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

Diplomová práce

Univerzální asistent pro Android

Plzeň, 2015 Bc. Jan Pavlík

Prohlášení

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

V Plzni dne 24. června 2015

Bc. Jan Pavlík

Děkuji vedoucímu práce Ing. Ladislavu Pešičkovi za odborné vedení a cenné rady při vypracování diplomové práce.

Abstract

This diploma thesis focuses on time management applications for mobile devices. The first part deals with possibilities and methods for collecting information from different sources that are available from Android devices. The theoretical part also includes an overview of the best-selling and top rated applications for time management.

The main goal of this thesis is to design and develop a time management application for operating system Android. The application will collect information from different sources and help users with managing their time by processing these information.

Abstrakt

Tato diplomová práce se zaměřuje na aplikace pro mobilní zařízení, které se zabývají organizací a plánováním času. První část práce popisuje možnosti integrace informací z různých informačních zdrojů, které jsou dostupné pro zařízení na platformě Android. Teoretická část práce také obsahuje srovnání nejlépe prodávaných a nejlépe hodnocených aplikací pro správu času.

Hlavním cílem této práce je navrhnout a vytvořit aplikaci pro organizaci a plánování času pro operační systém Android. Aplikace bude integrovat informace z různých zdrojů a na základě těchto informací bude usnadňovat uživatelům organizaci času.

Obsah

1 Úvod

Úvodní kapitola pojednává o motivaci diplomové práce a stanovuje cíle, kterých má být v práci dosaženo.

1.1 Motivace

V současné době je na trhu s mobilními aplikacemi velké množství různých aplikací, které si kladou za cíl usnadnit uživateli organizaci a plánování času. Problémem těchto aplikací je zejména to, že se většinou zaměřují pouze na jeden aspekt organizace času, ať už se jedná o kalendář, úkoly, poznámky nebo budíky a připomenutí.

Díky tomu je uživatel nucen používat více aplikací najednou, což degraduje jejich výsledné použití. Jednotlivé aplikace spolu totiž nijak nespolupracují, i když zde existuje potenciál pro integraci informací ze zmíněných oblastí organizace času, který se dá mimo jiné využít k efektivnějšímu plánování, či k lepšímu přehledu uživatele o jeho nadcházející agendě. Kromě tohoto důvodu je používání samostatné aplikace zároveň uživatelsky příjemnější.

Dnešní chytrá zařízení poskytují široké možnosti k získávání informací o jejich uživatelích. Využití těchto personalizovaných informací přímo v aplikacích pak může vést k přesnějším a pro uživatele relevantnějším výsledkům (v tomto případě při organizaci času) podobně, jako je tomu u osobních asistentů některých mobilních platforem (Siri - iOS, Cortana - Windows Phone).

1.2 Cíle práce

V teoretické části práce bude nejprve proveden průzkum a analýza vybraných aplikací na organizování a plánování času. Hlavním cílem této části je analyzovat vybrané aplikace s ohledem na integraci informací. Na základě analýzy bude následně navržena funkcionalita vytvářené aplikace s přihlédnutím na možnosti platformy Android.

Druhá část práce se věnuje vlastní tvorbě aplikace podle funkcinality stanovené v předchozí fázi. Aplikace by měla nabízet základní moduly pro správu kalendářových událostí, úkolů a budíků. Nejvýznamnějším přínosem aplikace pro uživatele pak budou chytré funkce, které budou například informovat o nadcházející agendě, předvyplňovat formulářová pole při vkládání událostí nebo navrhovat vložení nových infromací. K tomu se budou využívat mimojiné kalendářové informace, informace o poloze zařízení nebo o aktuálním počasí.

Praktická část práce bude postupně popisovat celý vývoj aplikace včetně návrhu, implementace a testování. Na závěr bude aplikace konfrontována se stanovenými požadavky na funkcionalitu případně budou navrženy možnosti rozšíření.

2 Integrace informací

Potřeby uživatelů jsou realizovatelné jen tehdy, pokud má aplikace k dispozici informace, na jejichž základě je později schopna rozhodovat nebo provádět akce, které zefektivní organizaci času (notifikace, návrhy). V této sekci budou postupně vyjmenovány informační zdroje, ze kterých mohou aplikace čerpat potřebné informace, a zároveň budou obecně popsány i metody, jak tyto informace získat.

2.1 Informační zdroje

Aplikacím se jako základ pro čerpání informací nabízí tři nezávislé oblasti. Na prvním místě je samozřejmě vlastní **uživatelský vstup**, jenž z hlediska následného vyhodnocování poskytuje asi nejrelevantnější informace. Dalším zdrojem informací je velké množství **hardwarových senzorů**, které jsou zabudovány v dnešních chytrých zařízeních. Poslední kategorií jsou **internetové zdroje**, jenž poskytují mnoho služeb, které v reakci na konkrétní dotaz (ve službou definované formě) vrací požadováné informace (počasí, doprava, atd.).

2.1.1 Uživatelský vstup

Zcela základním a na první pohled zjevným zdrojem informací je uživatelský vstup. Data, která aplikaci předá přímo její uživatel mají **největší vypovídací hodnotu** a měla by být vyhodnocována vždy s největší prioritou. Z tohotu pohledu je velmi důležité, aby aplikace nabízely co nejvíce vstupních formulářů a prvků pro možnost zadání kompletní definice např. kalendářové události či úkolu.

Na druhou stranu je třeba zvolit vhodný poměr mezi povinnými a nepovinnými vstupními poli tak, aby uživatel nebyl nucen zdlouhavě vyplňovat informace, které pro něj nejsou v daný moment důležité. S tímto problémem se občas aplikace (nejen ty mobilní) potýkají a zbytečně je tak zhoršeno uživatelské pohodlí (user experience).

S předchozí myšlenkou souvisí schopnost aplikací **předvyplňovat for-**

mulářová pole na základě již dostupných informací. Pokud je systém schopen zjistit některé informace bez zadání uživatele, dochází ke zkrácení doby interakce s aplikací a tím pádem komfortnějšímu používání. Příkladem u kalendářových aplikací může být přednastavené datum nebo čas v závisloti na aktuálním čase či přednastavený kalendář při přidávání události v závislosti na aktuální poloze (pokud se přidává událost a uživatel je doma, s nemalou pravděpodobností chce událost přidat do osobního kalendáře a pokud je v práci, tak naopak do kalendáře pracovního).

2.1.2 Hardwarové senzory

Dalším důležitým zdrojem dat jsou zabudované hardwarové senzory. Některé senzory, jako například geolokační (GPS[1] senzor) nebo akcelerometr, se staly takřka standardem a lze je nalézt v drtivé většině dnešních chytrých mobilních zařízení [Ope01].

Naopak se také ale můžeme setkat se senzory, které se vyskytují jen u vybavenějších modelů. Uživatelé mobilních zařízení, která tyto senzory nemají pak nemohou využívat funkcionalitu, jenž je na nich závislá. Při vývoji aplikací je tedy důležité vzít v potaz míru rozšíření daného senzoru v současných zařízeních či pravděpodobnou hardwarovou vybavenost cílové skupiny uživatelů.

Pomineme-li běžně používané senzory, jako jsou mikrofony, kamery, kapacitátor (dotyková vrstva displeje) nebo zařízení typu Wi-Fi, Bluetooth, NFC[2], atd., můžeme v aplikacích využívat také [Ope01][And08]:

- **Akcelerometr** - slouží k určení relativní polohy v prostoru (orientace na výšku nebo na šířku, displejem dolů nebo vzhůru). Poloha se určuje na základě gravitace.
- **Barometr** - určuje aktuální atmosférický tlak, který se používá při určení nadmořské výšky (využití v aplikacích pracujících s přesnou polohou).
- **Geolokační senzor** - typicky GPS čip pro určení aktuální geografické polohy.

[1] Global Positioning System

[2] Near Field Communication - technologie pro komunikaci na krátkou vzdálenost

- **Gyroskop** - podobně jako akcelerometr poskytuje informace o natočení zařízení v prostoru. Pro zjištění polohy využívá setrvačník.
- **Hallův senzor** - tento senzor měří magnetické pole. Jedno z možných použití je detekce otevření a zavření pouzder na magnet a následné odemknutí nebo zamknutí telefonu.
- **Krokoměr** - počítání kroků lze zajistit i pomocí akcelerometru, ale integrovaný krokoměr bývá přesnější a méně energeticky náročnější.
- **Magnetometr (kompas)** - na základě detekce magnetického pole Země určuje světovou stranu, na kterou zařízení směřuje.
- **Senzor okolního osvětlení** - reaguje na intenzitu okolního osvětlení a může tak upravit jas displeje.
- **Senzor otisků prstů** - vyskytuje se u vybavenějších modelů a slouží k autorizaci uživatele.
- **Senzor přiblížení** - zpravidla bývá umístěn nad displejem telefonu a rozpoznává přiblížení k nějakému objektu. Čidlo obsahuje IR[3] diodu a detektor. Po přiblížení k objektu (např. lidské ucho při telefonování) detektor zachytí odražený IR paprsek a systém na tuto událost zareaguje vypnutím displeje.
- **Senzor srdečního tepu** - skládá se z LED diody, která prosvítí přiložený prst, a senzoru pulzu, jenž monitoruje pohyb červených krvinek. Není zatím příliš rozšířen.
- **Senzor UV záření** - senzor nasměrovaný ke slunci určuje intenzitu UV záření.
- **Teploměr** - používá se především pro monitoring teploty hardwarových částí telefonů (vypnutí při přehřátí). Měření okolní teploty vyžaduje několik minut v ustálené poloze mimo ostatní zdroje tepla a není přesné (příliš se nepoužívá).
- **Vlhkoměr** - určuje vlhkost okolního prostředí (není příliš využíván).

Vzhledem k povaze navrhované aplikace přichází v úvahu využití především geolokačních senzorů. Nabízí se také využití akcelerometru pro zjištění pohybu uživatele. V momentě, kdy je pohyb uživatele detekován, může být

[3]Infrared - infračervené záření

spuštěna služba pro monitorování geografické polohy, která je potřeba například pro upozornění, že se uživatel nachází v blízkosti místa, ve kterém má naplánovaný nějaký úkol.

2.1.3 Internet

Jelikož většina dnešních uživatelů chytrých telefonů má nějaké datové připojení, nelze ve výčtu možných zdrojů vynechat internetové zdroje. S internetovým připojením pak mohou aplikace s využitím dostupných služeb získávat potřebné informace z různých odvětví. Typickými příklady mohou být např. informace o počasí, aktuální dopravní situaci nebo také o místech (restauracích, kinech, obchodech, památkách, atd.) v blízkosti uživatele.

Zmíněné služby a API[4] jsou většinou nabízeny jak v placené, tak i v neplacené verzi. Volné verze jsou zpravidla omezeny počtem přístupů ke službě za jednotku času (např. 1000 přístupů za minutu na jeden API klíč), mírou dostupnosti, případně mohou být služby omezeny oproti placeným verzím i obsahově.

Pro navrhovanou aplikaci dává smysl použití služeb s geografickou lokací a nejbližšími místy (Google Places API), která se dají přiřadit ke kalendářovým událostem a úkolům. Velké množství užitečných informací nabízí i služba Google Now (např. doprava a výpočet časové náročnosti cesty, vyhledání spojení veřejnou dopravou, hotely, atd.) [Gde01].

2.2 Metody získání informací

Po definici možných informačních zdrojů budou v této sekci obecně popsány metody, jak získávat nezbytné údaje, přičemž zde budou uvedena také možná úskalí jednotlivých přístupů k informacím.

[4]Application Programming Interface

2.2.1 Uživatelský vstup

Aby aplikace typu osobní asistent naplňovala očekávání uživatele, musí o něm bezpodmínečně znát některé základní údaje (např. lokaci domova a práce, pracovní dny a dny pracovního klidu, atd.). Jednou možností je získání těchto údajů periodickým sledováním chování uživatele, podle kterého se po nějaké době pravděpodobně podaří zmíněné údaje vypozorovat (použití hardwarových senzorů).

Alternativou k tomuto postupu (z hlediska aplikační logiky neúměrně jednodušší) je **přímé dotazování** uživatele na jeho základní údaje. Vhodným nástrojem může být jednoduchý formulář, který se uživateli zobrazí při prvním spuštění aplikace. Zároveň by měla existovat možnost případné změny těchto údajů (typicky v globálním nastavení aplikace).

Přímé dotazování lze použít i pro méně důležité informace, a to třeba v momentě, kdy aplikace rozpozná nějaké nestandardní zadání. Může se například jednat o vložení vícedenní události v pracovních dnech do osobního kalendáře. Reakcí aplikace může být krátký dotaz, jestli uživatel nemá v dané dny dovolenou, aby se mohl odpovídajícím způsobem změnit režim (např. defaultní nastavení budíků).

Je však důležité zmínit, že by aplikace měla kontrolovat četnost těchto dotazů a návrhů, případně nabídnout možnost nastavení četnosti.

2.2.2 Hardwarové senzory

Před použitím API pro přístup k hardwarovým senzorům platformy Android je nejprve nutné definovat povolení (permissions), aby byl uživatel při instalaci aplikace informován o tom, jaké senzory hodlá aplikace využívat. Dalším krokem je kontrola dostupnosti příslušného senzoru a v případě kladné odezvy je možné získávat požadovaná data [And08].

Důležitým aspektem je při přístupu k hardwarovým senzorům **energetická náročnost**. V první fázi by měl programátor pečlivě zvážit volbu vhodných časových intervalů pro opakovaný přístup k danému senzoru tak, aby se k senzorům nepřistupovalo příliš často.

K šetření baterie lze také dospět získáním informací jiným (méně energeticky náročným) způsobem. Tento postup se často volí například v případě,

kdy se zjišťuje poloha uživatele, ale není potřeba přesnost na jednotky metrů. Místo energeticky náročného GPS senzoru je poloha zjištěna s využitím internetu. Vždy ovšem záleží na požadavcích uživatelů a povaze aplikace, přičemž vývojáři by se měli pokusit najít kompromis mezi přesností, spolehlivostí či aktuálností zpracovávaných informací a náročností na spotřebu energie.

2.2.3 Internet

Při používání internetového připojení je taktéž třeba zohlednit energetickou náročnost vhodnou volbou frekvence odesílání požadavků na data ze služeb.

Omezení četnosti odesílání požadavků souvisí také s **limity datových připojení** (FUP[5]). Současné limity datových připojení se pohybují v řádech stovek MB a aplikace je tak třeba navrhovat i s ohledem na datovou náročnost. Pokud tedy existuje více verzí jedné služby, které se liší obsahově (např. předpověď počasí na den, 3 dny, týden, atd.), měli bychom v aplikaci požadovat jen ta data, jenž budou skutečně využita.

Vlastní požadavky se pak ve službou definované formě a zpravidla s unikátním API klíčem jako jedním z parametrů odešlou na požadovanou URI[6], přičemž serverová strana zašle po zpracování požadavku zpátky na klientské zařízení odpověď typicky v jednom z často využívaných formátů pro přenos dat po síti (JSON[7], XML[8], atd.). Konkrétní použití některých služeb bude popsáno v praktické části práce.

[5]Fair Use Policy
[6]Uniform Resource Identifier
[7]JavaScript Object Notation
[8]eXtensible Markup Language

3 Současné aplikace

Celá práce vychází z průzkumu možností a funkcionality, jenž uživatelům nabízejí aplikace dostupné na oficiálních obchodech. V provedeném průzkumu se objevují téměř výhradně aplikace pro OS Android.

Zmíněné jsou i dvě aplikace pro systém iOS, které poskytují některé zajímavé funkce, jimiž se liší od konkurenčních řešení. Aplikace pro Windows Phone nejsou v průzkumu uvedeny, jelikož oproti výše uvedeným platformám nenabízí žádnou přidanou hodnotu a obecně lze říci, že vzhledem ke svému minoritnímu podílu na trhu s mobilními zařízeními nabízí obchod pro Windows Phone nejméně aplikací.

Nejprve je v této kapitole popsána metodika průzkumu a také jsou definovány různé typy aplikací (podle nabízené funkcionality), které byly do průzkumu zahrnuty. Poté následuje srovnání nejprodávanějších a nejlépe hodnocených aplikací z kategorie produktivita obchodu Google Play, jenž spadají do okruhu aplikací definovaného v kapitole 3.2. Nalezené aplikace či aplikace s neobvyklou funkcionalitou budou popsány podrobněji.

3.1 Metodika

Při zkoumání aplikací se nejprve vycházelo z informací, které jsou dostupné na oficiálních obchodech a na webu vývojáře. Z těchto zdrojů byly převzaty zejména faktické informace o aplikacích, protože je zřejmé, že popis vytvořený přímo vývojářem nebude příliš objektivní.

Žebříčky aplikací pro OS Android byly sestaveny taktéž na základě dat z oficiálního obchodu Google Play. Jendá se o data platná k březnu 2015, která se v čase samozřejmě mění (např. vydání nové verze či vydání další jazykové mutace aplikace může ovlivnit prodejnost i hodnocení uživatelů). Dá se očekávat, že se může změnit pořadí jednotlivých aplikací, ale jejich výčet by měl být víceméně ustálený (jedná se většinou o populární aplikace s několikaletou historií).

Aplikace vyhodnocené podle zvolených kritérií (především prodejnost a hodnocení uživatelů) jako nejúspěšnější (Digical, aCalendar), byly v jejich bezplatných verzích nainstalovány na testovací zařízení a odzkoušeny. U ně-

kterých aplikací (zejména pro iOS) pak pro lepší seznámení s nabízenými funkcemi posloužily i dostupné videorecenze.

3.2 Typy aplikací

Zkoumané aplikace lze dělit podle toho, jaké oblasti správy a plánování času se věnují. Až na drobné výjimky nabízí všechny aplikace **kalendář** s náhledy různých časových úseků od dnů až po roky. V těchto aplikacích si může uživatel vytvořit různé kalendáře (pracovní, osobní, atd.), jež mají zpravidla přidělenou barvu pro přehledné oddělení událostí podle jejich příslušnosti k danému kalendáři.

Další kategorií jsou aplikace pro **správu úkolů**, které umožňují úkoly řadit do seznamů podle různých kritérií jako například čas provedení (deadline), priorita, kategorie, atd. Aplikace, jenž se specializují primárně na organizaci úkolů, často mají vlastní systém (např. bodový, kdy je uživatel hodnocen podle toho, jak je schopen plnit stanovené úkoly), který má uživatele motivovat k řádnému plnění svých úkolů.

V oficiálních obchodech se také vyskytují aplikace, které jsou určeny výhradně na tvorbu a **organizaci poznámek**, ať už se jedná o poznámky textové, obrazové nebo například hlasové. Poznámky lze taktéž přidávat v některých aplikacích, které se na tuto problematiku primárně nezaměřují (např. poznámky k událostem nebo úkolům), ale možnosti jsou v tomto případě omezené.

Dalším typem aplikací, jenž spadají do kategorie produktivita, jsou **budíky a připomenutí**. Jednoduché budíkové aplikace s možností nastavení času a vyzváněcí melodie nabízí v základní výbavě všechny dnešní mobilní platformy. Existují však také sofistikovanější aplikace, které umožňují uživateli nastavovat například maximální počet odložení, vzrůstající hlasitost, automatické odložení o předem nastavený čas, pokud uživatel na budík nereaguje nebo vestavěné funkce jako baterka s využitím LED diody nebo zobrazení informací o počasí přímo po probuzení.

Do průzkumu byly taktéž zahrnuty takzvané **automatizační aplikace**, které otevírají uživatelům nové možnosti, jak co nejlépe využít možnosti dané platformy. Princip těchto aplikací je velmi jednoduchý. Nejprve je nutné nastavit podnět (podmínku), kdy se má provést nějaká akce a poté se už jen

definuje, co se má provést v momentě, kdy je podmínka splněna. Jedním z mnoha příkladů může být třeba pravidlo na automatické vypnutí vyzvánění po desáté hodině večer nebo zapnutí vibračního módu po dobu meetingů, které jsou naplánovány v kalendáři.

3.3 Potřeby uživatelů

Cílem, od něhož se odvíjí potřeby uživatelů na zkoumané aplikace, je jednoduchá a efektivní organizace času, která má uživatelům dopomoci k co nejvyšší produktivitě.

Ideální aplikace by měla maximálně využívat možnosti platformy tak, aby si její uživatel musel pamatovat co nejméně informací o svém programu, událostech nebo úkolech a zároveň byl schopen všechny své povinnosti plnit v co nejkratší době a především v rámci stanovených termínů.

Jedním z hlavních požadavků na tyto aplikace je požadavek na **komplexnost**. Aplikace může být skutečně efektivní jen tehdy, pokud dokáže integrovat informace z různých zdrojů a pokud na základě těchto informací dokáže činit závěry, které lze v dnes přijaté terminologii považovat za chytré (tzn. funkce, které uživateli usnadňují plánování a rozhodování). Lze říci, že je z tohoto pohledu žádoucí, aby aplikace byla schopna pokrýt všechny oblasti správy času. Současně je také uživatelsky příjemnější používat pouze jednu aplikaci namísto několika různorodých a vzájemně nespolupracujících aplikací.

Aby uživatel nebyl nucen stále myslet na svoji agendu, měla by aplikace informovat o jednotlivých událostech pomocí **notifikací** nebo **dialogů**. Typickým příkladem může být dialog pro nastavení budíku, který se uživateli zobrazí po vložení raního meetingu do kalendáře. Na druhou stranu je zjevné, že tyto automatické nabídky a upozornění nesmí být příliš frekventované, aby uživatele neobtěžovaly. Z tohoto pohledu je vhodné poskytnout uživateli možnost nastavení četnosti notifikací.

Jak bylo uvedeno v motivaci práce, je třeba klást důraz také na to, aby aplikace získávaly co nejvíce informací o jejich uživatelích a mohly tak poskytovat výstupy, které budou odpovídat **osobním preferencím** uživatelů. Pokročilé systémy (např. osobní asistenti Siri, Cortana, atd.) jsou schopné s využitím různých metod umělé inteligence v průběhu dlouhodobějšího po-

užívání zjistit různé návyky, zájmy, oblíbená místa, atd. a podle nich pak přizpůsobit např. vyhledávání nebo návrhy [App01][Mic01].

3.4 Žebříčky aplikací

Pro srovnání nejúspěšnějších aplikací slouží následující dvě tabulky, které zároveň ukazují jakým oblastem se dané aplikace věnují. V tabulce 3.1 jsou nejprodávanější aplikace pro organizaci času obchodu Google Play. Jedná se o placené aplikace, které ale většinou nabízí i bezplatnou verzi s omezenou funkcionalitou. Tabulka 3.2 pak ukazuje žebříček deseti nejlepších aplikací (včetně bezplatných) podle hodnocení uživatelů.

Aplikace	**Kalendář**	**Úkoly**	**Poznámky**	**Budíky**
Digical+	ano	ano[1]	ne	ne
Business Calendar Pro	ano	ano[1]	ne	ne
Today Calendar Pro	ano	ano[1]	ne	ne
aCalendar+	ano	ano	ne	ne
Informant 3	ano	ano	ano	ne
Day by Day Organizer	ano	ano	ne	ne
Tasks (Úkoly)	ne	ano	ano	ne
Note Everything Pro	ne	ne	ano	ne
CalenMob	ano	ano[1]	ne	ne
Kalendář +	ano	ano[1]	ne	ne

Tabulka 3.1: Nejprodávanější aplikace.

Jak je z obou tabulek vidět, nejlepší aplikací pro organizaci času je podle statistik oficiálního obchodu Google Play aplikace **Digical+** od vývojáře Digibites. Mezi velmi úspěšné můžeme řadit aplikace, jenž se nacházejí v obou žebříčcích. Konkrétně se jedná o Bussines Calendar, aCalendar+, Day by Day Organizer a Kalendář+.

Všechny zmíněné aplikace mají kalendář s událostmi a také dokážou spravovat úkoly. Některé aplikace však úkoly reprezentují jako klasické kalendářové události s nastaveným připomenutím a nenabízejí tak pokročilé možnosti organizace úkolů, jako například sdružování do skupin nebo projektů, členění podle priorit, atd.

[1] jako událost

Aplikace	**Kalendář**	**Úkoly**	**Poznámky**	**Budíky**
Digical+	ano	ano[1]	ne	ne
Jorte kalendář	ano	ano	ano	ne
aCalendar+	ano	ano	ne	ne
Wunderlist	ne	ano	ne	ne
Business Calendar	ano	ano[1]	ne	ne
Any.DO To Do	ne	ano	ano	ne
Kalendář +	ano	ano[1]	ne	ne
ToDo list	ano	ano	ne	ne
Day by Day Organizer	ano	ano	ne	ne
Upomínky kal. událostí	ano	ano[1]	ne	ne

Tabulka 3.2: Nejlépe hodnocené aplikace.

Mezi nejlepší se dostaly i čtyři aplikace, které neobsahují modul s kalendářem. Jedná se o aplikace, které se věnují organizaci úkolů, a v jednom případě (Note Everything Pro) se do top 10 dostala aplikace, jenž se specializuje výhradně na tvorbu a správu poznámek.

Relativně překvapivým zjištěním je, že žádná z výše uvedených aplikací **neobsahuje modul pro správu budíků**. Je sice samozřejmě pravda, že budíkovou aplikaci mají v základní aplikační výbavě všechna dnešní chytrá zařízení, ale v kombinaci s aplikací pro organizaci času se otvírají další možnosti využití budíků. Jako jednoduchý příklad, který potvrzuje tuto tezi, může sloužit situace, kdy uživatel vloží do kalendáře ranní událost, přičemž v reakci na toto vložení proběhne kontrola nastavení budíků a případný návrh vložení nového budíku v momentě, kdy pro daný den není žádný budík nastavený. Zároveň je také uživatelsky příjemnější spravovat události, jejich připomenutí a budíky v jedné aplikaci.

Minimum zmíněných aplikací se společně s kalendářovými událostmi a úkoly zabývá i tvorbou a **organizací poznámek**. Využití poznámek už není z pohledu integrace informací tak zajímavé, jako například využití budíků, protože aplikace na základě nějaké poznámky (ať už textové, či multimediální) může jen velmi obtížně generovat akce (notifikace, návrhy, připomenutí), které by uživateli napomohly při organizování času. Přesto však může mít modul pro poznámky v těchto aplikacích opodstatnění, jako jednoduchý nástroj pro rychlé zaznamenání myšlenek, nápadů nebo jakýchkoliv jiných informací, které může uživatel později využít jako podklad pro tvorbu nových událostí a úkolů.

3.5 Nejúspěšnější aplikace

V této sekci budou podrobněji popsány aplikace, které se v žebříčcích prodejnosti a popularity vyskytovaly na předních příčkách.

3.5.1 Digical

Vývojář:	Digibites
Cena:	96,69 Kč
Instalace:	100 tis. - 500 tis. (placená), 1 mil. – 5 mil. (zdarma)
Verze OS:	Android 2.2 a vyšší

Aplikace Digical+ se umístila v obou zmíněných statistikách na prvním místě. Jedná se o placenou verzi, přičemž vývojář nabízí i verzi bezplatnou, ve které např. není widget, nelze volit mezi zobrazovanými kalendáři a v aplikaci jsou zobrazené reklamy [Dig01][Gpl03].

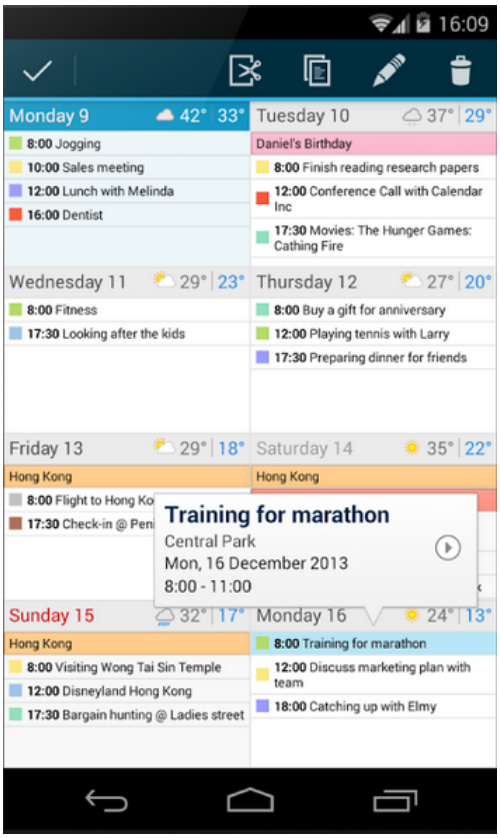
Klíčové funkce:

- různá kalendářová zobrazení,
- přizpůsobitelné widgety,
- možnost přidání kontaktů k událostem,
- integrované vyhledávání lokality, zobrazení na mapě (Google Maps),
- integrovaná předpoveď počasí,
- podpora Google Now pro sledování aktuální dopravní situace,
- oznámení se zkratkami k odložení, mapám a navigaci,
- synchronizace s Google Calendar, Outlook.com a Exchange,
- Smart Action Bar pro snadné kopírování, vkládání a editování událostí.

Aplikace je stejně jako několik dalších z uvedeného výčtu postavena na Google Calendar. Tento kalendář používá velké množství uživatelů, takže možnosti nastavení událostí v aplikaci jsou pro většinu uživatelů důvěrně známé.

Velmi užitečnou funkcí jsou notifikace, které uživatele podle vytvořených událostí informují, kdy je vhodné odejít, nebo jakou trasu v závislosti na aktuálním provozu zvolit. Tato funkcionalita je zajištěna pomocí systému Google Now.

Podobně jako někteří konkurenti nabízí i Digical+ integrované vyhledávání lokací s automatickým dokončováním. Zvolenou lokaci lze následně zobrazit na mapě, což u ostatních řešení (s výjimkou Google Calendar[2]) přímo uvnitř aplikace možné není.



Obrázek 3.1: Týdenní náhled kalendáře s lištou Smart Action Bar [Gpl03]

Ovládání aplikace je na velmi dobré úrovni. Přepínání mezi jednotlivými náhledy je intuitivní a provádí se z jednoho místa (Action Bar), případně klepnutím na konkrétní den, týden, atd. v zobrazeném kalendáři. K dobré ovladatelnosti napomáhá i tzv. Smart Action Bar (viz obr. 3.1). Jedná se o lištu nástrojů, která nahradí klasický Action Bar při označení libovoné kalendářové události. Na liště se nachází tlačítka pro vyjmutí, vložení a kopírování

[2]ne ve všech verzích

události na jiné místo v kalendáři nebo také tlačítko pro editaci dalších detailů konkrétní události.

Závěr

Aplikace Digical+ představuje velmi dobrou nadstavbu nad Google Calendar s velkým množstvím přidaných funkcí. Z pohledu této práce je však zajímavá jen funkce, která na základě polohy přiřazené k události dokáže uživateli naplánovat ideální časový harmonogram včetně nejméně časově náročné cesty do cíle.

3.5.2 Business Calendar

Vývojář:	Appgenix Software
Cena:	58 Kč
Instalace:	100 tis. - 500 tis. (placená), 5 mil. - 10 mil. (zdarma)
Verze OS:	Android 2.0.1 a vyšší

Mezi nejúspěšnější aplikace lze zařadit také Business Calendar. Stejně jako většina konkurentů nabízí jak placenou verzi Pro, tak i bezplatnou alternativu pro vyzkoušení. V současné době je na Google Play dostupný Businnes Calendar 2, který kromě vylepšení stávajících nabízí i některé nové funkce [Apg01][Gpl04].

Klíčové funkce:

- různá kalendářová zobrazení,
- přizpůsobitelné widgety,
- přidávání kontaktů k událostem,
- vytváření šablon událostí,
- automatické dokončování při vyhledávání lokací,
- import a export kalendářových dat (.ical, .ics).

Stejně jako Digical, je i Business Calendar rozšířením Google kalendáře. Vyjma několika definovaných kalendářových zobrazení (měsíc, týden, atd.), je

možné jednoduchým tahem označit dny, pro které se vytvoří vlastní view, kde budou zobrazeny pohromadě. V aplikaci se také využívají informační pop-up okna (např. s denní agendou), aby nedocházelo k častému přepínání zobrazení, jenž může být v některých případech zmatečné. Právě snadné a účelné ovládání je hlavní vyhodou této aplikace.

Oproti konkurenci nabízí Business Calendar např. šablony pro kalendářové události, které uživateli usnadní vytváření podobných událostí podle dříve uloženého vzoru. V kategorii úkolů je užitečná možnost přidávání podúkolů nebo organizování úkolů podle priorit.

Závěr

Business Calendar je bezesporu jednou z nejlepších kalendářových aplikací v obchodě Play, nicméně z hlediska integrace informací a chytrých funkcí toho příliš nenabízí.

3.5.3 aCalendar+

Vývojář:	Tapir Apps GmbH
Cena:	99 Kč
Instalace:	100 tis. - 500 tis. (placená), 5 mil. - 10 mil. (zdarma)
Verze OS:	Android 2.1 a vyšší

Mezi tři nejúspěšnější aplikace se dostal i aCalendar+ od Tapir Apps, který se na svoji stranu snaží získat mladší uživatele [Tap01][Gpl01].

Klíčové funkce:

- různá kalendářová zobrazení,
- přizpůsobitelné widgety,
- přidávání kontaktů k událostem,
- přidávání fotografií k událostem,
- podpora Google Tasks,
- automatické dokončování při vyhledávání lokací,

- sms a e-mail připomenutí.

Všechny aplikace, které se nacházejí v žebříčku na předních místech, mají spoustu různých zobrazení, widgetů a funkcí, přičemž aCalendar není výjimkou. Velmi praktické je např. malé view s konkrétním měsícem, které se zobrazuje při náhledu dne nebo týdne.

Z funkcí, jenž u předchozích aplikací chybí, lze zmínit např. zasílání připomenutí přes sms nebo e-mail[3]. Dále je též možné přidávat k událostem fotografie a pokud u nějakého kontaktu je nastavený obrázek, objeví se v kalendáři v den jeho narozenin. Užitečnou funkcí je také sdílení událostí technologií NFC nebo pomocí QR kódů.

Závěr

Jak už bylo řečeno, aCalendar je aplikace bohatá na možnosti zobrazení, funkce a nastavení, avšak podobně jako v Bussines Calendar zde chybí chytré funkce, které by uživateli ještě více usnadnily správu času.

3.6 Vybrané aplikace

V následující sekci budou zmíněny aplikace, které sice nemají celkové hodnocení nebo počet instalací na takové úrovni jako předchozí aplikace, ale nabízejí nějakou zajímavou funkcionalitu, díky níž se odlišují od konkurenčních řešení. Budou zde představeny i zástupci aplikací pro iOS.

3.6.1 AutomateIt

Vývojář:	SmarterApps Ltd
Cena:	68,95 Kč
Instalace:	50 tis. - 100 tis. (placená), 500 tis. - 1 mil. (zdarma)
Verze OS:	Android 2.2 a vyšší

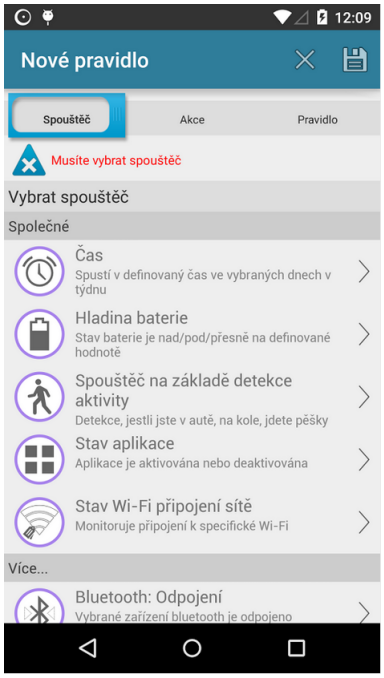
AutomateIt je zástupcem automatizačních aplikací, jejichž smyslem je

[3]jen v placené verzi

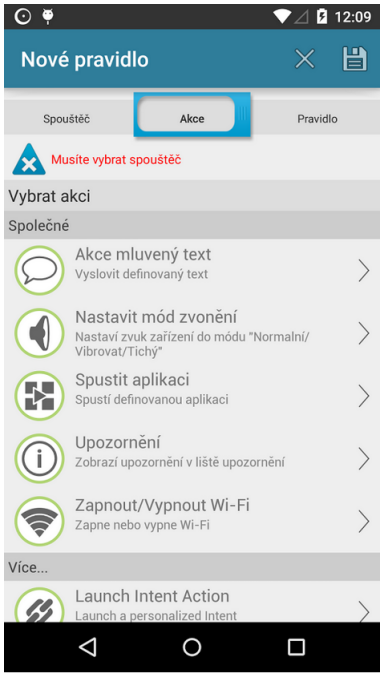
provádět různé, předem nastavené akce v okamžiku splnění definované podmínky - tzv. spouštěče (viz sekci 3.2) [Aut01][Gpl02].

V současné době lze v základní verzi použít 28 různých spouštěčů a v reakci na ně si může uživatel vybrat z 26 různých akcí s tím, že některé funkce mohou být omezeny jak zařízením, tak i verzí systému Android. V Pro verzi lze vytvářet složitější pravidla za použití složených spouštěčů a složených akcí, což znamená, že v rámci jednoho pravidla může být najednou spuštěno více akcí, které jsou závislé na více spouštěčích (akce se spustí, když se všechny spouštěče aktivují).

Na webu vývojáře lze nalézt základní užitečná pravidla zdarma. Uživatel si také samozřejmě může definovat pravidla vlastní (viz obr. 3.2). Výhodou je, že kolem této aplikace vznikla početná, aktivní komunita, která produkuje nová pravidla, s nimiž se obchoduje na tzv. Rules Market.



Obrázek 3.2.a: Výběr spouštěče



Obrázek 3.2.b: Výběr akce

Obrázek 3.2: Definice nového pravidla [Gpl02]

Vývojář se také snaží neustále rozšiřovat možnosti aplikace. Uživatel si může do aplikace doinstalovat nabízené pluginy, které otevírají nové možnosti při tvorbě pravidel. Jako příklad lze uvést Facebook plugin, který umožňuje aktualizovat status na základě libovolného spouštěče, nebo LIFX plugin, jenž

dokáže ovládat LIFX osvětlení v domě podle definovaných pravidel.

Závěr

Myšlenka automatizačních aplikací obecně je sice velmi jednoduchá, ale poskytuje silný nástroj k lepšímu využití širokých možností dnešních mobilních zařízení. Toto řešení může být inspirací při realizaci zamýšlené funkcionality vytvářené aplikace, kdy má např. dojít ke spuštění budíku v závislosti na aktuálním počasí.

3.6.2 Timeful

Vývojář:	Timeful, Inc.
Cena:	zdarma
Instalace:	počet není dostupný
Verze OS:	iOS 7.0 a vyšší

Timeful je kalendářová aplikace, pomocí které lze spravovat události, úkoly a zvyky (habits). V současné době je aplikace dostupná pro systém iOS, přičemž verze pro Android je ve fázi testování [Tim01][App03].

Klíčové funkce:

- různá kalendářová zobrazení,
- návrhy pro umístění událostí,
- schopnost přizpůsobovat se uživateli,
- habits a jejich historie,
- drag'n'drop pro snadné umístění událostí do kalendáře,
- synchronizace s nejpoužívanějšími kalendářovými službami.

Timeful je aplikace, která vyniká jednak svým jednoduchým vzhledem a ovládáním, ale především je nutné zmínit její chytré notifikace a návrhy plánování jednotlivých událostí, k čemuž využívá poznatky o chování a zvycích jejích uživatelů.

Po instalaci se aplikace uživatele dotáže na základní údaje jako například časové rozmezí, kdy je většinou v práci nebo kdy spí, nejproduktivnější část dne, atd. Tyto informace jsou následně použity k zpřesnění návrhů tak, aby uživateli co nejvíce vyhovovaly. Kromě těchto údajů je aplikace schopna v průběhu používání rozpoznat opakující se zvyky uživatelů a jejich preference, jenž opět vedou k lepšímu rozhodování při plánování.

Průběžně získané informace aplikace využívá například v situaci, kdy uživatel přidá úkol, který má být hotový do týdne a nezadá přitom přesný čas jeho provedení do kalendáře. Aplikace určí nejvhodnější místo v kalendáři a akci naplánuje, přičemž uživatel může posléze akci přeplánovat.

Další velmi zajímavou částí aplikace je správa zvyků (habits). Aplikace tak uživateli napomáhá udělat si čas na aktivity, které ho baví nebo jsou mu nějak prospěšné (posilovna, četba, atd.). Stačí přidat aktivitu a zamýšlenou četnost (např. třikrát v týdnu) a aplikace bude plánovat tyto události do volných míst v kalendáři. Po určité době je navíc možné sledovat historii jednotlivých aktivit, ve které je vidět, jak se uživatel skutečně svým zálibám věnoval.

Závěr

Timeful nabízí velmi zajímavé funkce, které u podobných aplikací nejsou běžné. Zejména chytré plánování, notifikace a schopnost učit se uživatelské preference mohou velmi zefektivnit správu času.

3.6.3 Omnifocus

Vývojář:	The Omni Group
Cena:	19.99 $ (cca 490 Kč)
Instalace:	počet není dostupný
Verze OS:	iOS 8.0 a vyšší

Společnost The Omni Group se věnuje především vývoji aplikací pro organizaci času a v této kategorii nabízí tři aplikace. Jedná se o plánovací aplikaci OmniPlan, aplikaci na poznámky a organizaci informací OmniOutliner a aplikaci na správu úkolů OmniFocus. Omnifocus je aplikace, která je postavena

na metodě organizace práce GTD[4]. V současné době je dostupná ve třech verzích, a to pro mobilní telefony iPhone, tablety iPad a osobní počítače Mac [Omn01][App02].

Klíčové funkce:

- organizace úkolů podle projektů, složek, priorit, kontextů,
- vytváření úkolů z jiných aplikací (Safari, Twitterlator)
- plánování a upozorňování na úkoly podle lokace,
- spolupráce s osobním asistentem Siri,
- přidávání multimediálních příloh k úkolům,
- funkce Forecast pro přehled o aktivitách v následujícím týdnu.

Getting Things Done

GTD je populární metoda, jenž dělí řízení nějakého pracovního úkolu do pěti kroků. Cílem tohoto dělení je přizpůsobení se schopnostem lidského mozku tak, aby člověk nemusel myslet na to, co a kdy má splnit a soustředil se primárně na samotné provedení úkolu [Omn01].

V první fázi GTD dochází ke **sběru informací** a jejich vložení do inboxu. Do inboxu uživatel vkládá různé úkoly, které ho právě napadnou, úkoly na základě přijatých e-mailů nebo hovorů, atd.

Druhým krokem je **zpracování položek**, které se nacházejí v inboxu. Zpracování probíhá podle jednoduchého klíče. Pokud je uživatel schopen úkol provést v krátkém čase (např. do dvou minut), doporučuje se úkol provést ihned a odškrtnout ho v inboxu. Druhou možností je delegování úkolu na někoho jiného v případě, že se nejedná o zodpovědnost daného uživatele. Poslední a nejčastější možností je odložení úkolu na pozdější zpracování.

Položky, jenž byly uživatelem odloženy, je nutné kvůli lepší přehlednosti **zorganizovat**. V aplikaci je možné vytvářet různé projekty, do kterých se následně jednotlivé položky vloží. Některé úkoly nemusí logicky náležet do žádného projektu. Takové úkoly se vkládají do tzv. seznamů jednoduchých

[4]Getting Things Done

akcí. Nejvyšší úrovní členění jsou složky, které dělí jednotlivé projekty podle kategorií (např. práce, škola, osobní, atd.).

V momentě, kdy jsou všechny úkoly zorganizované, zbývá je jen začít podle různých priorit nebo např. aktuálnosti dílčích projektů **provádět**. Poslední a velmi důležitou součástí GTD je **zhodnocení** (review). Doporučený interval zhodnocení je jeden týden, ale především záleží na důležitosti projektu. Aplikace nabízí na hodnocení zvláštní perspektivu, ve které je vidět pokrok na jednotlivých projektech, stav položek, jenž byly odloženy na pozdější datum, nebo připomenutí úkolů závislých na ostatních lidech.

Ostatní funkce

Velmi užitečnou funkcí je zobrazení úkolů podle lokací. Stačí jen úkolům přidat lokaci, na které jeho provedení zavisí, a poté se v tzv. nearby view zobrazují úkoly, podle aktuální polohy uživatele.

S možností přiřazení lokací souvisí i použití kontextů. V aplikaci totiž existuje také zobrazení, které dělí úkoly podle jejich kontextu. Kontext je definován jako závislost potřebná k dokončení úkolu. Typickým příkladem kontextu pak může být např. lokace, osoba (nadřízený, člen rodiny) nebo zařízení (telefon, osobní počítač).

Závěr

Omnifocus je velmi propracovaná aplikace, jejíž vývoj pro různé platformy probíhá již od roku 2007. Uživateli nabízí vhodný nástroj pro organizaci času metodou GTD. Skutečnost, že The Omni Group poskytuje tři různé aplikace pro organizaci času, je v rozporu s jedním bodem motivace této práce, ale vzhledem ke komplexnosti jednotlivých aplikací je zvolený způsob distribuce odůvodnitelný.

4 Navrhovaná funkcionalita

V této sekci bude na základě provedeného průzkumu současných nejúspěšnějších (tzn. nejvíce prodávaných nebo nejlépe uživatelsky hodnocených) aplikací pro organizaci času definována funkcionalita vytvářené aplikace. V první fázi je třeba definovat, jakými oblastmi organizace času se bude aplikace zabývat. Následně budou zmíněny konkrétní funkce společně se stručným nástinem jejich realizace (viz 2.1).

4.1 Oblasti organizace času

Jak bylo uvedeno v sekci 3.2, můžeme aplikace, jenž se zabývají organizací času rozdělit na kalendářové, úkolové a na aplikace, které se věnují organizaci poznámek. První dvě kategorie se pro navrhovanou aplikaci přímo nabízí a neměly by v návrhu chybět. Poznámky minimálně v první verzi aplikace nebudou, jelikož většinou slouží jen jako rychlý nástroj pro zaznamenání myšlenek uživatele a v aplikaci by se jejich vlastní obsah dal jen těžko dále využít.

4.1.1 Kalendář

Naprostou samozřejmostí a základem vytvářené aplikace je **kalendářový modul**. Uživatel bude moci přidávat nové kalendáře, k nimž je možné přiřadit barvu pro snadné odlišení událostí v jednotlivých náhledech. Přidané kalendářové události pak bude možné přesouvat mezi dostupnými kalendáři.

V první verzi aplikace budou k dispozici dva kalendářové náhledy, a to týdenní a denní s tím, že v týdenním náhledu bude zobrazen malý náhled dnů v měsíci dle aktuálního zobrazeného týdne. Denní náhled pak zobrazí události (začínající, pokračující nebo končící v daný den) i s podporou překrývání.

4.1.2 Úkoly

Podobně, jako je tomu u mnoha zkoumaných aplikací, bude i navrhovaná aplikace obsahovat modul pro **správu úkolů**. Vkládané úkoly bude možné organizovat do předem vytvořených projektů, jenž uživatel bude moci přiřadit k libovolnému kalendáři (osobní, pracovní, atd.).

Aplikace nabídne různé náhledy úkolů podle jejich nejdůležitějších parametrů, kterými jsou priorita, čas splnění úkolu a příslušnost k nějakému projektu. Součástí modulu s úkoly bude i jednoduchý nákupní seznam s možností přidávání položek a jejich odškrtáváním např. v průběhu nákupu.

4.1.3 Budíky

Přestože správa budíků s organizací času jako takovou úzce souvisí, neposkytuje žádná z výše uvedených aplikací ani elementární modul pro práci s budíky.

Vytvářená aplikace nabídne možnost nastavení budíků pro jednotlivé režimy dne (viz 4.2) a také možnost vkládat vlastní budíky s individuálním nastavením pro zvolené dny v týdnu.

4.2 Režimy dne

Aplikace zavádí nad rámec klasických kalendářových aplikací tzv. režimy jednotlivých dnů, které se primárně řídí tím, jestli je daný den dnem pracovním či nikoliv. Mimo tyto dva režimy bude možné použít speciální režim pro velmi důležité dny (státní zkouška, důležitá schůzka, atd.).

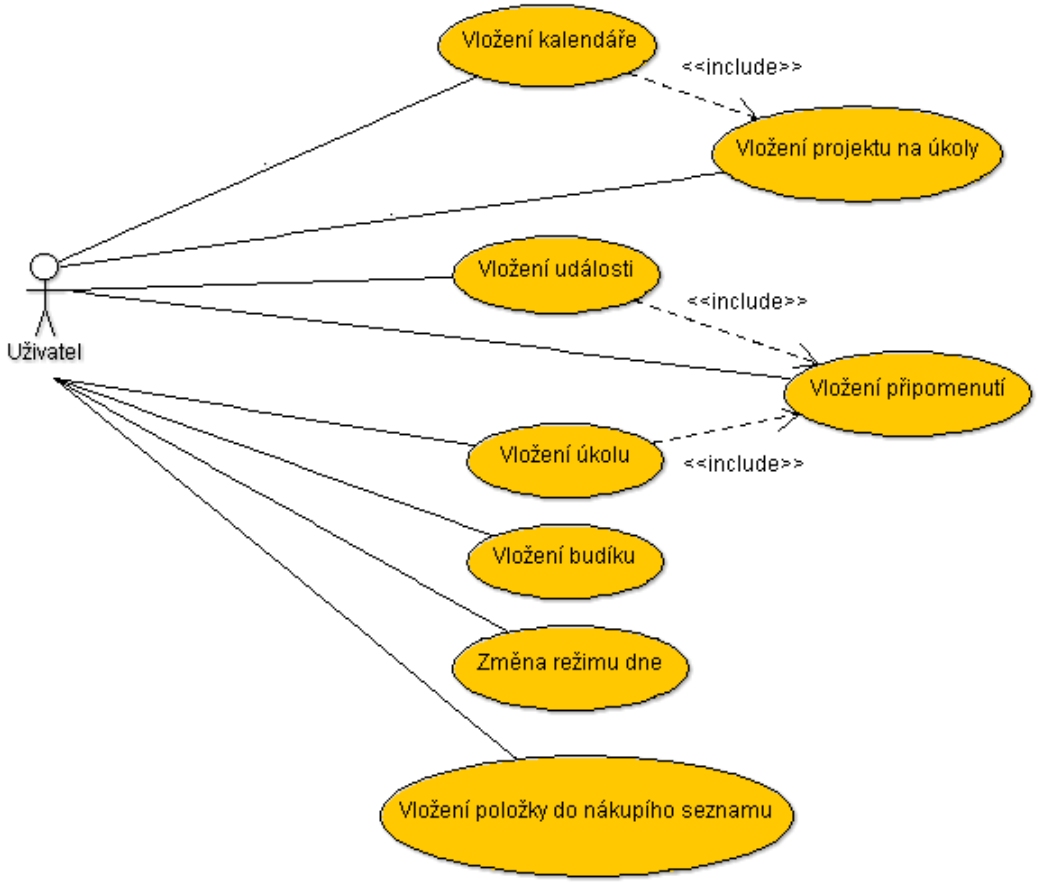
Zvolený režim dne pak přímo ovlivňuje například použití přednastavených budíků nebo spouštění podmíněných akcí (viz 4.4), které jsou aktivovány na základě vyhodnocení předdefinovaných podmínek.

V případě režimu důležitého dne bude aplikace v předstihu uživatele informovat o tom, že by si mohl přidat zvláštní budík (nezávislý na defaultních budících pro pracovní a nepracovní dny), pokud žádný jiný nebude nastaven. Jestliže pak uživatel na budík v důležitý den nebude reagovat, odešle se

automaticky na číslo kontaktní osoby uvedené v nastavení aplikace textová zpráva, která vyzývá kontaktní osobu, aby se pokusila spojit s uživatelem a ověřit, zda ještě nespí.

4.3 Případy užití

Z dosud popsaných funkcionalit, jenž bude vytvořená aplikace nabízet, vyplývá několik případů užití (use case), které se týkají jednotlivých oblastí správy času a nastavování režimů dne. Přestože k aplikaci přistupuje pouze jeden aktivní aktér (uživatel aplikace), je na obrázku 4.1 znázorněn diagram užití, který je zde uveden zejména pro ucelený pohled na možnosti aplikace a k dovysvětlení některých vztahů mezi případy užití.



Obrázek 4.1: Diagram případů užití

Nejprve je třeba zmínit, že jsou pro jednoduchost z diagramu vypuštěny případy užití pro editaci nebo mázání kalendářů, událostí, úkolů, atd., i když samozřejmě v aplikaci tyto případy užití budou implementovány.

V diagramu se vyskytují kromě jiného i vazby typu `include`, které značí, že případ, jenž je napojen touto vazbou, je proveden vždy, když je proveden původní případ (tj. případ, ze kterého vazba vede) [Sch01]. Konkrétně to pak v tomto případě znamená, že vždy když je vložen nový kalendář, vloží se pro tento kalendář defaultní projekt na úkoly. Samozřejmě je ale možné přidávat do kalendáře libovolně další projekty.

Podobná situace jako u projektů nastává v případě vkládání kalendářových položek (události a úkoly). Při vložení události nebo úkolu se automaticky vloží i všechna připomenutí, která k vkládané položce přísluší. Kdykoliv poté je zároveň možné přidávat další připomenutí jednotlivě.

4.4 Podmíněné akce

Další plánovanou funkcionalitou jsou tzv. podmíněné akce, které se po vzoru automatizačních aplikací (viz 3.6.1) provedou v momentě splnění všech nastavených podmínek. Podmínky budou definovány pro různé kategorie správy času, přičemž jejich splnění většinou vyústí v zobrazení notifikace nebo dialogu.

Vyhodnocování podmínek budou provádět pravidelné služby naplánované s využitím `AlarmManager`u (viz 5.1.3) nebo budou podmínky vyhodnocovány ihned po zpracování uživatelského vstupu. Jednou z možností je i spuštění vyhodnocení v reakci na událost nějakého senzoru (např. GPS).

Možnosti podmínek jsou omezeny v zásadě jen **datovým modelem** navrhované aplikace. Systém totiž vždy při vyhodnocení podmínek získá potřebná data a porovná je s nějakou mezní hodnotou. Uživatel bude moci nastavovat některé mezní hodnoty podmínek (v nastavení aplikace), ale implementace nástroje pro tvorbu vlastních pravidel jako je tomu u automatizačních aplikací plánována není (není cílem této aplikace).

Implementace nových podmíněných akcí je však poměrně snadno realizovatelná (jednoduchý princip získání dat a porovnání) a díky širokým možnostem platformy Android zde existuje velký prostor pro tvorbu nových pravidel.

Pokud by ale o splnění podmínky mělo rozhodovat kritérium, které nebude možné vyhodnotit za použití navrženého datového modelu, bude nutné model upravit.

V následujícím textu budou uvedeny příklady podmínek (dle jednotlivých kategorií organizace času) a reakcí aplikace na jejich splnění. Většina z navržených podmíněných akcí bude později realizována.

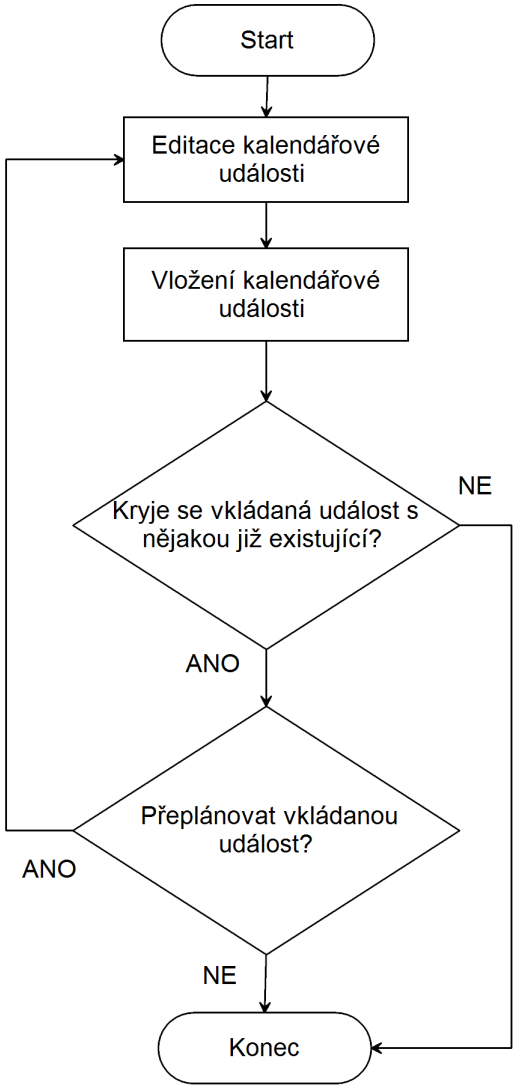
4.4.1 Kalendářové události

- Začátek události se blíží (např. následující den) a účast je nerozhodnutá - notifikace s možností přechodu na editaci události.
- Událost nemá žádné připomenutí a účast je potvrzená - notifikace s dostatečným předstihem o tom, zda o této události uživatel ví.
- Přidání události, která se kryje s jinou, dříve přidanou událostí - dialog s dotazem, jestli to tak má skutečně být (případně možnost přeplánovat).
- Přidání události s místem konání mimo dosah domova - dialog s možností přejít na webovou stránku s vyhledáváním hotelů v dané lokalitě.

4.4.2 Úkoly

- Uživatel se přiblíží k lokaci, která je uvedena u nějakého úkolu - notifikace o tom, zda nechce uživatel daný úkol rovnou splnit.
- Uživatel se přiblíží k obchodu a jeho nákupní seznam není prázdný - notifikace o tom, že by mohl nakoupit položky v seznamu, pokud má čas.
- Deadline úkolu již vypršel - dialog s možností přejít na editaci úkolu (přeplánování času provedení úkolu).
- Úkol již po nějakou dobu nemá určený čas provedení - dialog s možností přejít na editaci úkolu (naplánování nebo splnění úkolu).

Situace z podmíněné akce, jenž kontroluje případné kolize mezi událostmi, je naznačena vývojovým diagramem na obrázku 4.2.

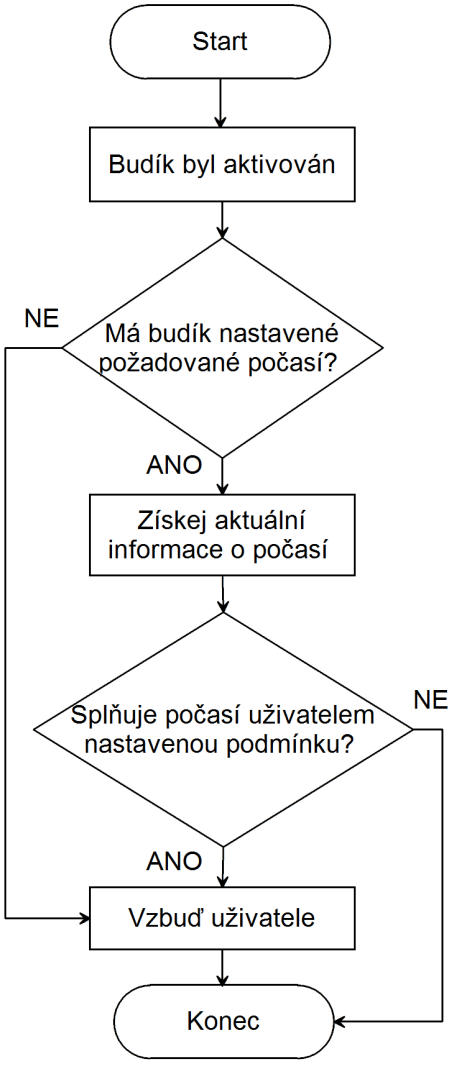


Obrázek 4.2: Přeplánování kalendářové události

4.4.3 Budíky

- Pracovní den a budík není nastavený nebo je nastavený na pozdní dobu - notifikace s možností přechodu na nastavení budíků.

- Den pracovního klidu a budík je nastavený na neobvykle časnou dobu - notifikace s možností přechodu na nastavení budíků.
- Přidání ranní události a budík pro daný den není nastaven - dialog s možností přejít na editaci budíků.
- Nastaveny dva (nebo více) budíky na jeden den - notifikace s možností přechodu na nastavení budíků.
- Budík s možností nastavení požadovaného počasí (např. vzbudit uživatele jen když nebude v čase budíku pršet).



Obrázek 4.3: Vyhodnocení budíku

Na obrázku 4.3 je znázorněno vyhodnocení budíku s přihlédnutím na aktuální počasí. Pokud se například z důvodu nedostupnosti internetového připojení nepodaří získat data o počasí, musí být uživatel vzbuzen pouze na základě nastaveného času.

4.5 Periodické vyhodnocování

Podmínky definované v předchozí sekci mohou být vyhodnocovány na základě třech různých situací, a to:

1. **Po uživatelském vstupu.** Kontrola splnění definovaných podmínek se spustí např. v momentě, kdy uživatel vkládá událost do kalendáře.
2. **Událost ze senzorů.** Sledováním polohy uživatele aplikace zjistí, že se uživatel nachází v blízkosti obchodu a může tak kontrolovat, zda má nějaké položky v nákupním seznamu a případně informovat o tom, že by je mohl nakoupit.
3. **Plánované služby.** Vyhodnocování podmínek může být také spouštěno v předem nastavených časových intervalech.

Pro potřeby periodického vyhodnocování je vhodný `AlarmManager` (viz 5.1.3), s jehož využitím lze naplánovat spuštění jakékoliv aplikační komponenty [And07][Lac01]. V praxi by tak naplánovaný alarm spouštěl jednotlivé služby, které by na pozadí nebo i mimo životní cyklus aplikace vyhodnocovaly definované podmínky.

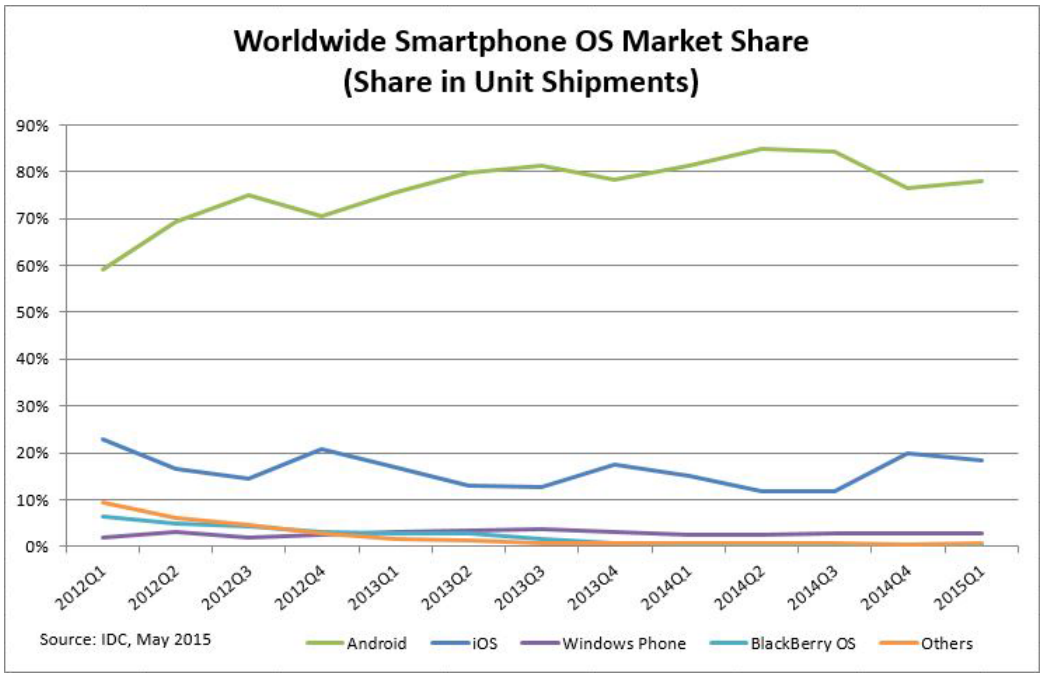
Služby mohou být plánovány s libovolým časovým intervalem opakování. Pro kontrolu nastavení budíků je např. vhodný denní interval, kdy aplikace každý večer zkontroluje, zda je na další den nastavený nějaký budík. Příkladem delšího intervalu opakování může být plánování služby pro upozornění na úkoly bez nastaveného času nebo na úkoly s vypršelou dobou provedení, u které může být dostačující týdenní interval.

5 Technologie

V této kapitole budou popsány technologie použité při vývoji aplikace. Nejprve jsou zmíněny některé části platformy Android. V následujících sekcích jsou představeny další knihovny a API.

5.1 Android

Platforma Android je už po několik let jasně nejrozšířenější platformou pro mobilní zařízení na světě. Pokud budeme brát v potaz mobilní zařízení jako celek (chytré telefony a tablety zároveň), tak platí, že přibližně každé druhé zařízení běží na OS Android (přesný podíl 51.59%[1]) [Net01]. Vezmeme-li v úvahu pouze mobilní telefony je podíl platformy Android na trhu ještě větší, a to 78.0% (viz obr.5.1) [Idc01].

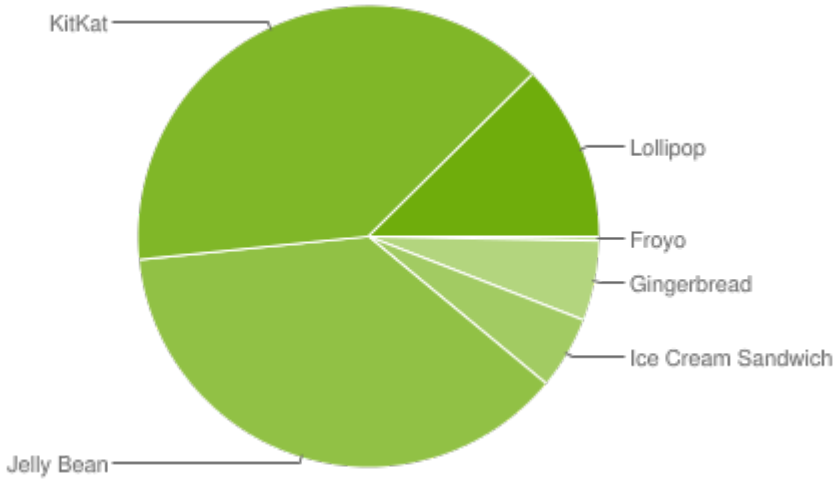


Obrázek 5.1: Podíl mobilních platforem na trhu [Idc01]

Momentálně nejnovější verzí OS Android je verze 5.1 s kódovým označe-

[1] platí pro květen 2015

ním Lollipop, která byla uvedena v březnu roku 2015. Jak je vidět na obrázku 5.2, nejvíce používanou verzí je aktuálně 4.4 (KitKat) s podílem 39.2%, následují verze s kódovým označením Jelly Bean (4.1, 4.2, 4.3), jenž mají podíl 37.4%, a nejnovější Lollipop je již používán na 12.4% zařízení[2] [And04].



Obrázek 5.2: Používané verze OS Android [And04]

Vzhledem k popsanému podílu verzí u momentálně používaných zařízení bude vyvíjená aplikace podporována ve všech verzích od 4.0.3 (Ice Cream Sandwich, API level 15) včetně.

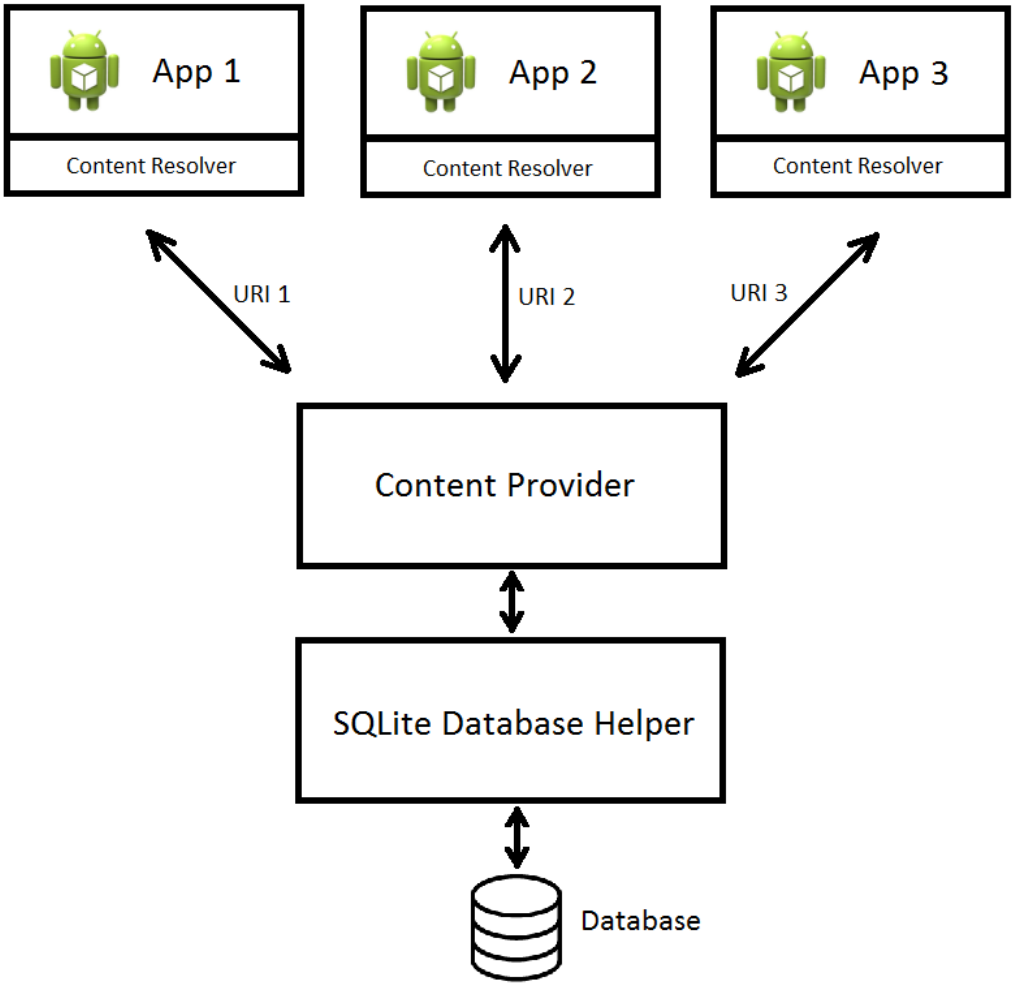
5.1.1 Content Provider

Content provider je aplikační rozhraní, které spravuje přístup k strukturovaným datům. Toto rozhraní zapouzdřuje potřebná data (ať už jsou uložena ve formě souborů či v SQLite databázi), poskytuje k nim přístup a také umožňuje definovat mechanismy pro ochranu dat [And03][Gra01].

Primárním účelem Content Providerů je poskytovat data ostatním aplikacím. Navrhovaná aplikace sice minimálně v její první verzi nebude poskytovat svá data, ale použití Content Provideru bylo zvoleno proto, že v budoucích verzích by bylo vhodné implementovat mechanismus synchronizování dat na server (záloha dat, případně poskytnutí dat pro webovou aplikaci), pro který je vhodný právě Content Provider v kombinaci se synchronizačním adaptérem (Sync Adapter) platformy Android.

[2]platí k 1. 6. 2015

Dalším důvodem pro volbu Content Provideru byla skutečnost, že Android nabízí některé předdefinované providery, mezi nimiž je i **Calendar Provider**. Použití Calendar Provideru výrazně usnadňuje vývoj při práci s kalendářovými daty, jelikož už je zde implementována např. podpora opakujících se událostí i s definovanými pravidly pro jejich vkládání, nebo možnost vkládat k událostem připomentí. Výhodou pak je, že připomenutí stačí vložit do databáze a vlastní spuštění dialogu s připomenutím (se zvukem nebo vibrací) už kompletně zajistí operační systém dle globálního nastavení. Mimo zmíněného Calendar Provideru byl v aplikaci vytvořen i jeden vlastní provider, který spravuje přístup k datům o projektech (organizace úkolů), budících nebo nákupním seznamu.



Obrázek 5.3: Schéma přístupu ke Content Provideru

Z obrázku 5.3 je vidět, že v aplikaci se ke Content Provideru přistupuje

pomocí klientského objektu **Content Resolver**, který je dosptupný z **Context**u aplikace (např. z jakékoliv aktivity). Metody tohoto objektu poskytují základní CRUD (create, retrieve/read, update, delete) funkce pro přístup k úložišti dat. Na základě URI[3], která je parametrem volaných metod pak Content Resolver volá stejnojmenné metody Content Provideru (odpovídajícího předané URI), jenž následně vrací požadovaná data [Gra01].

Konkrétní Content Provider tedy musí implementovat metody insert, update, delete a query, které již přistupují s využitím database helper objektu k databázi.

5.1.2 Service

Service neboli služba je komponenta aplikace, která neposkytuje žádné uživatelské rozhraní a běží na pozadí aplikace, takže se hodí například na provádění dlouhotrvajících operací. Ostatní komponenty (včetně komponent jiných aplikací, pokud služba není definována v manifestu jako `private`) mohou definované služby spouštět nebo s nimi komunikovat na principu komunikace klient-server. Obecně lze službu implementovat tak, aby poskytovala obě možnosti interakce [And09][Lac01].

Startované služby (started)

Služba je považována za spuštěnou po zavolání metody `startService()` jakoukoliv jinou aplikační komponentou. V momentě, kdy je služba spuštěná, může běžet nezávisle na komponentě, která ji spustila (tzn. i po ukončení životního cyklu (destroy) spouštěcí komponenty). Tento typ služeb se používá většinou pro provedení jedné operace (stažení souboru z internetu) s tím, že po dokončení operace se služba sama zastaví (obr. 5.4 vlevo).

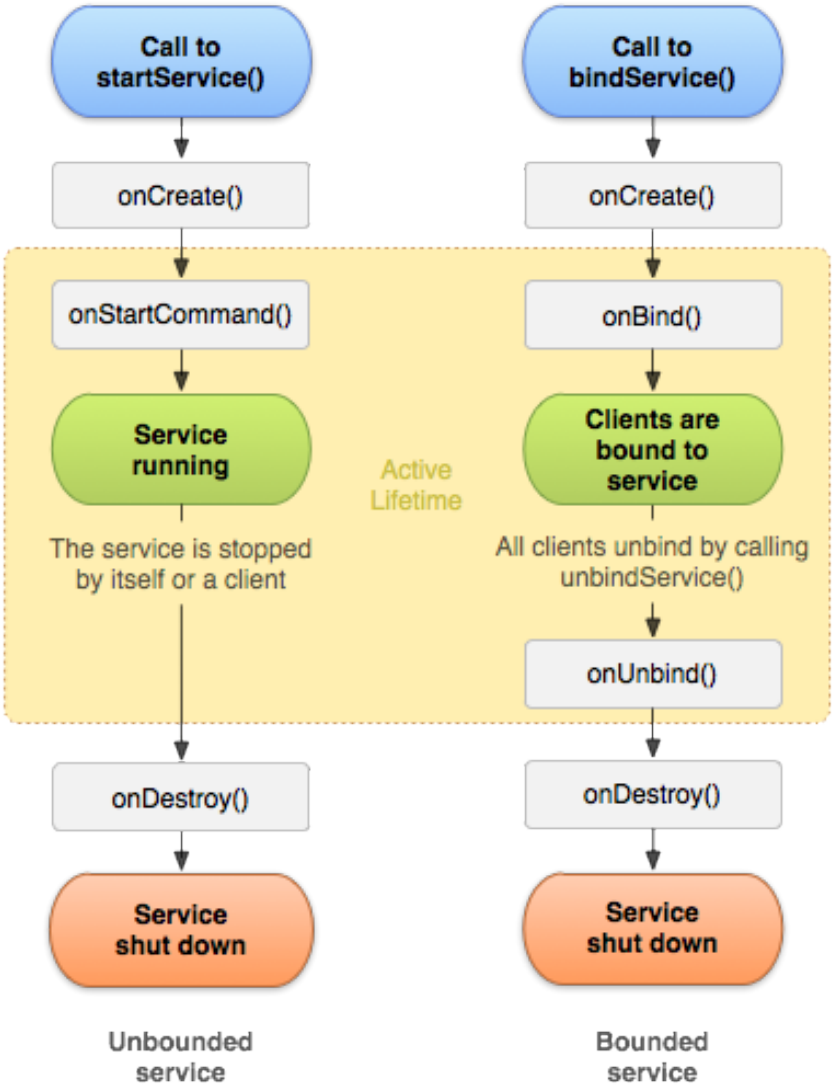
Vázané služby (bounded)

Tento typ služby se „naváže“ na libovolnou aplikační komponentu voláním metody `bindService()`. Taková služba poté nabízí rozhraní pro klient-server komunikaci. Komponenty tak mohou na službu libovolně posílat požadavky

[3]Unified Resource Identifier

a získávat výsledky. Na jednu službu se může vázat zároveň více komponent, ale rozdílem oproti spouštěným službám je, že v momentě kdy ukonční svojí činnost poslední navázaná komponenta, služba se automaticky ukončí.

Na obrázku 5.4 je znázorněn životní cyklus obou typů služeb, ze kterého je vidět, že spouštěné služby (na obrázku jako unbounded) mohou běžet nezávisle na komponentách, ale mohou být také na druhou stranu jakoukoliv komponentou zastaveny. Častější však bývají případy, kdy služba zastaví sama sebe po dokončení své činnosti.



Obrázek 5.4: Životní cyklus služby [And09]

Před použitím služeb je důležité uvědomit si, že pokud programátor neurčí jinak, daná služba po svém spuštění nevytváří nové vlákno ani nový proces. Pokud tedy služba má provádět operace náročné na výpočetní výkon, měla by si pro svůj běh vytvořit zvláštní vlákno a předejít tak možným problémům s odezvou aplikace.

5.1.3 Alarm Manager

Alarm Manager je nástroj platformy Android, který umožňuje plánovat a dle nastaveného času spouštět operace nezávisle na životním cyklu aplikace. Pokud je ale jisté, že operace proběhne v rámci běhu aplikace, měla by být namísto Alarm Manageru použita třída `Handler`, jelikož poskytuje operačnímu systém lepší kontrolu nad přístupem k systémovým zdrojům [And07][And02].

Při plánování alarmu za použití Alarm Manageru je zapotřebí provést čtyři základní kroky [And07].

1. **Volba typu alarmu.** Programátor může volit ze dvou základních typů alarmu. Prvním typem je alarm, jehož čas spuštění se počítá od posledního zavedení operačního systému. Další možností je alarm, který je závislý na aktuálním systémovém čase (spustí se v zadaný čas). Oba zmíněné typy mají ještě tzv. `WAKE_UP` verze, které zajistí probuzení zařízení (resp. procesoru) přesně v čas spuštění alarmu.
2. **Nastavení času spuštění.** Pokud je zadaný čas v minulosti, spustí se alarm okamžitě.
3. **Nastavení intervalu opakování.** V případě, že se má alarm v určitých intervalech opakovat.
4. **Vytvoření `PendingIntentu`.** Posledním krokem je vytvoření intentu (záměru), který specifikuje jaké akce se mají provést po spuštění alarmu.

Vzhledem k tomu, že používání alarmů s sebou nese i jistá rizika, je dobré dodržovat následující doporučení.

- Pokud je po spuštění alarmu odeslán požadavek na server, měl by být čas spuštění nastavený s nějakou náhodnou časovou složkou, aby se zamezilo přetížení serveru velkým množstvím požadavků ve stejný čas.

- Frekvence opakování by měla být nastavena na minimální možnou hodnotu (tzn. čím delší interval opakování, tím lépe).
- Zařízení by se nemělo probouzet, pokud to není nezbytně nutné.
- Přesný čas spuštění alarmu by měl být použit pouze v nutných případech. V momentě, kdy není alarm nastaven na přesný čas (metodou `setInexactRepeating()`), může operační systém naplánovat více alarmů na stejný čas a šetřit tak spotřebu baterie.

Při správě alarmů je třeba vzít na vědomí, že po vypnutí zařízení se všechny nastavené alarmy ruší. Přenastavení alarmů může být provedeno opět po spuštění zařízení. Postup je většinou takový, že se naimplementuje tzv. `BroadcastReceiver`, který bude nastaven tak, aby byl aktivován při zavedení operačního systému (nutné definovat v manifestu aplikace).

5.2 Google Play Services

Google Play Services poskytuje vývojářům aplikací rozhraní pro přístup k různým službám od společnosti Google. Balíček Google Play Services APK[4] obsahuje jednotlivé služby, které v operačním systému běží jako služby na pozadí [Gde01].

Google Play Services APK je uživatelům dostupný na oficiálním obchodě Google Play Store, z nějž je i automaticky aktualizován. Vývojáři pak z jednotlivých aplikací přistupují ke službám Google s využitím klientské knihovny, jenž je součástí Android SDK[5].

Z nejpoužívanějších služeb Google, které jsou součástí Google Play Services můžeme jmenovat například:

- **Location APIs** - Pro tvorbu aplikací, jenž pracují s lokací uživatele.
- **Google+** - Propojení aplikací se sociální sítí Google+ pro možnosti personalizace a sdílení.
- **Google Play Game Services** - Pro vývojáře herních aplikací.
- **Drive** - Přístup ke cloudovému úložišti Google.

[4]Android Application Package
[5]Software Development Kit

5.3 OpenWeatherMap Weather API

Weather API od společnosti OpenWeatherMap Inc. poskytuje jednoduché rozhraní pro přístup k informacím o počasí po celém světě. Vývojáři mají k dispozici jak verzi zdarma, tak i několik placených verzí, které se liší počtem přístupů, dostupností či nabízenou podporou [Owm01].

Po získání API klíče lze získat informace o aktuálním počasí, předpovědi počasí a také historická data. V neplacené verzi je možné obsloužit 1200 požadavků za minutu, dostupná jsou kompletní data o počasí a historická data lze získat pouze pro předchozí den.

HTTP[6] požadavek má přesně definovanou formu. Místo, pro které se mají získat informace o počasí se dá vyhledat čtyřmi způsoby, a to:

1. dle názvu města:
   `api.openweathermap.org/data/2.5/weather?q=<název_města>`,
2. dle id města:
   `api.openweathermap.org/data/2.5/weather?id=<id_města>`,
3. dle zeměpisné šířky a délky:
   `api.openweathermap.org/data/2.5/weather?lat=<šířka>&long=<délka>`,
4. dle poštovního směrovacího čísla (zip) v dané zemi:
   `api.openweathermap.org/data/2.5/weather?zip=<zip_kód,stát>`.

Dalšími parametry požadavku lze určit například fyzikální jednotku teploty (Kelviny, stupně Celsia, stupně Fahrenheita), nebo formát odpovědi, přičemž je možné vybírat z formátu JSON, XML a HTML.

Odpověď serveru na odeslaný HTTP požadavek obsahuje následující informace:

- informace o místě: id, název města, kód země, zeměpisná šířka a délka,
- časové údaje: aktuální čas, čas východu slunce, čas západu slunce,
- teplota vzduchu: aktuální teplota, interval minimální a maximální teploty,

[6]Hypertext Transfer Protocol

- atmosférický tlak,
- rychlost a směr větru,
- počasí: kód, název, popis, kód ikony počasí.

V aplikaci pak lze pracovat s kódy jednotlivých druhů počasí dle následující tabulky 5.1.

Kód počasí	**Typ počasí**
200	Bouřky
300	Mžení, mrholení
500	Déšt
600	Sníh
700	Stav ovzduší (mlha, kouř, atd.)
800	Oblačnost (jasno, oblačno, atd.)
900	Extrémní počasí (tornádo, hurikán, atd.)

Tabulka 5.1: Kategorie počasí. [Owm01]

Tyto základní kategorie jsou ještě zpřesněny úpravou kódu v řádech desítek a jednotek. Například tabulka pro kategorii oblačnost pak má následující podobu (tab. 5.2).

Kód počasí	**Typ počasí**
800	Jasno
801	Skorojasno
802	Polojasno
803	Oblačno (téměř zataženo)
804	Zataženo

Tabulka 5.2: Zpřesňující kódy počasí. [Owm01]

5.4 Joda time

Joda Time je volně dostupná knihovna (licence Apache 2.0 [Apa01]) která je vyvíjena v rámci Joda project, což je projekt pro tvorbu nízkoúrovňových knihoven pro platformu Java [Jod01].

Knihovna poskytuje velmi dobrou náhradu standardních tříd Javy pro práci s časem. Základní třídy, které nabízí Java SE[7], nejsou dostačujícím nástrojem pro potřeby kalendářových aplikací zejména proto, že postrádají jednoduché metody pro manipulaci s časovými hodnotami.

Hlavní výhody knihovny jsou shrnuty v následujícím výčtu:

- Snadný přístup k časovým údajům (např. metody `getDayOfMonth()`, `getMinuteOfHour()`, atd.).
- Metody pro časové výpočty jako např. `plusDays()`, `minusHours()`.
- Metody pro porovnání časových hodnot (např. `isBeforeNow()`).
- Formátovaný výpis s podporou `java.util.Locale`.
  Např. `dateOfBirth.monthOfYear().getAsText(Locale.ENGLISH)`.
- Podpora osmi kalendářových systémů (Juliánský, Gregoriánský, Islámský, atd.).
- Podpora časových pásem.

[7]Standard Edition

6 Realizace vlastní aplikace

Tato kapitola se věnuje realizaci navržené aplikace (Personal Assistant) pro platformu Android. Aplikace je vytvářena primárně pro použití v mobilních telefonech. Spuštění na tabletu je také možné, ale pro lepší vzhled a uživatelský komfort by bylo vhodné definovat speciální layout pro tento typ zařízení (není součástí této práce).

Postupně bude popsán datový model, navržené uživatelské rozhraní včetně jednotlivých aktivit, potřebná oprávnění a některé další důležité části řešení.

6.1 Datový model

Jak bylo uvedeno v předchozích sekcích, základem prvkem, který poskytuje přístup ke kalendářovým datům, je **Calendar Provider** platformy Android. V aplikaci je Calendar provider použit jak pro ukládání kalendářových událostí, tak pro ukládání úkolů. Je zde využito určité podobnosti a některých společných vlastností obou těchto kalendářových položek.

Pro ostatní potřebná data (např. budíky, projekty nebo položky nákupního seznamu) byl vytvořen vlastní Content Provider, jenž bude popsán v druhé části této sekce.

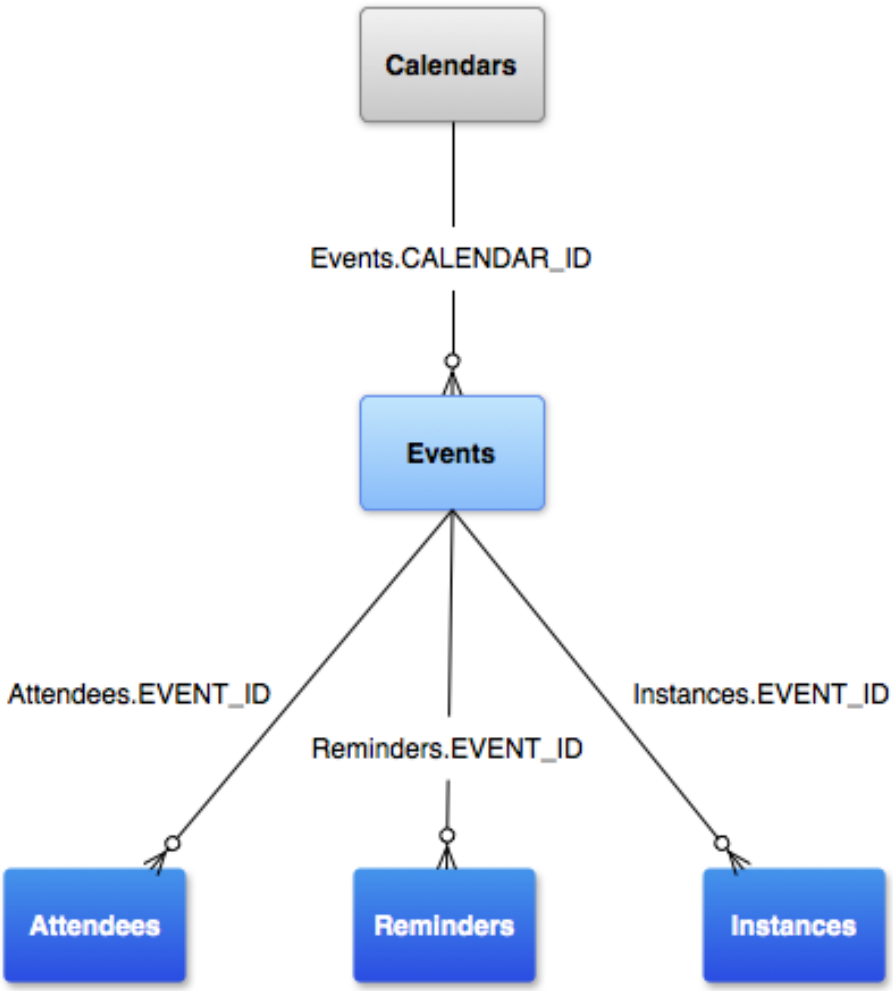
6.1.1 Calendar Provider

Jedním z důvodů proč byl zvolen Calendar Provider i pro úkoly, byla možnost využití vnitřního mechanismu provideru pro opakované události. Do databáze totiž stačí vložit událost s definovaným pravidlem pro opakování a provider už se o jednotlivé instance (opakování) události postará sám [And01][Gra01].

Na obrázku 6.1 je již znázorněn datový model Calendar Provideru. Model obsahuje celkem pět tabulek, z nichž tabulka Attendees (účastníci události) není v realizované aplikaci využita (přidávání různých kontaktů k událostem je možné implementovat v dalších verzích aplikace).

K tabulkám Calendars (kalendáře), Events (události), Attendees a Remin-

ders (připomenutí) může aplikce přistupovat s některými menšími omezeními, která budou později popsána, ale tabulka Instances sloužící pro správu opakovaných událostí je dostupná pouze ke čtení. Calendar Provider totiž spravuje jednotlivé instance událostí podle definovaných pravidel pro opakování svým vnitřním mechanismem.



Obrázek 6.1: Datový model Calendar Provideru [And01]

Na uvedeném datovém modelu můžeme popsat vazby, jenž jsou definovány mezi jednotlivými entitami. Každá událost náleží právě jednomu kalendáři, přičemž kalendář může obsahovat více různých událostí. Další vazby jsou obdobné. K jedné události může být přidáno více účastníků, může mít více různých připomenutí a také (v případě opakované události) více instancí.

Tabulka Calendars

V tabulce Calendars jsou uloženy informace o jednotlivých kalendářích. U kalendářů je třeba zmínit, že i přesto, že se v aplikaci pracuje pouze s lokálními kalendáři (synchronizaci dat na server je možné implementovat v dalších verzích), může kalendáře přidávat, editovat a mazat pouze synchronizační adaptér (Sync Adapter). Při maninupaci s kalendáři v aplikaci je tedy nutné vystupovat jako synchronizační adaptér, což se zajistí nastavením parametrů `Uri` pro přístup k tabulce (viz kód 6.1).

Kód 6.1: Úprava URI pro použití jako synchronizační adaptér

```
Uri calendarsUri = Calendars.CONTENT_URI
    .buildUpon()
    .appendQueryParameter(CALLER_IS_SYNCADAPTER, "true")
    .appendQueryParameter(Calendars.ACCOUNT_NAME,
        PERSONAL_ASSISTANT_ACCOUNT_NAME)
    .appendQueryParameter(Calendars.ACCOUNT_TYPE,
        ACCOUNT_TYPE_LOCAL)
    .build();
```

Samotná tabulka má velké množství sloupců z nichž ty nejdůležitější jsou:

- `NAME` - název kalendáře,
- `CALENDAR_DISPLAY_NAME` - zobrazovaný název kalendáře,
- `VISIBLE` - údaj o tom, zda mají být události daného kalendáře zobrazené,
- `SYNC_EVENTS` - údaj o tom, zda se mají události kalendáře synchronizovat,
- `ACCOUNT_NAME` - název účtu kalendáře,
- `ACCOUNT_TYPE` - typ účtu,
- `CALENDAR_COLOR` - barva kalendáře.

Tabulka Events

Tabulka Events obsahuje informace o kalendářových událostech. Při vkládání nové události musí být vyplněno několik základních údajů, kterými jsou id

kalendáře, čas začátku události, časové pásmo a čas ukončení události (případně délka trvání události společně s pravidlem opakování u opakujících se událostí). V následujícím výčtu jsou vyjmenovány nejdůležitější sloupce této tabulky.

- `CALENDAR_ID` - id kalendáře,
- `TITLE` - název události,
- `EVENT_LOCATION` - místo konání,
- `DESCRIPTION` - popis události,
- `ACCOUNT_NAME` - název účtu kalendáře,
- `DTSTART` - čas začátku události v milisekundách,
- `DTEND` - čas ukončení události v milisekundách,
- `EVENT_TIMEZONE` - časové pásmo,
- `ALL_DAY` - údaj o tom, zda se jedná o celodenní událost,
- `RRULE` - pravidlo pro opakované události (viz 6.8),
- `STATUS` - účast na události (účastní se, neúčastní se, nerozhodnutý),
- `AVAILABILITY` - dostupnost při konání události (volný, zaneprázdněn, nerozhodnutý).

Tabulka Reminders

Ke každé kalendářové události lze přiřadit libovolné množství (resp. množství omezené nastavením kalendáře) připomenutí různého typu. V současné době lze nastavit několik metod připomentí (alert, sms, e-mail), přičemž nativní Android kalendář provádí jen připomenutí typu alert (default). Tabulka je velmi jednoduchá a obsahuje pouze tři sloupce.

- `EVENT_ID` - id kalendářové události,
- `METHOD` - metoda, kterou bude událost připomenuta (alert, sms, e-mail),
- `MINUTES` - počet minut před stanoveným časem události, kdy má být připomenutí aktivováno.

Tabulka Instances

V této tabulce jsou uložena jednotlivá opakování určité kalendářové události. V případě, že se jedná o jednorázovou událost, je vztah mezi záznamy v tabulce Events a Instances 1:1. Pokud je událost opakovaná, připadá na jedno id události N záznamů v tabulce Instances. Z této tabulky jsou nejdůležitější sloupce:

- `EVENT_ID` - id kalendářové události,
- `BEGIN` - začátek konkrétního výskytu (instance) události v milisekundách,
- `END` - konec konkrétního výskytu (instance) události v milisekundách.

6.1.2 PersonalAssistant Provider

Pro další potřeby aplikace, které už nebylo možné pokrýt Content Providery platformy Android, byl vytvořen provider vlastní. Tento provider poskytuje přístup k informacím o projektech, do kterých se organizují úkoly, budících, položkách nákupního seznamu a také o nastavení režimů jednotlivých dnů.

Postup tvorby Content Provideru lze shrnout do třech hlavních bodů, kterými jsou [And03]:

1. **Návrh datového úložiště.**
2. **Konkrétní implementace třídy** `ContentProvider`**.**
3. **Definování řetězce autority, přístupových URI a tabulek.**

Pro ukládání strukturovaných dat se v rámci platformy Android ve většině případů používá relační databázový systém SQLite [Adi01]. Vytvoření databáze se provádí za pomoci třídy `SQLiteOpenHelper`. Na ukázce kódu 6.2 je vidět vytvoření všech potřebných tabulek v metodě `onCreate()`.

Kód 6.2: Vytvoření databáze

```
@Override
public void onCreate(SQLiteDatabase db) {
    db.execSQL(SQL_CREATE_PROJECT_TABLE);
```

```
        db.execSQL(SQL_CREATE_SHOPPING_TABLE);
        db.execSQL(SQL_CREATE_ALARM_TABLE);
        db.execSQL(SQL_CREATE_SPECIAL_DAYS_TABLE);
}
```

Tabulka Projects

Všechny tabulky modelu mají implicitní primární klíč `BaseColumns._ID`. Dále jsou v tabulce Project tyto sloupce:

- `CALENDAR_ID` - id nadřazeného kalendáře,
- `NAME` - název projektu,
- `DESCRIPTION` - popis projektu,
- `PRIORITY` - priorita.

Tabulka Shopping_items

Tabulka pro uložení položek nákupního seznamu je velmi jednoduchá a obsahuje kromě primárního klíče jen tyto dva sloupce:

- `NAME` - název položky,
- `COMPLETED` - údaj o tom, zda už je položka nakoupená.

Do dalších verzí se nabízí např. přidání sloupce s informací o obchodě, ve kterém je možné položku koupit. Nákupní seznam by se pak mohl filtrovat dle různých obchodů.

Tabulka Alarms

Tabulka Alarms uchovává údaje o uložených budících. Kromě typických informací o budících jako je čas nebo dny opakování, jsou zde uloženy i informace o požadovém počasí, pokud má být budík vyhodnocen dle zjištěného počasí v čas jeho aktivace.

- `NAME` - název budíku,
- `SOUND` - melodie, která se má přehrát,
- `HOURS` - čas budíku - hodiny,
- `MINUTES` - čas budíku - minuty,
- `DAYS_OF_WEEK` - celočíselná reprezentace vybraných dnů v týdnu (jeden bit pro každý den v týdnu),
- `WEATHER_CODE` - kód počasí (viz tab. 5.1),
- `TEMPERATURE` - hraniční hodnota teploty vzduchu,
- `TEMP_GREATER` - údaj o tom, zda má být zjištěná hodnota teploty vyšší nebo nižší než hraniční hodnota,
- `ENABLED` - údaj o tom, zda je budík aktivní.

Tabulka Special_days

Poslední tabulkou je tabulka pro uložení nastavení režimů jednotlivých dnů, která má tyto tři sloupce:

- `START` - čas počátečního dne v milisekundách,
- `END` - čas konečného dne v milisekundách,
- `MODE` - zvolený režim dne.

Po návrhu dátabáze je třeba implementovat `ContentProvider`, resp. jeho abstraktní metody. Jedná se o metody zajišťující CRUD operace (`insert(), query(), update() a delete()`), dále pak metodu `getType()`, která vrací MIME[1] typ dle zadané Uri, a nakonec metodu `onCreate()`, jenž inicializuje provider.

Na ukázce kódu 6.3 je vidět, jak se definuje MIME typ pro tabulky (konkrétně pro tabulku Alarms).

Kód 6.3: Definice MIME typu tabulky

```
public static final String CONTENT_TYPE =
    ContentResolver.CURSOR_DIR_BASE_TYPE +
    "/vnd.cz.pavlik.dp." +
    AssistantDatabaseHelper.ALARM_TABLE_NAME;
```

[1]Multipurpose Internet Mail Extensions

Nakonec zbývá už jen definovat řetězec autority a Uri pro přístup ke konkrétním tabulkám provideru. Řetězec autority je základem všech Uri provideru. Zpravidla se pro jeho tvorbu používá název balíku Android projektu (viz kód 6.4). Z `CONTENT_URI` zmíněné v ukázce se pak přidáním názvu tabulky vytvoří Uri pro přístup přímo k dané tabulce.

Kód 6.4: Definice Uri

```
public static final String AUTHORITY =
    "cz.pavlik.dp.assistant.provider";
public static final Uri CONTENT_URI =
    Uri.parse("content://" + AUTHORITY);
```

Veškeré přístupové Uri, MIME typy, názvy tabulek, sloupců, atd. se nachází v tzv. kontrakt třídě `AssistantContract`.

6.2 Datově orientované třídy

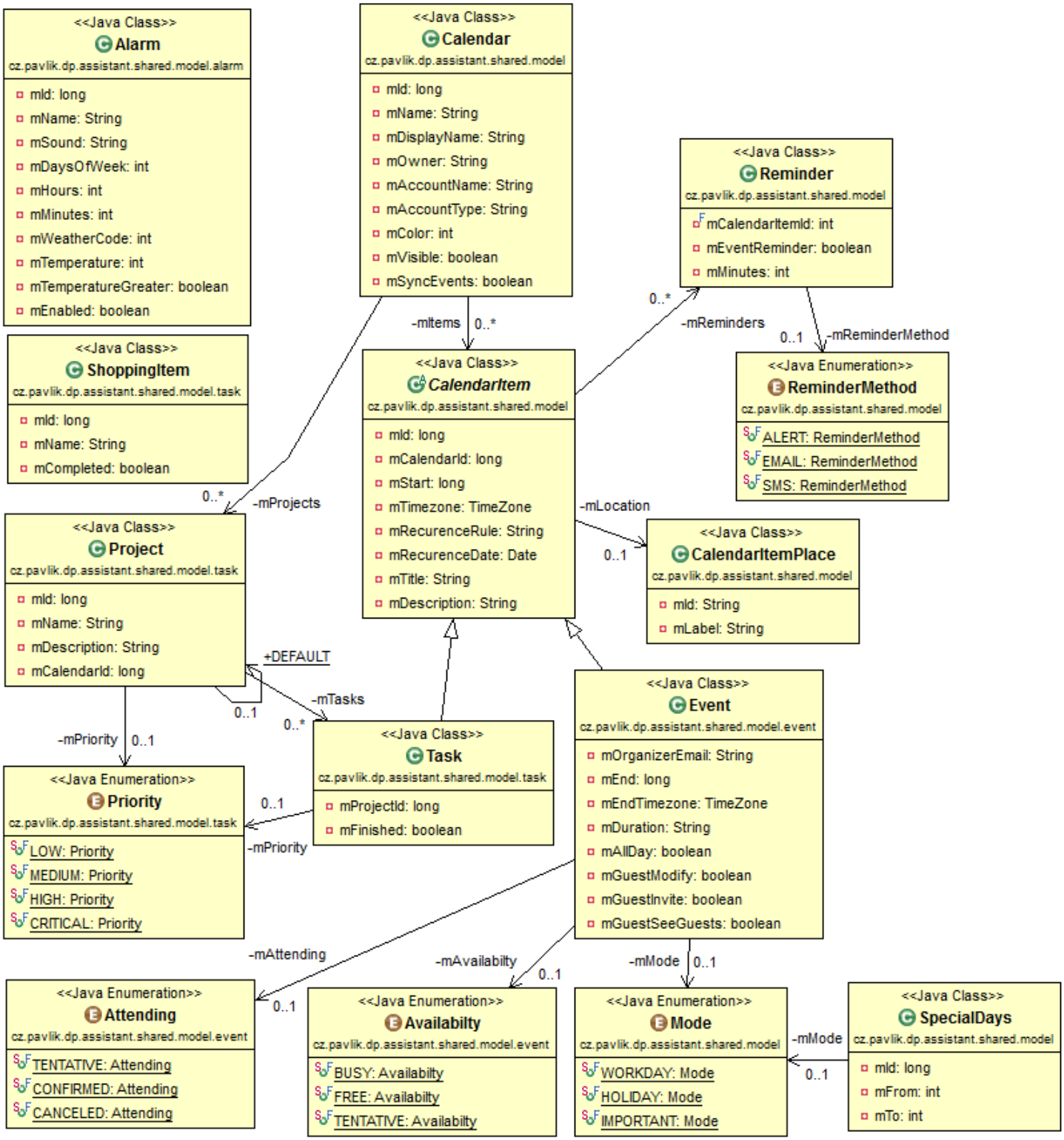
Pro snazší manipulaci s kalendářovými i ostatními daty aplikace při vyhodnocování a přístupu do databáze byly vytvořeny třídy, které do jisté míry napodobují datový model (viz obr. 6.2).

Při zápisu dat z instancí těchto tříd do databáze se pak v aplikaci využívá pomocných metod, které na základě předaných objektů vytvoří `ContentValues`, což jsou objekty, jenž se následně už jen předají `ContentResolver`u, který zajistí jejich vložení do databáze [Adi01][And03]. Na ukázce kódu 6.5 je vidět vytvoření `ContentValues` a naplnění daty pomocí metody `put()`, která má jako parametr název sloupce a vkládanou hodnotu.

Kód 6.5: Tvorba ContentValues

```
ContentValues values = new ContentValues();
values.put(Projects.NAME, project.getName());
values.put(Projects.DESCRIPTION, project.getDescription());
values.put(Projects.PRIORITY, project.getPriority().ordinal());
values.put(Projects.CALENDAR_ID, project.getCalendarId());
```

V případě čtení dat z databáze je `ContentResolver`em vrácen objekt typu `Cursor`, který obsahuje všechny záznamy odpovídající provedenému SQL dotazu. V aplikaci pak podobně jako je tomu u zápisu dat, existují pomocné metody, jenž na základě instance `Cursor`u vytvoří kolekci (typicky `List`) datových objektů.



Obrázek 6.2: Diagram tříd

Tyto datově orientované třídy jsou součástí zvláštního projektu (`personal-assistant-shared`) a jsou klientskou Android aplikací přidány ve formě jar[2] knihovny. Tento způsob distribuce byl zvolen proto, že by mohla být tato knihovna v budoucnosti využita například při tvorbě webové aplikace ve webových technologiích založených na jazyce Java.

6.3 Nutná oprávnění

K tomu, aby mohla aplikace přistupovat k různým zdrojům a informacím v zařízení, musí ve svém manifestu definovat tzv. oprávnění (permission). Uživatel je pak při instalaci aplikace informován o tom, jaké informace aplikace vyžaduje, a v případě, že s těmito požadavky nesouhlasí, může instalaci zrušit [And10][Lac01].

Kód 6.6: Deklarace oprávnění v manifestu aplikace

```
<uses-permission
  android:name="android.permission.READ_CALENDAR" />
<uses-permission
  android:name="android.permission.WRITE_CALENDAR" />
<uses-permission
  android:name="android.permission.INTERNET" />
<uses-permission
  android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission
  android:name="android.permission.ACCESS_FINE_LOCATION" />
<uses-permission
  android:name="android.permission.ACCESS_COARSE_LOCATION" />
<uses-permission
  android:name="android.permission.VIBRATE"  />
<uses-permission
  android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission
  android:name="android.permission.WAKE_LOCK" />
<uses-permission
  android:name="android.permission.SEND_SMS" />
```

První dvě oprávnění se týkají čtení a zápisu kálendářových dat s využitím Calendar Provideru (viz 6.1.1). Přístup k Internetu je vyžadován např. při

[2] Java Archive

získávání informací o počasí. S tím souvisí i oprávnění k získání informací o stavu připojení k Internetu (`ACCESS_NETWORK_STATE`). Pokud aplikace zjistí, že zařízení nemá přístup k Internetu, neodesílá např. požadavky na data o počasí.

Data o poloze zařízení se dají získat více způsoby. Pro přesnější metodu (GPS) je třeba povolit oprávnění `ACCESS_FINE_LOCATION` a pro data s menší přesností `ACCESS_COARSE_LOCATION`.

Android vyžaduje oprávnění také v momentě, kdy chce aplikace používat vibrace (`VIBRATE`). Ty jsou v aplikaci využity společně se zvukovým upozorněním při notifikacích, které vznikají na základě periodického vyhodnocování.

Tři poslední oprávnění souvisí především s budíky a službami. Jak bylo uvedeno v sekci 5.1.3, všechny naplánované alarmy se po vypnutí zařízení ruší. Pro jejich opětovné naplánování se používá tzv. `BroadcastReceiver`, který může být díky oprávnění `RECEIVE_BOOT_COMPLETED` spuštěn ihned po zavedení operačního systému.

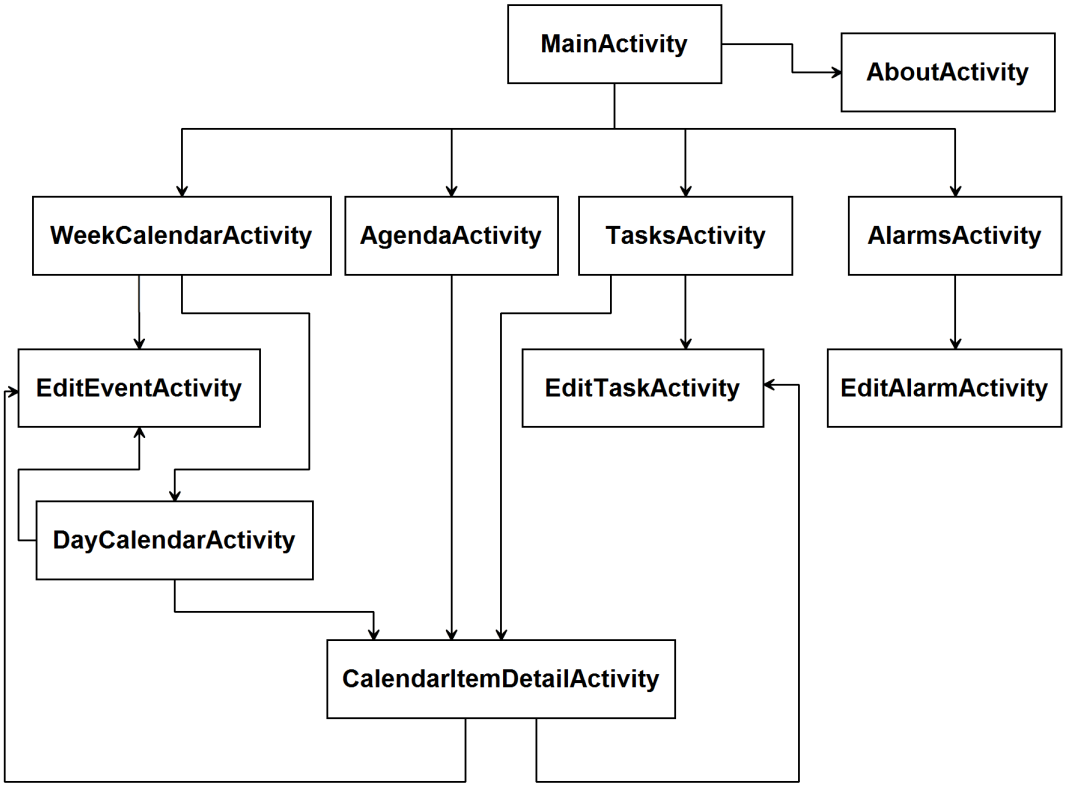
Oprávnění `WAKE_LOCK` je vyžadováno kvůli nutnosti probudit zařízení v momentě, kdy je aktivován budík. V případě, že se jedná dle nastavení režimu dne o důležitý den a uživatel na budík nereaguje, je na kontaktní číslo uvedené v nastavení aplikace odeslána SMS informující o této situaci. Proto musí být v manifestu uvedeno oprávnění `SEND_SMS`.

6.4 Uživatelské rozhraní

Aplikace je určena primárně pro mobilní telefony. Spuštění na tabletu je také možné, ale layout není pro zobrazení na větších displejích uzpůsoben. Dohromady je v aplikaci 13 různých aktivit (`android.app.Activity`), ve kterých může uživatel spravovat informace o událostech, úkolech a budících.

Všechny aktivity mají v manifestu nastavenou orientaci na `portrait` (na výšku), aby se zamezilo přetáčení obrazovky, protože zobrazení kalendáře nebo formulářů na vložení událostí v režimu `landscape` ubírá na přehlednosti. Formuláře pro vkládání událostí vyžadují na většině testovaných zařízení vertikální skrolování i při režimu `portrait`.

V následujícím výčtu jsou vyjmenovány konkrétní aktivity aplikace. Na obrázku 6.3 je pak vidět schéma přechodů mezi aktivitami.



Obrázek 6.3: Přechody mezi aktivitami

- `MainActivity` - Spouštěcí aktivita aplikace. Zobrazuje informace o počasí a o počtech událostí a úkolů v následujících dnech.
- `WeekCalendarActivity` - V této aktivitě jsou zobrazeny všechny události pro daný týden.
- `DayCalendarActivity` - Po kliknutí na konkrétní den ve `WeekCalendarActivity` se zobrazí jeho náhled s událostmi pro tento den.
- `CalendarItemDetailActivity` - Zobrazuje detaily jednotlivých událostí nebo úkolů.
- `EditEventActivity` - Aktivita na vkládání nových a editování existujících kalendářových událostí.
- `TasksActivity` - Aktivita pro zobrazení úkolů. Úkoly se zobrazují podle jednotlivých kritérií (priorita, čas, projekt), mezi kterými se dá v této aktivitě pohybovat (za použití `ViewPageru`). Na poslední stránce této aktivity se nachází nákupní seznam.
- `EditTaskActivity` - Aktivita na vkládání nových a editování existujích

úkolů.

- `AgendaActivity` - Zobrazuje události i úkoly seřazené podle času pro příštích sedm dnů.
- `AlarmsActivity` - Umožňuje měnit nastavení budíků a přidávat nové budíky.
- `EditAlarmActivity` - Aktivita na vkládání nových a editaci existujících budíků.
- `AlarmActivity` - Tato aktivita se zobrazí po aktivaci budíku a je doprovázena přehráváním vybraného zvuku budíku.
- `SettingsActivity` - Umožňuje měnit globální nastavení aplikace. Tato aktivita je dostupná ze všech ostatních aktivit.
- `AboutActivity` - Poskytuje souhrnné informace o aplikaci.

6.5 Služby

Velmi důležitou součástí aplikace jsou služby, které se starají např. o periodické vyhodnocování kalendářových událostí, úkolů a budíků nebo také o spuštění budíku v nastavený čas.

V sekci 5.1.2 byl popsán obecný postup při používání služeb, jenž bude nyní ilustrován na službě `AlarmService`, která zajišťuje spouštění budíků. Pro současné potřeby aplikace dostačují tzv. started služby, které se samy ukončí po dokončení své práce [And07][And09]. Na ukázce kódu 6.7 už můžeme vidět implementaci metody `onStartCommand()`.

Kód 6.7: Implementace služby AlarmService

```
@Override
public int onStartCommand(Intent intent, int flags, int startId){
  if(intent != null) {
    Intent alarmIntent = new Intent(getBaseContext(),
        AlarmActivity.class);
    alarmIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
    alarmIntent.putExtras(intent);
    getApplication().startActivity(alarmIntent);

    Alarm alarm = (Alarm) intent.getExtras().get("alarm");
    AlarmManagerHelper.setAlarm(getApplicationContext(), alarm);
  }
```

```
    stopSelf();
    return super.onStartCommand(intent, flags, startId);
}
```

Všimněme si, že má tato služba dva hlavní úkoly. Prvním úkolem je zapnutí aktivity, která má vzbudit uživatele (`AlarmActivity`). Následně je třeba naplánovat další spuštění budíku podle nastavených dnů opakování. Samotné naplánování příštího spuštění už je provedeno pomocí `AlarmManagerHelper`u. Po výpočtu dalšího času spuštění dochází k naplánování služby (resp. `Intent`u, který službu spustí).

Kód 6.8: Plánování služby pro přesný čas spuštění

```
@SuppressLint("NewApi")
private static void setExactAlarm(Context context, long time,
    PendingIntent intent) {

    AlarmManager manager = (AlarmManager) context
        .getSystemService(Context.ALARM_SERVICE);

    if(Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
        manager.setExact(AlarmManager.RTC_WAKEUP, time, intent);
    }
    else {
        manager.set(AlarmManager.RTC_WAKEUP, time, intent);
    }
}
```

Z ukázky kódu 6.8 je vidět, že ve verzi 4.4 (KitKat) došlo ke změně API pro nastavování alarmů. Pro přístroje se starší verzí OS Android musíme použít původní způsob nastavení.

6.6 Získání geolokačních dat

Platforma Android nabízí vlastní lokační API (balík `android.location`), nicméně vývojářům je doporučeno používat Google Location Services API, které je součástí Google Play Services (viz 5.2)[And05].

Tento framework např. automaticky na základě zvolených parametrů o spotřebě energie vybírá způsob získání geolokčních dat, umožňuje plánovat aktualizace polohy, atd. [And05][Gde01].

Přístup k Google Location Services API se provádí vždy přes klienta `GoogleApiClient`. Vytvoření klienta je vidět v následující ukázce (6.9).

Kód 6.9: Vytvoření klienta Google Play Services

```
protected synchronized void buildGoogleApiClient() {
  mGoogleApiClient = new GoogleApiClient.Builder(this)
    .addConnectionCallbacks(this)
    .addOnConnectionFailedListener(this)
    .addApi(LocationServices.API)
    .build();
}
```

Dalším krokem je vytvoření požadavku (`LocationRequest`) na aktualizaci polohy. Zde je nutné specifikovat interval, ve kterém se budou požadavky na aktualizaci polohy opakovat. Důležitým parametrem je také priorita, jenž značí, jestli je v daný moment důležitější přesnost lokace nebo energetická náročnost.

Pro získání informací o aktuálním počasí byla zvolena hodnota `PRIORITY_BALANCED_POWER_ACCURACY`, přičemž výsledná geolokační data mají odchylku přibližně 100 metrů [And05]. Zvolením této priority se vyloučí použití GPS při hledání polohy, což v tomto případě nevadí, protože pro přijetí dat o počasí je nutné připojení k Internetu (Wi-Fi, datové připojení), které poslouží i jako zdroj informací o poloze.

Naopak při monitoringu polohy zařízení za účelem informování uživatele, že se nachází např. v blízkosti místa, se kterým je spojený nějaký úkol, je zvolena priorita `PRIORITY_HIGH_ACCURACY`. Pro tuto funkcionalitu je totiž využito Geofencing API[3] Google Play Services, jenž vyžaduje přesné informace o poloze [And05].

Nakonec stačí už jen implementovat rozhraní `LocationListener` pro příjem aktualizací o poloze a zahájit vyhledávání (viz kód 6.10).

Kód 6.10: Spuštění aktualizací polohy

```
protected void startLocationUpdates() {
  mLastLocationRequestTime = DateTime.now().getMillis();
  LocationServices.FusedLocationApi.requestLocationUpdates(
      mGoogleApiClient, mLocationRequest, this);
}
```

[3]API pro sledování, zda se zařízení nachází v předem definovaných místech

Třetím parametrem metody `requestLocationUpdates()` je zmiňovaná konkrétní implementace `LocationListener`u.

6.7 Informace o počasí

Aplikace pracuje i s informacemi o aktuálním počasí. Pro získání těchto informací je nutné znát aktuální polohu zařízení (viz předchozí sekci). Další podmínkou je dostupnost připojení k Internetu.

Pokud jsou obě podmínky splněny, může aplikace odeslat požadavek na Weather API popsané v sekci 5.3. Operace, jenž využívají připojení k síti by neměly být prováděny v hlavním vlákně aplikace [And06]. Pro operace tohoto typu se v platformě Android používá `AsyncTask`, který provádí výpočetně náročný kód mimo hlavní vlákno [Gra01]. Odeslání požadavku a přijetí odpovědi se odehrává v metodě `doInBackground()`. V ukázce kódu 6.11 je naznačeno odeslání požadavku a přijetí odpovědi s informacemi o počasí.

Kód 6.11: Přijetí dat o aktuálním počasí

```
HttpURLConnection connection = (HttpURLConnection)url
    .openConnection();
connection.addRequestProperty("x-api-key", API_KEY);

BufferedReader reader = new BufferedReader(
    new InputStreamReader(connection.getInputStream()));
StringBuffer json = new StringBuffer(1024);
String tmp;
while((tmp = reader.readLine()) != null) {
  json.append(tmp);
}
reader.close();
```

Pro přijetí odpovědi ve formátu JSON dojde k jejímu parsování, které je provedeno ještě na pozadí. Aplikace poté může využít zpracovaná data v metodě `onPostExecute()`, která již běží v hlavním vlákně.

6.8 Opakované události

Aplikace podporuje přidávání opakujících se kalendářových událostí i úkolů. Je možné volit celkem z pěti možností opakování, kterými jsou denní, týdenní, roční a dvě varianty měsíčního opakování. Měsíční opakování lze nastavit na konkrétní den v měsíci (např. každý desátý den v měsíci) nebo na pořadí dne v týdnu pro daný měsíc (např. každá druhá sobota v měsíci).

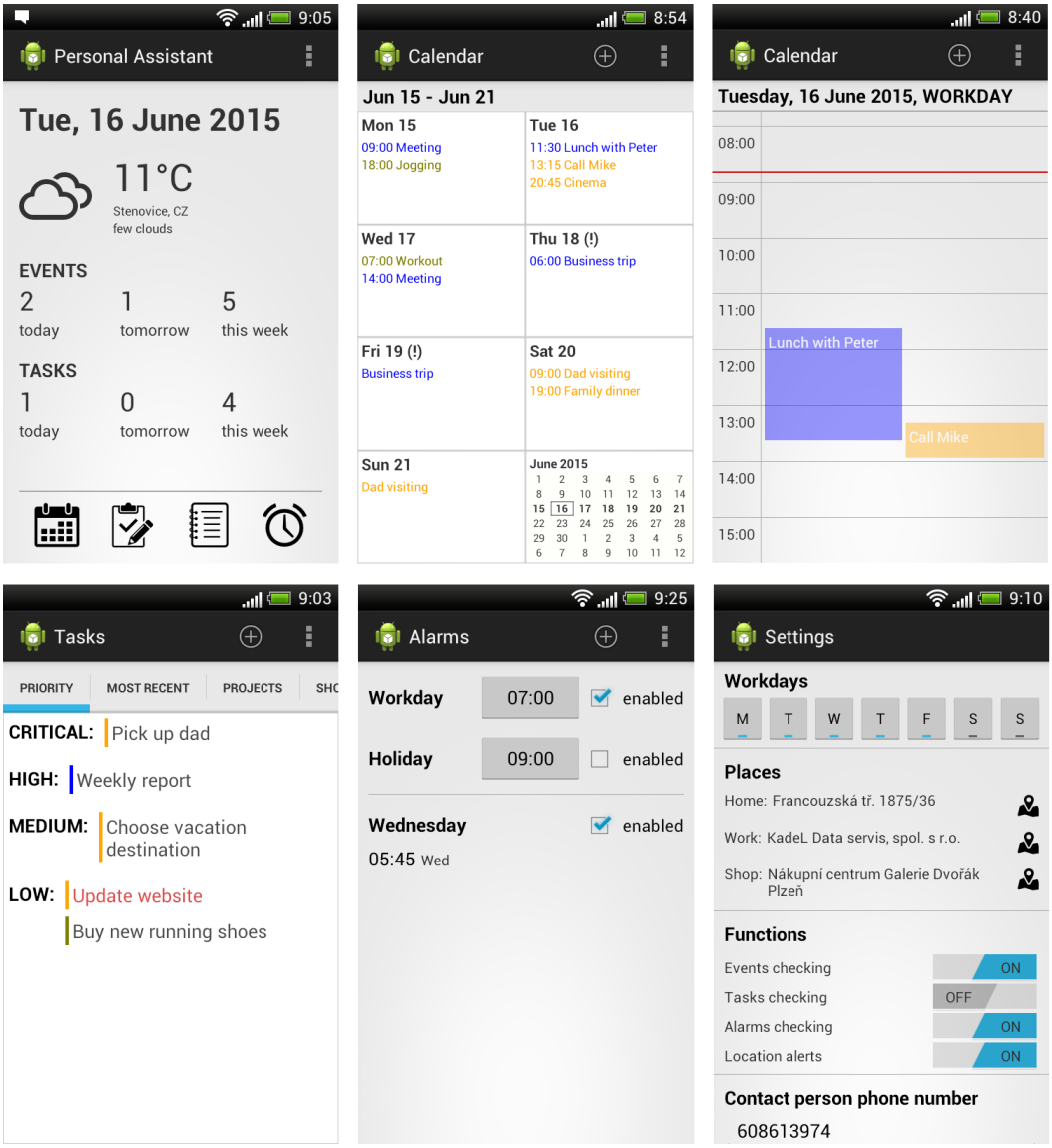
Tato funkcionalita je zajištěna s využitím Calendar Provideru. Jak bylo uvedeno v sekci 6.1.1, tabulka `Events` obsahuje sloupce `RRULE` a `RDATE` pro definici opakujících se událostí. Sloupec `RDATE` se nastaví na čas události a pravidla se pro výše uvedená opakování definují následujícím způsobem [Ora01][And01]:

- Denní opakování: `FREQ=DAILY`.
- Týdenní opakování: `FREQ=WEEKLY;BYDAY=%s`, kde `%s` je dvouznakový kód dne v týdnu (MO, TU, WE, atd.).
- Měsíční opakování (1.varianta): `FREQ=MONTHLY;BYMONTHDAY=%d`, kde `%d` je pořadí dne v měsíci.
- Měsíční opakování (2.varianta): `FREQ=WEEKLY;BYDAY=%d%s`, kde `%d` je pořadí dne v týdnu pro daný měsíc a `%s` dvouznakový kód dne v týdnu.
- Roční opakování: `FREQ=YEARLY`.

Na závěr je důležité zmínit, že kalendářové události se vkládají do tabulky `Events`, ale při čtení dat je třeba provádět SQL dotazy nad tabulkou `Instances`, která obsahuje jednotlivé výskyty události.

6.9 Ukázky aplikace

Na obrázku 6.4 jsou pro ilustraci ukázány některé snímky z aplikace. Na prvním podobrázku je vidět vstupní obrazovka se sumářem nejbližších událostí a úkolů. Kalendářové události lze zobrazovat v týdenním a denním náhledu. Ve druhém řádku obrázků jsou zobrazené úkoly dle nastavených priorit, budíky a možnosti nastavení aplikace včetně spouštění různých funkcí.



Obrázek 6.4: Ukázky aplikace

7 Ověření funkcionality

V této kapitole budou popsány způsoby, kterými byla vytvořená aplikace testována v průběhu vývoje i po jejím dokončení. Budou zde uvedeny také některé scénáře, pomocí kterých lze ověřit funkconalitu jednotlivých částí aplikace.

7.1 Testovaná zařízení

Pro účely testování aplikace byla použita dvě reálná zařízení a dvě emulovaná zařízení (jeden telefon a jeden tablet). Specifikace fyzických zařízení jsou v tabulce 7.1 a údaje o emulovaných zařízeních jsou v tabulce 7.2.

	HTC One V	**Sony Xperia miro**
Displej	3,7”	3,5”
Procesor	1x Scorpion, 1 000 MHz	1x Cortex-A5, 800 MHz
Operační paměť	512 MB	512 MB
Grafický čip	Adreno 205	Adreno 200
Verze OS Android	4.0.3	4.0.4

Tabulka 7.1: Specifikace reálných zařízení

	Google Nexus 5	**Android Generic Tablet**
Displej	4,95”	10,1”
Procesor	Intel Atom (x86)	ARM-v7a
Operační paměť	2048 MB	2048 MB
Grafický čip	host GPU	host GPU
Verze OS Android	5.0.1	4.4.2

Tabulka 7.2: Specifikace emulovaných zařízení

Tablet byl do testování zahrnut kvůli ověření, zda aplikace půjde spustit a používat na tomto typu zařízení, což se potvrdilo. Primárně je vytvořená aplikace určena pro použití na mobilních telefonech.

7.2 Jednotkové testy

Pro ověření správné funkcionality jednotlivých částí celé aplikace byly ve frameworku JUnit vytvořeny jednotkové testy. Pro testování částí aplikace, které jsou specifické pro OS Android, lze použít speciální nástroje Android SDK, jenž jsou nadstavbou JUnit frameworku.

Klasické JUnit testy byly použity např. při testování tříd z projektu `personal-assistant-shared`. Na ukázce kódu 7.1 můžeme vidět testování budíků. Konkrétně se zde testují dny v týdnu, pro které je budík aktivní či nikoliv. Informace je kódována do sedmi bitů (sedm dní v týdnu), přičemž hodnota 1 znamená, že budík je v daný den aktivní.

Kód 7.1: Testování budíků

```
@Test
public void testDaysOfWeek() {
  Alarm alarm = new Alarm(1);

  // žádný den v týdnu není aktivní (0000000)
  alarm.setDaysOfWeek(0);
  for(int i = 0;i < 7;i++) {
    // budík by neměl být aktivní
    Assert.assertFalse(alarm.isActiveInDay(i));
  }

  // všechny dny v týdnu aktivní (1111111)
  alarm.setDaysOfWeek(127);
  for(int i = 0;i < 7;i++) {
    // budík by měl být aktivní každý den
    Assert.assertTrue(alarm.isActiveInDay(i));
  }
}
```

Na ukázce je vidět testování extrémních hodnot (žádný den aktivní a všechny dny aktivní). Pro každý den (0 - pondělí, 6 - neděle) se v testu kontroluje stav budíku voláním metody `isActiveInDay()`.

Testování kódu, který souvisí s OS Android, bude ilustrováno na ověření funkcionality vytvořeného Content Provideru. Využívá se zde možnosti získání mock objektů jako je například `MockContentResolver`, pro přístup k provideru během testování.

V ukázce kódu 7.2 je nejprve v přípravné části testu (metoda `setUp()`) vidět získání mock objektu, který je následně použit při testu vložení záznamu do databáze.

Kód 7.2: Testování Content Provideru

```
@Before
protected void setUp() throws Exception {
  super.setUp();
  mMockResolver = getMockContentResolver();
}

@Test
public void testInsertShoppingItem() {
  Uri uri = mMockResolver.insert(
    AssistantContract.ShoppingItems.CONTENT_URI,
    getContentValues());

  assertEquals(1L, ContentUris.parseId(uri));
}
```

Podobné možnosti jako při testování Content Providerů nabízí testovací projekty Android SDK i pro testování aktivit nebo služeb.

7.3 Manuální testování

Během vývoje i po jeho skončení byla aplikace na výše uvedených zařízeních testována také manuálně. U manuálního testování je důležité uvést, že by ho neměl provádět jen vývojář aplikace, ale také další osoby, které neznají impletační detaily.

Za tímto účelem byla aplikace nainstalována na několik dalších zařízení, kde byla běžně používána. Při testování občas docházelo k pádům aplikace, které se však následně podařilo reprodukovat, zalogovat a adekvátně opravit. Manuálním testováním byla ověřena správna funkcionalita důležitých částí aplikace:

- přidávání, editace a mazání kalendářů, událostí, úkolů, budíků, atd.,
- plánování a spouštění budíků,

- plánování a spouštění opakujících se služeb,
- změna denních režimů,
- spouštění geolokačních služeb,
- automatické odesílání SMS,
- změna globálního nastavení aplikace.

Průběh manuálního testování bude popsán na případech testování budíků a geolokačních služeb. V případě **testu budíků** lze scénář shrnout do těchto kroků:

1. Nastavení budíku na různé dny.
2. Uložení budíku.
3. Posun systémového času na čas, kdy je budík aktivní (přesný čas budíku nebo pozdější).
4. Posun systémového času na čas, kdy budík není aktivní.

Čas je nutné posouvat vždy dopředu, protože jinak může dojít k nekonzistencím při plánování služeb. Po testování tohoto typu (s posouváním času) je pro správnou funkci služeb vhodné zařízení restartovat, jelikož po restartu dojde k inicializaci všech naplánovaných služeb.

Výsledkem testu by mělo být spuštění budíku po provedení třetího kroku. Naopak po provedení čtvrtého kroku by budík neměl být spuštěn. Pokud by byl opakován třetí krok (tzn. nastavení času opět na některý další aktivní čas budíku), měl by být budík opět spuštěn. Podobným způsobem lze testovat plánování periodických služeb. S denní periodou se provádí vyhodnocení úkolů a budíků. Týdenní periodu má vyhodnocování úkolů.

Testování geolokačních služeb bude popsáno na případu přiblížení se k obchodu. Testovací scénář má následující body:

1. Nastavení polohy obchodu v globálním nastavení aplikace.
2. Vložení nějaké nákupní položky do nákupního seznamu.
3. Aktivace geolokačních služeb v globálním nastavení aplikace.
4. Vypnutí aplikace a displeje zařízení.
5. Fyzický přesun k nastavenému místu.

K testování polohy lze samozřejmě využít automatické nástroje, které předávají zařízení údaje o poloze dle nastavení, ale testování v reálných podmínkách může odhalit problémy, jenž simulovat nelze.

Výsledkem testu by měla být notifikace po přiblížení k místu obchodu (okruh 500 metrů). Důležitým krokem testu je vypnutí aplikace a displeje, protože u služeb tohoto typu lze předpokládat, že aplikace nebude vždy spuštěna a telefon může být zároveň i uspán. V případě rychlejšího pohybu (průjezdu automobilem) nemusí být notifikace spuštěna, protože z důvodu šetření baterie nejsou aktualizace lokačních dat tak časté (nejméně po pěti minutách, častěji v případě, že ostatní aplikace také vyžadují aktualizace).

7.4 Zhodnocení

Aplikace se, jak už bylo uvedeno v předchozím textu, věnuje organizaci kalendářových událostí, úkolů a budíků. Pro každou z těchto kategorií nabízí aplikace chytré funkce, které uživateli mají usnadňovat správu času. V následujícím výčtu budou jednotlivé funkce zmíněny.

Kalendářové události:

- kontrola kolizí událostí,
- kontrola budíků při vložení dopolední události,
- nabídka vyhledání hotelů po vložení události vzdálené min. 200 km od domova,
- upozorňování na události, které mají nerozhodnutý stav účasti,
- upozorňování na události, které mají potvrzenou účast, ale nemají nastavené připomenutí.

Úkoly:

- upozorňování na úkoly, které nemají nastavený čas splnění,
- upozorňování na úkoly, jejichž čas splnění již vypršel,
- upozorňování na úkoly v momentě přiblížení k místu úkolu,

- upozorňování na možnost provedení nákupu v momentě přiblížení k nastavenému obchodu.

Budíky:

- upozornění, pokud na následující den není nastaven budík,
- upozornění na kontrolu nastavení, pokud je na následující den nastaveno více budíků,
- upozorňování na nastavení budíků pro důležité dny,
- možnost nastavení počasí pro spuštění budíku,
- automatické odeslání SMS kontaktní osobě, pokud uživatel nereaguje na budík v důležitý den.

Zmíněné funkce byly otestovány na uvedených referenčních zařízeních a také na zařízeních dalších uživatelů, kteří zároveň používáním aplikace napomohli ke komplexnějšímu testování. Implementované funkce odpovídají návrhu (viz kapitolu 4), přičemž možná rozšíření současné aplikace jsou popsána v následující sekci.

7.5 Možná rozšíření

Aplikace v její první verzi nabízí některé zajímavé funkce pro usnadnění organizace času. Existuje zde však potenciál k dalšímu rozvoji, který bude v této sekci nastíněn.

V aplikaci se pracuje s lokálními kalendáři, které si uživatel pro svoje potřeby vytvoří. V budoucnu by bylo vhodné implementovat **synchronizaci dat** na server. Jedním důvodem pro tuto funkcionalitu je záloha dat. Zároveň je také možné vytvořit webovou aplikaci pro náhled na agendu např. z osobního počítače. Synchronizaci dat lze na platformě Android zajistit pomocí synchronizačních adapterů (viz 5.1.1), případně je možné využít některou z knihoven pro serializaci dat a odeslat data na server jiným způsobem. Pro vzájemnou serializaci a deserializaci na straně klienta i serveru může sloužit vytvořená knihovna datově orientovaných tříd `personal-assistant-shared`.

Jak bylo uvedeno v předchozím textu, aplikace je určena pro mobilní telefony. Spuštění na tabletech je také možné, ale layout není těmto zařízením přizpůsoben. Jednou z dalších variant rozšíření by tedy mohla být **podpora zařízení s větší obrazovkou**. Toho lze dosáhnout navržením a vytvořením nových layoutů (nezávisle na aplikační logice) pro jednotlivé typy zařízení, kterých je na platformě Android velké množství.

Z pohledu dalších funkcí aplikace je zde ještě prostor pro návrh a tvorbu nových podmíněných akcí či zavedení dalších služeb pro periodické vyhodnocování kalendářových dat. Princip podmíněných akcí je velmi jednoduchý a platforma Android nabízí mnoho informačních zdrojů, které mohou být pro tento účel využity. V budoucím vývoji tak například lze využít další senzory (viz 2.1.2) pro tvorbu nových nebo zpřesnění stávajících pravidel.

8 Závěr

Cílem této diplomové práce bylo nejprve prozkoumat a analyzovat současné mobilní aplikace pro organizaci a plánování času a poté na základě provedené analýzy navrhnout a vytvořit aplikaci univerzálního asistenta pro platformu Android.

Do průzkumu byly zahrnuty aplikace pro OS Android a také iOS, přičemž jako kritérium úspěšnosti aplikací byla zvolena prodejnost v oficiálních obchodech a uživatelské hodnocení aplikací. Podrobněji byly analyzovány aplikace, které se dle zvolených kritérií umístily na předních příčkách, a dále také aplikace nabízející nějakou zajímavou funkcionalitu, kterou se odlišují od konkurenčních řešení.

S využitím závěrů plynoucích z provedeného průzkumu byla v další fázi práce navržena mobilní aplikace pro platformu Android. Nejprve bylo definováno, jakým oblastem organizace času se bude aplikace věnovat tak, aby byla dostatečně komplexní. Dále byly popsány konkrétní metody, na jejichž základě byly později implementovány některé chytré funkce pro usnadnění správy uživatelské agendy (např. podmíněné akce nebo periodické vyhodnocování).

Aplikace univerzálního asistenta byla dle návrhu vytvořena pro použití na mobilních telefonech. Ověření funkcionality proběhlo jak na skutečných, tak na emulovaných zařízeních. Úspěšně byly testovány různé scénáře, které se zaměřovaly na správnou funkci použitých algoritmů vyhodnocování i na práci s informacemi pocházejícími ze senzorů zařízení (např. geolokační data).

Po úspěšné realizaci navrhované aplikace byla v práci ještě uvedena možná rozšíření aplikace. Z technického pohledu se nabízí implementace mechanismu pro synchronizaci dat aplikace na server nebo také optimalizace uživatelského rozhraní pro použití na tabletech. Z hlediska funkcionality je zde např. potenciál pro implementaci dalších podmíněných akcí, jejichž základní myšlenka byla popsána při návrhu aplikace.

Seznam obrázků

Seznam tabulek

Přehled použitých zkratek a značení

API	Application Programming Interface. Rozhraní pro vývoj aplikací.
CRUD	Create, Read, Update, Delete. Základní operace nad daty.
FUP	Fair User Policy. Omezení internetových připojení.
GPS	Global Positioning System. Družicový polohový systém.
GTD	Getting Things Done. Metoda pro organizaci času.
HTML	HyperText Markup Language. Jazyk pro tvorbu webových stránek.
HTTP	Hypertext Transfer Protocol. Protokol pro výměnu dokumentů.
IR	InfraRed. Infračervené záření.
JSON	JavaScript Object Notation. Datový formát pro přenos dat.
LED	Light-Emitting Diode. Dioda emitující světlo.
LIFX	LED žárovka s Wi-Fi konektivitou.

MIME	Multipurpose Internet Mail Extensions. Standard pro definici formátu dat.
NFC	Near Field Communication. Technologie pro komunikaci na krátkou vzdálenost.
OS	Operační systém.
QR	Quick Response code. Kód pro automatizovaný sběr dat.
SQL	Structured Query Language. Dotazovací jazyk pro práci s relačními databázemi.
URI	Uniform Resource Identifier. Jednoznačný identifikátor zdroje.
UV	UltraViolet. Ultrafialové záření.
Wi-Fi	Wireless LAN. Standard pro bezdrátovou komunikaci.
XML	Extensible Markup Language. Obecný značkovací jazyk.

Literatura

[Adi01] ADITYA, S. K., KARN, V. K. *Android SQLite Essentials.* Packt Publishing, 2014. ISBN 9781783282951.

[And01] ANDROID DEVELOPERS. *Calendar Provider* [online]. 2015. [cit. 19.2.2015]. Dostupné z: `http://developer.android.com/guide/topics/providers/calendar-provider.html`.

[And02] ANDROID DEVELOPERS. *Communicating with the UI Thread* [online]. 2015. [cit. 22.4.2015]. Dostupné z: `https://developer.android.com/training/multiple-threads/communicate-ui.html`.

[And03] ANDROID DEVELOPERS. *Content Providers* [online]. 2015. [cit. 19.2.2015]. Dostupné z: `http://developer.android.com/guide/topics/providers/content-providers.html`.

[And04] ANDROID DEVELOPERS. *Dashboards* [online]. 2015. [cit. 16.6.2015]. Dostupné z: `http://developer.android.com/about/dashboards/index.html`.

[And05] ANDROID DEVELOPERS. *Location Strategies* [online]. 2015. [cit. 19.1.2015]. Dostupné z: `http://developer.android.com/guide/topics/location/strategies.html`.

[And06] ANDROID DEVELOPERS. *Processes and Threads* [online]. 2015. [cit. 21.4.2015]. Dostupné z: `http://developer.android.com/guide/components/processes-and-threads.html`.

[And07] ANDROID DEVELOPERS. *Scheduling Repeating Alarms* [online]. 2015. [cit. 25.1.2015]. Dostupné z: `https://developer.android.com/training/scheduling/alarms.html`.

[And08] ANDROID DEVELOPERS. *Sensors Overview* [online]. 2015. [cit. 9.5.2015]. Dostupné z: http://developer.android.com/guide/topics/sensors/sensors_overview.html.

[And09] ANDROID DEVELOPERS. *Services* [online]. 2015. [cit. 25.3.2015]. Dostupné z: http://developer.android.com/guide/components/services.html.

[And10] ANDROID DEVELOPERS. *System Permissions* [online]. 2015. [cit. 11.5.2015]. Dostupné z: http://developer.android.com/guide/topics/security/permissions.html.

[Apa01] THE APACHE SOFTWARE FOUNDATION. *Apache License Version 2.0* [online]. January 2004. [cit. 14.6.2015]. Dostupné z: http://www.apache.org/licenses/LICENSE-2.0.

[Apg01] APPGENIX SOFTWARE. *Business Calendar* [online]. 2015. [cit. 11.3.2015]. Dostupné z: http://www.businesscalendar.de/.

[App01] APPLE INC. *Apple iOS: Siri* [online]. 2015. [cit. 2.4.2015]. Dostupné z: https://www.apple.com/ios/siri/.

[App02] APPLE INC. *iTunes Preview: OmniFocus2* [online]. 2015. [cit. 14.1.2015]. Dostupné z: https://itunes.apple.com/us/app/omnifocus-2-for-ipad/id904071710?mt=8/.

[App03] APPLE INC. *iTunes Preview: Timeful* [online]. 2015. [cit. 14.1.2015]. Dostupné z: https://itunes.apple.com/us/app/timeful/id842906460?mt=8/.

[Aut01] AUTOMATEIT. *AutomateIt* [online]. 2015. [cit. 15.3.2015]. Dostupné z: http://automateitapp.com/.

[Dig01] DIGIBITES TECHNOLOGY. *DigiCal* [online]. 2015. [cit. 21.2.2015]. Dostupné z: http://digibites.nl/digical/.

[Gde01] GOOGLE DEVELOPERS. *Google Play Services* [online]. 2015. [cit. 21.2.2015]. Dostupné z: https://developers.google.com/android/guides/overview/.

[Gpl01] GOOGLE PLAY. *aCalendar+ Calendar & Tasks* [online]. 2015. [cit. 11.3.2015]. Dostupné z: https://play.google.com/store/apps/details?id=org.withouthat.acalendarplus/.

[Gpl02] GOOGLE PLAY. *AutomateIt Pro* [online]. 2015. [cit. 15.3.2015]. Dostupné z: https://play.google.com/store/apps/details?id=AutomateItPro.mainPackage/.

[Gpl03] GOOGLE PLAY. *Kalendář DigiCal* [online]. 2015. [cit. 12.3.2015]. Dostupné z: https://play.google.com/store/apps/details?id=com.digibites.calendar/.

[Gpl04] GOOGLE PLAY. *Business Calendar Pro* [online]. 2015. [cit. 11.3.2015]. Dostupné z: https://play.google.com/store/apps/details?id=mikado.bizcalpro/.

[Gra01] GRANT, A. *Beginning Android 4.* Apress, 2012. ISBN 9781430239840.

[Idc01] IDC CORPORATE USA *Smartphone OS Market Share* [online]. 2015. [cit. 16.6.2015]. Dostupné z: http://www.idc.com/prodserv/smartphone-os-market-share.jsp.

[Jod01] JODA.ORG *Joda Time* [online]. 2015. [cit. 2.2.2015]. Dostupné z: http://www.joda.org/joda-time/.

[Lac01] LACKO, L. *Vývoj aplikací pro Android.* Computer Press, 2015. ISBN 9788025143476.

[Mic01] MICROSOFT. *Windows Phone: Cortana* [online]. 2015. [cit. 2.4.2015]. Dostupné z: http://www.windowsphone.com/cs-cz/how-to/wp8/cortana/meet-cortana/.

[Net01] NET APPLICATIONS.COM *Mobile/Tablet Operating System Market Share* [online]. 2015. [cit. 16.6.2015]. Dostupné z: https://www.netmarketshare.com/operating-system-market-share.aspx?qprid=8&qpcustomd=1.

[Omn01] THE OMNI GROUP. *The Omni Group: Apps OmniFocus* [online]. 2015. [cit. 14.2.2015]. Dostupné z: https://www.omnigroup.com/omnifocus/.

[Ope01] OPENSIGNAL. *Mobile Sensors Database* [online]. 2015. [cit. 9.5.2015]. Dostupné z: http://tools.ietf.org/html/rfc5545.

[Ora01] ORACLE. *Internet Calendaring and Scheduling Core Object Specification (iCalendar)* [online]. September 2009. [cit. 18.3.2015]. Dostupné z: http://tools.ietf.org/html/rfc5545.

[Owm01] OPENWEATHERMAP INC. *Weather API* [online]. 2015. [cit. 13.4.2015]. Dostupné z: `http://openweathermap.org/api/`.

[Sch01] SCHADER, M., KORTHAUS, A. *The Unified Modeling Language: Technical Aspects and Applications.* Physica-Verlag, 1998. ISBN 0783790811056.

[Tim01] TIMEFUL INC. *Timeful* [online]. 2015. [cit. 2.1.2015]. Dostupné z: `http://www.timeful.com/`.

[Tap01] TAPIR APPS GMBH. *TapirApps: aCalendar+* [online]. 2015. [cit. 11.3.2015]. Dostupné z: `http://www.tapirapps.de/en/home.htm`

A Přílohy

A.1 Instalační příručka

Instalační balíček vytvořené aplikace (.apk) je dostupný na internetové adrese `http://home.zcu.cz/~pavlik14/app/PersonalAssistant.apk` nebo na přiloženém CD (viz A.3). Jelikož aplikce není umístěna na oficiálním obchodu s aplikacemi Google Play, je nutné v nastavení zařízení povolit instalaci aplikací z neznámých zdrojů. Toto nastavení lze ve většině případů nalézt v: *Nastavení* -> *Zabezpečení* -> v kategorii *Správa zařízení* je třeba zaškrtnout pole *Neznámé zdroje*.

Po stažení instalačního balíčku do zařízení jej stačí jen spustit a aplikace se nainstaluje. Aplikace může být nainstalována na zařízení s verzí OS Android 4.0.3 (Ice Cream Sandwich) a vyšší. Další podmínkou je instalace knihovny Google Play Services, která je zdarma dostupná v oficiálním obchodě Google Play. V případě, že knihovna Google Play Services nebude nainstalována, aplikace vyzve uživatele k jejímu stažení a instalaci (případně aktualizaci při zastaralé verzi).

A.2 Uživatelská příručka

Při prvním spuštění je vhodné aplikaci poskytnout některé základní údaje. Tyto údaje je možné zadávat v nastavení (Settings) aplikace, které je dostupné z menu Action Baru (případně u starších verzí OS Android po stisku pravého hardwarového tlačítka - Recents). V nastavení je možné zároveň spouštět a vypínat funkce (opakované služby). Po spuštění libovolné funkce dojde k jejímu provedení a zároveň naplánování na další spuštění, které bude provedeno automaticky.

Kalendář

Po stistku prvního tlačítka zleva na hlavní obrazovce aplikace (obrazovka, jenž se zobrazí po spuštění aplikace) se zobrazí týdenní kalendářový náhled. Mezi jednotlivými týdny lze přecházet swipováním (přejíždění prstem) do stran na buňce s měsíčním náhledem, která se nachází v pravém dolním rohu obrazovky. V menu této obrazovky se nachází několik položek, kterými jsou:

- **Přidání události. (Add event)** Po stisku tohoto tlačítka (ikona +) se otevře formulář pro vložení kalendářové události, ve kterém je nutné vyplnit minimálně název a čas události.
- **Přidání kalendáře (Add calendar).** Zobrazí jednoduchý dialog pro vložení nového kalendáře.
- **Smazání kalendáře (Remove calendar).** Zobrazí jednoduchý dialog pro smazání kalendáře.
- **Změna režimu dne (Change day mode).** Zobrazí jednoduchý dialog pro změnu režimu dne od zvoleného data na zadaný počet dní.
- **Vynulování režimu dne (Reset day mode).** Zobrazí jednoduchý dialog pro vynulování vložené změny režimu.
- **Nastavení (Settings).** Přechod na nastavení aplikace.

Po klepnutí na popisek dne v týdenním náhledu přejde aplikace na denní náhled, ze kterého lze přejít na detail události (klepnutí na událost). Na obrazovce s detaily události je možné opět z menu aplikace událost editovat nebo smazat. Podobně jako u týdenního náhledu lze přecházet mezi jednotlivými dny přejížděním do stran kdekoliv na časové mřížce, kde se nenacházejí kalendářové události.

Úkoly

Obrazovka s úkoly se zobrazí po stistku druhého tlačítka zleva na hlavní obrazovce. Tato obrazovka obsahuje několik náhledů na úkoly (podle priority, času a projektů) a také jednoduchý nákupní seznam.

Z menu obrazovky lze provádět následující akce:

- **Přidání úkolu. (Add task)** Po stisku tohoto tlačítka (ikona +) se otevře formulář pro vložení kalendářové události, ve kterém je nutné vyplnit minimálně název úkolu.
- **Přidání projektu (Add project).** Zobrazí jednoduchý dialog pro vložení nového projektu na úkoly.
- **Smazání projektu (Remove project).** Zobrazí jednoduchý dialog pro smazání projektu.
- **Nastavení (Settings).** Přechod na nastavení aplikace.

Klepnutím na libovolný úkol lze podobně jako u kalendáře přejít na detaily, kde je možné úkol editovat či smazat. Do nákupního seznamu se přidávají položky vyplněním textového pole v dolní části obrazovky a stisknutím tlačítka Add (přidat). Jednotlivé položky lze po jejich nákupu odškrtávat. Pro vyčištění již nakoupených položek ze seznamu slouží tlačítko v levém dolním rohu obrazovky.

Třetí tlačítko zleva na hlavní obrazovce aplikace slouží pro přechod na obrazovku s **agendou**. Agenda nabízí pohled na nejbližší (příštích 7 dní) události a úkoly, který je řazen podle času.

Tlačítkem v pravém dolním rohu úvodní obrazovky lze přejít na nastavení **budíků**. Zde je možné nastavovat budíky pro pracovní a nepracovní dny a dále také další uživatelem přidané budíky. Budík se přidává opět z menu aplikace (nutné vyplnit název, zvuk a čas budíku). Po přidání budíku lze přejít na jeho editaci klepnutím na jeho popisek. Na obrazovce, kde se budíky editují je možné zároveň budík smazat. Budík se po jeho spuštění vypíná tlačítkem Dismiss.

Aplikace v případě, že jsou v nastavení aktivovány některé funkce, generuje různé notifikace, které upozorňují uživatele na úkoly, události nebo budíky. Po klepnutí na notifikaci lze většinou přejít na detaily nebo náhledy, které jsou s notifikací spojené.

A.3 Obsah CD

Přiložené CD má následující adresářovou strukturu:

- **app**: Instalační balíček vytvořené aplikace (.apk).
- **doc**: Text diplomové práce včetně zdrojových souborů nutných k překladu textu.
- **javadoc**: Vygenerovaná JavaDoc dokumentace.
- **src**: Projekty se zdrojovými kódy aplikace. Tato složka obsahuje i zdrojové kódy automatických testů.
  - `personal-assistant-shared`: Projekt, který obsahuje datově orientované třídy, které je v případě potřeby možné použít v serverové aplikaci. Součástí projektu jsou i testy.
  - `PersonalAssistant`: Projekt se zdrojovými kódy a potřebnými knihovnami mobilní aplikace pro OS Android.
  - `PersonalAssistantTest`: Testovací projekt Android aplikace.