

UNIVERZITA PARDUBICE

Fakulta elektrotechniky a informatiky

Mobilní IS STAG

Lukáš Kupr

Diplomová práce

2014

## **Prohlášení autora**

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 21.8.2014

Lukáš Kupr

## **Poděkování**

Tímto bych chtěl poděkovat panu Ing. Jiřímu Zechmeisterovi za rady a připomínky, kterými mi pomohl při tvorbě této diplomové práce. Dále bych rád poděkoval své rodině a přítelkyni za trpělivost, kterou se mnou měli po dobu zpracování diplomové práce.

## **Anotace**

Diplomová práce má za úkol čtenáři přiblížit problematiku operačních systémů pro mobilní telefony. Podrobněji se pak zabývá operačním systémem Google Android, kde, mimo jiné, popisuje různorodost zařízení, na kterých se Android využívá. Dále se zaměřuje na popis webových služeb, včetně služeb poskytovaných IS/STAG. V závěru je popsána praktická část diplomové práce, kterou byla tvorba aplikace Mobilní IS/STAG komunikující s webovými službami. Aplikace byla vytvořena pro platformu Android.

## **Klíčová slova**

Android, Java, Windows Mobile, iOS, Apple, Microsoft, Webové služby, IS/STAG, UPCE, verze API

## **Title**

Mobile IS/STAG

## **Annotation**

The Thesis is aimed on the readers to approach the issue of operational systems for mobile phones. More specifically, it deals with the Google Android operating, which describes a variety of devices, among other things, which is utilized by Android. This Thesis focuses also on description of the Web services, including services provided by IS/STAG. Conclusion describes the practical part of the thesis, which was the creation of Mobile IS/STAG communicating with web services. The application was created for Android platform.

## **Keywords**

Android, Java, Windows Mobile, iOS, Apple, Microsoft, Web services, IS/STAG, UPCE, API level

# 1 Obsah

<b>1</b>	<b>Obsah .....</b>	<b>8</b>
	<b>Seznam zkratk.....</b>	<b>8</b>
	<b>Seznam obrázků.....</b>	<b>9</b>
	<b>Seznam tabulek .....</b>	<b>9</b>
<b>2</b>	<b>Úvod .....</b>	<b>10</b>
<b>3</b>	<b>Android.....</b>	<b>12</b>
3.1	Historie .....	12
3.2	Verze Android .....	12
3.2.1	Android 1.0 (Září 2008) .....	12
3.2.2	Android 1.1 (Únor 2009).....	13
3.2.3	Android 1.5 – Cupcake (Duben 2009) .....	13
3.2.4	Android 1.6 – Donut (Září 2009) .....	13
3.2.5	Android 2.0 až Android 2.1 – Eclair (Říjen 2009) .....	13
3.2.6	Android 2.2 – Froyo (Květen 2010) .....	13
3.2.7	Android 2.3 až Android 2.3.7 – Gingerbread (Prosinec 2010) .....	13
3.2.8	Android 3.0 až Android 3.2 – Honeycomb (Únor 2011) .....	13
3.2.9	Android 4.0 až Android 4.0.4 – IceCreamSandwich (Říjen 2011) .....	13
3.2.10	Android 4.1 až Android 4.3 – JellyBean (Červen 2012).....	14
3.2.11	Android 4.4 až Android 4.4.4 – KitKat (Září 2013).....	14
3.3	Běhové prostředí (runtime).....	14
3.3.1	Dalvik .....	14
3.3.2	ART (Android Runtime) .....	14
<b>4</b>	<b>Aplikace Android.....</b>	<b>16</b>
4.1	Android Manifest.....	16
4.1.1	Příklad AndroidManifest.xml .....	16
4.2	Activity.....	17
4.2.1	Životní cyklus <i>activity</i> .....	17
4.2.2	Ukládání a obnovení stavu <i>activity</i> .....	19
4.3	Fragmenty .....	20
4.4	Jednotky.....	20
4.5	Layout (rozložení) uživatelského rozhraní .....	21

4.6	Lokalizace.....	22
4.7	Další modifikátory .....	22
4.7.1	Orientace zařízení .....	22
4.7.2	Kratší strana.....	22
4.7.3	API level.....	22
4.8	Oprávnění .....	22
<b>5</b>	<b>Webové služby (WS).....</b>	<b>24</b>
5.1	Webové služby SOAP .....	24
5.1.1	WSDL (Web Services Description Language).....	24
5.1.2	SOAP (Simple Object Access Protocol) .....	24
5.1.3	Způsob komunikace.....	25
5.2	Webové služby REST (Representational State Transfer).....	25
5.2.1	GET .....	25
5.2.2	POST .....	26
5.2.3	DELETE .....	26
5.2.4	PUT.....	26
5.3	SOAP nebo REST?.....	26
5.4	Aplikací využívané webové služby .....	26
5.4.1	Pubtran.....	27
5.4.2	ČSFD.cz.....	27
5.4.3	Tabletenky .....	28
5.4.4	ShopsInTouch.....	29
<b>6</b>	<b>Webové služby nad IS/STAG .....</b>	<b>31</b>
6.1	Poskytované webové služby .....	31
<b>7</b>	<b>Mobilní IS/STAG.....</b>	<b>34</b>
7.1	Minimální verze Android .....	34
7.2	Komunikace s webovými službami.....	35
7.3	Optimalizace získávání dat.....	36
7.4	Uživatelské prostředí aplikace.....	37
7.4.1	Přihlášení do aplikace.....	38
7.4.2	Úvodní obrazovka uživatele .....	39
7.4.3	Rozvrh hodin .....	40
7.4.4	Termíny zkoušek .....	41

7.4.5	Známky .....	42
7.4.6	Veřejné rozvrhy .....	42
7.4.7	Přepnout uživatele a odhlásit .....	42
7.5	Další možná vylepšení .....	42
<b>Literatura .....</b>		<b>44</b>
<b>Příloha A – Ukázka WSDL dokumentu ze serveru IS/STAG UPCE .....</b>		<b>46</b>
<b>Příloha B – SOAP požadavek a odpověď .....</b>		<b>49</b>
<b>Příloha C – Obsah CD přiloženého k diplomové práci .....</b>		<b>50</b>

## Seznam zkratk

API	Application Programming Interface
DPI	Dot Per Inch
FUP	Fair Use Policy
HTML5	HyperText Markup Language verze 5
HTTP	HyperText Transfer Protocol
IS/STAG	Informační Systém Studijní Agendy
NFC	Near Field Communication
RAM	Random-Access Memory
SMS	Short Message Service
URL	Uniform Resource Locator
Wi-Fi	Wireless Fidelity
XLS	Office Open XML
XML	Extensible Markup Language



## Seznam obrázků

Obrázek 1 - Životní cyklus activity (8) .....	18
Obrázek 2 - Uživatelské prostředí aplikace Pubtran .....	27
Obrázek 3 - Uživatelské prostředí aplikace ČSFD.cz .....	28
Obrázek 4 - Uživatelské prostředí aplikace Tabletenky .....	29
Obrázek 5 - Uživatelské prostředí aplikace ShopsInTouch.....	30
Obrázek 6 - Aktuální procentuální zastoupení jednotlivých verzí systému Android v mobilních zařízeních .....	35
Obrázek 7 - Postupné načítání názvů předmětů k jednotlivým známám .....	37
Obrázek 8 - Úvodní obrazovka aplikace Mobilní IS/STAG .....	38
Obrázek 9 - Získání osobního čísla studenta .....	38
Obrázek 10 - Obrazovka přihlášení (bez klávesnice vlevo, s klávesnicí vpravo) .....	39
Obrázek 11 - Rozvrh hodin (klasický vlevo, seznam vpravo) .....	41

## Seznam tabulek

Tabulka 1 - Částečný seznam oprávnění aplikace (12) .....	23
Tabulka 2 - Seznam poskytovaných webových služeb nad Informačním systémem IS/STAG .....	32
Tabulka 3 - Seznam parametrů služby rozvrhy metody getRozvrhByStudent (17).....	33
Tabulka 4 - Význam ikon vyjíždějící nabídky .....	40

## 2 Úvod

Pryč jsou doby, kdy mobilní telefony sloužily výhradně k telefonování a zasílání textových zpráv. Dnes se od telefonů očekává mnohem více. Umožňují nám přijímat e-mailové zprávy, prohlížet internetové stránky, používat různé aplikace a hrát hry s grafikou, která před několika lety nebyla ještě ani na stolních počítačích. Dá se říci, že telefony v posledních letech stále více nahrazují stolní počítače a notebooky.

Prvním velkým průkopníkem operačních systémů, takzvaných „chytrých telefonů“ neboli „smartphonů“, byla firma Symbian Ltd., která vytvořila operační systém s názvem Symbian OS. Symbian OS byl od roku 2008 vlastněn firmou Nokia, která tento systém používala ve svých chytrých telefonech. V roce 2011 Nokia ukončila další vývoj svého systému a začala využívat operační systém společnosti Microsoft Windows Phone 7 a následně Windows Phone 8 (1). Třetím zástupcem na poli operačních systémů pro chytré telefony je iOS od společnosti Apple Inc. Apple vyvíjí tento systém výhradně pro svá zařízení. Poslední velký hráč na poli operačních systémů pro mobilní telefony je Android patřící Google.

V současnosti jsou na trhu mobilních zařízení tři z výše jmenovaných systémů. Jedná se o Windows Phone 8, iOS a Android. V posledním kvartálu roku 2013 byl operační systém Android zastoupen v necelých 78 % prodaných mobilních telefonů, iOS si z koláče ukousl zhruba 18% a Windows Phone 8 3 %. Pro srovnání, v prvním čtvrtletí roku 2009 byl Android součástí 1,6% prodaných mobilních telefonů. První příčku v té době držel Symbian OS s 49 % zastoupením (2).

Android se těší takto velkému zastoupení, protože je výrobcům mobilních telefonů nabízen zdarma, má obrovskou komunitu vývojářů her a aplikací, a také proto, že je možné jej nasadit i na levné, neboli „*low-end*“, mobilní telefony, které nemají příliš výkonný hardware, takže uživatel má možnost i na levném zařízení využívat pokročilých funkcí chytrého telefonu.

Operační systémy určené pro chytré telefony se postupně rozrostly o další příbuzná zařízení. V roce 2010 společnost Apple uvedla na trh svůj první iPad s operačním systémem iOS (3). Jednalo se o tablet, což je prakticky zvětšený mobilní telefon, který uživatelům poskytuje zpravidla větší displej. Postupně se přidali i další výrobci, kteří ovšem ve svých tabletech využívají operační systém Android. Motivací tabletů není nahradit mobilní telefony, ale doplnit je a poskytnout uživateli větší obrazovou plochu pro sledování filmů, čtení elektronických knih, surfování po Internetu, větší zábavu při hraní her a podobně. Dá se říci, že tablety vyplňují pomyslný prostor mezi mobilními telefony a notebooky. Oproti notebookům se liší tím, že zpravidla nemají fyzickou klávesnici a interakce mezi tabletem a uživatelem probíhá prostřednictvím dotykového displeje. Tyto přístroje jsou nabízeny s různou úhlopříčkou displeje od sedmi do dvanácti palců.

Mezi další zařízení, kam se mobilní operační systémy dostaly, patří chytré televize, neboli „smart“ TV. Funkčně se jedná o zařízení velmi podobné tabletu s tím rozdílem, že

úhlopříčky obrazovek nejsou kolem 10 palců, ale jsou omezené pouze technickými možnostmi.

V posledních letech se na trh dostala další kategorie mobilních elektronických zařízení. Jedná se o takzvanou nositelnou elektroniku, neboli wearables. Převážně jsou to přístroje příbuzné klasickým hodinkám, s tím rozdílem, že na místě ciferníku mají displej a ve svých útrobách jim netiká hodinkový mechanismus, nýbrž obsahují výčet komponent obdobný mobilním telefonům. V této kategorii opět svádí boj operační systém Android s konkurenčním iOS.

### 3 Android

Android je nejrozšířenější operační systém pro mobilní elektroniku na celém světě. Je založený na jádru Linux (konkrétně verze 2.6) a primárně se vyvíjí pro zařízení s omezenými zdroji, ať už se jedná o výkon procesoru, velikost dostupné paměti pro ukládání dat, či velikost operační paměti (RAM). Mezi tato zařízení patří všechna výše jmenovaná – mobilní telefony, tablety, televize a nositelná elektronika. Knihovny Androidu jsou psány v programovacích jazycích C a C++. Vývoj aplikací probíhá v programovacím jazyku Java, konkrétně za využití Android SDK (4).

#### 3.1 Historie

Historie Androidu sahá až do roku 2003, kdy byla založena firma Android Inc. Andy Rubinem, Richem Minerem, Nickem Searsem a Chrisem Whitem. Původním plánem bylo vytvoření operačního systému určeného pro digitální kamery.

V roce 2005 byla nepříliš známá firma Android Inc. koupena společností Google za účelem angažování se v oblasti mobilních telefonů. V témže roce začaly práce na zcela novém operačním systému s názvem Android, týmem, v jehož čele byl původní zakladatel Android Inc. Andy Rubin. První zprávy o možném nástupu Google na trh s mobilními telefony prosáklý na veřejnost v roce 2007, kdy získal několik patentů týkajících se mobilních technologií. V listopadu téhož roku se zákulisní informace potvrdily a došlo k veřejnému představení nového operačního systému pro mobilní telefony s názvem Android.

První telefon se na pultech obchodů objevil v říjnu roku 2008, tedy až téměř rok po oficiálním představení, a nesl označení HTC Dream (někde označován jako T-mobile G1). Android použitý v telefonu nesl verzi 1.0.

Už v roce 2010 se stal Android nejpoužívanějším operačním systémem v mobilních zařízeních a, jak již bylo zmíněno výše, v dnešní době zaujímá zhruba 80% trhu. (5)

#### 3.2 Verze Android

Android od prvního vydání až do dnešní doby prošel obrovským vývojem a dá se očekávat, že další vývoj bude pokračovat. Jednotlivé milníky ve vývoji systému zmíním níže.

##### 3.2.1 Android 1.0 (Září 2008)

Jedná se o první verzi, která se objevila v mobilních telefonech. Kromě základních funkcí mobilního telefonu nechyběla podpora Wifi, Bluetooth, surfování po Internetu, prohlížení videí na Youtube, fotografování pomocí integrovaného fotoaparátu a mnoho dalších funkcí. Už od tohoto vydání Androidu byl dostupný Android Market, kde si lidé mohli stahovat aplikace a rozšířit tím funkcionalitu svého telefonu.

### **3.2.2 Android 1.1 (Únor 2009)**

Toto vydání nepřinášelo žádné nové funkcionality, ale opravy chyb a stability.

### **3.2.3 Android 1.5 – Cupcake (Duben 2009)**

Zde se poprvé objevila podpora pro virtuální klávesnice. Další důležitou novinkou byla možnost vytváření takzvaných widgetů, což jsou miniaturní aplikace, které můžete umístit například na domovskou stránku a tím získat rychlý přístup k informacím, o které se zajímáte (počasí, zprávy, kontakty, atd).

Od této verze dále nesou všechna další vydání název podle nějaké sladké pochutiny. V tomto případě se jedná o Cupcake.

### **3.2.4 Android 1.6 – Donut (Září 2009)**

Mezi hlavní novinky patřil převod textu na řeč (text-to-speech), podpora WVGA rozlišení displeje (800 x 480 px) a doplnění dalšími drobnějšími úpravami.

### **3.2.5 Android 2.0 až Android 2.1 – Eclair (Říjen 2009)**

V této verzi uživatelé přivítali možnost přidat více účtů pro synchronizaci kontaktů a e-mailů. Dále přibyla, mimo jiné, podpora Bluetooth verze 2.1, více typů rozlišení displeje, Microsoft Exchange účtu, interní prohlížeč Internetu přinesl podporu HTML5 standardu.

### **3.2.6 Android 2.2 – Froyo (Květen 2010)**

Jedná se o první verzi systému Android s podporou technologie Adobe Flash. Mezi další novinky patří možnost sdílení internetového připojení přes Wifi hotspot nebo USB. Dále tato verze přišla s možností instalace aplikací na paměťovou kartu, čímž uživatelé ušetřili velké množství místa v telefonech, které v té době měly zpravidla interní úložiště velmi omezené.

### **3.2.7 Android 2.3 až Android 2.3.7 – Gingerbread (Prosinec 2010)**

Přinesla vylepšení pro práci s textem v podobě copy&paste (kopírovat a vložit). Dále přibyla podpora NFC technologie, více fotoaparátů (přední a zadní), možnost plateb přes službu Google Wallet.

### **3.2.8 Android 3.0 až Android 3.2 – Honeycomb (Únor 2011)**

Tato verze přinesla zcela přepracovaný design se zaměřením na tablety, tedy zařízení s větším displejem. Na mobilní telefony se toto vydání Androidu nikdy oficiálně nedostalo. Kromě grafických úprav se zde poprvé objevila podpora hardwarové akcelerace grafiky 3D. Příchod této verze rozdělil operační systém Android na dvě větve, kde jedna byla určena pro mobilní telefony a druhá pro tablety.

### **3.2.9 Android 4.0 až Android 4.0.4 – IceCreamSandwich (Říjen 2011)**

Jednalo se o poslední verzi, která oficiálně podporovala technologii Adobe Flash. Zásadní změnou v tomto vydání Androidu bylo sjednocení programátorských rozhraní pro mobilní telefony a tablety. Lépe řečeno – tato verze (a všechny další) byla nasazena na mobilních telefonech i tabletech. Dále přinesla hardwarovou akceleraci uživatelského

rozhraní, natáčení videa ve Full HD rozlišení, odemykání zařízení prostřednictvím detekce obličeje, integrovaný editor fotografií a mnoho dalších úprav.

### **3.2.10 Android 4.1 až Android 4.3 – JellyBean (Červen 2012)**

V tomto vydání bylo zapracováno na zvýšení rychlosti běhu operačního systému. Dále přibyla možnost přidávat widgety na zamknutou obrazovku, více uživatelů na jednom zařízení (pouze tablety), podpora 4K rozlišení<sup>1</sup>.

### **3.2.11 Android 4.4 až Android 4.4.4 – KitKat (Září 2013)**

V této verzi došlo k optimalizaci správy operační paměti na zařízeních s méně než 512 MB RAM. Každé zařízení, které má méně než 512 MB RAM, se musí hlásit jako „*low RAM*“ zařízení a operační systém upraví správu paměti pro lepší běh systému. Minimální velikost operační paměti, potřebná pro provoz, je 340 MB. Mezi další změny patří zákaz přístupu ke statistikám baterie z aplikací třetích stran a možnost tisku přes Wifi.

V době psaní této práce je poslední oficiálně vydanou verzí Android 4.4.4. Každá nově vydaná verze přináší mnoho nových funkcí, grafických změn, oprav chyb, vylepšení výkonu, úsporu baterie a další modifikace. Výše vypsané změny pokrývají pouze ty nejdůležitější v jednotlivých verzích. Od prvního vydání Androidu až dodnes proběhla obrovská evoluce a mobilní telefony, potažmo tablety, díky tomu mnohem lépe doplňují klasické počítače, či notebooky (5).

## **3.3 Běhové prostředí (runtime)**

Android i přes obrovský vývoj od svého prvního uvedení stále není dokonalý operační systém. Často je mu vytýkán fakt, že je pomalý v porovnání s konkurenčním iOS, a to zejména na zařízeních, které se řadí mezi „*low end*“. Co vlastně způsobuje tuto „*pomalost*“? Velký podíl na tom má způsob spouštění aplikací, o které se stará právě běhové prostředí. Od počátku Androidu je toto běhové prostředí zastoupeno Dalvik. Od Androidu verze 4.4 je u telefonů Google Nexus volitelně dostupné nové, experimentální, běhové prostředí s názvem ART, které je ve výchozím režimu vypnuto a přepnout jej lze v nástrojích pro vývojáře. Dá se očekávat, že další majoritní aktualizace systému Android bude obsahovat ART na všech zařízeních a Dalvik úplně zmizí.

### **3.3.1 Dalvik**

Toto běhové prostředí funguje jako JIT (Just-In-Time) kompilátor, což v praxi znamená, že po příkazu ke spuštění aplikace se nejdříve načte binární kód aplikace, který se následně zkompile do strojového kódu a až poté dojde ke spuštění. Je zřejmé, že nutnost kompilace aplikace při jejím spuštění je určitá prodleva, která se negativně projeví na rychlosti jejího spuštění.

### **3.3.2 ART (Android Runtime)**

Jedná se o zcela nové běhové prostředí, které vyvíjí Google přímo pro platformu Android. Hlavní rozdíl oproti Dalvik je v tom, že ART využívá AOT (Ahead-Of-Time)

---

<sup>1</sup> Horizontální rozlišení obrazu je zhruba 4000 pixelů

způsob kompilace. To znamená, že systém předkompiluje kód aplikace ve chvíli, kdy se instaluje do telefonu (či jiného zařízení). Výhoda je v tom, že se aplikace již nekompiluje při spuštění a tudíž je tato operace mnohem rychlejší a šetrnější na baterii. Tento způsob kompilace s sebou přináší i nějaké nevýhody. Instalace aplikací trvá o něco déle, což nás v důsledku nemusí tolik zajímat, protože aplikace neinstalujeme tak často jako je spouštíme. Druhou nevýhodou je větší paměťová náročnost instalované aplikace. Aplikace instalované v běhovém prostředí ART zabírají zhruba o 10 až 20 procent více místa v úložišti, což může být na zařízeních s menší pamětí problém, ale stále je to akceptovatelné vzhledem k výraznému zrychlení běhu telefonu a prodloužení výdrže baterie (6).

## 4 Aplikace Android

Aplikace pro platformu Android jsou vyvíjeny v programovacím jazyku Java a rozložení jednotlivých prvků (tlačítek, textových polí a podobně) v aplikaci se definuje pomocí souborů ve formátu XML.

### 4.1 Android Manifest

Nedílnou součástí každé aplikace je soubor `AndroidManifest.xml`, což je v podstatě konfigurační soubor, pomocí kterého operační systém zjistí několik důležitých informací:

- Verze Androidu, pro kterou je aplikace vyvíjena (tzv. API level),
- minimální verze Android, ve kterém je aplikace schopná provozu,
- obsahuje název aplikace,
- výčet komponent (*activity*), které jsou součástí aplikace,
- název hlavní *activity*, která se spustí při spuštění aplikace
- seznam oprávnění vyžadovaných aplikací (*uses-permission*),
- seznam oprávnění definovaných aplikací (*permission*),
- seznam broadcast receiverů (*receiver*),
- seznam služeb (*service*),
- seznam content providerů (*provider*),
- atd. (7)

#### 4.1.1 Příklad `AndroidManifest.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="cz.kupr.stagupce.stag" >

    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >

        <activity
            android:name=".MainActivity"
            android:label="@string/app_name" >

            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>

        <activity
            android:name=".LoginActivity"
            android:label="@string/title_login_activity" >
        </activity>

        <activity
            android:name=".UserStudentActivity"
            android:label="@string/title_activity_user_main" >
        </activity>

        <activity
            android:name=".SettingsActivity"
            android:label="@string/title_activity_settings" >
```



```
</activity>
</application>

<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_CALENDAR" />
<uses-permission android:name="android.permission.VIBRATE" />

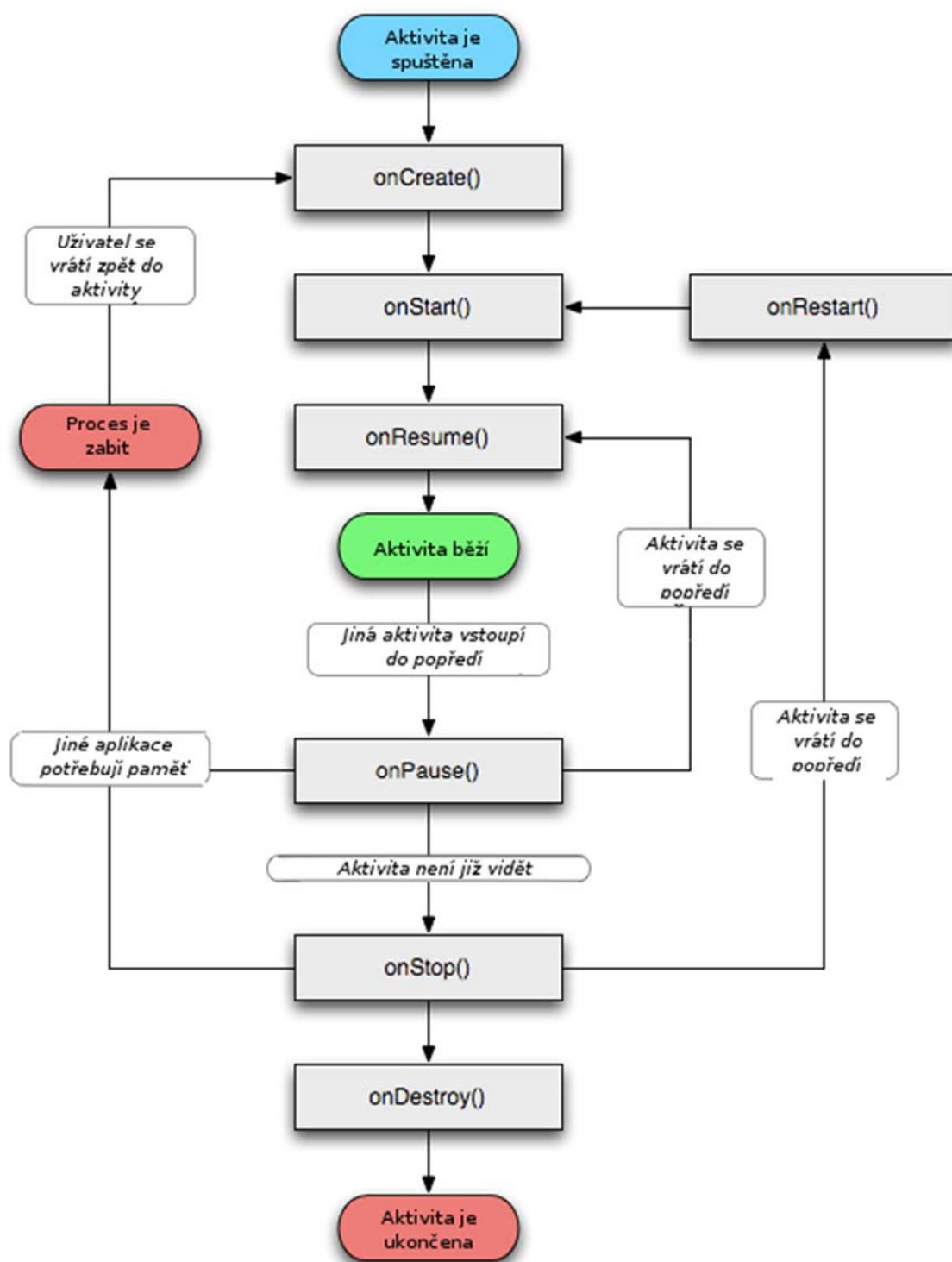
</manifest>
```

## 4.2 Activity

*Activity* je základní stavební kámen každé aplikace v Android, bez níž by nebylo možné ji spustit. Aplikace se může skládat z několika různých *activity*. Je to prezentační část aplikace, která je určena pro interakci s uživatelem. Každá *activity* má přiřazené své *view*, pomocí kterého je určen její vzhled. Z programátorského hlediska se jedná o jakoukoli třídu, která dědí od třídy s názvem *Activity*.

### 4.2.1 Životní cyklus *activity*

*Activity* se v průběhu svého života může nacházet ve třech stavech – může být **na popředí** a očekávat uživatelský vstup, nebo může být **částečně na popředí**, například překrytá nějakým dialogovým oknem, anebo může být **pozastavená na pozadí**. Při změně stavu *activity* dochází k systémovým zpětným voláním (*callback*).



Obrázek 1 - Životní cyklus activity (8)

Na obrázku výše je znázorněn diagram životního cyklu *activity*.

Metoda **onCreate()** je vstupní bod *activity* a volá se ve chvíli, kdy je *activity* vytvořena. Jedná se o místo, kde by mělo dojít k inicializaci *view*. Metoda má parametr *savedInstanceState*, pomocí kterého můžeme obnovit stav *activity* v případě, že byla

systémem ukončena a odstraněna z paměti. Po zavolání této metody je aktivita stále neviditelná a nepřijímá žádné vstupy od uživatele.

**onStart()** je volána ihned po metodě **onCreate()**. Je to vhodné místo pro spuštění například vlákna, které pravidelně stahuje data z internetu a zobrazuje je uživateli.

**onRestart()** je volána, když uživatel znovu zobrazí dříve skrytou *activity* (byla na ní zavolána metoda **onStop()**).

**onResume()** je metoda, která se volá, když se *activity* dostane zcela na povrch, tedy je možná interakce s uživatelem. Zde je vhodné spouštět například animace uživatelského prostředí.

Metoda **onPause()** je volána těsně před tím, než se *activity* dostane do pozadí, například tím, že dojde k překrytí dialogovým oknem. Na tomto místě by mělo docházet k uložení aktuálního stavu *activity*.

Další metodou životního cyklu je **onStop()**, která je volána ve chvíli, kdy přestane být viditelná pro uživatele. Zde bychom měli zrušit všechny prostředky, které jsme inicializovali v metodě **onStart()**

Poslední metodou životního cyklu je **onDestroy()**. *Activity* zde zcela končí a to buď z toho důvodu, že dokončila práci nebo se ji operační systém rozhodl dočasně odstranit z paměti, aby uvolnil místo v paměti pro další *activity*.

#### 4.2.2 Ukládání a obnovení stavu *activity*

Uložení stavu *activity* uvítají uživatelé například ve chvíli, kdy chtějí napsat SMS zprávu a někdo jim psaní přeruší například telefonním hovorem. Operační systém pro uskutečnění hovoru potřebuje určité množství systémových prostředků a tak se může stát, že bude muset uvolnit operační paměť, aby mohl telefonní hovor vůbec proběhnout. V tu chvíli se stane to, že *activity*, ve které psal uživatel zprávu, bude odstraněna z paměti. Pokud programátor SMS klienta na tuto situaci nemyslel, bude pravděpodobně uživatel, který psal SMS, nemile překvapen tím, že se mu pole pro text zprávy zobrazí bez jeho, dosud napsaného, textu.

O uložení stavu se můžeme postarat v metodě **onSaveInstanceState(Bundlestate)**, která je vždy volána dříve, než je *activity* zrušena a odstraněna z paměti. Do objektu typu **Bundle** uložíme jednotlivé hodnoty, které jsou potřebné pro uchování stavu *activity*. Uložení se provede jednoduchým voláním metody **state.putString(Stringkey, Stringvalue)**, kde **key** je název hodnoty, kterou ukládáme a **value** je hodnota.

Obnovení stavu *activity* můžeme provést na dvou místech. První způsob je v metodě **onCreate(Bundle savedInstanceState)**, kde si otestujeme, zda objekt **Bundle** obsahuje nějaká data. Pokud ano, tak to znamená, že *activity* již byla spuštěna a teď dochází k obnovení stavu. Druhým způsobem je přesun logiky pro obnovení stavu *activity* do metody **onRestoreInstanceState(Bundle savedInstanceState)**. Tato metoda je volána

těsně po `onStart()` a je volána pouze v situaci, kdy dochází k obnovení stavu před odstraněním *activity* z paměti. Pokud se *activity* spouští poprvé, metoda `onRestoreInstanceState` se nevolá (8) (9).

### 4.3 Fragmenty

Nástupem tabletů na trh se řešil problém velkých rozdílů mezi velikostí displejů jednotlivých zařízení. Tím vznikla potřeba na větším zařízení zobrazit více informací najednou, oproti menším zařízením. Řešením jsou takzvané fragmenty, což je nová vrstva aplikace, která je mezi *activity* a definicí grafického rozložení prvků (view). Fragmenty přinášejí zjednodušený způsob návrhu složitějších grafických layoutů. Princip spočívá v tom, že na jedné *activity* můžeme zobrazit více fragmentů. Nejčastěji se využívá takzvané master/detail zobrazení, což znamená, že v levé části displeje vidíte seznam (například e-mailů) a v pravé části se zobrazují obsahy jednotlivých e-mailů, na které kliknete. Tím také odpadá nutnost klikat na tlačítko zpět, pokud chcete opět zobrazit seznam e-mailů.

Díky tomu, že se fragmenty objevily až od verze Androidu 3.0, byly by všechny aplikace, které je podporují, na starších verzích nepoužitelné – ani by nešly spustit. Vývojáři Androidu na to mysleli a byli si vědomi, že by fragmenty nikdo nepoužíval, pokud by nevymysleli mechanismus, jak dostat fragmenty do starších verzí Androidu. Výsledkem jejich práce je takzvaná „Support Library“, která přenáší podporu fragmentů až do Androidu verze 1.6 (10).

### 4.4 Jednotky

Každé zařízení má různou velikost obrazovky a různé rozlišení (počet pixelů). Není výjimkou, že dvě zařízení se stejnou velikostí obrazovky mají různé rozlišení – jedno zařízení může disponovat rozlišením displeje 800 x 480 px a druhé 1920 x 1080 px. Tento rozdíl se udává hodnotou DPI (Dot Per Inch – počet pixelů na palec). Pokud na tento fakt nebudeme při vývoji aplikace myslet, může se stát, že některá tlačítka budou na zařízení s vysokým DPI tak miniaturní, že na ně nebude uživatel moci pohodlně kliknout, nebo naopak na zařízení s nízkou hodnotou DPI budou přetékat přes obrazovku.

Aby aplikace vypadala na všech zařízeních přibližně stejně, je doporučené nepoužívat jako jednotky velikosti pixely, ale dp, což je density-independent pixels, neboli pixel, který bere v potaz hustotu rozlišení displeje.

Jako standardní hustota rozlišení displeje je určena hodnota 160 pixelů na velikost jednoho palce. Podle vzorce

$$1dp = \frac{160px}{dpi}$$

lze vypočítat hodnotu dp pro jakýkoliv rozměr displeje a pro jakékoliv rozlišení. Pro vývojáře, ani pro uživatele není přesná hodnota dp nijak důležitá. Důležité je si uvědomit, že používání klasických pixelů jako jednotek velikosti, se u Androidu nedoporučuje.

Pro představu, jak dp fungují, uvedu příklad. Mobilní telefon s úhlopříčkou displeje 5 palců a rozlišením 1080 x 1920 pixelů má hodnotu DPI 380. To znamená, že podle vzorce uvedeného výše nám připadá na jeden pixel zhruba 0,4210526 dp, a to znamená, že na velikost 1 dp se uživateli ve skutečnosti vykreslí na obrazovku 2,375 px (11).

## 4.5 Layout (rozložení) uživatelského rozhraní

Rozložení prvků uživatelského prostředí by mělo probíhat pouze pomocí definic v XML souborech. Samozřejmě, že se dá stejného efektu dosáhnout i za použití programovacího jazyka Java, ale není to doporučováno. Důvodem je oddělení logiky aplikace od jejího vzhledu, díky čemuž je mnohem snazší v budoucnu měnit vzhled aplikace bez nutnosti zásahu do logické části. Další výhodou oddělené logiky je fakt, že layout může vytvářet člověk, který vůbec neprogramuje, ale zabývá se výhradně designem mobilních aplikací.

Jak již bylo zmíněno, operační systém Android je využíván na zařízeních s velmi malým displejem, ale i na přístrojích s obrazovkou nad deset palců. Z pohledu vývojáře je žádoucí, aby jejich aplikace byla co nejlépe využitelná na všech velikostech displeje. Android poskytuje mechanismus, díky kterému můžeme pro každý rozměr obrazovky definovat vlastní layout, nebo jen jeho část.

Při vývoji aplikace jsou všechny XML soubory definující vzhled naší aplikace uloženy ve složce res/layout. Co když ale chceme u některé části aplikace, aby vypadala jinak na mobilním telefonu a jinak na tabletu? Už dříve jsem zmiňoval, že na tabletu můžeme chtít zobrazit seznam e-mailů v levé části a obsah zvoleného e-mailu v pravé části displeje. To samozřejmě u telefonu není žádoucí, neboť by plocha pro obsah e-mailu byla malá. Odpovědí jsou takzvané modifikátory. Vedle složky layout vytvoříme novou složku, kterou pojmenujeme podle přesně daného schématu: layout-w<N>dp, kde N je minimální požadovaná šířka displeje. Tímto modifikátorem zajistíme, že XML šablony umístěné do této složky, se použijí pouze na zařízeních, které mají minimální šířku N dp (co znamená jednotka dp bylo již zmíněno dříve). Šířka displeje se odvíjí od aktuálního natočení displeje. Máte-li telefon v landscape<sup>2</sup> režimu, má displej větší šířku a tudíž může zobrazovat jiný layout, než v režimu portrait<sup>3</sup>. Takovýchto složek s modifikátorem můžeme vytvořit více a mít tak lépe pod kontrolou vzhled naší aplikace na různých velikostech displeje.

---

<sup>2</sup> Poloha telefonu na šířku

<sup>3</sup> Poloha telefonu na výšku

## 4.6 Lokalizace

Všechny texty, které se v aplikaci objevují, je doporučeno vkládat do, tomu určených, XML souborů, stejně jako je tomu u layoutu. Žádný řetězec by se neměl vyskytovat přímo ve zdrojovém Java kódu. Pro tyto texty je určen soubor `res/values/strings.xml`. V případě, že chceme aplikaci nabízet ve více jazycích, přichází na řadu druhý typ modifikátorů, který určuje jazyk aplikace. Jazyk můžeme určit podle země i regionu. Chceme-li například přidat překlady pro britskou angličtinu, vytvoříme nový soubor `res/values/strings-en-rGB.xml`, kde `en` znamená anglický jazyk a `rGB` určuje region Velké Británie, a následně do něj umístíme patřičné překlady. Můžeme také přidat anglické překlady bez přesnějšího určení regionu. Soubor by se pak jmenoval `res/values/strings-en.xml`. Celý systém určení správného překladu funguje tak, že systém Android vyhledává od nejspecifičtější složky až po nejobecnější. V tomto příkladu by pro britské uživatele nejprve prohledával soubor `/res/values/strings-en-rGB.xml`, kdyby tam daný řetězec nenalezl, podíval by se do `/res/values/strings-en.xml` a nakonec by prohledal `/res/values/strings.xml`. Obecně platí, že v souboru `/res/values/strings.xml` by měly být všechny řetězce, které se v aplikaci vyskytují a zároveň by měly být v jazyce, kterému rozumí nejvíce potenciálních uživatelů. Další lokalizace by měly být přidány pomocí dalších souborů s modifikátory.

## 4.7 Další modifikátory

V předchozích kapitolách jsme si popsali dva modifikátory, pomocí kterých můžeme přizpůsobit vzhled aplikace na základě velikosti zařízení anebo provádět lokalizaci do různých jazyků. Modifikátorů je ale mnohem více a některé si ještě popíšeme.

### 4.7.1 Orientace zařízení

Tento modifikátor určuje, že se zdroje umístěné ve složce s tímto modifikátorem použijí na základě orientace zařízení. Orientace může být *landscape*, tedy poloha na šířku, nebo *portrait*, na výšku. Modifikátory pro rozlišení těchto dvou stavů jsou *land* a *port*.

### 4.7.2 Kratší strana

Modifikátor sloužící pro určení zdrojů na základě kratšího rozměru displeje. Zapisujeme jej `sw<N>dp`, kde *N* je délka kratší strany zařízení.

### 4.7.3 API level

Díky tomuto modifikátoru můžeme určovat zdroje na základě toho, jakou verzi Androidu má uživatel ve svém zařízení. Tento modifikátor zapisujeme například *v3*, *v4*, nebo *v7*, atd. (11) (12)

## 4.8 Oprávnění

Systém Android se snaží uživatelům zajistit co nejvyšší míru bezpečnosti. Jedním ze způsobů je fakt, že každá aplikace musí jasně definovat seznam zvláštních oprávnění,

kteřá chce ke svému fungování využívat a uživatel musí při instalaci potvrdit, že s výčtem oprávnění aplikace souhlasí.

Chcete-li například v aplikaci využívat přístup na Internet, musíte do souboru AndroidManifest.xml přidat patřičné oprávnění. V tomto případě se jedná konkrétně o následující řádek kódu:

```
<uses-permission android:name="android.permission.INTERNET" />
```

Pro přidání dalšího oprávnění stačí přidat nový řádek po vzoru výše uvedeného a nahradit slovo „INTERNET“ za jiné oprávnění.

Název oprávnění	Popis oprávnění
ACCESS_NETWORK_STATE	Přístup k informacím o připojení k Internetu
ACCESS_WIFI_STATE	Přístup k informacím o dostupných bezdrátových sítích
BATTERY_STATS	Přístup k informacím ke statistikám baterie
BLUETOOTH	Umožňuje připojení ke spárovanému Bluetooth zařízení
BLUETOOTH_ADMIN	Možnost zobrazit a párovat nová Bluetooth zařízení
CALL_PHONE	Umožňuje aplikaci realizovat telefonní hovory, aniž by uživatel musel otevírat aplikaci Telefon
CAMERA	Přístup k fotoaparátu
CHANGE_NETWORK_STATE	Umožňuje aplikaci vypínat a zapínat připojení k síti
CHANGE_WIFI_STATE	Umožňuje aplikaci vypínat a zapínat připojení k bezdrátové síti
FLASHLIGHT	Přístup k osvětlovací diodě fotoaparátu
MODIFY_AUDIO_SETTINGS	Přístup k nastavení zvuku
NFC	Přístup k vstupně/výstupním operacím přes NFC
READ_CALENDAR	Možnost čtení dat z kalendáře uživatele
READ_CALL_LOG	Čtení historie telefonních hovorů
READ_CONTACTS	Čtení kontaktů
READ_SMS	Čtení SMS zpráv
RECEIVE_SMS	Umožňuje monitorovat příchozí SMS zprávy pro případné zpracování
RECORD_AUDIO	Možnost nahrávání zvuku
SET_TIME	Možnost nastavit systémový čas
SET_WALLPAPER	Možnost nastavit tapetu obrazovky
VIBRATE	Možnost aktivovat vibrace, například reakce na kliknutí
WRITE_CALENDAR	Možnost zápisu do kalendáře
WRITE_CALL_LOG	Možnost zápisu do historie telefonních hovorů
WRITE_CONTACTS	Možnost zápisu do kontaktů
WRITE_EXTERNAL_STORAGE	Možnost zápisu na externí paměťové úložiště (paměťovou kartu)

Tabulka 1 - Částečný seznam oprávnění aplikace (12)

## 5 Webové služby (WS)

Webová služba je software umožňující komunikaci různých aplikací prostřednictvím nějaké sítě, nejčastěji Internetu. Díky tomu si mohou aplikace pracující fyzicky vzdálené od sebe snadno vyměňovat informace a na základě těchto informací kooperovat.

Výhodou webových služeb je nezávislost na platformě systémů, mezi kterými probíhá komunikace. To znamená, že mezi sebou mohou komunikovat například aplikace psané v jazyce Java s aplikacemi psané v .NET frameworku od Microsoftu.

### 5.1 Webové služby SOAP

Tento typ webových služeb je procedurálně orientovaný. To znamená, že využívají model požadavek – odpověď. Pro komunikaci mezi dvěma systémy se nejčastěji využívá komunikační protokol HTTP, a to především z toho důvodu, že má podporu ve většině aplikací. Další výhodou je fakt, že HTTP patří mezi běžně používané protokoly a tudíž je ve firemní infrastruktuře povolen port, který protokol HTTP využívá – 80.

Webové služby splňují 2 základní podmínky:

- Poskytují klientům způsob, jak mají s danou WS komunikovat – jaké mají zasílat dotazy – WSDL.
- WS poskytují klientům (aplikacím) rozhraní, pomocí kterého klienti získávají data – SOAP zprávy.
- Třetí bod není nutný, ale většinou je dodržovaný. Jedná se o to, že WS jsou publikovány, aby mohly být následně vyhledávány potenciálními klienty. Publikování může být v rámci lokální sítě, či globálně v rámci Internetu. Pro publikování se využívá specifikace UDDI (Universal Description, Discovery, and Integration).

#### 5.1.1 WSDL (Web Services Description Language)

Je způsob, jakým server poskytuje informace o tom, co daná služba nabízí a jakým způsobem to získat, neboli klientovi říká, jaký formát odpovědi může od webové služby očekávat a co má obsahovat dotaz, který klient zasílá na adresu serveru webové služby. Naopak, nespecifikuje způsob implementace dané webové služby, protože to není pro klienta podstatná informace. Každá webová služba SOAP by měla být popsána pomocí WSDL.

WSDL je zapsán ve standardním XML formátu, což znamená, že je snadno čitelný a je nezávislý na platformě, což je důležité vzhledem k tomu, že i samotné webové služby nejsou závislé na konkrétní platformě.

#### 5.1.2 SOAP (Simple Object Access Protocol)

Simple Object Access Protocol se využívá pro specifikaci zpráv, pomocí kterých probíhá komunikace mezi dvěma systémy. SOAP byl původně navržen pro přenos zpráv



přes Internet, ale i tak je nezávislý na přenosovém protokolu. V současné době je pro přenos zpráv nejčastěji využíván protokol HTTP (případně HTTPS). SOAP zprávy se zasílají ve formátu XML.

### **5.1.3 Způsob komunikace**

První věc, kterou musí klient udělat, je zjistit adresu webové služby. Tu může dostat buď přímo, nebo tak, že ji dohledá v registru služeb UDDI. Na nalezenou adresu odešle požadavek na získání informací o webové službě, neboli WSDL. Po přečtení WSDL souboru je vygenerován požadavek ve formě SOAP zprávy, který je odeslán opět na tu samou adresu. Na serveru webové služby dojde k přečtení zprávy a zpracování požadavků, které zpráva obsahuje. Výsledek zpracování je následně odeslán zpět klientovi, který zprávu zaslal. Zpětná odpověď serveru klientovi je opět zpráva typu SOAP.

## **5.2 Webové služby REST (Representational State Transfer)**

Tento pojem poprvé publikoval Roy Fielding, jeden z autorů HTTP protokolu, v roce 2000 ve své disertační práci. Jedná se o konkurenční technologii webovým službám typu SOAP. Oproti SOAP jsou webové služby REST orientovány datově, nikoli procedurálně.

Každý zdroj ve webové službě je identifikován jedinečnou adresou URL. Přes rozhraní webových služeb REST můžeme na zdrojovém systému provádět až 4 druhy základních operací. Jedná se o operace CRUD. Jednotlivá písmena ve zkratce jsou počáteční písmena slov reprezentující jednotlivé operace. Jedná se o Create, Read, Update a Delete. Tyto operace bývají implementovány pomocí odpovídajících metod HTTP protokolu, viz níže.

Oproti webovým službám typu SOAP zde není přesně určen formát výsledku dat, takže není nutné přenášet XML data, která jsou v některých případech zbytečně datově náročná na přenos po síti, ale samozřejmě se XML výstup také používá. Stejně tak se používá výstup ve formátu JSON, který obsahuje méně nadbytečných dat, než je tomu u XML souboru. Další oblíbený formát výstupu webových služeb je CSV, který redukuje množství přenášených informací ještě více, než JSON. Někdy také může být poskytován výstup přímo do souboru ve formátu XLS, což je klasický formát známý z aplikace MS Excel.

Pro komunikaci pomocí webových služeb se většinou využívá bezstavový protokol HTTP. To znamená, že všechny požadavky jsou mezi sebou jasně odděleny a ani jeden není závislý na tom druhém. Což s sebou přináší nutnost ověřovat přihlášení při každém požadavku na server webových služeb.

### **5.2.1 GET**

Jedná se o metodu, pomocí které získáváme data ze serveru webové služby. V podstatě se jedná o klasický požadavek na URL adresu.

### **5.2.2 POST**

Tato metoda slouží pro odesílání dat na server webové služby a opět se jedná o metodu známou z každodenního používání Internetu. Stejná metoda se používá například k odesílání webových formulářů.

### **5.2.3 DELETE**

Metoda DELETE slouží k rušení dat na serveru webových služeb. Tato metoda již není tak rozšířená, jako v případě GET a POST a mnoho nástrojů pro komunikaci přes http protokol ji nepodporují, takže se velmi často využívá řešení, že se příkaz na smazání dat pošle metodou POST s indikátorem, který říká, že se mají data odstranit.

### **5.2.4 PUT**

Poslední metoda nese název PUT a slouží k úpravě existujících dat. Použití je stejné jako u metody POST s tím rozdílem, že v URL adrese se předává parametr konkrétního zdroje, který má být upraven. Opět je u této metody problém shodný s metodou DELETE – většina nástrojů ji nepodporuje (14).

## **5.3 SOAP nebo REST?**

Budujete aplikaci, ve které chcete poskytovat data pomocí webových služeb a nevíte, kterou variantu zvolit k implementaci? Jedná-li se spíše o volání procedur, pravděpodobně bude praktičtější zvolit webové služby typu SOAP. Naopak, pokud se jedná o získávání dat, jsou vhodnější služby typu REST.

K webovým službám typu REST je pro protistranu snazší vytvořit klienta, což může být také jeden z rozhodujících aspektů při výběru, jaké webové služby zvolit.

Řešením také může být implementace obou typů webových služeb a výběr nechat na případném klientovi, který si tak bude moci zvolit, jakým způsobem bude k webovým službám přistupovat (15) (16).

## **5.4 Aplikací využívané webové služby**

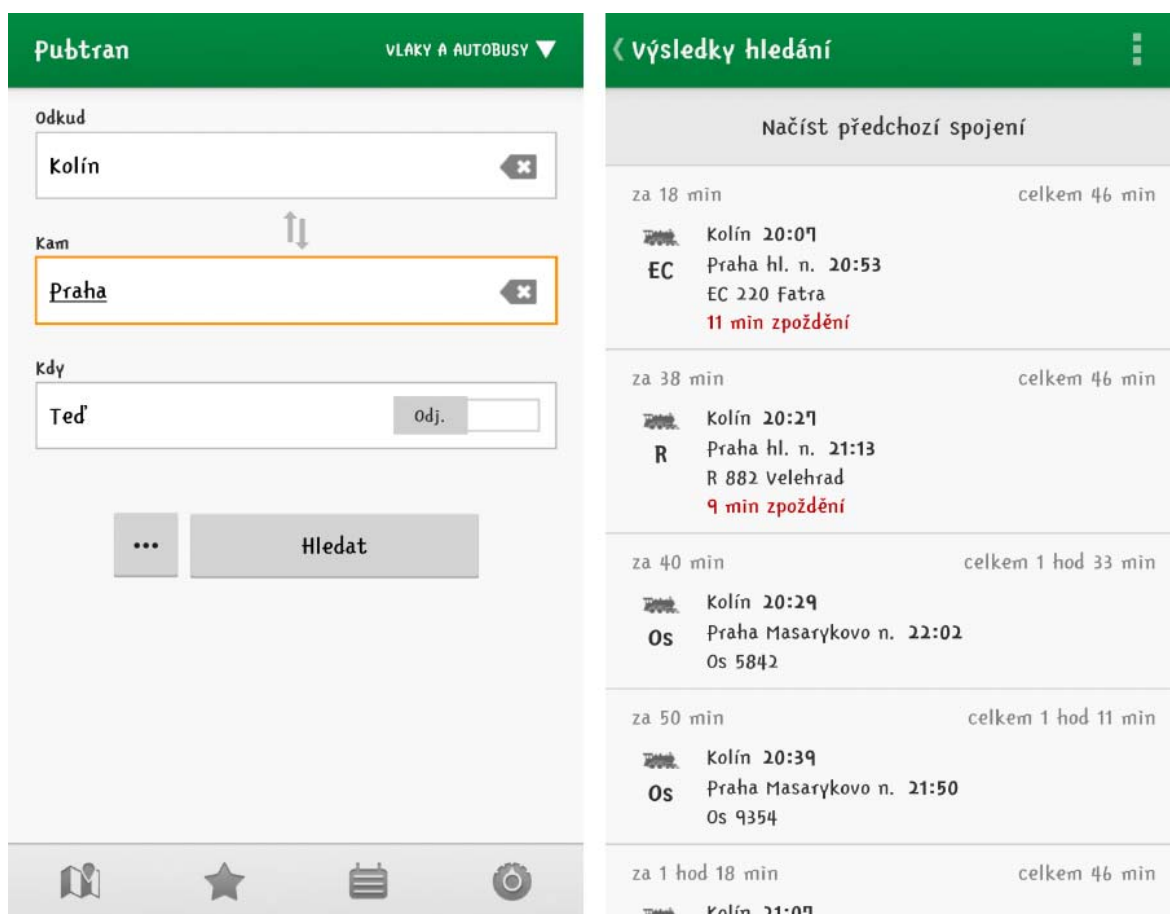
V zadání diplomové práce bylo, mimo jiné, popsat aplikace pro Android využívající webové služby. Na první pohled se to zdálo jako snadný úkol, na Internetu najdu několik aplikací využívající webové služby a popíši jejich účel a základní funkcionalitu. Po chvíli procházení různých stránek na Internetu jsem zjistil, že to úplně snadný úkol nebude. Důvodem je to, že žádná aplikace, respektive její vývojář, neuvádí způsob získávání dat pro svou aplikaci, takže není zcela jasné, zda využívá webové služby, přímý přístup do nějaké SQL databáze, či jiný způsob získávání dat. Tato informace není uveřejňována pravděpodobně z toho důvodu, že z uživatelského hlediska je jedno, jakým způsobem aplikace získává data.

Vzhledem k výše zmíněným faktům si dovoluji trochu zobecnit původní zadání a popíši aplikace, které pro své fungování vyžadují stahování dat z Internetu a s největší pravděpodobností využívají webové služby.

### 5.4.1 Pubtran<sup>4</sup>

Jedná se o jednoduchou, ale užitečnou, aplikaci pro uživatele, kteří cestují veřejnou dopravou. Pokud se někam chystáte, například vlakem, vyplníte město, ze kterého jedete, kam chcete jet a hned vidíte výpis nejbližších spojů, včetně detailních informací o daném spoji. Aplikace vyhledává i tramvaje, autobusy a další prostředky hromadné dopravy.

Na Internetu<sup>5</sup> je zmínka o tom, že jsou data čerpána z iDOS. Pomocí aplikace Network Log se mi podařilo zjistit adresu, se kterou aplikace komunikuje. Na zjištěné adrese jsem zjistil, že se jedná o server webových služeb typu SOAP.



Obrázek 2 - Uživatelské prostředí aplikace Pubtran

### 5.4.2 ČSFD.cz<sup>6</sup>

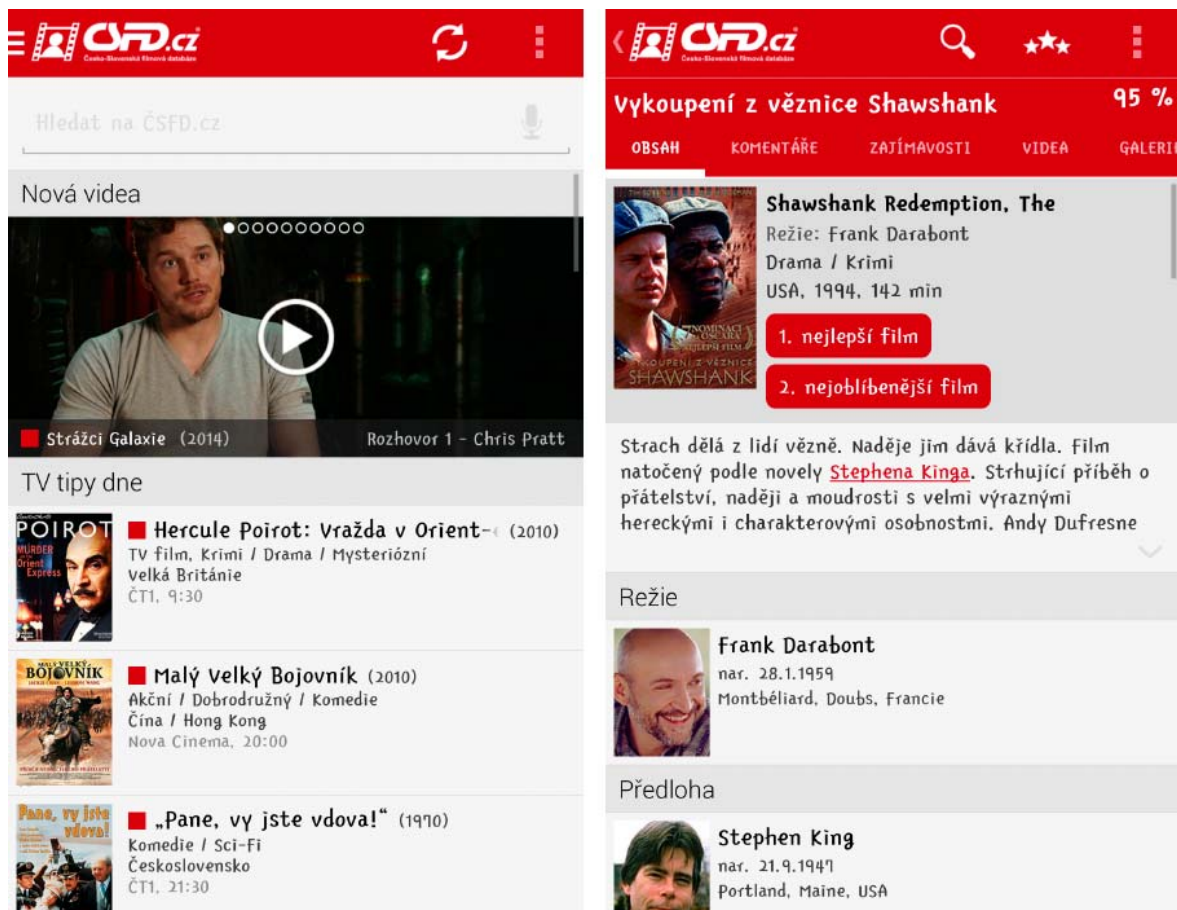
Tato aplikace je zcela jistě dobře známá všem uživatelům Androidu, kteří rádi sledují filmy. Jedná se o propracovanou aplikaci, která poskytuje mobilní alternativu za webové stránky Česko-Slovenské filmové databáze. Umožňuje vyhledávání filmů, zobrazení informací o filmu, seznam herců, zajímavosti, galerie a mnoho dalších. Za zmínku stojí také možnost vyhledat nejbližší kino na základě určení polohy Vaším telefonem.

<sup>4</sup> <https://play.google.com/store/apps/details?id=cz.fhejl.pubtran>

<sup>5</sup> <http://www.svetandroida.cz/pubtran-jizdni-rady-pro-android-201009>

<sup>6</sup> <https://play.google.com/store/apps/details?id=cz.csfd.csfdroid>

U této aplikace jsem pomocí Network Logu zjistil, že komunikuje se třemi různými servery, ale podrobnější informace o těchto serverech jsem nikde nenalezl. Opět se dá předpokládat, že jeden z nich bude server webových služeb. Další z nich pravděpodobně bude obsahovat obrázky.

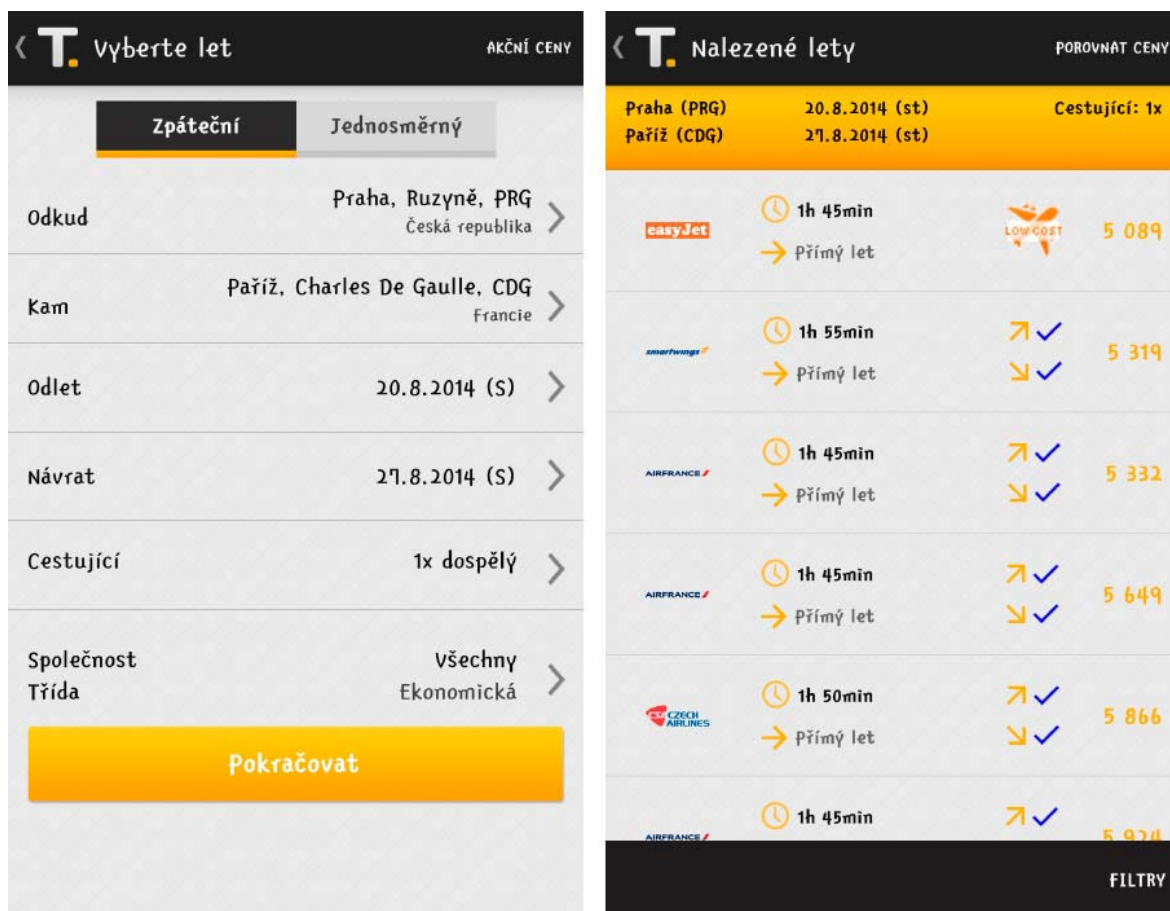


Obrázek 3 - Uživatelské prostředí aplikace ČSFD.cz

### 5.4.3 Tabletenky<sup>7</sup>

Tabletenky je aplikace, která slouží k vyhledávání nejlevnějších letenek na trhu, a to bez jakýchkoli poplatků za využívání. Poskytuje snadný přístup k akčním cenám letenek, porovnání cen jednoho letu od několika prodejců a spoustu dalších funkcí a přitom si zachovává snadnou použitelnost a přehlednost. U této aplikace jsem přímo od vývojáře, společnosti Aaron Group, dostal potvrzenou informaci, že aplikace využívá pro stahování dat k letenkám komunikaci prostřednictvím webových služeb.

<sup>7</sup> <https://play.google.com/store/apps/details?id=cz.aag.tabletenky>

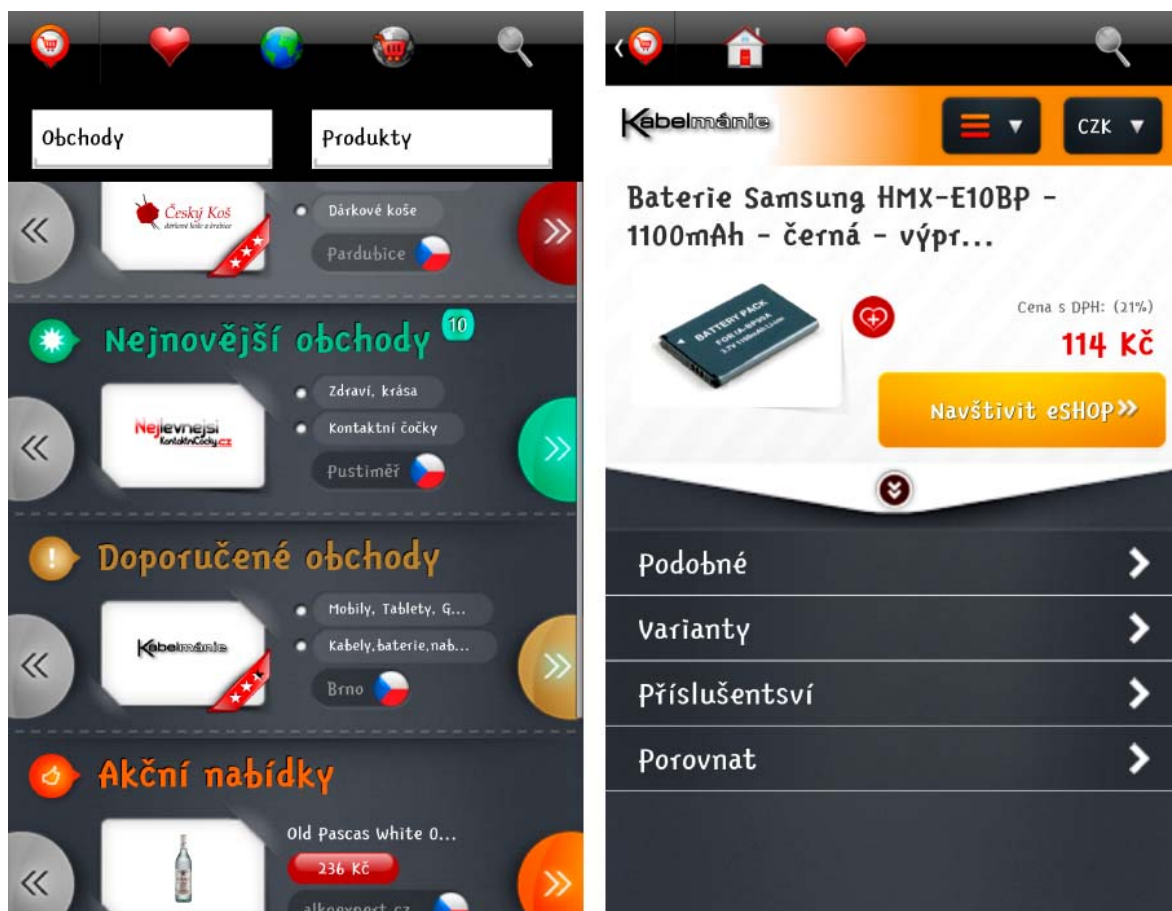


Obrázek 4 - Uživatelské prostředí aplikace Tabletka

#### 5.4.4 ShopsInTouch<sup>8</sup>

Užitečná aplikace, která agreguje několik desítek e-shopů. To znamená, že můžete na jednom místě procházet produkty zhruba dvou stovek elektronických obchodů. ShopsInTouch agreguje celé e-shopy, včetně struktury kategorií, novinek, kontaktů a dalších informací. Snadno si lze vytvořit seznam oblíbených elektronických obchodů pro snazší přístup k produktům, které Vás zajímají. U této aplikace jsem dostal opět potvrzeno od vývojáře, že aplikace využívá server webových služeb, ve kterém jsou uložena data z jednotlivých e-shopů.

<sup>8</sup> <https://play.google.com/store/apps/details?id=com.apigro.shopsintouch>



Obrázek 5 - Uživatelské prostředí aplikace ShopsInTouch

## 6 Webové služby nad IS/STAG

IS/STAG je informační systém, který využívá 16<sup>9</sup> vysokých škol v České republice, včetně Univerzity v Pardubicích. Systém, včetně rozhraní webových služeb, je vyvíjen a udržován Západočeskou univerzitou v Plzni.

Každá škola, využívající IS/STAG má možnost si funkčnost přizpůsobit dle vlastních požadavků. Systém je modulární. Jádru systému obsahuje tyto základní bloky:

- Studijní programy, obory, plány, předměty,
- student,
- přijímací řízení,
- rozvrhy,
- předzápis,
- zkoušky,
- semestrální práce,
- mobility studentů,
- evaluace,
- předpisy služeb,
- absolvent.

Přes webové rozhraní portálu IS/STAG je dostupné velké množství užitečných informací, kterou jsou ovšem dobře čitelné pro uživatele, ale ne pro případné strojové aplikace, které mají za úkol využívat data z IS/STAG. Může se jednat například o webové stránky jednotlivých kateder, kde je žádoucí vypsát seznam předmětů, studentů a podobně. Z tohoto důvodu byly nad IS/STAG vybudovány webové služby, které tyto informace poskytují ve strojově čitelné podobě.

Nad daty IS/STAG byly, po analýze požadavků, implementovány oba základní standardy webových služeb, tedy SOAP i REST. Důvod implementace SOAP byly požadavky na propojení s jinými počítačovými systémy. Na druhé straně byla snaha o co největší jednoduchost vytváření případného klienta, který potřebuje získat nějaká základní data a neprovádět žádné složité operace nad webovými službami, proto došlo i na implementaci REST.

### 6.1 Poskytované webové služby

Seznam dostupný na webovém rozhraní aktuálně poskytovaných webových služeb je automaticky generován serverem, takže není závislý na ruční aktualizaci správcem.

Adresa	Popis	Typ
/ws/services/rest/absolventi	Modul Absolventi	REST
/ws/services/soap/absolventi		SOAP
/ws/services/rest/ciselniky	Číselníky STAGu	REST
/ws/services/soap/ciselniky		SOAP

<sup>9</sup><http://cs.wikipedia.org/wiki/IS/STAG#Historie>



/ws/services/rest/ects	ECTS služby	REST
/ws/services/soap/ects		SOAP
/ws/services/rest/help	Různé pomocné a interní služby	REST
/ws/services/soap/help		SOAP
/ws/services/rest/kalendar	Kalendář, harmonogram ak. roku, atd.	REST
/ws/services/soap/kalendar		SOAP
/ws/services/rest/kvalifikacni prace	Kvalifikační práce (diplomky, bakalářky)	REST
/ws/services/soap/kvalifikacni prace		SOAP
/ws/services/rest/mistnost	Místnosti	REST
/ws/services/soap/mistnost		SOAP
/ws/services/rest/platby	Služby týkající se plateb.	REST
/ws/services/soap/platby		SOAP
/ws/services/rest/predmety	Předměty	REST
/ws/services/soap/predmety		SOAP
/ws/services/rest/prijimacky	Přijímací řízení	REST
/ws/services/soap/prijimacky		SOAP
/ws/services/rest/programy	Studijní programy, obory, segmenty atd.	REST
/ws/services/soap/programy		SOAP
/ws/services/rest/rozvrhy	Rozvrhy	REST
/ws/services/soap/rozvrhy		SOAP
/ws/services/rest/student	Studenti.	REST
/ws/services/soap/student		SOAP
/ws/services/rest/terminy	Termíny zkoušek	REST
/ws/services/soap/terminy		SOAP
/ws/services/rest/ucitel	Učitelé	REST
/ws/services/soap/ucitel		SOAP
/ws/services/rest/users	Uživatelé - Spojení mezi externím uživatelským jménem a identitami v IS/STAG.	REST
/ws/services/soap/users		SOAP
/ws/services/rest/znamky	Známky	REST
/ws/services/soap/znamky		SOAP
/ws/services/soap/ess/bbm	Elektronická spisová služba - BBM SPS	SOAP
/ws/services/rest/podporaVyuky	Podpora výuky	REST
/ws/services/rest/rss	RSS export termínů zkoušek	REST
/ws/services/rest/reports	Služby pro tvorbu reportů (TeX a Jasper).	REST

**Tabulka 2 - Seznam poskytovaných webových služeb nad Informačním systémem IS/STAG**

Každá výše uvedená webová služba nabízí skupinu metod, které vrací odpovídající data. Jednotlivé metody přijímají určité parametry, na základě kterých provádí filtrování dat. Některé parametry jsou povinné, jiné volitelné. Pro názorný příklad uvedu jednu metodu z webové služby „rozvrhy“. Metoda se jmenuje „getRozvrhByStudent“ a jak název napovídá, vrací rozvrh studenta.

Parametr	Typ	Povinný	Popis
stagUser	String	Ne	
osCislo	String	Ano	Osobní číslo studenta
Rok	String	Ne	Akademický rok. Pokud není zadán, použije se aktuální
Semestr	String	Ne	Semestr (LS, ZS)
vsechnyAkce	Boolean	Ne	Je-li false, nebo nevyplněn, vybere klasický rozvrh, jinak zobrazí všechny typy akcí
vsechnyCasyKonani	Boolean	Ne	Je-li false, nebo nevyplněn, zobrazí klasický rozvrh



			(jedna rozvrhová akce = jeden záznam). Jinak zobrazí všechny jednotlivé data konání rozvrhových akcí
<b>Lang</b>	String	Ne	Jazyk výstupu

**Tabulka 3 - Seznam parametrů služby rozvrhy metody getRozvrhByStudent (17)**

## 7 Mobilní IS/STAG

Mobilní IS/STAG je aplikace pro operační systém Android a jedná se o praktickou část této diplomové práce. Aplikace částečně nahrazuje webovou verzi IS/STAG a poskytuje studentům a vyučujícím základní informace o jejich harmonogramu. Důvodem vývoje takovéto aplikace je usnadnění přístupu studentům a vyučujícím k informacím, které potřebují mít denně při ruce. Samozřejmě mohou využít internetový prohlížeč ve svém mobilním telefonu, ale získání potřebných informací je časově mnohem náročnější, uživatelsky velmi nepohodlné, protože webové stránky nejsou přizpůsobeny malým displejům, a hlavně bývá problém s množstvím přenášených dat na klasických webových stránkách. V dnešní době se nejčastěji velikost FUP pro mobilní připojení pohybuje mezi 300 – 500 MB na měsíc. To se může zdát jako dostatečné množství, ale každodenní zjišťování rozvrhu prostřednictvím Internetového prohlížeče si dokáže ukousnout velkou část z tohoto limitu.

Při testování v mobilním prohlížeči Google Chrome sebralo prvotní zobrazení rozvrhu zhruba 3 MB dat. Celý proces zahrnoval načtení úvodní stránky, přihlášení následné zobrazení rozvrhu. Po tom, co prohlížeč uložil obrázky, styly a skripty do dočasné paměti (cache), bylo další načítání „levnější“ z pohledu konzumace dat. Další čtyři kontroly rozvrhu (opět stejný proces, od přihlášení po zobrazení rozvrhu) již spotřebovaly dohromady necelé 3 MB dat. Uděláme-li z toho dlouhodobý průměr, dostaneme se zhruba na 1 MB za jednu kontrolu rozvrhu. To není tak velká hodnota, ale přidáme-li k tomu uživatelskou přívětivost procházení stránek IS/STAG v mobilním telefonu, je zřejmé, že aplikace přizpůsobená mobilním telefonům bude pro uživatele velkým přínosem.

V mobilní aplikaci jsem provedl stejný test – tedy zobrazení rozvrhu hodin. Proces zahrnoval přihlášení, zobrazení úvodní obrazovky a následné vykreslení rozvrhu hodin. Po prvním pokusu se stáhlo necelých 49 kB dat. Poté jsem provedl ještě 4 pokusy a celková konzumace dat se zastavila na 140 kB. Při dalších pokusech se ušetřil nějaký přenos, protože aplikace využívá cache, kam si ukládá například informace o předmětech.

Srovnáme-li výsledky internetového prohlížeče a aplikace postavené nad webovými službami IS/STAG, zjistíme, že spotřeba dat na zjištění rozvrhu hodin klesne zhruba třicetkrát.

### 7.1 Minimální verze Android

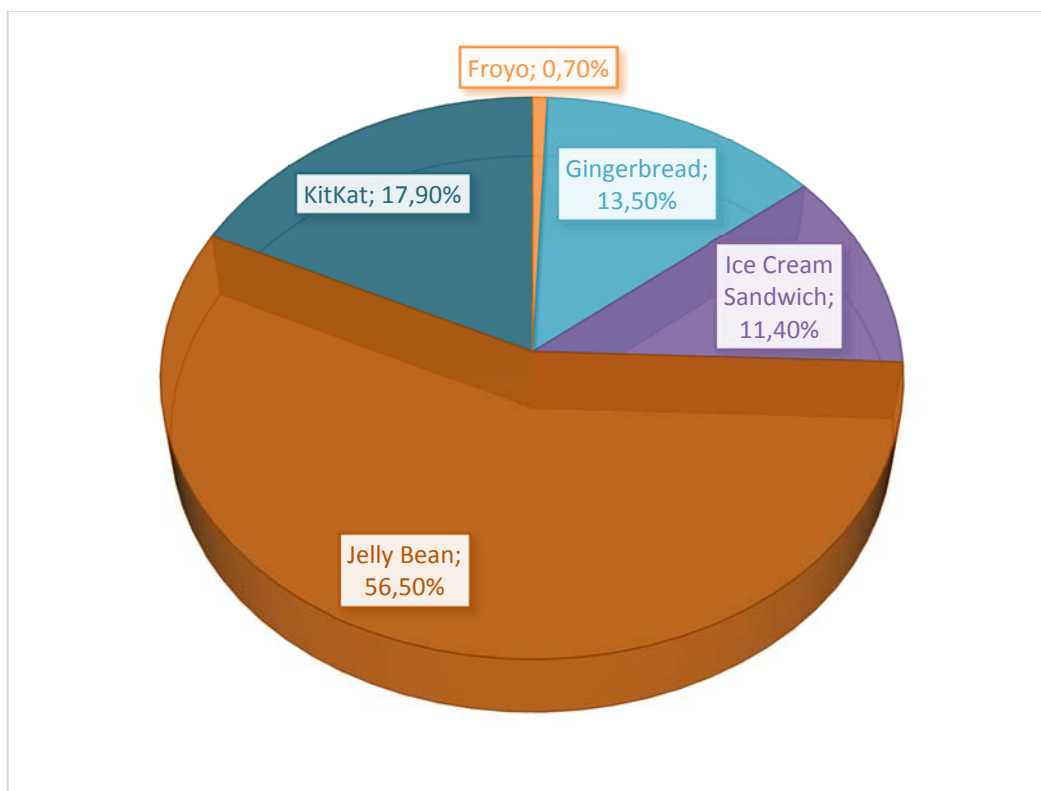
Při vývoji aplikace musí být zvaženo, pro jakou nejstarší verzi systému bude aplikace funkční. Obecně platí, že čím starší verze chceme podporovat, tím je vývoj náročnější. Důležitým aspektem při řešení tohoto problému, je zjištění, jaké verze jsou aktuálně nejvyužívanější. Na stránkách určených pro vývojáře<sup>10</sup> je k dispozici graf, který je každý týden aktualizován a ukazuje procentuální zastoupení jednotlivých verzí ve většině zařízení po celém světě. Aktuální data ukazují, že nejvyužívanější verzí Android je

---

<sup>10</sup>[https://developer.android.com/about/dashboards/index.html?utm\\_source=ausdroid.net](https://developer.android.com/about/dashboards/index.html?utm_source=ausdroid.net)

aktuálně JellyBean, tedy API verze 16 až 18, s více než polovičním podílem. Na druhém místě je doposud nejnovější verze, tedy KitKat (API 19), který využívá necelých 18 % mobilních zařízení. Za zmínku stojí verze Gingerbread (API 10), která i přes to, že je relativně stará, stále funguje na 13,5 % zařízení.

Aplikace je primárně psaná pro zařízení s verzí Android 4.4 (KitKat), protože jsem, jako autor aplikace, neměl k dispozici zařízení se starší verzí systému. Odkoušená je na většině verzí od 4.0, a to pomocí Android emulátoru na počítači. Starší verze, než 4.0 prozatím nejsou podporovány, i přes to, že s nimi bylo v průběhu vývoje počítáno. Například pro zobrazení fragmentů byla použita knihovna „Support library“, která umožňuje jejich zobrazení na zařízení od API verze 4. Starší verze nejsou podporovány proto, že se v Android emulátoru objevily různé chyby při běhu aplikace, které se mi nepodařilo zatím odstranit.



Obrázek 6 - Aktuální procentuální zastoupení jednotlivých verzí systému Android v mobilních zařízeních

## 7.2 Komunikace s webovými službami

Jako zdroj dat byl použit server webových služeb typu REST, a to především proto, že není nutné přenášet XML soubory v podobě SOAP zpráv při odesílání požadavku na server. Naopak, formát výstupu byl zvolen XML.

Pro komunikaci s webovými službami byl, v rámci této diplomové práce, vytvořen klient, který poskytuje API pro přístup k potřebným informacím. Klient byl napsán v programovacím jazyce Java a ke svému fungování nevyužívá žádnou externí knihovnu.

Klient se skládá z několika vrstev. Jádro klienta tvoří základní funkcionality pro komunikaci s webovými službami. Nad jádrem je další vrstva, která poskytuje specifické metody odpovídající metodám webových služeb, čímž případnému uživateli klienta usnadňuje práci s webovými službami, protože nemusí zkoumat názvy metod, které IS/STAG poskytuje. Paralelně s těmito vrstvami existuje další vrstva, která se stará o převod XML na objekty poskytující „get“ metody, což opět usnadňuje uživateli klienta práci v tom smyslu, že nemusí zkoumat obsah jednotlivých XML souborů, ale jednoduše zavolá příslušnou metodu objektu, například metoda uvedená níže vrátí název předmětu.

```
predmet.getNazev();
```

Díky těmto pomocným třídám a velkému množství metod se klient z původních 10 kB zvětšil na 70 kB.

Klient svou funkcí nepokrývá výčet všech webových služeb poskytovaných IS/STAG, nýbrž pouze ty, které byly využity v mobilní aplikaci.

### 7.3 Optimalizace získávání dat

Při práci na aplikaci se objevil problém v délce stahování informací z webových služeb. Například seznam známek, které student obdržel za dobu svého studia, se načítal déle, než 10 sekund. Důvodem je fakt, že webové služby jsou psány obecně a nejsou tudíž přizpůsobené pro potřeby konkrétní aplikace. To znamená, že výše zmíněný příklad načtení všech známek ve skutečnosti znamenalo stáhnout jedním dotazem seznam všech známek studenta a podle toho, kolik měl student známek, se prováděl odpovídající počet dotazů, které měly za úkol získat název předmětu k dané známce. Měl-li student 15 známek, provedlo se 1 + 15 dotazů.

První krok optimalizace spočíval v tom, že se uživateli dočasně zobrazil seznam známek, který nezobrazoval celý název předmětu, ale pouze zkratku, která se vrátí hned při prvním dotazu na seznam známek, a poté se postupně načítaly názvy jednotlivých předmětů. Tuto optimalizaci dokumentuje obrázek níže, kde je vidět, že část známek má zobrazený celý název předmětu a na zbytek se teprve čeká.

Subject	2011 ZS	2011 LS
Zpracování obrazu	1	
Pokročilé techniky programování	2	
INADS	1	
Elektronická příprava dokumentů	1	
INTP	1	
INZUI	1	
INPSW	1	
INPRS	1	
INAM	1	
Datové sklady	2	
Programování internetových aplikací	1	
Datové struktury a algoritmy	2	
INPDS	1	
INPG2	1	

**Obrázek 7 - Postupné načítání názvů předmětů k jednotlivým známkám**

Díky první optimalizaci uživatel nemusel čekat několik vteřin, než se mu objeví na displeji nějaká data, ale stále trvalo delší dobu, než se načetly všechny názvy předmětů. Musel tedy přijít na řadu další krok optimalizace a tím bylo využití úložiště v mobilním telefonu, kam se výsledky stažené z webových služeb IS/STAG uložily. Jako úložiště byla zvolena databáze SQLite, která je součástí systému. Celé to funguje tak, že se jednou stažené informace uloží do databáze a při příštím požadavku na název předmětu již nebude nutné opětovně stahovat informace z webových služeb, nýbrž bude stačit získání záznamu z lokální databáze. To znamená, že zmíněných několik vteřin uživatel počká jen jednou, příště už bude mít data uložená ve svém telefonu. Takto uložená data mají platnost jeden týden. Po týdnu se znovu automaticky aktualizují ze serveru IS/STAG. Touto optimalizací se celkově zrychlila práce s aplikací, jelikož stejný princip byl použit na rozvrh hodin, seznam termínů a další. Bude-li uživatel chtít získat aktuální data, stačí kliknout na tlačítko „aktualizovat“ v horní části displeje a tím vynutí stažení informací ze serveru IS/STAG.

## 7.4 Uživatelské prostředí aplikace

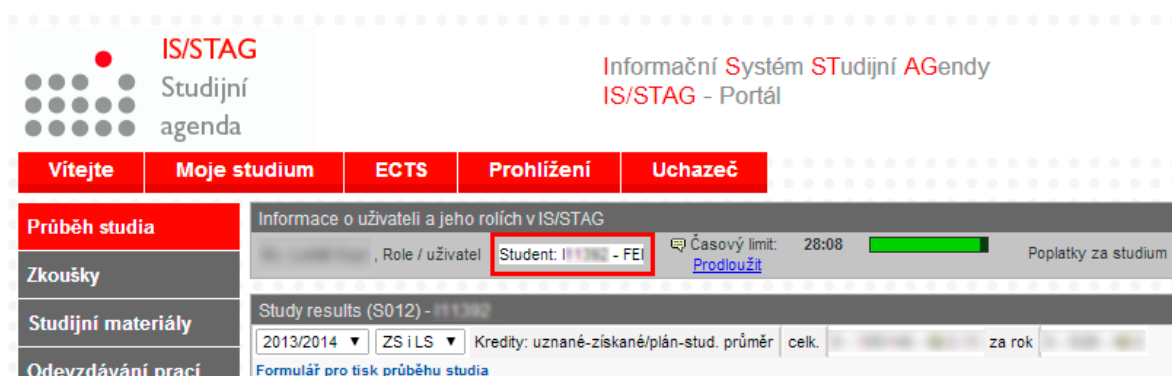
Po otevření aplikace se zobrazí úvodní obrazovka, kde uživatel vybere, zda je student, nebo vyučující. To znamená, že aplikaci nemůže využívat kdokoli, ale pouze zaměstnanci, či studenti Univerzity Pardubice.



Obrázek 8 - Úvodní obrazovka aplikace Mobilní IS/STAG

#### 7.4.1 Přihlášení do aplikace

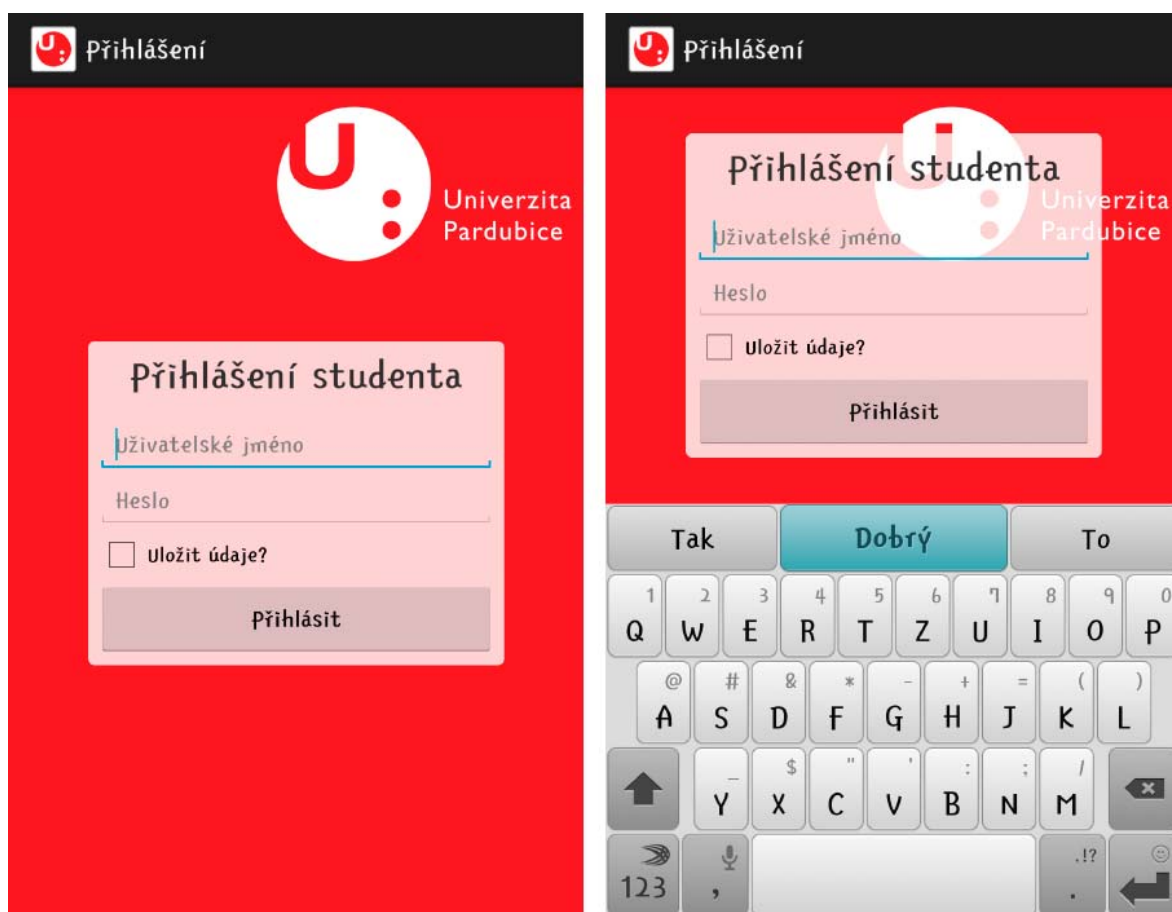
Přihlášení do aplikace je v podstatě přihlášení do webových služeb IS/STAG, což způsobuje drobný problém. Studenti jsou zvyklí se přihlašovat do webového portálu pomocí svého NetID, což je údaj ve tvaru „stXXXXX“, kde „XXXXX“ je číslo specifické pro každého studenta. Webové služby toto přihlašování nepodporují a uživatel je nucen se přihlašovat pomocí takzvaného osobního čísla studenta, které je ve tvaru „IXXXXX“, kde „XXXXX“ je opět číslo specifické pro každého studenta, ale jiné, než v případě NetID. Své osobní číslo studenta lze získat po přihlášení do webové aplikace IS/STAG v záložce Průběh studia v horní části stránky. Na obrázku níže je to vyznačené v červeně ohraničeném obdélníku.



Obrázek 9 - Získání osobního čísla studenta

Osobní číslo studenta uživatel získal, ale jaké má použít heslo? To je druhý údaj, který pravděpodobně nebude znát. Heslo je pro každého studenta nastaveno automaticky a nepodařilo se mi jej nikde změnit, pravděpodobně to nejde. Automatické heslo je ve tvaru „stagXXXXXXXXXX“, kde „XXXXXXXXXX“ je rodné číslo studenta včetně čísel za lomítkem, ale bez znaku lomítka.

Útěchou pro uživatele může být fakt, že si může v aplikaci, při přihlašování, nastavit, aby byly jeho přihlašovací údaje uloženy a tudíž je nebude muset příště znovu zadávat. Přihlašovací údaje jsou v telefonu uloženy v šifrované podobě, takže je nelze získat přečtením nějakého souboru. Případný útočník by musel získat zašifrované údaje a znát šifrovací algoritmus, který byl použit k šifrování.



Obrázek 10 - Obrazovka přihlášení (bez klávesnice vlevo, s klávesnicí vpravo)








#### 7.4.2 Úvodní obrazovka uživatele

Další popis uživatelského prostředí se bude týkat studentské verze. Část aplikace pro vyučující je shodná, s tím rozdílem, že vyučující nemají zobrazený seznam obdržených známek.

Po přihlášení uživatel, v tomto případě student, na obrazovce vidí své osobní číslo studenta, jméno, příjmení, fakultu a obor, který studuje. Nad těmito informacemi jsou zobrazeny různé ikonky, které si popíšeme dále. Pod základními informacemi o studentovi

je zobrazen jeho rozvrh v podobě seznamu, který je seřazen podle času rozvrhových akcí. Přejede-li uživatel prstem z levého kraje obrazovky doprava do prostoru, zobrazí se mu vyjíždějící nabídka, která obsahuje stejné ikonky, které jsou na úvodní obrazovce a k tomu ještě nějaké další.

Jednotlivé ikonky vyjíždějící nabídky jsou pojmenovány, ale pro jistotu zde uvedu, co se pod nimi skrývá.

<b>Ikonka</b>	<b>Název</b>	<b>Význam</b>
	Úvod	Kliknutím na tuto ikonku se zobrazí úvodní obrazovka uživatele
	Rozvrh hodin	Zobrazí rozvrh hodin přihlášeného uživatele
	Termíny zkoušek	Zobrazí termíny zkoušek a zápočtů přihlášeného uživatele
	Známky	Vypíše seznam obdržených známek přihlášeného uživatele
	Veřejné rozvrhy	Umožňuje zobrazení rozvrhů hodin ostatních studentů, či vyučujících
	Přepnout uživatele	Možnost přepnout z režimu student na vyučujícího a naopak
	Odhlásit	Provede odhlášení aktuálně přihlášeného uživatele a odstraní přihlašovací údaje, takže je bude nutné při dalším přihlášení vyplnit znovu

**Tabulka 4 - Význam ikon vyjíždějící nabídky**

### 7.4.3 Rozvrh hodin

Rozvrh hodin se zobrazí po kliknutí na příslušnou ikonku vyjíždějící nabídky. Uživatel si může vybrat, jaký ročník a semestr si přeje zobrazit. Aplikace si vždy pamatuje poslední volbu, takže při příštím zobrazení rozvrhu se zobrazí ten, který byl zobrazen naposledy.

Mobilní IS/STAG umožňuje dva způsoby zobrazení rozvrhu. Prvním způsobem je klasické zobrazení, známé například z webové aplikace IS/STAG. Druhý způsob je zobrazení formou seznamu, kdy se jednotlivé rozvrhové akce vypisují pod sebe chronologicky seřazené.

Po kliknutí na jakoukoli rozvrhovou akci se zobrazí její detail, který čítá základní informace o dané akci:

- čas konání,
- učebna,
- zkratka předmětu,
- název předmětu,



- přednášející,
- cvičící,
- doporučená literatura,
- anotace,
- přehled látky,
- požadavky,
- předpoklady a
- hodnotící metody.

Rozvrh hodin						
2011/2012 Letní semestr						
	7:00 1	8:00 2	9:00 3	10:00 4	11:00 5	12:00
Po		07:45 KIT/INDSK CA-PC104 Žák		10:45	11:00 KS CA k	
Út					11:30	
St			09:00 KST/INPG2 CA-SEM406 Veselý	11:00		
Čt						
Pá					11:00 K Š	
So						
Ne						

Rozvrh hodin						
2011/2012 Zimní semestr						
Pondělí						
15:00 – 17:00 INEPD CA-IT402 Hudec Elektronická příprava dokumentů						cv
17:00 – 19:00 INTP CA-H2 Rak Teorie pravděp. a mat. statistika						př
Úterý						
09:00 – 11:00 INPTP CA-SEM402 Greiner Pokročilé techniky programování						př
11:00 – 13:00 INZUI CA-SEM301 Taufner Základy umělé inteligence						př
13:00 – 15:00 INZO CA-SEM404 Fribert Zpracování obrazu						př
15:00 – 17:00 INADS CA-SEM402 Žák Architektura a techn. databáz. systémů						př
17:00 – 19:00 INPSW CA-SEM402 Šimerda Projektování SW systémů, UML						př

Obrázek 11 - Rozvrh hodin (klasický vlevo, seznam vpravo)

#### 7.4.4 Termíny zkoušek

Tato část aplikace zobrazuje termíny zkoušek aktuálního školního roku. Zkoušky jsou zobrazeny v podobě seznamu. Při kliknutí na jakýkoli termín se zobrazí podrobnosti termínu, podobně jako je tomu na rozvrhu, s tím rozdílem, že zde je navíc zobrazeno jméno vyučujícího, který termín vystavil a typ termínu (zkouška, zápočet, ...)

V detailu termínu také přibyl tlačítko „Vložit do kalendáře“. Po kliknutí na toto tlačítko se uživateli zobrazí dialog pro uložení upozornění do osobního kalendáře s předvyplněnými informacemi o daném termínu. Stejnou akci může uživatel vyvolat při delším stisku na termín v seznamu, kdy se mu zobrazí kontextové menu s dvěma položkami. Jednou je právě vložení do kalendáře a druhou je zobrazení detailu termínu.

#### **7.4.5 Známky**

Tato část aplikace poskytuje studentovi přehled o obdržených známkách v průběhu studia. Ke každé známce je informace o názvu předmětu, ročníku, ve kterém známku obdržel, semestr a hodnocení. Jsou zde vypsány i zápočty, u kterých je znak „S“ (splnil). Náhled této části aplikace je na obrázku Obrázek 7.

#### **7.4.6 Veřejné rozvrhy**

V této části si může uživatel vyhledat ostatní studenty, nebo vyučující. Stačí zadat jméno, příjmení, nebo obojí a vybrat, zda chcete hledat vyučujícího, či studenta. Po dokončení vyhledávání se zobrazí seznam nalezených položek, nebo hláška informující o tom, že žádný záznam nebyl nalezen.

Pro vyhledávání lze využít i zástupný znak „%“, který nahrazuje libovolný počet jakéhokoli znaku. Zadá-li uživatel například „L%“ do pole „Jméno“, aplikace zobrazí seznam, kde budou studenti, či učitelé se jmény Leoš, Lukáš, Luboš a případně nějaká další.

Pro zobrazení rozvrhu hodin nějakého ze studentů/učitelů stačí kliknout na jeho jméno a pak se zobrazí již známá obrazovka, která je shodná s dříve popisovaným rozvrhem hodin.

#### **7.4.7 Přepnout uživatele a odhlásit**

Předposlední položkou nabídky je „Přepnout uživatele“, která umožňuje přepínat mezi studentem a vyučujícím.

Poslední volbu, „Odhlásit“, využije uživatel, přeje-li se odhlásit z IS/STAG a odstranit přihlašovací údaje. To se hodí zejména při přihlášení na cizím mobilním zařízení.

### **7.5 Další možná vylepšení**

Aplikace splňuje požadavky zadání diplomové práce, přidává nějaké funkcionality navíc, ale zdaleka nepokrývá všechny možnosti, které systém Android a webové služby IS/STAG poskytují.

Jednou z věcí, na kterých by mohlo být zapracováno, je lepší optimalizace pro zařízení s větším displejem, aby aplikace lépe využívala větší obrazovou plochu.

Další vylepšení se netýká přímo aplikace, ale webových služeb IS/STAG. Konkrétně se jedná o způsob přihlašování, kdy se uživatel musí přihlašovat pomocí jiných údajů, než je zvyklý z webové verze. Po konzultaci s podporou webových služeb IS/STAG jsem byl informován, že by webové služby mohly do budoucna podporovat klasické přihlašování, ale v tuto chvíli tomu tak není.

V zadání diplomové práce byla funkcionalita exportu termínů zkoušek do kalendáře. Jako možné vylepšení by mohlo být poskytnutí uživatelům možnost zobrazit systémovou notifikaci bez nutnosti exportu do kalendáře.

Jak bylo zmíněno, verzi Android 2.3.3 využívá stále kolem 13,5 % uživatelů. Stálo by za úvahu, zda aplikaci neoptimalizovat i pro tuto verzi systému. Ještě záleží na tom, zda je procento mezi studenty podobné. Je jasné, že mladší generace uživatelů mobilních telefonů obměňují své přístroje častěji, takže by procento využívání starší verze mohlo být mezi studenty nižší.

Do budoucna by mohla aplikace poskytovat i veřejné informace bez nutnosti přihlašování. Problém je v tom, že webové služby IS/STAG neposkytují žádné informace, ani ty veřejně dostupné, bez přihlášení.

## Literatura

1. **Wikipedie.** Symbian OS. *Wikipedie, otevřená encyklopedie*. [Online] 18. Duben 2014. [Citace: 20. Červenec 2014.] [http://cs.wikipedia.org/wiki/Symbian\\_OS](http://cs.wikipedia.org/wiki/Symbian_OS).
2. **Dr. Friedrich Schwandt, Tim Kröger.** Global market share 2009-2013. *Statista*. [Online] Únor 2014. [Citace: 7. Srpen 2014.] <http://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/>.
3. **Wikipedie.** iPad. *Wikipedie, otevřená encyklopedie*. [Online] 5. Srpen 2014. [Citace: 10. Srpen 2014.] <http://cs.wikipedia.org/wiki/IPad>.
4. —. Android (operační systém). *Wikipedie, otevřená encyklopedie*. [Online] 7. Srpen 2014. [Citace: 10. Srpen 2014.] [http://cs.wikipedia.org/wiki/Android\\_%28opera%C4%8Dn%C3%AD\\_syst%C3%A9m%29](http://cs.wikipedia.org/wiki/Android_%28opera%C4%8Dn%C3%AD_syst%C3%A9m%29).
5. **Mysliveček, David.** Krátké ohlédnutí za historií Androidu. *Svět Androida*. [Online] 11. Květen 2013. [Citace: 20. Červenec 2014.] <http://www.svetandroida.cz/kratke-ohlednuti-za-historii-androidu-201305>.
6. **VÁCLAVÍK, LUKÁŠ.** Android 5.0 bude rychlejší. Google definitivně vymění Dalvik za ART. *Cnews.cz*. [Online] 20. Červen 2014. [Citace: 8. Červenec 2014.] <http://www.cnews.cz/android-50-bude-rychlejsi-google-definitivne-vymeni-dalvik-za-art>.
7. **Allen, Grant.** *Android 4 Průvodce programováním mobilních aplikací*. Brno : Computer Press, 2013. 978-80-251-3782-6.
8. **AndroidWiki.** Aktivita a intenty. *AndroidWiki*. [Online] 9. Listopad 2012. [Citace: 20. Červenec 2014.] [http://www.androidwiki.cz/Aktivita\\_a\\_intenty](http://www.androidwiki.cz/Aktivita_a_intenty).
9. **Android Developers.** Activity. *Android Developers*. [Online] 12. Srpen 2014. [Citace: 13. Srpen 2014.] <http://developer.android.com/reference/android/app/Activity.html#onRestoreInstanceState>.
10. **Konečný, Matěj.** Vyvíjíme pro Android: Fragmenty a SQLite databáze. *Zdroják*. [Online] 20. Červenec 2012. [Citace: 10. Srpen 2014.] <http://www.zdrojak.cz/clanky/vyvijime-pro-android-fragmenty-a-sqlite-databaze/>.
11. —. Vyvíjíme pro Android: Suroviny, Intenty a jednotky. *Zdroják*. [Online] 29. Červen 2012. [Citace: 11. Srpen 2014.] <http://www.zdrojak.cz/clanky/vyvijime-pro-android-suroviny-intenty-a-jednotky/>.
12. **Android Developers.** Providing Alternative Resources. *Android Developers*. [Online] [Citace: 21. Červenec 2014.]

<http://developer.android.com/guide/topics/resources/providing-resources.html#AlternativeResources>.

13. —. Manifest.permission. *Android Developers*. [Online] 12. Srpen 2014. [Citace: 13. Srpen 2014.] <http://developer.android.com/reference/android/Manifest.permission.html>.

14. **Malý, Martin**. REST: architektura pro webové API. *Zdroják*. [Online] 3. Srpen 2009. [Citace: 8. Srpen 2014.] <http://www.zdrojak.cz/clanky/rest-architektura-pro-webove-api/>.

15. **Wikipedie**. Webová služba. *Wikipedie, otevřená encyklopedie*. [Online] 10. Březen 2013. [Citace: 5. Červenec 2014.] [http://cs.wikipedia.org/wiki/Webov%C3%A1\\_slu%C5%BEba](http://cs.wikipedia.org/wiki/Webov%C3%A1_slu%C5%BEba).

16. **Churý, Lukáš**. Základy XML webových služeb. *Programujte.com*. [Online] 6. Srpen 2005. [Citace: 8. Srpen 2014.] <http://programujte.com/clanek/2005081704-zaklady-xml-webovych-sluzeb/>.

17. **Univerzita Pardubice**. Webové služby nad IS/STAG. *Univerzita Pardubice*. [Online] [Citace: 10. Duben 2014.] <https://stag-ws.upce.cz/ws/help/>.

## Příloha A – Ukázka WSDL dokumentu ze serveru IS/STAG UPCE

Jedná se o WSDL dokument webové služby „termíny“

```
<?xml version='1.0' encoding='UTF-8'?><wsdl:definitions
name="TerminyServiceImplService" targetNamespace="http://stag-ws.zcu.cz/"
xmlns:ns1="http://cxfl.apache.org/bindings/xformat"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://stag-ws.zcu.cz/"
xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<wsdl:types>
<xs:schema attributeFormDefault="unqualified"
elementFormDefault="unqualified" targetNamespace="http://stag-ws.zcu.cz/"
xmlns:tns="http://stag-ws.zcu.cz/"
xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:element name="termin" type="tns:terminType" />
<xs:element name="terminy" type="tns:terminyType" />
<xs:element name="ucitel" type="tns:ucitelType" />
<xs:complexType name="terminyType">
<xs:sequence>
<xs:element maxOccurs="unbounded" minOccurs="0" name="termin"
nillable="true" type="tns:terminType" />
</xs:sequence>
</xs:complexType>
<xs:complexType name="terminType">
<xs:sequence>
<xs:element name="termIdno" type="xs:long" />
<xs:element name="ucitIdno" type="xs:long" />
<xs:element name="ucitel" type="tns:ucitelType" />
<xs:element name="predmet" type="xs:string" />
<xs:element name="katedra" type="xs:string" />
<xs:element name="rok" type="xs:string" />
<xs:element name="semestr" type="xs:string" />
<xs:element name="datum" type="tns:xmlDateType" />
<xs:element minOccurs="0" name="obsazeni" type="xs:string" />
<xs:element minOccurs="0" name="limit" type="xs:string" />
<xs:element name="casOd" type="xs:string" />
<xs:element name="casDo" type="xs:string" />
<xs:element name="budova" type="xs:string" />
<xs:element name="mistnost" type="xs:string" />
<xs:element minOccurs="0" name="poznamka" type="xs:string" />
<xs:element name="opravny" type="xs:string" />
<xs:element minOccurs="0" name="deadlineDatumOdhlaseni"
type="tns:xmlDateType" />
<xs:element minOccurs="0" name="deadlineDatumPrihlaseni"
type="tns:xmlDateType" />
<xs:element minOccurs="0" name="zacatekPrihlasovani"
type="tns:xmlDateType" />
<xs:element name="platnost" type="xs:string" />
<xs:element name="typTerminu" type="xs:string" />
</xs:sequence>
</xs:complexType>
<xs:complexType name="ucitelType">
<xs:sequence>
<xs:element name="ucitIdno" type="xs:long" />
<xs:element name="jmeno" type="xs:string" />
<xs:element name="prijmeni" type="xs:string" />
<xs:element minOccurs="0" name="titulPred" type="xs:string" />
```

```

<xs:element minOccurs="0" name="titulZa" type="xs:string" />
<xs:element name="platnost" type="xs:string" />
<xs:element name="zamestnanec" type="xs:string" />
</xs:sequence>
</xs:complexType>
<xs:simpleType name="xmlDateType">
<xs:restriction base="xs:string" />
</xs:simpleType>
<xs:element name="getTerminyZkousek" type="tns:getTerminyZkousek" />
<xs:complexType name="getTerminyZkousek">
<xs:sequence>
<xs:element minOccurs="0" name="termIdno" nillable="true" type="xs:long" />
<xs:element minOccurs="0" name="ucitIdno" nillable="true" type="xs:long" />
<xs:element minOccurs="0" name="katedra" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="zkratka" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="semestr" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="rok" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="osCislo" nillable="true" type="xs:string" />
<xs:element minOccurs="0" name="zobrazitProsle" nillable="true" type="xs:boolean" />
<xs:element minOccurs="0" name="zobrazitAktualni" nillable="true" type="xs:boolean" />
<xs:element minOccurs="0" name="zobrazitZrusene" nillable="true" type="xs:boolean" />
<xs:element minOccurs="0" name="zobrazitBlokovane" nillable="true" type="xs:boolean" />
<xs:element minOccurs="0" name="lang" nillable="true" type="xs:string" />
</xs:sequence>
</xs:complexType>
<xs:element name="getTerminyZkousekResponse" type="tns:getTerminyZkousekResponse" />
<xs:complexType name="getTerminyZkousekResponse">
<xs:sequence>
<xs:element name="terminy" type="tns:terminyType" />
</xs:sequence>
</xs:complexType>
</xs:schema>
</wsdl:types>
<wsdl:message name="getTerminyZkousekResponse">
<wsdl:part element="tns:getTerminyZkousekResponse" name="parameters">
</wsdl:part>
</wsdl:message>
<wsdl:message name="getTerminyZkousek">
<wsdl:part element="tns:getTerminyZkousek" name="parameters">
</wsdl:part>
</wsdl:message>
<wsdl:portType name="TerminyService">
<wsdl:operation name="getTerminyZkousek">
<wsdl:input message="tns:getTerminyZkousek" name="getTerminyZkousek">
</wsdl:input>
<wsdl:output message="tns:getTerminyZkousekResponse" name="getTerminyZkousekResponse">
</wsdl:output>
</wsdl:operation>

```

```

</wsdl:portType>
<wsdl:binding name="TerminyServiceImplServiceSoapBinding"
type="tns:TerminyService">
<soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http" />
<wsdl:operation name="getTerminyZkousek">
<soap:operation soapAction="" style="document" />
<wsdl:input name="getTerminyZkousek">
<soap:body use="literal" />
</wsdl:input>
<wsdl:output name="getTerminyZkousekResponse">
<soap:body use="literal" />
</wsdl:output>
</wsdl:operation>
</wsdl:binding>
<wsdl:service name="TerminyServiceImplService">
<wsdl:port binding="tns:TerminyServiceImplServiceSoapBinding"
name="TerminyServiceImplPort">
<soap:address location="https://stag-ws.upce.cz/ws/services/soap/terminy"
/>
</wsdl:port>
</wsdl:service>
</wsdl:definitions>

```



## Příloha B – SOAP požadavek a odpověď

Požadavek:

```
<SOAP-ENV:Envelope
  xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:SOAP-ENC="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <SOAP-ENV:Body>
    <m:getStudentiByPredmet xmlns:m="http://stag-ws.zcu.cz/">
      <katedra>KIV</katedra>
      <predmet>DTP1</predmet>
      <rok>2004</rok>
      <semestr>LS</semestr>
      <lang>String</lang>
    </m:getStudentiByPredmet>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Odpověď:

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Body>
    <ns1:getStudentiByPredmetResponse xmlns:ns1="http://stag-
ws.zcu.cz/">
      <ns2:studenti xmlns:ns2="http://stag-ws.zcu.cz/">
        <student>
          <osCislo>A02196</osCislo>
          <jmeno>Lukáš</jmeno>
          <prijmeni>BALINT</prijmeni>
        </student>
        <student>
          <osCislo>A04504</osCislo>
          <jmeno>Jan</jmeno>
          <prijmeni>BÁRTA</prijmeni>
          <titulPred>Bc.</titulPred>
        </student>
      </ns2:studenti>
    </ns1:getStudentiByPredmetResponse>
  </soap:Body>
</soap:Envelope>
```

## **Příloha C – Obsah CD přiloženého k diplomové práci**

CD obsahuje:

- Zdrojové kódy klienta komunikujícího s webovými službami IS/STAG v podobě projektu v NetBeans IDE verze 8.0,
- zkompileovaný projekt klienta pro okamžité použití,
- zdrojové kódy aplikace Mobilní IS/STAG v podobě projektu ve vývojovém prostředí Android Studio verze 0.8.2,
- zkompileovaná aplikace připravená pro okamžitou instalaci do mobilního zařízení s operačním systémem Android,
- nechybí ani textová část této diplomové práce.