se něco vykonává na pozadí. V dialogu je zobrazen nekonečný progressbar (komponenta ukazující postup při vykonávání nějaké delší úlohy), který by mohl být předělán na procentuální.
- **Dělení událostí přes půlnoc** - spíše možnost nastavení při generování. Nyní je trvání událostí započítáno do dne, kdy událost začala. Tímto rozdělením by statistiky byly z určitého pohledu přesnější.
- **Prázdné řádky** - pokud v současné verzi aplikace nastaví uživatel nějaké počáteční nebo koncové datum statistik a zaškrtně vložení prázdných řádků, dojde k vložení pouze do oblasti před poslední událost. Nově by mohly být prázdné řádky vkládány na začátek či konec zvoleného období. Může být opět nastaveno při generování.
- **Formát PDF** - formát PDF by mohl být zařazen do seznamu volitelných výstupních formátů. Jelikož statistiky ale většinou zabírají na šířku mnoho místa, nemuselo by výsledné zobrazení v PDF vypadat rozumně.
- **Lemmatizace² klíčových slov** - pokud by byla klíčová slova lemmatizována na svůj základní tvar, nemusely by se ve statistikách vyskytovat duplicity klíčových slov lišících se v několika písmenech (za předpokladu absence vizuální kontroly uživatele).
- **Optimalizace rychlosti** - jádro aplikace by prošlo důkladnou analýzou za účelem odhalení slabých míst, jejichž oprava by přispěla k rychlejšímu řešení náročnějších úloh.
- **Google API** - pokud uživatel využívá pro správu kalendářů aplikaci Google calendar, nabízí se využití Google API k získání kalendářů uživatele po zadání přístupových údajů.

¹nice-to-have = doplňkové rozšíření aplikace, které není k jejímu bezchybnému chodu nezbytné

²lemmatizace = proces převedení slova do jeho základního tvaru

Příloha

7.3 Uživatelská dokumentace

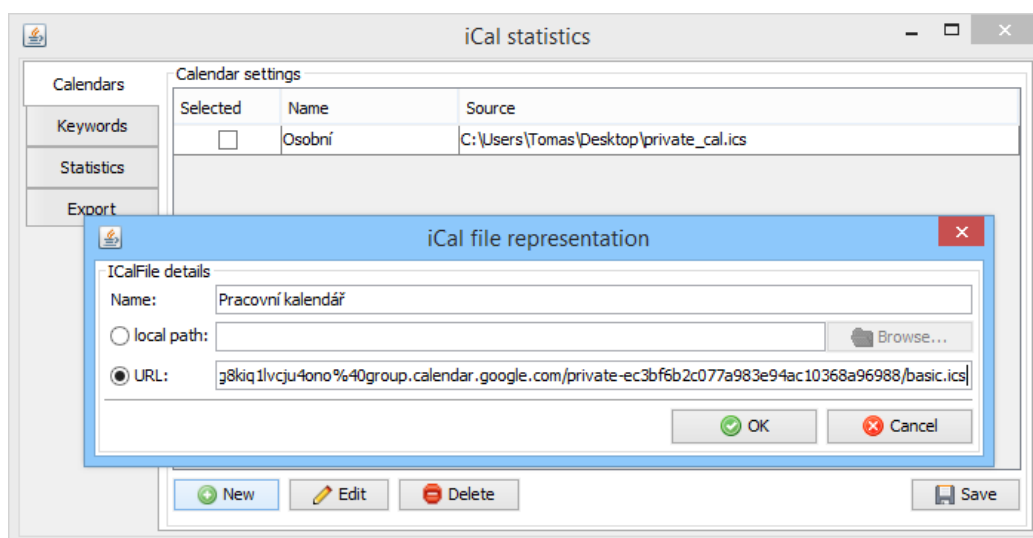
Zde je popsán ukázkový postup generování statistik a vysvětlení veškerého nastavení.

První podmínkou je mít v aplikaci přidané kalendáře. Kalendář lze přidat buď z lokálního umístění nebo z adresy URL. V záložce *Calendars* stiskneme tlačítko *New*. Zobrazí se okno pro detail kalendáře neboli reprezentaci kalendáře. Vyplní se jméno a následně se buď vybere cesta nebo vloží URL. Každý kalendář má v prvním sloupci zaškrtačací pole, které říká, že tento kalendář bude zdrojem dat při generování statistik.

Tyto reference na kalendáře je možné ukládat tlačítkem *Save* do souboru *data_icr.xml*. Pokud se provedou v tabulce kalendářů nějaké změny, je nutné je uložit, aby se při příštím spuštění aplikace projevíly.

Následujícím úkolem je získání klíčových slov. Klíčová slova je možné do aplikace nahrát ze souboru *CSV*, ve kterém je na každé řádce uloženo slovo a jeho výskyt oddělené středníkem. Další možností je parsování klíčových slov z kalendáře. V menu *From calendar* vybereme kalendář, který bude zdrojem dat a nastavíme požadované vlastnosti parsování. Proces spustíme stiskem tlačítka *PARSE*. Po stisku se zobrazí okno informující uživatele o probíhající činnosti. Aplikace umožňuje parsování klíčových slov z kalendáře a současně jejich vložení ze souboru.

Nastavení parsování zahrnuje volbu minimálního výskytu klíčového slova. Dále regulární výraz, který se zadává do pole *Matches regexp*, proti kterému se budou vyparsována slova testovat (nevyhovující budou odstraněna). Dalším nastavením je časový interval. Pokud si uživatel přeje parsovat z celého

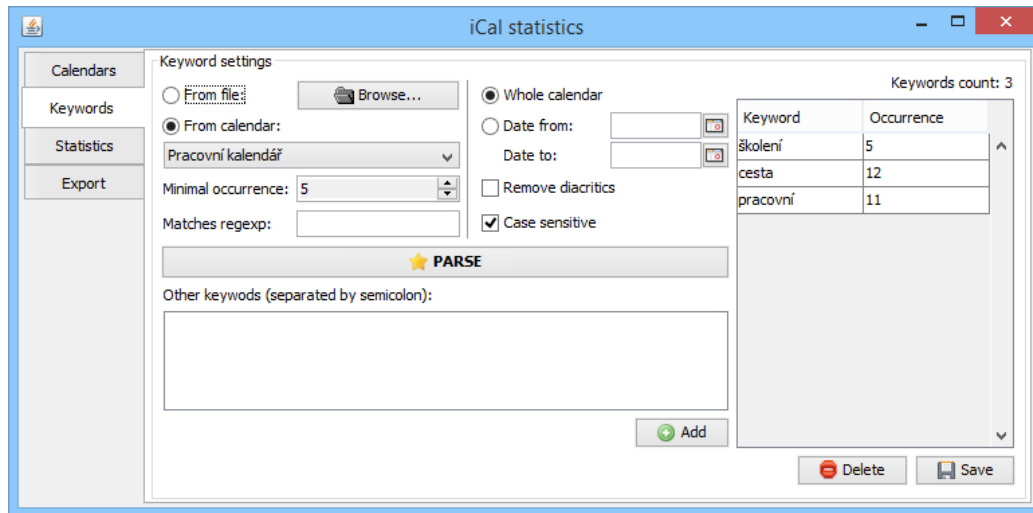


Obrázek 7.1: Vkládání kalendářů do aplikace

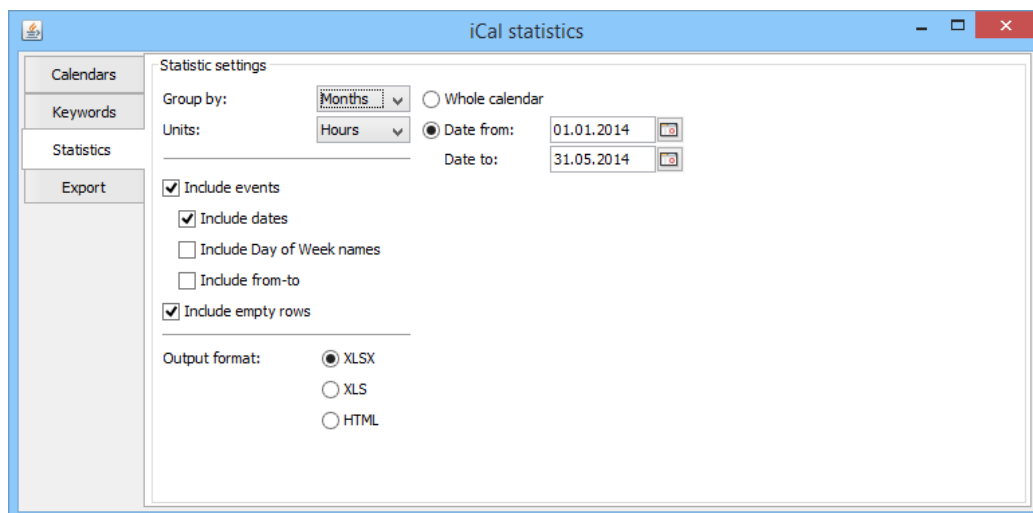
kalendáře, nechá nastaveno *Whole calendar*, v opačném případě přepne na *Date from* a nastaví datum od a do. Dále je možné zaškrtnout odstranění diakritiky a ignoraci velkých a malých písmen.

Po skončení parsování se výsledná klíčová slova zobrazí s jejich výskytem v tabulce. Nad tabulkou je informace o jejich počtu a pod tabulkou jsou tlačítka na smazání vybraných slov a na jejich uložení do souboru. Sloupce tabulky lze řadit a označené záznamy lze mazat klávesou *Delete*. Poslední význam hraje pole *Other keywords*. Tam uživatel může vložit klíčová slova (oddělená středníkem), která při parsování neprošla filtrem, ale z nějakého důvodu jsou v seznamu požadována.

Následujícím krokem je nastavení statistik na kartě *Statistics*. První položkou je granularita. Ta říká, po jakých skupinách budou události ve statistikách děleny. Možnosti jsou: jednotlivé události, dny, týdny, měsíce a roky. Následuje jednotka výpočtů statistik říkající, že veškeré součty a veškerá trvání událostí ve statistikách budou buď v minutách, hodinách nebo dnech. Následující pole *Include events* zobrazí jednotlivé události v dané granularitní skupině. *Include dates* zobrazí datum začátku události, *Include Day of Week names* zobrazí den události, *Include from-to* zobrazí čas začátku a konce a *Include empty rows* zobrazí granularitní skupiny spadající do zvoleného časového intervalu neobsahujícího žádnou událost. Následuje zvolení výstupního formátu a poté časový interval statistik: buď celý kalendář, nebo jen uvedené období.



Obrázek 7.2: Parsování klíčových slov

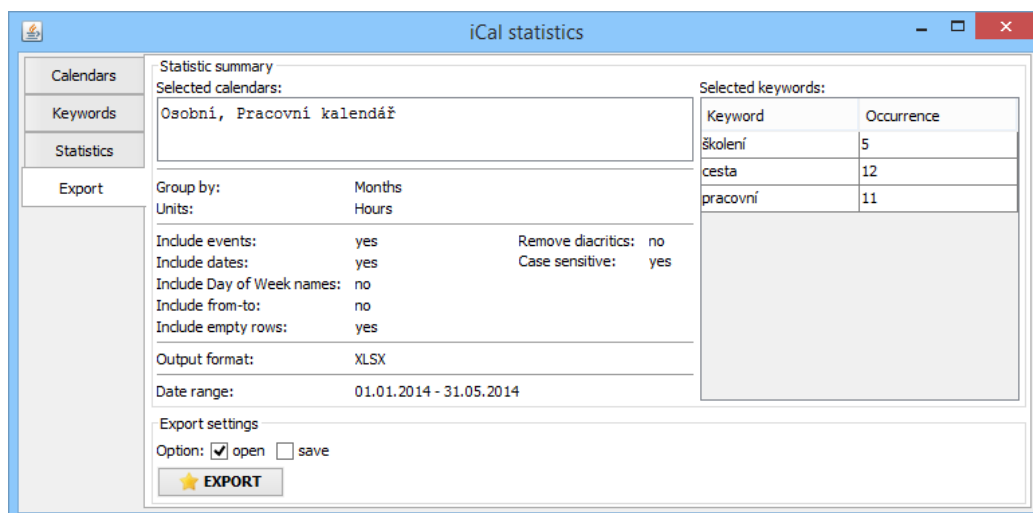


Obrázek 7.3: Nastavení statistik

Posledním krokem je kontrola nastavení a export statistik, k čemuž slouží karta *Export*. Po kontrole vybraných kalendářů, klíčových slov a zvoleného nastavení může uživatel zvolit, jestli chce výstupní soubor otevřít, uložit nebo obojí. Podmínkou spuštění generování je: zvolený alespoň jeden zdrojový kalendář, časový rozsah nastaven buď na celý kalendář, nebo alespoň *datum od* nebo *datum do* a alespoň jedno klíčové slovo. Pokud toto nebude splněno, bude o tom uživatel informován chybovou hláškou.

Před spuštěním generování je nutné dbát na zvolený rozsah generování statistik a na velikost obsahu kalendáře, protože pokud kalendář obsahuje velké množství událostí a rozsah je zvolen na celý kalendář, může generování trvat dlouhou dobu. Totéž platí pro parsování klíčových slov, kde se může doba výpočtu taktéž prodloužit.

Samotný export se spouští stisknutím tlačítka *Export*.



Obrázek 7.4: Export statistik

Literatura

- [1] Stephen Covey, A. Roger Merrill, and Rebecca R. Merrill. *First Things First*. Simon and Schuster, New York, 1994.
- [2] Bernard Desruisseaux. *Internet Calendaring and Scheduling Core Object Specification*.
<https://tools.ietf.org/html/rfc5545>, 2009.
- [3] Ulrika Hedlund. *Effective Calendar Management*.
<http://www.businessproductivity.com/effective-calendar-management/>, 2012.
- [4] Pavel Herout. *Java - grafické uživatelské prostředí a čeština*. KOOP, České Budějovice, 2007.
- [5] Bill Hoberecht. *Calendar Management: Getting Control of Your Appointments and Meetings*.
<http://www.pinnacleprojects.com/index.php/time-management/147-calendar-management-getting-control-of-your-appointments-and-meetings>, 2014.
- [6] Jiří Kolařík. *Jak na time management v 10 bodech*.
<http://www.jiri-kolarik.cz/clanek/jak-na-time-management/>, 2011.
- [7] Heinz Tschabitscher. *Google Calendar - Free Online Calendar*.
http://email.about.com/od/bestfreeonlinecalendars/gr/Google_Calendar_ReviewFree_Online_Calendar.htm.