TECHNICKÁ UNIVERZITA V LIBERCI
Ekonomická fakulta

APLIKACE PRO OPERAČNÍ SYSTÉM ANDROID A JEJICH VYUŽITÍ V PODNIKOVÝCH INFORMAČNÍCH SYSTÉMECH

Bakalářská práce

Studijní program: B6209 – Systémové inženýrství a informatika
Studijní obor: 6209R021 – Manažerská informatika

Autor práce: **Petr Apeltauer**
Vedoucí práce: Ing. Vladimíra Zádová, Ph.D.

Liberec 2014

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Economics

APPLICATIONS FOR THE ANDROID OPERATING SYSTEM AND ITS USAGE IN ENTERPRISE INFORMATION SYSTEMS

Bachelor thesis

Study programme: B6209 – System Engineering and Informatics
Study branch: 6209R021 – Managerial Informatics

Author: **Petr Apeltauer**
Supervisor: Ing. Vladimíra Zádová, Ph.D.

Liberec 2014

TECHNICKÁ UNIVERZITA V LIBERCI
Ekonomická fakulta
Akademický rok: **2014/2015**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Petr Apeltauer**
Osobní číslo:	**E12000457**
Studijní program:	**B6209 Systémové inženýrství a informatika**
Studijní obor:	**Manažerská informatika**
Název tématu:	**Aplikace pro operační systém Android a jejich využití v podnikových informačních systémech**
Zadávající katedra:	**Katedra informatiky**

Zásady pro vypracování:

1. Vývojové prostředí NetBeans
2. Specifika vývoje aplikací pro platformu Android
3. Vývoj mobilních aplikací - trendy a rizika
4. Analýza požadavků a tvorba mobilní aplikace
5. Zhodnocení přínosu řešení

Rozsah grafických prací:

Rozsah pracovní zprávy: **30 normostran**

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

Seznam odborné literatury:

GRANT, Allen. Android 4: Průvodce programováním mobilních aplikací. 1. vyd. Brno: Computer Press, 2013. ISBN 978-80-251-3782-6.
MEIER, Reto. Professional Android 4 Application Development. 1st ed. Indianapolis: Wiley, 2012. ISBN 978-1-118-23722-9.
COHEN, Ryan a Tao WANG. GUI Design for Android Apps. 1 st ed. New York: Apress, 2014. ISBN 978-1-4842-0383-5.
TRAVIS, Brian E. XML a SOAP Programování serverů BizTalk. 1. vyd. Praha: Computer Press, 2000. ISBN 80-7226-303-X.
Elektronická databáze článků ProQuest (knihovna.tul.cz).

Vedoucí bakalářské práce: **Ing. Vladimíra Zádová, Ph.D.**
Katedra informatiky

Konzultant bakalářské práce: **Ing. Petr Motl**
OR-CZ, spol. s r. o.

Datum zadání bakalářské práce: **31. října 2014**

Termín odevzdání bakalářské práce: **7. května 2015**

doc. Ing. Miroslav Žižka, Ph.D.
děkan

L.S.

doc. Ing. Jan Skrbek, Dr.
vedoucí katedry

V Liberci dne 31. října 2014

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

Podpis:

Poděkování

Poděkování za podporu a odborné poradenství při psaní bakalářské práce patří vedoucí práce Ing. Vladimíře Zádové, Ph.D., a Ing. Petru Motlovi, který zastupoval funkci konzultanta. Velký dík patří také všem rodinným příslušníkům, za stálou podporu během studia.

Anotace

Bakalářská práce Aplikace pro operační systém Android a jejich využití v podnikových informačních systémech se zabývá využitím mobilních zařízení s operačním systémem Android jako koncových zařízení komunikujících s informačním systémem podniku.

Práce popisuje vývoj, rizika a aktuální dění ve světě mobilního byznysu, a předkládá náhled do vývoje a testování aplikací pro systém Android s využitím vývojového prostředí NetBeans. Dále stručně popisuje důležité prvky aplikací pro Android, projektovou strukturu, webové služby a protokol pro výměnu zpráv – Simple Object Access Protocol.

Hlavním přínosem práce je vývoj aplikace komunikující s informačním systémem podniku, analýza požadavků pro možnost jejího vzniku a zhodnocení přínosu daného řešení.

Klíčová slova

Aktivita, Android, aplikační rozhraní, emulátor, fragment, informační systém, integrované vývojové prostředí, layout, mobilní zařízení, operační systém, platforma, view, virtuální zařízení.

Annotation

The bachelor thesis Applications for the Android Operating System and its Usage in Enterprise Information Systems deals with the usage of mobile devices – running the Android operating system – as terminals communicating with the enterprise information system.

This thesis describes the development, risks and the current trends in the world of mobile business, and presents the development and testing of applications for Android using the NetBeans IDE. It also briefly describes the essential elements of Android applications, project structure, web services and messaging protocol – Simple Object Access Protocol.

The main contribution of this work is the development of an application communicating with the enterprise information system, requirements analysis for creation of the application, and evaluation of benefits of the proposed solution.

Key Words

Activity, Android, Application Programming Interface, emulator, fragment, information system, Integrated Development Environment, layout, mobile device, operating system, platform, view, virtual device.

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

3G	Third Generation Wireless, mobilní datová síť třetí generace
API	Application Programming Interface, aplikační programové rozhraní
AVD	Android Virtual Device, Android virtuální zařízení
ČVUT	České vysoké učení technické
ERP	Enterprise Resource Planning, plánování podnikových zdrojů
EXE	executable, spustitelný
GUI	Graphical User Interface, grafické uživatelské rozhraní
HTTP	Hypertext Transfer Protocol, protokol pro přenos hypertextových dokumentů
IDE	Integrated Development Environment, integrované vývojové prostředí
IP	Internet Protocol, internetový protokol
IS	informační systém
JDK	Java Development Kit, Java vývojová sada
JIT	Just-In-Time, metoda kompilace při spuštění
MVC	Model-View-Controller, softwarová architektura model-pohled-řadič
NFC	Near Field Communication, radiová bezdrátová komunikace
OS	operační systém
REST	Representational State Transfer, architektura pro distribuované prostředí
SDK	Software Development Kit, vývojová sada
SIP	Session Initiation Protocol, protokol pro inicializaci relací
SOAP	Simple Object Access Protocol, protokol pro výměnu zpráv
TUL	Technická univerzita v Liberci
USB	Universal Serial Bus, univerzální sériová sběrnice
VPN	Virtual Private Network, virtuální privátní síť
XML	Extensible Markup Language, rozšiřitelný značkovací jazyk

Úvod

Android je operační systém (dále jen OS) vyvinutý americkou společností Google, který byl představen v roce 2007. První volně dostupná verze platformy byla uvedena na trh v roce 2008. Primárně je zaměřen na přenosná zařízení s dotykovými displeji, jako jsou chytré telefony a tablety. Tento systém je založen na jádře OS Linux a je licencován zdarma, jako veřejně dostupný, otevřený software (open-source). Android jako produkt náleží alianci OHA (Open Handset Alliance), do které patří mnoho organizací jako je Google, HTC, Motorola, Samsung a jiné. Na přelomu roku 2014 a 2015 byla vydána jeho zatím poslední verze Android 5.0, označující se také jako Lollipop.

Z hlediska oblíbenosti OS mobilních zařízení se Android ujímá prvního místa. Existuje enormní počet vyvinutých aplikací, který se každým dnem zvyšuje.

Na stránce oficiálního webu [1], věnovaného vývojářům pro Android je uvedeno: *„Android powers hundreds of millions of mobile devices in more than 190 countries around the world. It's the largest installed base of any mobile platform and growing fast – every day another million users power up their Android devices for the first time and start looking for apps, games, and other digital content.“* To lze volně přeložit jako: „Android používá stovky miliónů přístrojů, ve více než 190 zemích světa. Je to nejvíce instalovaný základ jakékoliv mobilní platformy a jeho používání roste velice rychle – každý den další a další miliony uživatelů zapínají své první Android zařízení a začínají hledat aplikace, hry a jiný digitální obsah.“

Na systému Android běží více než 70 % chytrých telefonů, jak se zmiňuje Paul Elias [2]. Pro toho, kdo chce, aby byla možnost využití jeho aplikací co nejvyšší, je jasnou volbou. Navíc vznikají stále nové, volně dostupné, knihovny třetích stran, které podstatně zjednodušují práci vývojářů a významně rozšiřují možnosti používání mobilních aplikací pro Android.

Uvedené skutečnosti měly podstatný vliv na volbu tématu bakalářské práce. Jejím hlavním cílem je vývoj mobilní aplikace pro OS Android ve vývojovém prostředí NetBeans propojené s informačním systémem (dále jen IS) podniku. Teoretická část práce je

zaměřena na vývojové prostředí NetBeans z hlediska programování pro Android a specifika vývoje aplikací pro Android, jsou zmíněny i aktuální trendy a rizika související s užíváním mobilních zařízení. Praktická část obsahuje postup zavedení podpory programování pro systém Android s vytvořením jednoduché aplikace, popis analýzy požadavku zákazníka s následnou tvorbu mobilní aplikace umožňující komunikaci uživatele s IS podniku a zhodnocení přínosu řešení.

1. Zhodnocení současného stavu

Vývoj aplikací pro OS Android, zaměřených na komunikaci s IS podniku, je poměrně nový trend, který se rychle rozrůstá. Nejčastěji se vytváří aplikace na chytré telefony a tablety. Zařízení s tímto cílem se v organizacích vyskytují skoro všude, užívají je manažeři, vedení, řádoví pracovníci, používány jsou dokonce i ve výrobních dílnách podniků. Zatím neexistuje mnoho odborných knih věnovaných kombinaci systému Android s IS organizací, nicméně je patrné, že tomu tak do budoucna nebude.

Rozbor OS Android, z hlediska jeho využití na vytváření aplikací jako klientů IS, je popsán v článku Ondřeje Bergera, s názvem *„Analýza systému Android ve vztahu ke klientské části informačních systémů“*. Dokument rozebírá části, ze kterých je možné vytvořit chtěný program, a uvádí s tím související prvky a technologie.

O detailním popisu ukázky využití aplikace pro platformu Android spojené s IS se zaměřením na řízení skladu a evidenci prodejců, pojednává diplomová práce Vojtěcha Šobáňa, s názvem *„Informační systém pro řízení skladu a návrh mobilní aplikace pro Android“*. Další ukázkou využití aplikace s IS se zabývá bakalářská práce Martina Šestáka *„Klient pro studijní informační systém KOS na platformě Android“*, v rámci které byla vytvořena a popsána aplikace pro usnadnění přístupu ke konkrétnímu IS univerzity ČVUT (České vysoké učení technické), umožňující vyhledávání osob, zapsaných předmětů, kopírování kalendáře ze systému apod. Kromě ČVUT se mobilnímu využití IS věnují i studenti jiných institucí, jako Lukáš Novák z Univerzity Palackého v Olomouci, se svou bakalářskou prací *„Aplikace na objednávání jídel pro platformu Android“* a Jan Barcík z Unicorn College, jehož bakalářská práce nese název *„Vývoj e-commerce platformy pro mobilní zařízení s podporou webového backendu“*.

Také na Technické univerzitě v Liberci (TUL), vzniklo na dané téma mnoho kvalifikačních prací, jako je například diplomová práce *„Mobilní aplikace pro práci s univerzitním elearningovým portálem na platformě Android“*, jejíž autor Ondřej Vacek, vyvinul mobilní aplikaci, která pomocí využití klientské a serverové části komunikuje přes webovou službu s e-learningovým portálem založeným na systému Moodle. Komunikace je zde zpracována přes SOAP (Simple Object Access Protocol). Zajímavou diplomovou

prací je též *„Synchronizace databáze mobilního zařízení OS Android se serverovou databází“*, kde Leoš Dostál kromě jiného popisuje vývoj klientské části ve formě knihovny a synchronizaci databáze mobilního zařízení s databází serveru. V diplomové práci *„Systém řízení chytrého domu s inteligentní elektroinstalací na základě OS Android“*, tvůrce Mikhail Yudakhin, píše o mobilní aplikaci vytvořené v rámci systému umožňujícího kontrolu a řízení systému inteligentního domu. O dalším využití platformy Android, v souvislosti s pokladním systémem, pojednává diplomová práce Martina Hozáka s názvem *„Pokladní systém pro restaurace a bary“*. Jinou zajímavou diplomovou prací, vytvořenou na TUL, za účelem navržení a implementace aplikace pro správu procesů a úkolů, je *„Klient server aplikace pro Activiti workflow“*, v rámci které Jiří Sojka vyvinul klientskou část na mobilní platformu Android.

O akademickém IS se lze dočíst v odborném článku *„Implementation of MVC (Model-View-Controller) Architectural to Academic Management Information System with Android Platform Base“*, kde autoři Pravina Utpatadevi, Oka Sudana a Agung Cahyawan, popisují implementaci zajímavé metody utváření aplikací pomocí separace dat, stavící na architektuře MVC (Model-View-Controller), do IS založeném na platformě Android.

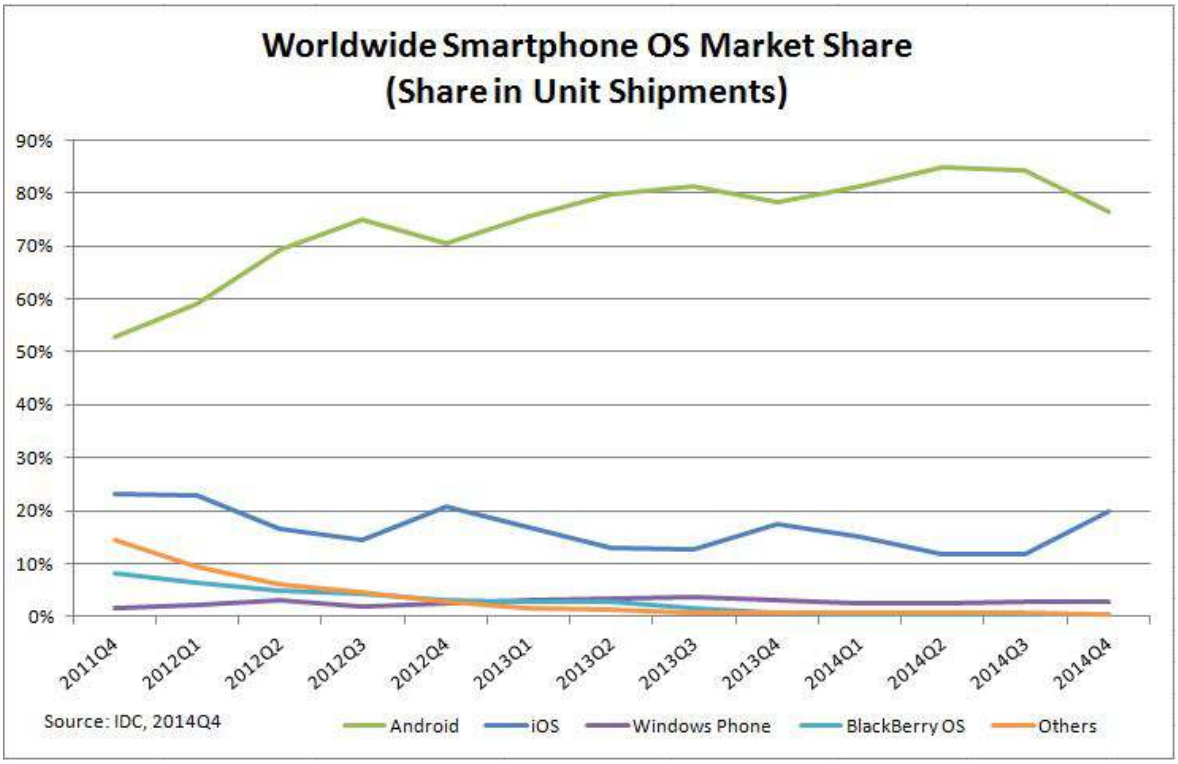
Všechny uvedené publikace nemají příliš velký rozsah, jde spíše o akademické práce a odborně zaměřená díla pojednávající o platformě Android v souvislosti s IS. Výjimkou není ani dílo *„Livestock Information System using Android Smartphone“*, kde M. H. Ariff a I. Ismail, využívají platformy Android s technologií Bluetooth k získání informací o zdravotním stavu hospodářských zvířat, měřeného pomocí nositelných senzorů.

V uvedených dílech byla komunikace s IS prostřednictvím procedurálně založeného SOAP prakticky využita pouze v rámci bakalářské práce Ondřeje Vacka, valná většina ostatních autorů k tomuto účelu využila datově orientované softwarové architektury REST (Representational State Transfer). Vývojové prostředí NetBeans pro vytváření aplikací nepoužil nikdo z autorů, častým bylo však využití prostředí Eclipse.

Na nynějším trhu je nabízeno již mnoho aplikací komunikujících s podnikovými IS (Helios Mobile, SAP Business One, AuditPro, erpDroid aj.), hromadně vydávány začaly být však teprve nedávno a opravdový vzestup teprve zažijí.

2. Vývoj, trendy a rizika

Významnými OS pro mobilní zařízení jsou Android, iOS, Windows Phone a BlackBerry. Jednoznačně nejdominantnějším je však Android, který je nejrozšířenějším a nejvíce využívaným základem mobilních platforem. Na následujícím obrázku lze vidět vývoj podílu na celosvětovém trhu z hlediska operačních systémů pro chytré telefony (počítáno podle prodaných kusů). Největším konkurentem OS Android se jeví iOS. Vývoj jejich křivek si navzájem zrcadlově odpovídá.



Obr. 1: Celosvětový podíl na trhu OS pro chytré telefony
Zdroj: http://www.idc.com/prodserv/smartphone-ms-img/chart-ww-smartphone-os-market-share.png [vid. 2015-04-20]

Hlavní vliv na popularitu systému Android u vývojářů má pravděpodobně jeho volná licence a možnost programování v jazyce Java se snadno dostupnými vývojovými nástroji, díky kterým je vytváření aplikací rychlé a jednoduché. Změnu kódového obsahu lze okamžitě zobrazit na vlastním zařízení či na virtuálním zařízení v počítači. S jeho popularitou souvisí i vývoj jednotlivých verzí OS Android, které k sobě postupně přitáhly velké množství vývojářů i běžných uživatelů.

2.1 Významné milníky OS Android

Aplikace pro Android se vyvíjí velmi rychle, a s tím i požadavky na jejich funkcionalitu. Kromě chytrých telefonů a tabletů je jisté jeho rozšiřování mezi jiná zařízení a stroje (hodinky, roboti, automobily), nicméně jak uvádí Grant Allen [3 s. 27]: *„Hlavní oblastí uplatnění systému Android však budou i nadále zařízení s menšími obrazovkami a s hardwarovou klávesnicí či bez ní. Ze statistického hlediska pak bude systém Android v blízké budoucnosti pravděpodobně i nadále spojován především s chytrými telefony.“*

Mezi nejvýznamnější milníky vývoje systému Android patří verze Cupcake, Donut, Eclair, Froyo, Gingerbread, Honeycomb, Ice Cream Sandwich, Jelly Bean, KitKat a nejnovější Lollipop. Verze jsou anglicky pojmenované podle pochutin a seřazeny abecedně. Každé z verzí systému Android odpovídá určité API (Application Programming Interface).

API je zkratka označující aplikační programové rozhraní. Jedná se o soubor protokolů, rutin a nástrojů, umožňujících vytváření různých aplikací. Specifikuje vzájemnou interakci softwarových komponentů a používá se pro vytváření grafického uživatelského rozhraní prezentovaného na displeji zařízení, neboli GUI (Graphical User Interface) [4]. Pro vývojáře to znamená, že jako programátor může využívat zabudované funkce platformy Android. Tyto funkce přibývají a rozšiřují se společně se zvyšováním stupně API.

Stupeň API je číselná hodnota, jednoznačně identifikující aplikační rámec API (framework), který poskytuje určitá verze Android platformy [5]. Zjednodušeně řečeno, stupeň API vyznačuje konkrétní formu API.

Mezi významné verze OS Android patří ([3], [6], [7], [8], [9] a [10]):

- *Android 1.5, Cupcake (API 3)* – Verze Cupcake přinesla ochranu kopírování aplikací, a základní funkcionalitu domovské obrazovky a uživatelského prostředí (GUI). Zlepšila funkce videokamery, Bluetooth a zavedla služby YouTube.
- *Android 1.6, Donut (API 4)* – Donut měla velký vliv na rozmach. Android byl rozšířen do více typů telefonů, a zlepšil podporu fotoaparátů a galerií. Byla také představena dostupnost API poskytujících alternativní interakční metody pro uživatele

s vizuálními, fyzickými, a jinými handicapy, kvůli lepší interakci s dotykovými obrazovkami.

- *Android 2.0, Eclair (API 5)* – Verze Eclair zpřístupnila vývojářům možnost práce s kontakty v mobilu a s běžícími procesy aplikace. Dále přinesla nový vzhled pro odlišné displeje, zlepšení vyhledávače, Google Maps a položila základy navigaci.
- *Android 2.2, Froyo (API 8)* – Vydání Froyo přivedlo pomocné funkce pro možnost ukládání dat a zápisu do souboru, spravování chování aplikací odsunutých do pozadí (chování audio přehrávačů), možnost interakce s mobilem pomocí hlasu, a především byl zvýšen výpočetní výkon procesorů díky JIT kompilátoru (Just-In-Time).
- *Android 2.3, Gingerbread (API 9)* – Verze Gingerbread zoptimalizovala využitelnost baterie, zinovovala menu, dialogová okna a jiné vzhledové prvky. Přinesla překlad do nových světových jazyků a hlavně podporu zajímavých technologií, jako je radiová bezdrátová komunikace NFC (Near Field Communication) a internetové volání pomocí SIP (Session Initiation Protocol).
- *Android 3.0, Honeycomb (API 11)* – Nové možnosti přizpůsobení GUI od API 11 způsobili rozšíření OS Android na již zcela běžně užívané tablety, pro které byla verze specificky určená. A to hlavně díky zavedení fragmentů (viz kapitolu 4.2) díky kterým lze přizpůsobit aplikace větším obrazovkám.
- *Android 4.0, Ice Cream Sandwich (API 14)* – Velmi elegantní design vznikl s verzí 4.0, která zpřístupnila telefonům prvky určené pro Honeycomb, a další komponenty související hlavně se vzhledem (např. podporu formátu WEBP pro obrázky).
- *Android 4.1, Jelly Bean (API 16)* – Jelly Bean rozšířila grafické vymoženosti, přinesla možnost nastavitelných účtů pro více uživatelů, a především zrychlila reakce zařízení na podněty uživatele. Stala se opravdu obdivovanou verzí se spoustou stoupenců.
- *Android 4.4, KitKat (API 19)* – Zvýšený výkon, optimalizaci paměti, inteligentnější chování a trochu pozměněnou grafickou stránku uvedla verze KitKat, dovolující ještě lepší a rychlejší komunikaci.
- *Android 5.0, Lollipop (API 21)* – Nejaktuálnější a zatím nejrozsáhlejší změny přináší verze Lollipop, jež imponuje svým GUI přizpůsobeným na různé displeje, do čehož patří jak nositelná elektronika s velice malými obrazovkami, tak i tablety či televize s obrovskými úhlopříčkami. Verze podporuje 64 bitové architektury platforem,

animace, novější 3D grafiku se stíny, materiální design pro přizpůsobení na různě velká zařízení, stylové efekty a mnoho jiných nových prvků.

2.2 Dění ve světě mobilního byznysu

Jednou z význačných novinek ze světa mobilního byznysu je studie, která naznačuje, že Android zažije v roce 2015 ještě větší rozmach, přičemž iOS naopak utrpí. Jako mobilní platforma je totiž Android na svém vrcholu a stále konstantně roste [11].

Dalším trendem je růst používaných objektů v mobilním byznysu a zvyšující se možnosti využití mobilních aplikací. Nedávno například Samsung představil automatickou pračku kontrolovatelnou mobilním zařízením prostřednictvím Wi-Fi, a Google vydal aplikaci na detekci kouře. Významným trendem jsou také cloudové aplikace, které nepracují lokálně na mobilu a lze k nim přistupovat odkudkoliv [11], a rozšiřování nositelné elektroniky označované jako „wearables“, pod niž se řadí elektronika, kterou může mít člověk upevněnou na těle. Patří sem například chytré hodinky, náramky, brýle, kamery, a jiná elektronika s různorodou funkcí (měřící teplotu, tlak, tep aj.) [12].

Stále rozvíjejícím se trendem ve světě mobilního byznysu je vývoj klientských aplikací pro komunikaci s IS organizace. Velký vliv na popularitu tohoto vývojového směru má potřeba mobility, jejíž přínos významně zefektivňuje systémy podniků. Aplikace komunikující s IS šetří čas uživatelům a umožňují okamžitou práci se systémem. S tím souvisí i nutnost pracovního off-line režimu, díky kterému může zaměstnanec pracovat s daty, i když není datové připojení k dispozici. A ve chvíli, kdy je připojení obnoveno, se data aktualizují do centrálního systému. Toho využívá například aplikace Helios Mobile vyvinutá společností Asseco Solutions, a.s. pro komunikaci s IS Helios typu ERP (Enterprise Resource Planning) [13].

2.2.1 Bezpečnost

Přenosná zařízení, především chytré telefony, se stávají našimi osobními asistenty v životě a pomocníky pro uchování důležitých informací. To s sebou přináší i stinnou stránku,

kterou představuje možnost ztráty důležitých dat, jejich změnu, ukradení identity apod. Sám OS Android má jistá bezpečností opatření, jako je například zamknutí telefonu po uvedení do stavu spánku či kliknutím na tlačítko. Pro aktivaci telefonu je tak nutno zadat číselný kód, heslo nebo nějaké gesto. Na mobilní zařízení existují také antivirové programy, jejichž použití je rozhodně dobré zvážit. Proti nebezpečným hackerům to však mnohdy nestačí. Jak se mimo jiné zmiňuje Stuard Dredge [14], právě jejich útoky na sociální média rozpoutaly širokou debatu, která má a bude mít vliv i na software mobilních telefonů. Mnoho aplikací v nynější době chabě střeží svůj obsah proti narušitelům. Mathumuniasamy Sivanpillai [15] uvádí, že přibližně 75 % vyvinutých mobilních aplikací neprojde ani základními bezpečnostními testy a hackeři z nich dokážou získat citlivé informace.

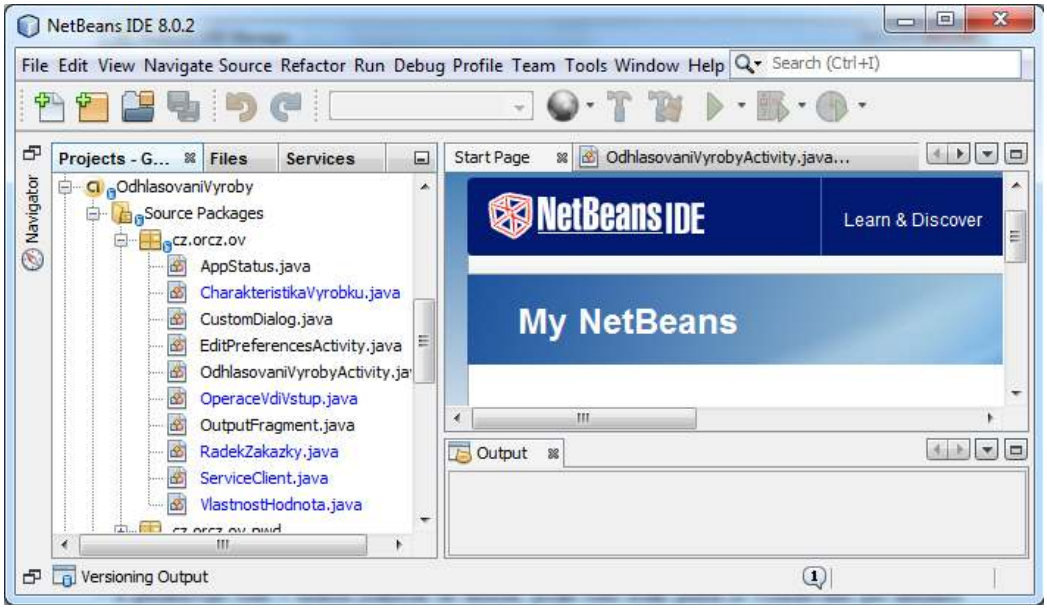
Používání mobilních zařízení s IS podniku vyžaduje ještě větší obezřetnost. Krádež zařízení může znamenat získání přístupu do systému, kde by si narušitel dělal, cokoliv chce. To by mohlo mít silný dopad na celý podnik a jeho zaměstnance. Nicméně správným využitím platformy Android se dá rizikovým problémům předejít.

Jedním z opatření je vyhovující definování aplikace s omezenými funkčnostmi pro určitý okruh pracovníků. Pokud bude například aplikace moci data jen zapisovat, nemusíme se bát datového úniku ani mazání. Na druhou stranu vytvářet specifikované aplikace zvlášť pro každou skupinu lidí podle jejich příslušných práv je neefektivní.

Zajímavou možností, se kterou přišla společnost TruconneXion, a.s [13], je zavedení centralizace vzdálené správy služebních mobilních zařízení, díky které lze ovládat koncové přístroje (např. mobily, tablety) a tím se bránit nepříznivým situacím. Jejich aplikace AuditPro umožňuje dálkové vymazání dat, vyfocení zloděje, spuštění alarmu a jiná bezpečnostní opatření.

3. Vývojové prostředí NetBeans a jeho podpora pro Android

NetBeans IDE (Integrated Development Environment) je integrované vývojové prostředí, primárně využívané pro práci s programovacím jazykem Java (lze ho však využít i pro JavaScript, C++, XML aj.). Jedná se o silně využívaný, bezplatný, otevřený software. Obsahuje velké množství komponent pro zjednodušení práce programátora, což zahrnuje vyznačené řádkování, doplňování textu, podtrhávání syntaktických či sémantických chyb, šablony a nápovědy. Přehledně zobrazuje struktury aplikací, které mohou obsahovat stovky složek a souborů, a nabízí podpůrné funkce, jako je například režim ladění, díky kterému lze snadněji najít chyby v programu. Tak reprezentují produkt na stránce oficiálního webu [16]. Aktuálně nejnovější dostupnou verzí je NetBeans 8.0.2 (viz Obr. 2).



Obr. 2: Integrované vývojové prostředí NetBeans
Zdroj: vlastní

Aby bylo možné v NetBeans IDE vytvářet aplikace pro Android, je nutné udělat několik úprav a nainstalovat doplňky, mezi které patří zásuvný modul *NBAndroid* a vývojová sada *Android SDK* (Software Development Kit), která mimo jiné obsahuje *emulátor* (virtuální zařízení). Jednotlivé kroky k přizpůsobení prostředí NetBeans pro programování a testování aplikací na OS Android jsou uvedeny v kapitolách 3.3 a 3.4.

3.1 Modul NBAndroid a Android SDK

NBAndroid je zásuvný modul podporující vývojový cyklus Android aplikací, do čehož patří podpora projektů založených na pomoci nástrojů SDK, užití emulátoru a zlepšené editory pro Android XML soubory [17].

Vývojová sada SDK pro Android obsahuje vzorový projekt se zdrojovým kódem, ladící a vývojové nástroje, emulátor a základní potřebné knihovny pro vytváření aplikací [18]. Součástí vývojové sady je i SDK Manager poskytující možnost instalace doplňků, nástrojů a komponent podporujících odlišné verze OS Android a jejich aktualizace.

Android SDK mimo jiné obsahuje i API pro podporu hardwaru založeného na lokaci (např. GPS), fotoaparátu, zvuku, síťového připojení, Wi-Fi, Bluetooth, senzorů, NFC, dotykových obrazovek, řízení spotřeby a jiných vymožeností, jak uvádí Reto Meier [6].

3.2 Základní emulátor a jeho alternativy

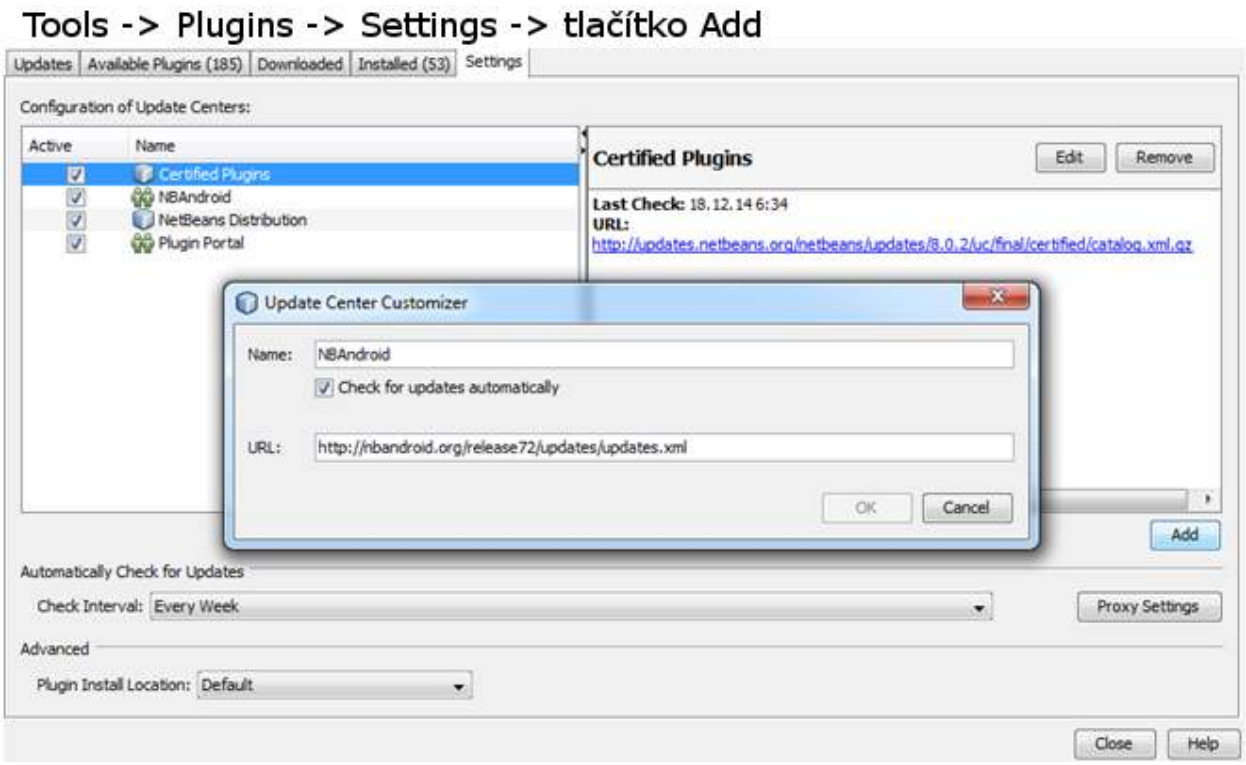
Emulátor je software reprezentující virtuální zařízení umožňující testování a ladění aplikací pomocí simulace jejího chování na zařízeních s odlišnou verzí OS Android, odlišnou velikostí a rozlišením displeje. Základní emulátor je součástí nástrojů SDK, stejně jako AVD Manager (Android Virtual Device Manager), k vytváření a správě virtuálních zařízení. Používání základního emulátoru však může být značně neefektivní.

Jednou z alternativ pro testování aplikací je využití vlastního zařízení s OS Android, druhou je pak emulátor Genymotion, což je program volně dostupný pro osobní užití, mající jisté hardwarové a softwarové požadavky. Start i odezvy emulátoru Genymotion jsou rychlejší a tím je vývoj aplikací efektivnější (záleží však i na parametrech počítače).

3.3 Zavedení podpory pro vývoj aplikací

Kapitola částečně vychází z dokumentace [19] Petra Motla, který uvádí, že pro zavedení podpory pro Android je potřeba mít vývojové prostředí minimální verze NetBeans 7.2, a také sadu JDK (Java Development Kit) verze 7 a vyšší. Po jejich instalaci je nutné provést několik kroků, aby bylo možné začít vyvíjet aplikace na OS Android:

1. Doinstalovat zásuvný modul s názvem NBAndroid. V menu vývojového prostředí (viz Obr. 2) je potřeba vybrat kartu *Tools* a následnou cestu, jako je v horním popisu na Obr. 3. Zde se musí vyplnit do políček název modulu[1] i adresa, na které je modul dostupný[2], a po té vše potvrdit tlačítkem *OK*.



Obr. 3: Získání modulu NBAndroid
Zdroj: vlastní

[1] NBAndroid

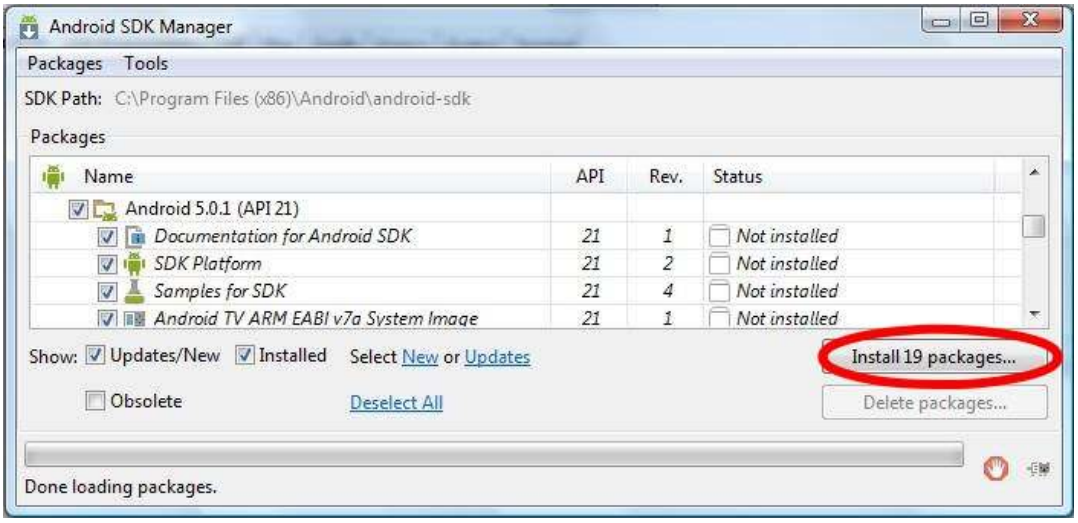
[2] http://nbandroid.org/release72/updates/updates.xml

2. V záložce *Available Plugins* zadat do políčka *Search* slovo „Android“. Tím se zobrazí moduly, ze kterých je potřeba vybrat Android, NBAndroid Extension a NBAndroid Gradle Support, následně kliknout na tlačítko *Install* a po té *Next*.
3. V rozbalovacím menu vybrat nabízenou verzi Groovy Support a zahájit instalaci. Je možné, že se objeví chyba, protože je potřebná nějaká další komponenta. Nejčastěji je vyžadována ruční instalace podpůrného modulu Groovy and Grails . Ten lze také najít v záložce *Available Plugins*.
4. Restartovat NetBeans IDE.
5. Získat vývojovou sadu SDK. Tu lze najít na příslušné adrese[3], pod nadpisem *SDK Tools Only*. Pro uživatele OS Windows je doporučeno vybrat soubor, u kterého je v závorce napsáno „Recommended“.
6. Stažený soubor nástrojů SDK rozbalit do adresáře *C:\android-sdk*, v případě stažení instalačního EXE souboru se do něj musí provést instalace. Pokud ji nebude možno provést, je nutno před tím ještě nastavit systémovou proměnnou *JAVA_HOME* s hodnotou nesoucí cestu k adresáři s nástroji JDK, a popřípadě i upravit systémovou proměnnou *Path*.[4]
7. Měl by se automaticky spustit SDK Manager. V případě, že se tak nestane, je potřeba ho vyhledat a pustit ručně. Dále se musí zatrhnout chtěné verze Android a kliknout na *Install X*[5] *packages,* čímž se po akceptaci licencí nainstalují doplňky (viz Obr. 4). Později bude možno spustit SDK Manager z prostředí NetBeans (karta *Tools* a záložka *Android SDK Manager*) a dodatečně nakonfigurovat další potřebné verze OS Android.

[3] http://developer.android.com/sdk/index.html#Other

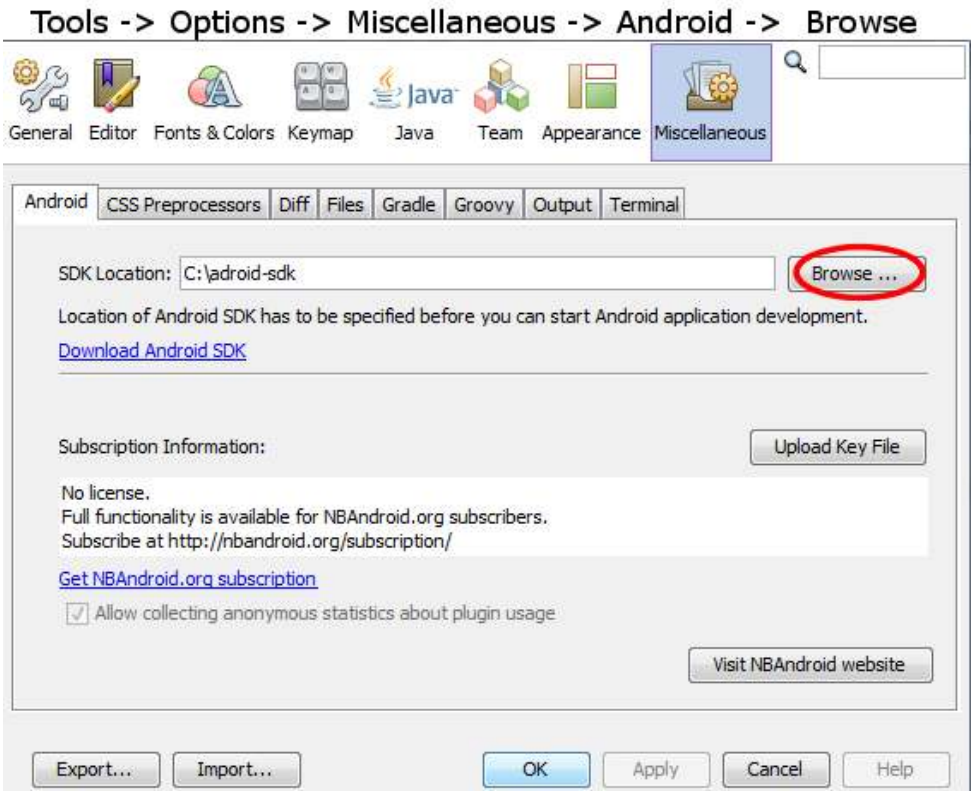
[4] Detailní postup využitelný pro většinu ze systémů Windows: http://zaantar.eu/2013/10/nastaveni-promenne-java_home-ve-windows-7/

[5] X představuje číslo v řádech jednotek až desítek, podle toho kolik příloh je vyžadováno pro instalaci podpory dané verze OS Android.



Obr. 4: Instalace podpory verze Android 5.0.1
Zdroj: vlastní

8. V menu NetBeans IDE je potřeba vybrat kartu *Tools* a postupovat dle horního popisu na Obr. 5. Po té se musí do políčka *SDK Location* nastavit cesta k SDK v adresáři *C:\android-sdk*.
9. Nakonec je nutné restartovat vývojové prostředí NetBeans.



Obr. 5: Definice cesty k adresáři s nástroji SDK
Zdroj: vlastní

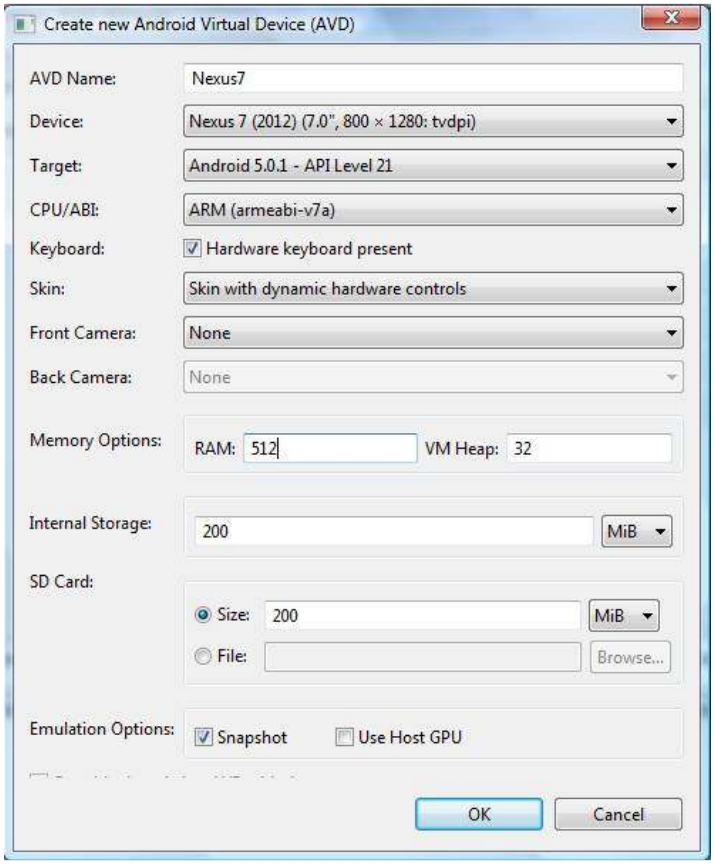
3.4 Zavedení podpory pro testování aplikací

Díky krokům v kapitole 3.3 lze začít vytvářet aplikace pro OS Android. Nicméně důležitou částí vývoje je i testování, bez kterého se kompletní vývojový cyklus aplikací neobejde. To je možné provádět několika způsoby.

3.4.1 Základní emulátor SDK

Pro používání základního emulátoru k testování aplikací je třeba ho vytvořit, musí se tedy:

1. Na kartě *Tools* v NetBeans IDE otevřít AVD Manager a kliknout na *Create*. Otevře se okno, ve kterém je díky předchozím krokům v nabídce *Target* verze Android 5.0.1.
2. Další parametry je potřeba nastavit dle vlastního uvážení (typ procesoru, paměť, kapacita SD karty atd.) a potvrdit. Výhodné je zatrhnout položkou *Snapshot*, ta způsobí spouštění emulátoru ve stavu, v jakém byl před vypnutím (viz Obr. 6).



Obr. 6: Nastavení emulátoru
Zdroj: vlastní

3.4.2 Využití vlastního zařízení

Pro vývojáře, kteří chtějí testovat aplikace na vlastním zařízení s OS Android (mobil, tablet aj.), je nutno:

1. Připojit zařízení k počítači a v jeho nastavení vybrat položku *Možnosti pro vývojáře* či anglicky *Developer options*. Zde zatrhnout *Ladění USB* (Universal Serial Bus) neboli *USB debugging*.
2. Získat a nainstalovat *USB ovladač* (*USB driver*). Pokud ho OS počítače nenainstaluje automaticky nebo pro jeho instalaci nenabídne řešení, musí se ovladač na konkrétní zařízení stáhnout ze stránek výrobce či jinde[6].
3. Nakonec restartovat počítač. Při následném spouštění projektu z NetBeans IDE by mělo být po provedení kroků v kapitole 3.5.1 nabízeno i vlastní zařízení.

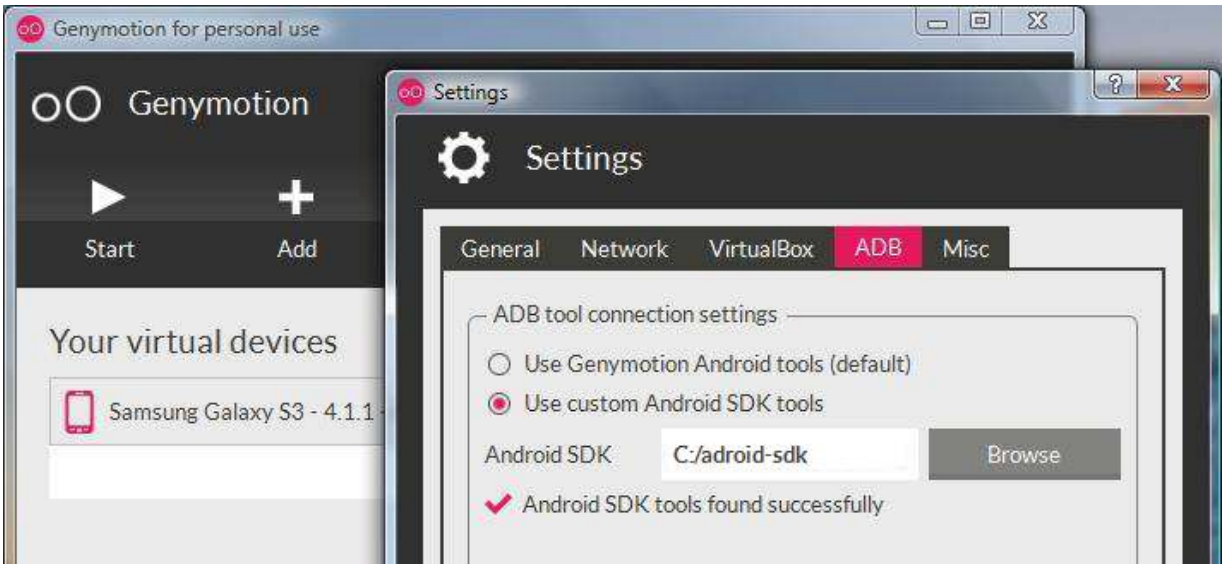
3.4.3 Genymotion

Efektivnější a rychlejší emulátor, než je ten obsažený v SDK, nabízí Genymotion. Pro jeho získání a nastavení je potřeba udělat následující:

1. Zaregistrovat se na oficiálních stránkách[7], přihlásit se a stáhnout instalační soubor. Emulátory Genymotion pracují na bázi vizualizačního nástroje Oracle Virtual Machine VirtualBox, který je ve stažitelném souboru obsažen také.
2. Nainstalovat a spustit Genymotion.
3. V jeho prostředí kliknout na tlačítko *Add* a přihlásit se dle údajů zadaných při registraci na oficiálních stránkách (tlačítko *Sign in*).
4. Vybrat si ze seznamu zařízení, potvrdit a po stažení kliknout na *Settings*, kde se na kartě *ADB* musí nastavit cesta k nainstalovanému SDK (viz následující obrázek).

[6] Doporučujeme vyhledat ovladač podle typu telefonu na stránce: http://singledrivers.blogspot.cz/

[7] Stránky k registraci a stažení programu Genymotion: https://www.genymotion.com/#!/pricing



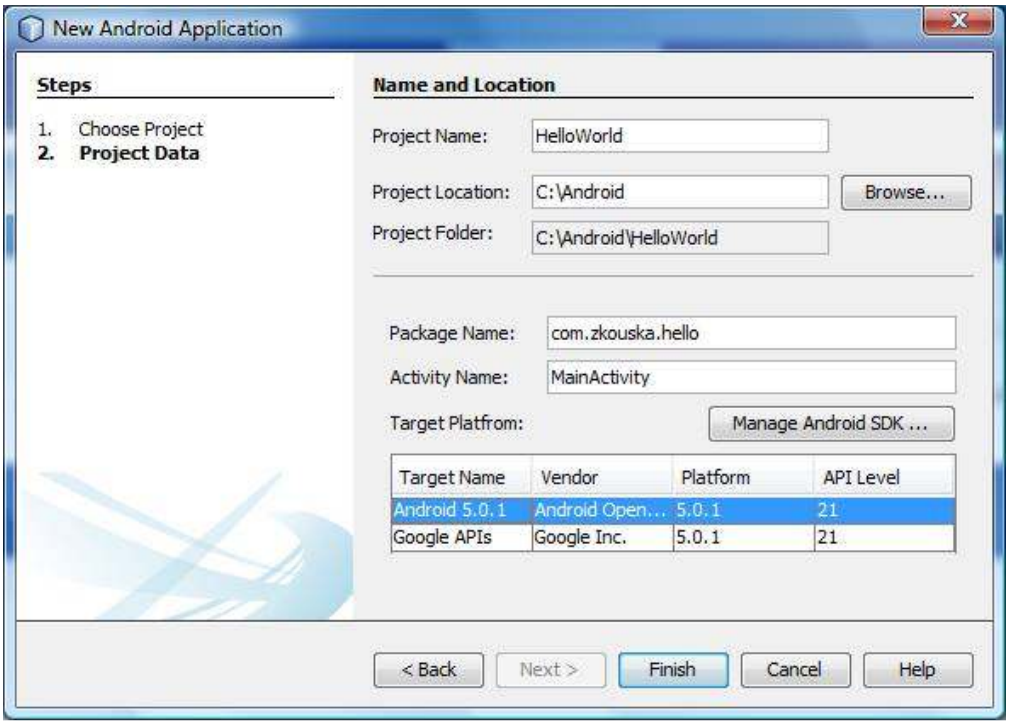
Obr. 7: Definice cesty k adresáři s SDK v programu Genymotion
Zdroj: vlastní

5. Spustit emulátor tlačítkem *Start*. Ten bude viditelný i při spouštění projektu v prostředí NetBeans, po provedení kroků v kapitole 3.5.1. Pokud používaný počítač nesplňuje systémové požadavky programu, nemusí se emulátor spustit.

3.5 Zkušební aplikace

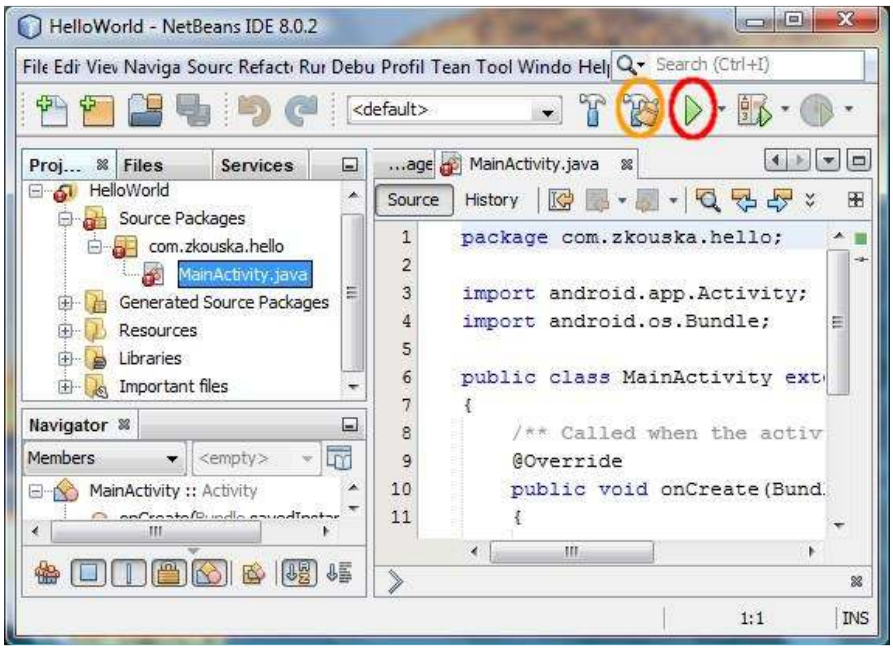
Správné provedení předchozích kroků lze zkontrolovat vytvořením první zkušební aplikace HelloWorld:

1. V prostředí NetBeans je nutno zadat kartu *File* a položku *New Project*. Otevře se okno se dvěma sloupci. Ve sloupci *Categories* je potřeba označit *Android*, ve sloupci *Projects* vybrat *Android Project*, a po té kliknout na *Next*.
2. Políčka název projektu (*Project Name*), lokace (*Projet Location*), název balíčku (*Package Name*) a název hlavní aktivity (*Activity Name*) lze vyplnit dle nadcházejícího obrázku (viz Obr. 8) a potvrdit tlačítkem *Finish*. Po provedení se vytvoří nový projekt. Pojmy aktivita a balíček budou rozvedeny v další části dokumentu.



Obr. 8: Vytváření nového projektu v NetBeans IDE
Zdroj: vlastní

3. Dále se musí, jako na Obr. 9, provést *Clean and Build* (vyznačeno žlutě), čímž se vytvoří či aktualizují příslušné složky a soubory projektu, a kliknout na *Run* (vyznačeno červeně) pro zkompilování a spuštění. Nastartuje se virtuální zařízení a spustí aplikace.



Obr. 9: Přeložení projektu a spuštění emulátoru
Zdroj: vlastní

První start virtuálního zařízení může trvat delší dobu. Emulátor s verzí Android 5.0.1 a vyšší může zatížit počítač. Pokud se nechce načíst, je lepší raději vytvořit virtuální zařízení s nižší verzí (např. 4.1.2). K tomu musí být nainstalována příslušná podpora verze OS v programu SDK Manager (karta *Tools*, záložka *Android SDK Manager*) nebo stažena jiná verze emulátoru Genymotion. V případě, že ani základní emulátor s nižší verzí OS nenastartuje, je možná chyba v nekompatibilitě s grafickou kartou.

3.5.1 Výběr z dostupných zařízení

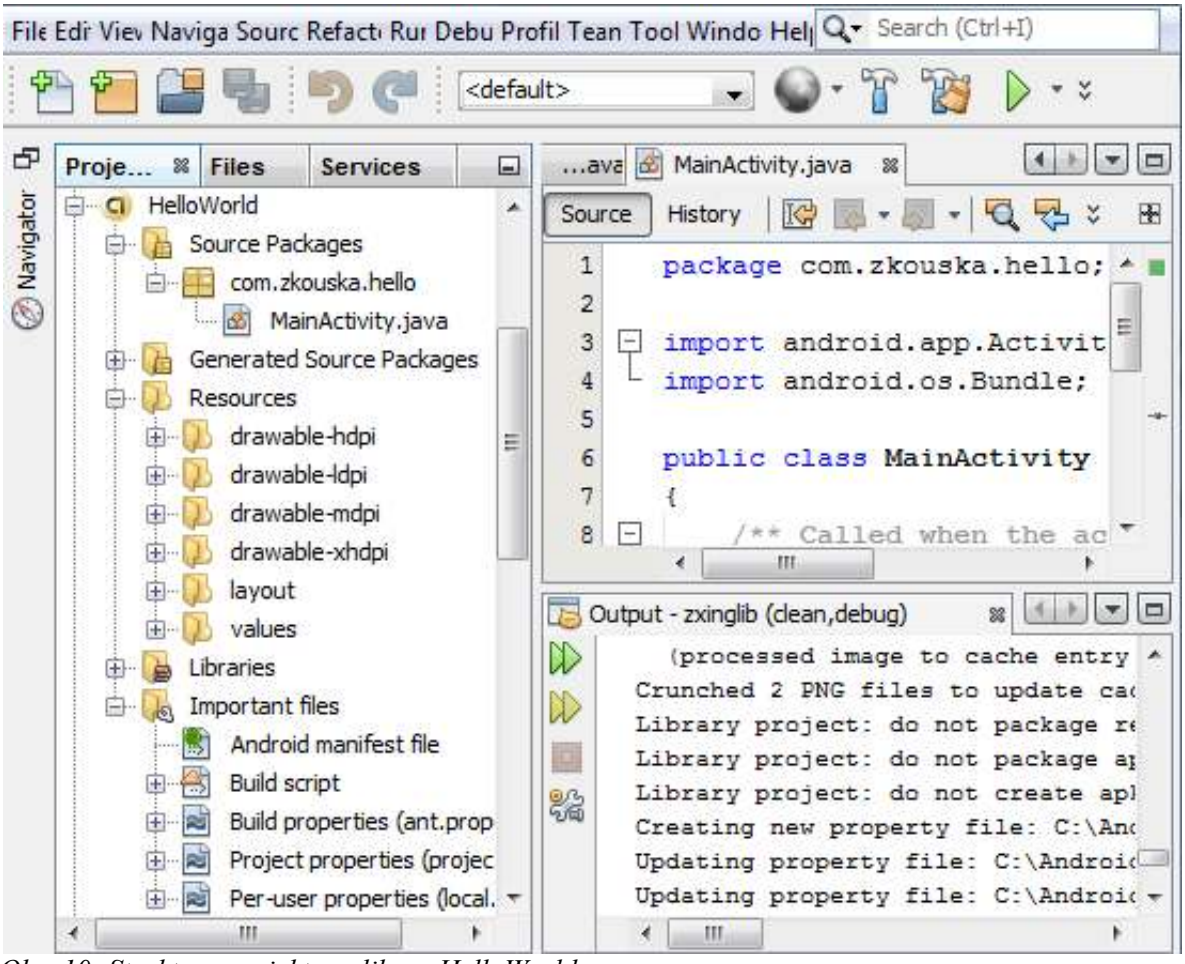
Užitečnou funkcí je možnost si při spouštění aplikace z prostředí NetBeans vybrat z více dostupných zařízení. Ve vývojovém prostředí se toho dá docílit v menu na kartě *Run*, kde je potřeba vybrat záložku *Set Project Configuration*, a *Customize*. Zde se musí vybrat *Manual* (u *Target Device*) a potvrdit.

4. Specifika vývoje aplikací

Při vytváření projektů (aplikací) pro Android je mnohdy potřeba pracovat s rozsáhlými projektovými strukturami a získat znalosti o nových prvcích, které se ve zdrojovém kódu objevují a tím odlišují vývoj pro Android od obecného programování v Javě.

4.1 Struktura projektu aplikace pro Android

Strukturu projektu aplikace vytváří velké množství složek obsahujících soubory různých typů s odlišným obsahem, jež definují funkčnost aplikace a GUI. Navážeme-li na aplikaci HelloWorld, jejíž vytvoření je popsáno v kapitole 3.5, můžeme si ve vývojovém prostředí všimnout její projektové struktury (viz Obr. 10).



Obr. 10: Struktura projektu aplikace HelloWorld
Zdroj: vlastní

Mezi adresáře vytvářející strukturu projektu patří:

- *Source Packages* – Obsahuje soubory se zdrojovým kódem aplikace. Ty mohou být uchovávány v odlišných balíčcích. Hlavní částí programu je tzv. *aktivita* (na Obr. 10 definovaná pomocí třídy *MainActivity*), bez které by nešla aplikace spustit. S touto částí struktury obvykle pracuje vývojář nejvíce.
- *Generated source packages* – Při překládání kódu se zde automaticky generují soubory, obsahující základní konfigurační třídy a cesty ke zdrojům aplikace. Do tohoto adresáře není potřeba nijak zasahovat.
- *Resources* – Zde jsou uloženy zdroje (suroviny, prostředky), se kterými aplikace pracuje. Zdroje jsou rozděleny do jednotlivých složek (layout, values atd.).
  Názvy podadresářů mají jistý význam, neboť se podle nich určuje, jak mají být určité prostředky použity [20].
- *Drawable* – Adresář nejčastěji uchovávající obrázky (s příponami png, jpg, gif aj.). Znaky za pomlčkou určují, jaké obrázky mají být užity při jakém rozlišení obrazovky daného zařízení. Přičemž ldpi (low) je nejnižší rozlišení, pak následuje mdpi (medium), hdpi (high) a xhdpi (extra-high).
- *Layout* – Obsahuje soubory ve formátu XML. Ty jsou přidělovány aktivitám či fragmentům, kterým vytvářejí masku (layout) a tím mohou rozvrhnout, co a jak se bude zobrazovat na displeji přístroje. Soubory definují tedy vzhled aplikace, což zahrnuje tlačítka, obrázky, textová pole, pozadí, styl textu a jiné prvky.
- *Values* – Zde jsou uloženy XML soubory obsahující řetězce, barvy, styly apod. Nejvíce využívané jsou řetězce, neboli textové výrazy, uložené v souboru strings.xml. Jejich způsob zobrazování se definuje v kódu či v masce (layoutu). To je velice výhodné pro překládání aplikace do světových jazyků. Jejich rozdělení se dá specifikovat příponou, např. values-cs je určen pro češtinu, dále existuje values-en (angličtina), values-sk (slovenština), values-ru (ruština) atd.
- *Xml* – Častou součástí zdrojů je i adresář xml, v němž jsou uchovány ostatní XML soubory, využívané jako zdroje aplikace [3].
- *Libraries* – Zde se mohou ukládat knihovny, díky kterým je práce vývojářů jednodušší, umožňují totiž využívání dalších doplňků a vymožeností.

- *Important files* – Jak název napovídá, adresář uchovává důležité soubory pro funkčnost aplikace. Za zmínku stojí hlavně *AndroidManifest.xml*, popisující utvářenou aplikaci, její součásti a vyžadovaná práva. Jsou v něm uvedeny i softwarové a hardwarové požadavky pro aplikaci, a musí zde být definovány všechny třídy dědící ze třídy *Activity*. Další soubory v adresáři popisují cestu k nástrojům sady SDK, stupeň použitého API při vývoji a jiné informace o projektu.

4.2 Základní charakteristické prvky

Strukturu projektu aplikace pro platformu Android (viz kapitolu 4.1) může, v případě komunikace s IS, rozšiřovat webová služba přistupující k databázovému serveru, často definovaná vývojářem také prostřednictvím jazyka Java. Takovou aplikaci pak lze z hlediska logiky rozdělit mezi 3 oblasti:

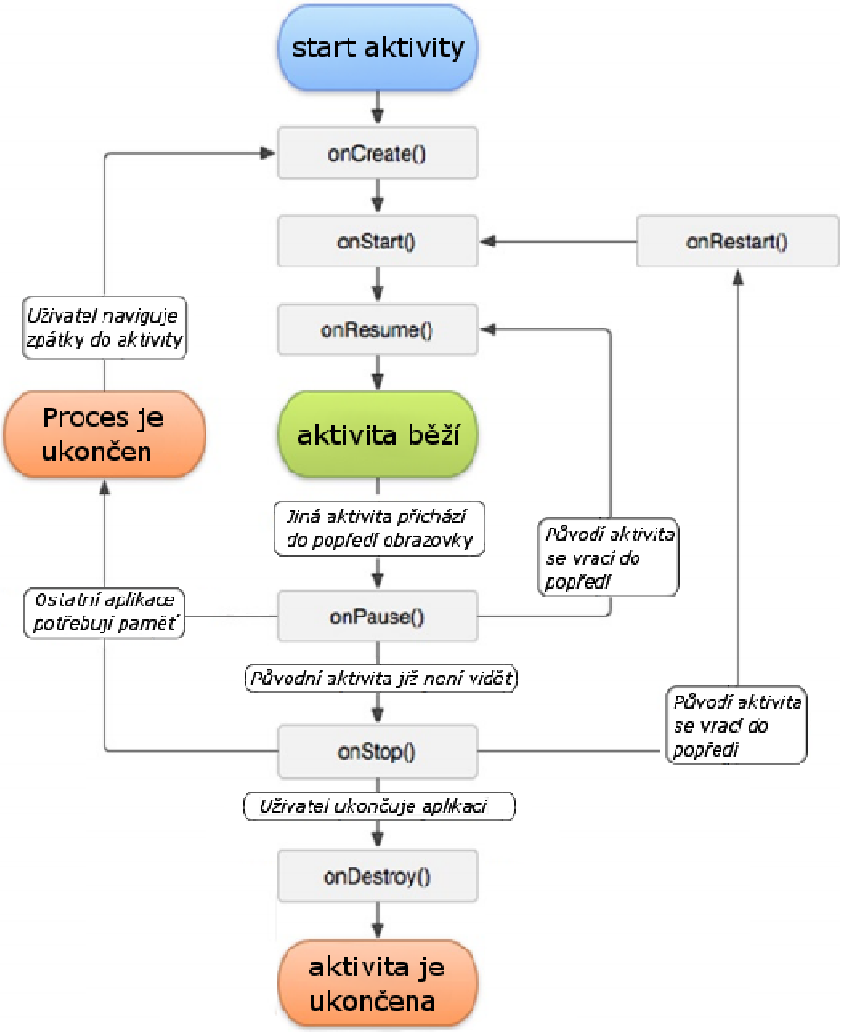
1. *Prezentační* – Tuto oblast reprezentuje část aplikace pracující na platformě Android. Komponenty zajišťující funkci prezentace jsou např. aktivity, fragmenty, view a layouty obsahující další prvky (komponenty rozvedeny v další části kapitoly).
2. *Aplikační* – Vlastní logiku aplikace definuje webová služba aplikačního serveru IS.
3. *Datová* – Přístup k datům je zajištěn díky databázovému serveru.

Architektura výše uvedené aplikace je třívrstvá. Aplikační i datová oblast však může být řešena i ve zdrojovém kódu samotné aplikace na platformě Android. A to například interakcí s funkcemi či jinými aplikacemi mobilního zařízení (platformy), a využitím poskytovatelů obsahu (rozvedeny níže).

Vývoj pro Android v jazyce Java má také své odlišnosti, specifické prvky, jejichž využití poskytuje API určené pro Android a bez kterých se nelze obejít. Nicméně z velké části se vychází ze syntaxe Javy, proto se předpokládá její základní znalost. Při vývoji pro Android lze používat i některé běžně rozšířené knihovny Java.

Mezi specifické prvky[8] utvářející aplikace pro Android patří:

- *Aktivita (activity)* – Je stavebním blokem uživatelského prostředí. Lze si ji představit jako entitu OS Android analogickou k oknu webového prohlížeče, klepnutím na definované tlačítko je možné z ní otevřít aktivitu novou a klepnutím na tlačítko Zpět se do ní zase vrátit [3]. Je možné v ní nadefinovat funkční obsahovou stránku, která je provázána s jednotlivými view v layoutu (například co se má stát po stisknutí tlačítka).



Obr. 11: Životní cyklus aktivity
Zdroj: GUI Design for Android Apps – přeloženo

[8] Pro rozlišení mezi třídou a jejími instancemi (objekty, prvky utvářející aplikaci) jsou používána malá a velká písmena (např. třída Fragment vs. fragment jako instance).

Životní cyklus aktivity je zobrazen výše, na Obr. 11, kde lze vidět metody volající se automaticky v daných situacích. Jak se zmiňují Ryan Cohen a Tao Wang [7], pokud je spuštěno více aplikací vyžadujících paměť telefonu či jiného zařízení, mohou být vynuceně ukončeny běžící procesy aktivit na pozadí (např. odesílání dat).

- *View* – Je základním prvkem pro vytváření GUI, někdy označovaný jako widget, jak ho nazývá například Allen Grant [3]. Jedná se o *viditelný* obsah, který může být zobrazen na displeji, například tlačítko, textové pole, rozbalovací seznam, oblasti s odlišným pozadím a tvarem, s textem apod.
- *Layout* – Layout je neviditelný kontejner obsahující jedno či více view. Používá se jako maska, která se přiřazuje aktivitě nebo fragmentu.
- *Fragment* – Představuje chování či část uživatelského prostředí aktivity. Musí být umístěn v aktivitě a jeho životní cyklus je přímo ovlivňován životním cyklem hostící aktivity. Díky fragmentům lze vytvořit více dynamické a flexibilní uživatelské prostředí optimalizované pro různě velké obrazovky [21].

  Typickým příkladem jejich uplatnění je přizpůsobení aplikací na chytré telefony i tablety, kdy se na každém ze zařízení může zobrazovat prostředí aplikace jinak.
- *Identifikátory* – Jsou to statické vlastnosti automaticky generované třídy R, pomocí kterých se mohu odkazovat na zdroje (prostředky) [22]. Ve zdrojovém kódu se například na soubor my_layout.xml odkazuji pomocí identifikátoru R.layout.my_layout (dle názvu adresáře a souboru) a na řetězcovou položku v souboru strings.xml pomocí odkazu R.strings.app_name (dle názvu souboru a jména položky). Nejčastěji se využívají identifikátory jako parametry v metodách. Často se ve zdrojovém kódu vyskytuje i R.id.id_daneho_view, jenž odkazuje na view uvnitř layoutu v určitém XML souboru. K tomu však musí mít dané view mezi atributy identifikační klíč, jako součást své deklarace.
- *Záměry (intenty)* – Určují komunikaci mezi prvky aplikací. Popisují akci, která má být provedena [6]. Pomocí metod třídy Intent nejčastěji ve zdrojovém kódu definujeme spouštění nové aktivity (z hlavní aktivity), které můžeme předávat parametry. Tím se většinou změní prostředí obrazovky na zařízení (mají-li aktivity jiné layouty).
- *Posluchač událostí (listener)* – Poskytuje reakci na nějakou událost. V aktivitách či fragmentech například dovoluje definovat, co se má stát po kliknutí na tlačítko nebo

view, po dlouhém přidržení prstu, po přejetí prstem do určitého směru apod., což umožňuje mnoho tříd a metod, zpravidla končících na slovo Listener.

Využitím metod třídy Intent v posluchači událostí (který je v hl. aktivitě) lze například nadefinovat spouštění nové aktivity po kliknutí na tlačítko.

- *Řešitel událostí (event handler)* – Rozšiřuje možnosti využití reakce na danou událost. V této souvislosti často pracuje ruku v ruce s posluchačem události. Dokáže se vypořádat s děním různorodé situace. Reprezentují ho metody, které lze přiřadit i vlastnoručně navrženému, specifickému view [23]. Například pomocí něj lze vyřešit, co se má dít po podržení prstu na vlastním view a zároveň co se má stát po jeho uvolnění.
- *Služba (service)* – Služby jsou procesy, které jsou v případě potřeby užívány k dlouhodobé činnosti nezávisle na aktivitách [3]. Umožňují například neustálou kontrolu stavu zařízení, přehrávání hudby na pozadí, stahování a nahrávání dat apod.
- *Poskytovatel obsahu (content provider)* – Poskytovatelé obsahu se používají jako spojovník aktivit či aplikací, kde slouží ke sdílení dat. Jak popisují Ryan Cohen a Tao Wang [7], můžeme si je představit jako databázové servery poskytující neustálý přístup k datům. Allen Grant [3] zase uvádí, že poskytovatele obsahu může reprezentovat cokoliv, například webový kanál, integrovaná SQLite databáze, kterou poskytuje samotná platforma Android, či jiné složitější řešení.

Layout a jeho view lze nadefinovat v XML souborech (ve zdrojích) nebo přímo ve zdrojovém kódu aktivity či fragmentu. Layout s view pak lze přiřadit k aktivitě (fragmentu), čímž po instalaci a spuštění aplikace vzniká GUI viditelné na obrazovce zařízení.

5. Webové služby a SOAP protokol

Komunikaci mezi aplikací (klientem) a IS umožňují webové služby (Web Services) vyměňující informace mezi klientem a serverem prostřednictvím SOAP protokolu založeném na zprávách ve formátu XML.

Brian E. Travis ve své knize [24 s. 138] vysvětluje: *„SOAP je syntaxe umožňující vytvářet aplikace pro vzdálené volání metod objektů. SOAP již nevyžaduje, aby na obou počítačích běžely stejné systémy nebo aby aplikace byly napsány ve stejném jazyku. Namísto volání metod skrze proprietární binární protokol používá balíček SOAP otevřenou standardní syntaxi pro volání metod. A tou syntaxí je XML.“* SOAP dobře definuje i Margaret Rouse ve svém příspěvku [25], kde uvádí, že SOAP je protokol umožňující zasílání a zpracování zpráv, který dovoluje programům běžícím na odlišných OS komunikovat prostřednictvím HTTP (Hypertext Transfer Protocol) a XML (Extensible Markup Language).

Webová služba může být zase vnímána jako program (aplikace) umožňující přístup k datům dostupným na webu, ke kterým může naše aplikace přistupovat jako k datům v lokálním souboru [24]. V jazyce Java ji lze vytvořit pomocí webové aplikace (v NetBeans IDE projekt Web Application) využívající operací označovaných jako webové metody.[9] Těm je možné předávat určitý parametr (vstup, request) a zpátky získávat odpověď (výstup, response). Vstup i výstup může být v jednodušším případě hodnota proměnné určitého typu (String, Integer aj.) či pole, a v tom složitějším celý objekt nebo skupina objektů.

[9] Ve zdrojovém kódu lze webovou službu rozeznat pomocí označení „@WebService“ a webovou metodu jako „@WebMethod“.

6. Aplikace Odhlašování výroby

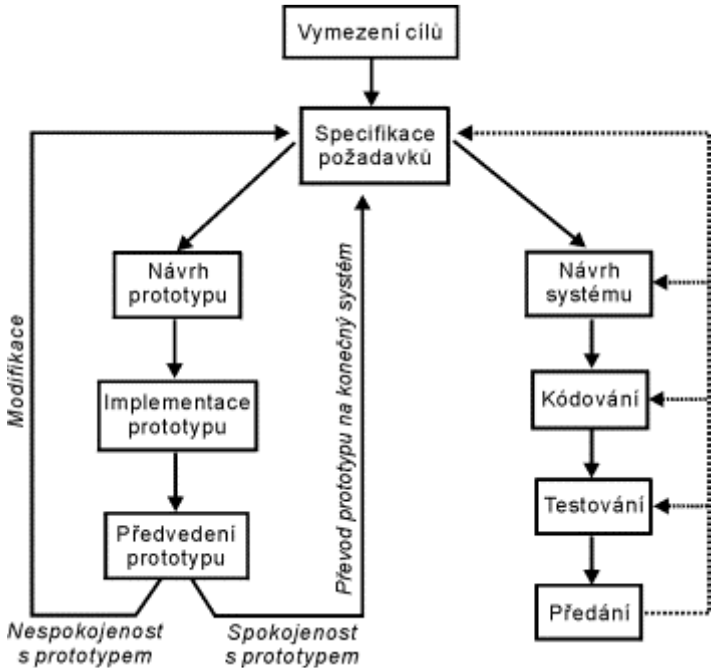
V rámci bakalářské práce byla vyvinuta aplikace Odhlašování výroby sloužící pro podporu ve výrobě, odpovídající výpočetnímu modelu klient/server. Aplikace komunikuje s IS OR-SYSTEM Open[10] vytvořeným organizací OR-CZ spol. s r.o. K realizaci projektu bylo potřeba provést specifikaci požadavku zákaznické organizace Obzor, výrobní družstvo, Plzeň. Aby mohla aplikace na platformě Android komunikovat s IS podniku, musela být vytvořena klientská a serverová část programu, dohromady vytvářející komunikační architekturu aplikace (viz Obr. 12). Aplikace byla vyvinuta ve spolupráci s Ing. Petrem Motlem, který měl na starosti část serverovou.

Obr. 12: Komunikační architektura aplikace Odhlašování výroby
Zdroj: vlastní

[10] OR-SYSTEM Open je nejnovější verzí IS OR-SYSTEM určeného pro podporu řízení podniku, který je zaměřen na výrobní a obchodní organizace (IS typu ERP).

6.1 Specifikace požadavku

Specifikaci požadavku pro tvorbu aplikace umožnily tři hlavní etapy. První etapa představuje podání požadavku a jeho základní formulaci z hlediska zákazníka. Do druhé etapy patří analýza požadavku pomocí odborných konzultací zastupitelů klientské a řešitelské firmy. A poslední třetí etapa zahrnuje prototyp, který byl představen zákazníkům, a na základě jeho odzkoušení se upřesňovalo budoucí řešení. Životní cyklus tvorby aplikace nejvíce vystihuje prototypový model znázorněný na následujícím obrázku.



Obr. 13: Prototypový model životního cyklu systému
Zdroj: http://www.fi.muni.cz/~smid/image304.gif
[vid. 2015-04-22]

6.1.1 Požadavek zákazníka

Prvotním podnětem pro vytvoření aplikace bylo zadání požadavku na odvádění výroby pomocí mobilní aplikace. Zákaznická firma popsala svůj současný stav a představu řešení.

Původní stav zákaznické firmy byl takový, že mistr musel obejít jednotlivé výrobní dílny, kde sesbíral průvodní listiny obsahující čárové kódy pro odhlášení výrobní operace, čárové kódy pracovníků a plánovaného množství. Při tom si ručně zaznamenal, které operace (činnosti) ve výrobě byly provedeny, kdo je provedl a v jak velkém množství. Po té přešel

do své kanceláře v jiné části budovy a tam provedl hlášení výkonů pracovníků pomocí snímače čárových kódů připojeného k počítači. Z průvodní listiny byly po přihlášení do IS načteny čárové kódy vykonaných operací, číselný kód pracovníka a ručně zadáno skutečné množství představující velikost rozsahu operací (v různých měrných jednotkách), tedy například obrobení kolika kusů určitých polotovarů, nastříhání kolika metrů plechu apod. Po zápisu těchto informací v IS vznikl záznam o hlášení výkonů a došlo k automatické změně příjmů a výdejů ze skladu.

Pro zlepšení situace a zjednodušení procesu pro své zaměstnance zákazník podal požadavek na aplikaci v telefonu nahrazující původní řešení, která měla navíc zobrazovat informace o operaci. Jak měl vypadat vstup a výstup na obrazovce telefonu je uvedeno v následující tabulce.

Tabulka 1: Představa zákazníka o vstupu a výstupu aplikace na obrazovce telefonu

Vstup		**Výstup**
Snímané údaje	**Ručně zadávané údaje**	Zakázka, jméno odběratele zakázky, řádek zakázky, název operace, typ stroje, název stroje, typ výrobku, název výrobku, měrná jednotka a plánované množství.
Čárový kód pracovníka a operace.	Skutečně vyrobené množství.	

Zdroj: vlastní

6.1.2 Analýza požadavku

Pro analýzu požadavku byly provedeny konzultace, specifikace potřeb, dotazů a řešení dalších nabízejících se otázek. Důležitý poznatek byl, že v klientské firmě existují výrobní dílny, kde nechtějí zprostředkovat Wi-Fi pokrytí. Bylo by to příliš nákladné, v halách je spousta kovu a pokrytí by muselo být řešeno velmi složitě a nákladně, aby kov nepohlcoval signál.

Na základě konzultace pověřená organizace, OR-CZ spol. s r.o., stanovila dvě hlavní varianty režimu funkčnosti aplikace:

- On-line fungování aplikace prostřednictvím sítě mobilního operátora (datový tarif) a připojení do lokální sítě zákazníka pomocí VPN (Virtual Private Network).
- Off-line režim provozu programu. Po zadání vstupu by aplikace uchovala data, a po návratu do kanceláře nebo jiné části areálu s pokrytím Wi-Fi by se provedla synchronizace dat.

Poté byla ve výrobní hale prověřena dostupnost sítě firmy O2, která je dodavatelem zákazníka. Dostupnost sítě byla dostačující, proto byla vybrána první varianta on-line režimu s tím, že aplikace poběží na OS Android, a to díky jeho rozšíření a dostupnosti (viz kapitolu 2). Kvůli autentizaci bylo stanoveno, že musí aplikace používat ke spojení s IS uživatelský login a heslo. Taktéž měla aplikace navracet zpětnou vazbu s případnými upozorněními pro uživatele (špatné heslo, login, neexistující operace apod.) a automaticky se po spuštění připojovat k firemní VPN.

Pověřená organizace nabídla jako průmyslový přístroj do výroby zařízení Motorola TC55, které bylo nakonec odkoupeno díky svým parametrům a možnosti rychlého lineárního skenování (parametry telefonu rozvedeny v kapitole 6.5).

6.1.3 Prototyp

Pro dodatečné upřesnění požadavku zákazníka byl navržen a vytvořen prototyp aplikace. Zákazník si tak mohl vyzkoušet chování aplikace nad demonstračními daty. Dohromady bylo vytvořeno 7 verzí prototypu. Na základě zpětné vazby zákazníka má výsledná aplikace doplňující funkce v nastavení (viz kapitolu 6.2.2) a jak lze vidět v následující tabulce, změnil se vstup i výstup aplikace. Díky tomu je umožněno zadávání údajů ručně i snímačem.

Tabulka 2: Konečný vstup a výstup aplikace na obrazovce telefonu

Vstup		**Výstup**
Snímané údaje	**Ručně zadávané údaje**	Pracovník (vykonavatel operace), zakázka, řádek zakázky, operace, název výrobku, plánované množství a skutečné množství (s měrnou jednotkou).
Čárový kód pracovníka, operace či množství.	Čárový kód pracovníka, operace či množství.	

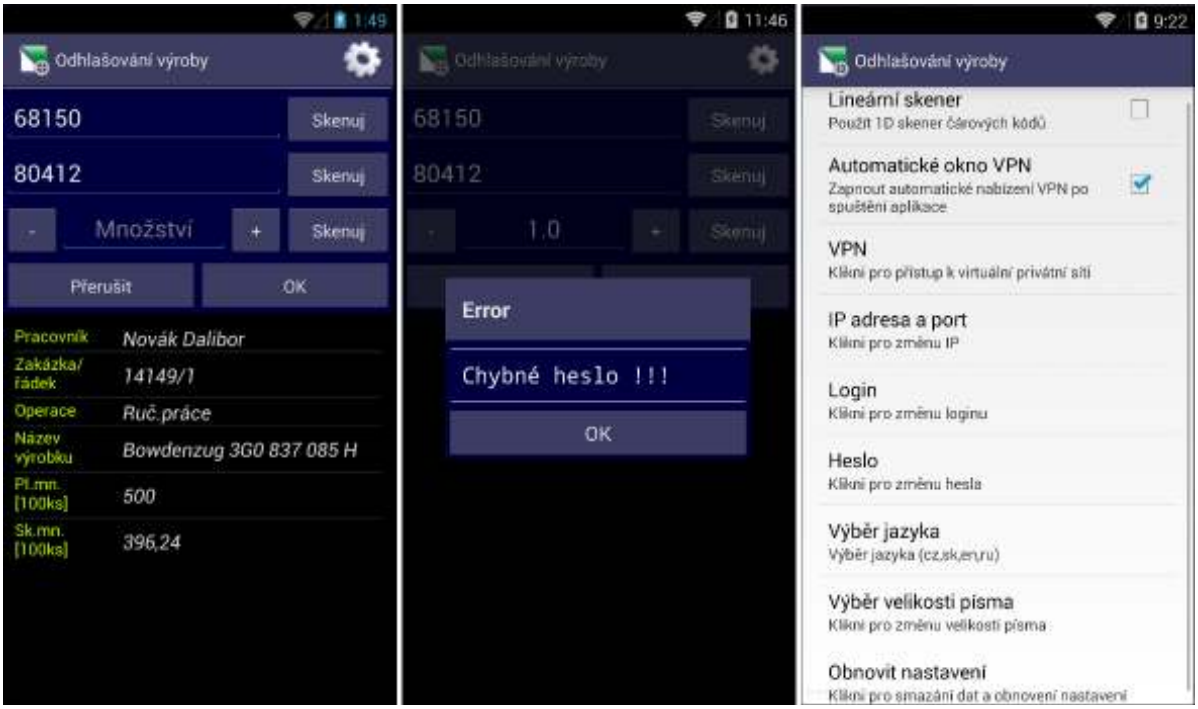
Zdroj: vlastní

Automatické připojení k VPN zákaznické firmy aplikace však nepodporuje, neboť platforma Android zatím neumožňuje zacházení s třídami, které mají na starost ovládání, vytváření a připojení k VPN (jedná se o interní skryté třídy).

Po schválení poslední verze prototypu byla aplikace rozšířena o komunikaci s webovou službou. Návrh systému předávání dat prostřednictvím SOAP protokolu byl již předem určen, vycházelo se ze starších vytvořených projektů. Testování probíhalo průběžně během kódování aplikace i po něm. Po předání aplikace ještě proběhlo pár nepatrných úprav (např. možnost nahlášení skutečného množství nad plánované). Ty však neměly žádný zásadní vliv na změnu stávajícího řešení. Popis chování výsledné aplikace na platformě Android z hlediska uživatele zachycují diagramy případů užití v kapitole 6.2.

6.2 Charakteristika a funkce vytvořené aplikace

Aplikace Odhlašování výroby podporuje platformy s API 11 a vyšší. Důvodem je používání jistých metod, které starší API nepodporují. Klientská část komunikační architektury vytváří *prezentační* oblast aplikace a v serverové části je uplatněna oblast *aplikační* a *datová* (viz kapitolu 4.2). Bezdrátové odesílání dat mezi oběma částmi je v zákaznické firmě umožněno pomocí datových služeb mobilního operátora[11], aplikace může však s IS komunikovat i prostřednictvím Wi-Fi. V souvislosti s touto komunikační architekturou může být Android aplikace na konkrétním mobilním zařízení brána jako klient IS. Na následujícím obrázku je uvedeno uživatelské prostředí aplikace, lze zde vidět tři různé stavy jednoho displeje.

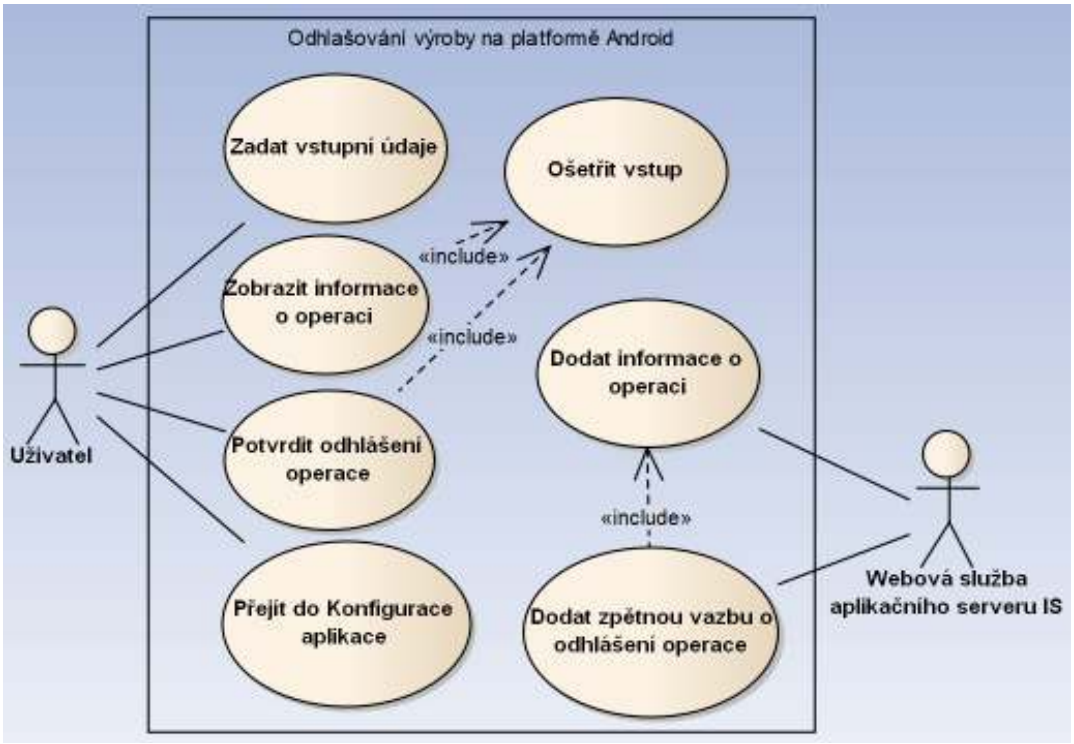


Obr. 14: Uživatelské prostředí aplikace Odhlašování výroby
Zdroj: vlastní

[11] Díky přenosu malého objemu dat je možné používat i méně výkonné sítě, např. u zákaznické firmy využívají 3G (Third Generation Wireless).

6.2.1 Základní funkcionalita

Hlavním cílem aplikace je ohlášení vykonaných operací (činností) ve výrobě do IS OR-SYSTEM Open. Operaci představuje například obrobení polotovarů, ruční práce, stříhání plechu apod. Na následujícím diagramu lze vidět funkcionalitu klientské části, ke které mimo uživatele přistupuje i část serverová (webová služba).



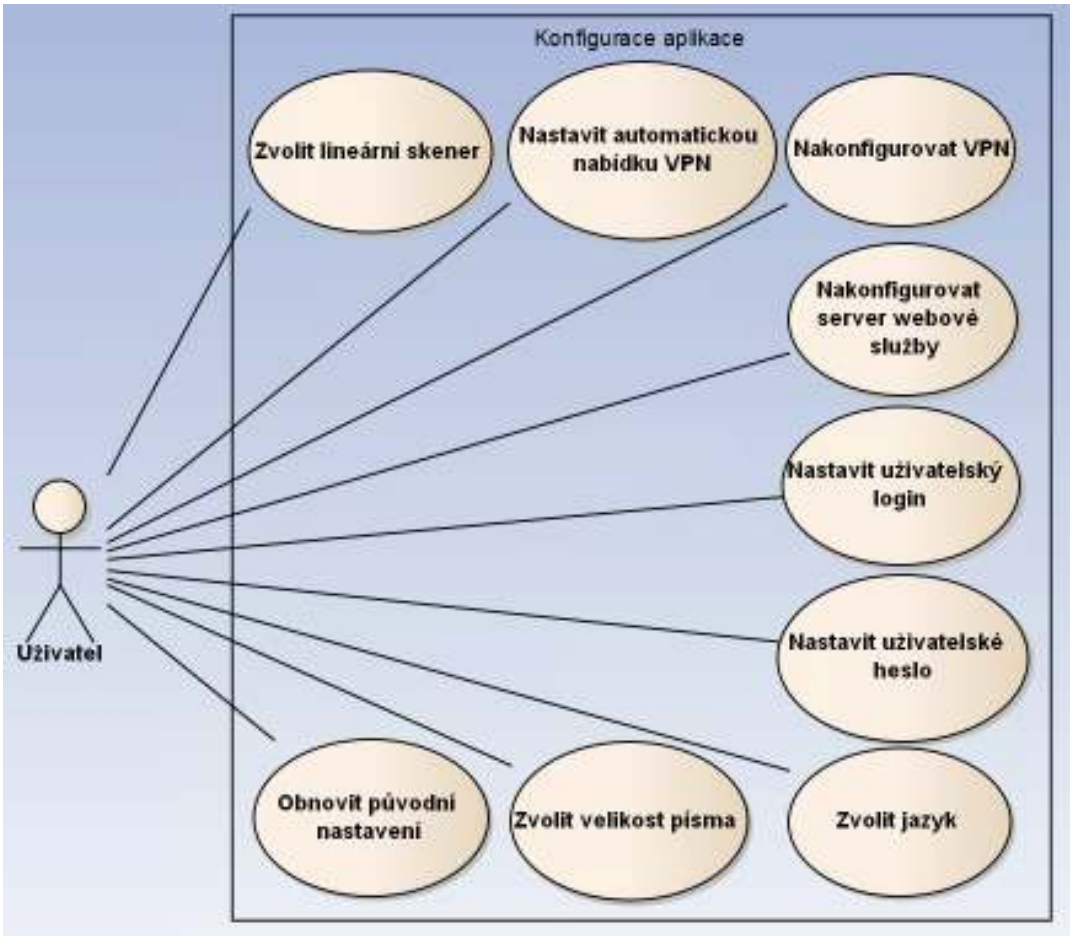
Obr. 15: Diagram případů užití aplikace Odhlašování výroby na platformě Android
Zdroj: vlastní

- *Zadat vstupní údaje* lze ručně nebo skenováním čárových kódů operace, pracovníka a množství (kamerou mobilního zařízení či lineárně laserem).
- *Zobrazit informace o operaci* lze stiskem klávesového tlačítka Enter, čímž se zavolá metoda webové služby, která *dodá informace o operaci*. Není-li kurzor v textovém poli Množství, tak stisk tlačítka Enter způsobí zaměření kurzoru na textové pole níže.
- *Potvrdit odhlášení operace* je možné stiskem tlačítka OK. Tím se zavolá další metoda webové služby, která *dodá zpětnou vazbu o odhlášení operace* i aktualizované informace o operaci. Na displeji se tedy zobrazí potvrzení o zápisu operace a aktualizované informace, nebo určitá chybová hláška (viz Obr. 14 uprostřed). Obě webové metody společně s klientskou částí *ošetřují vstup* od uživatele.

- *Přejít do Konfigurace aplikace* může uživatel stiskem obrázku ozubeného kola na výchozí obrazovce vpravo nahoře (viz Obr. 14).

6.2.2 Konfigurace aplikace

Položky konfigurace aplikace (nastavení) jsou zobrazeny na Obr. 14 vpravo. Následující diagram znázorňuje chování systému v konfiguraci z hlediska uživatele.



Obr. 16: Diagram případů užití pro konfiguraci aplikace
Zdroj: vlastní

- *Zvolit lineární skener* lze pouze, podporuje-li jej platforma. Zatržením odpovídající položky v konfiguraci se zapne možnost lineárního skenování čárových kódu (laserem) a vypne skenování prostřednictvím kamery.
- *Nastavením automatické nabídky VPN* se rozumí nabízení připojení k VPN ihned po zapnutí aplikace. To je dosaženo prostřednictvím dialogového okna odkazujícího na aktivitu platformy umožňující přístup k VPN.

- *Nakonfigurovat VPN* uživatel může kliknutím na položku VPN, která ho odkáže na stejnou aktivitu jako dialogové okno v předešlém případě.
- *Nakonfigurovat server webové služby* lze zadáním IP (Internet Protocol) adresy a portu. Pro komunikaci s webovou službou však musí uživatel ještě *nastavit uživatelský login a heslo.*
- *Zvolit jazyk* si uživatel může z nabídky češtiny, angličtiny, slovenštiny a ruštiny. Taktéž lze *zvolit velikost písma* (málé, normální, velké) a *obnovit původní nastavení*, což obnoví výchozího stav konfigurace a smaže zapsaná data (IP adresa a port, login aj.).

6.3 Klientská část

Reprezentuje ji projekt, jehož struktura zahrnuje důležité soubory, zdroje, knihovny a třídy se zdrojovým kódem (např. aktivity, fragmenty), který je následně zkompilován a vznikne z něj část aplikace využitelné na hmatatelném zařízení, pomocí níž uživatel komunikuje s IS. Zatupuje tedy samotnou aplikaci na platformě Android, jako klienta IS.

6.3.1 Aktivity a fragmenty

Přístup ke GUI a komunikaci uživatele s aplikací umožňuje pět tříd zahrnujících tři aktivity a dva fragmenty:

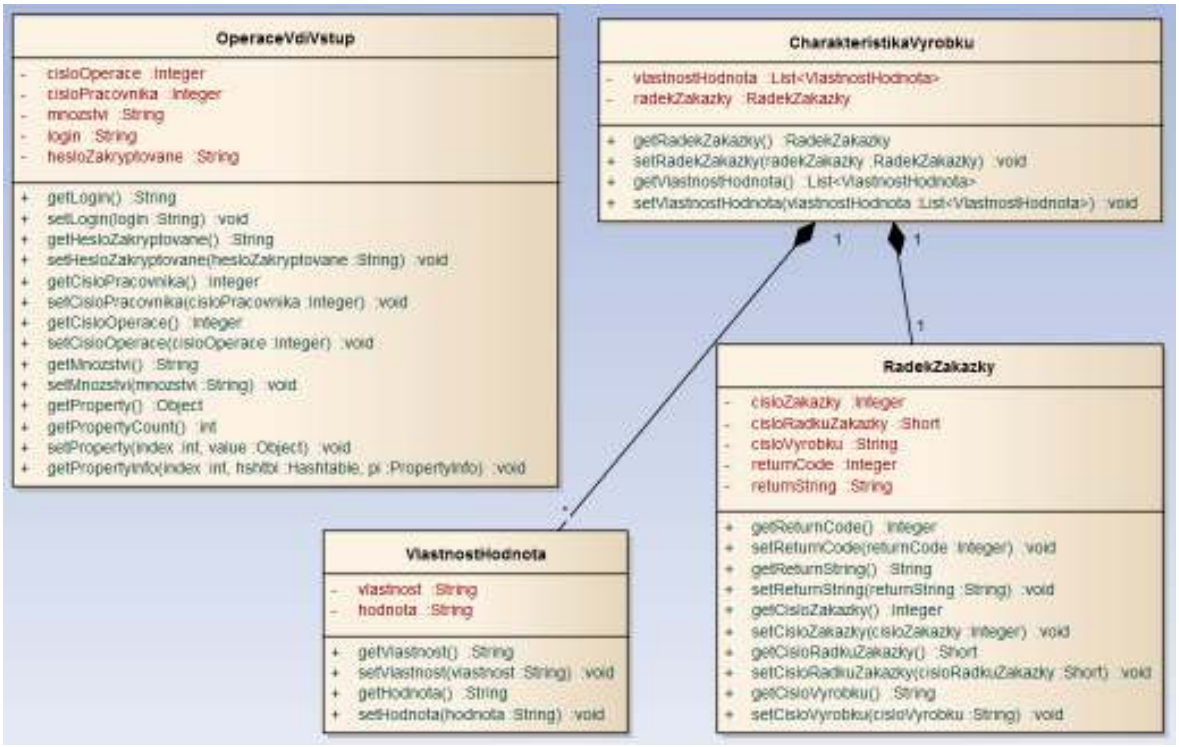
- *OdhlasovaniVyrobyActivity* – Klíčová třída s nejrozsáhlejším kódem definující aktivitu, která se vytváří ihned po zapnutí aplikace. Poskytuje základní funkcionalitu aplikace (viz kapitolu 6.2.1), přebírá a ošetřuje vstup od uživatele, umožňuje skenování čárových kódů (využita knihovna *android-zxinglib*), spouští (vytváří) fragment zobrazující výstup aplikace a odkazuje na ostatní aktivity. Navíc filtruje sekvenci vstupních znaků tak, že uživateli umožňuje zadat pouze znaky od 0 do 9 (u textového pole s množstvím i znak tečky).
- *EditPreferencesActivity* – Spouštěna z klíčové aktivity jako záložka s nastavením. Obsahuje fragment poskytující přístup ke konfiguraci aplikace (viz kapitolu 6.2.2).
- *CaptureActivity* – Spouštěna z klíčové aktivity tlačítkem *Skenuj*. Slouží pro práci se snímačem kódu. Je součástí knihovny *android-zxinglib.*
- *OutputFragment* – Poskytuje informace zobrazované jako výstup aplikace v dolní části obrazovky telefonu či tabletu (pracovník, zakázka, operace aj.). Kvůli zobrazení výstupu nahrazuje prázdný layout, zpracovává vstup od uživatele a využívá metody klientské třídy *ServiceClient* umožňující komunikaci s IS.
- *EditPreferencesFragment* – Spouští se ihned po vytvoření *EditPreferencesActivity* a nahrazuje její obsah. Třída je provázána s prvky souboru *preferences.xml* a implementuje určitá rozhraní, díky čemuž lze definovat konfiguraci aplikace rozvedenou v kapitole 6.2.2.

6.3.2 Třídy pro podporu komunikace s webovou službou

Propojení s webovou službou je na klientské části umožněno pomocí pěti tříd:

- *ServiceClient* – Definuje komunikaci se serverem (provázání s webovými metodami), k čemuž využívá knihovnu *ksoap2* s využitím SOAP protokolu. Třída obsahuje dvě metody spouštějící asynchronní úlohy, které běží paralelně s hlavním vláknem programu, odesílají a získávají data ze serveru. Jedna metoda umožňuje zobrazení informací o operaci ve výrobě a pracovníkovi, a cílem druhé metody je umožnit aktualizaci dat v databázi a tím nahlásit provedenou operaci (obrobení deseti polotovarů, ruční práce apod.), a zobrazit aktualizované informace.
- *OperaceVdiVstup* – Vstupní třída jejíž objekt se přenáší na webovou službu. Ten může nést hodnoty více proměnných (login, šifrované heslo, množství, číslo operace a pracovníka). Díky využití rozhraní z knihovny *ksoap2* uchovává informace o proměnných a jejich vlastnostech (počet, datový typ, index aj.).
- *CharakteristikaVyrobku* – Kopie výstupní třídy, jejíž objekt se přenáší z webové služby zpět do klientské aplikace. Její přítomnost v projektu aplikace je povinná dle náležitostí využití knihovny *ksoap2* a kvůli potřebě pracovat se získaným objektem.
- *RadekZakazky* – Její instanci jsou z objektu třídy *CharakteristikaVyrobku* přijatého z webové služby přiřazeny informace o zakázce a zpětná vazba, díky níž může aplikace v případě potřeby zobrazovat uživateli určitá upozornění pomocí dialogových oken (pokud byla operace ukončena, neexistuje zakázka apod.).
- *VlastnostHodnota* – Třída jejíž objekty nesou informace o operaci (popř. i jméno pracovníka) získané z objektu *CharakteristikaVyrobku* webové služby. Jde o informace zobrazující se jako výstup aplikace.

Vstupní a výstupní třídu lze vidět v následujícím diagramu tříd. Instance třídy *OperaceVdiVstup* se prostřednictvím SOAP protokolu odesílá na webovou službu, ze které se přijímá instance třídy *CharakteristikaVyrobku* skládající se ze tříd *RadekZakazky* a *VlastnostHodnota.*



Obr. 17: Diagram tříd zobrazující vstupní a výstupní třídu
Zdroj: vlastní

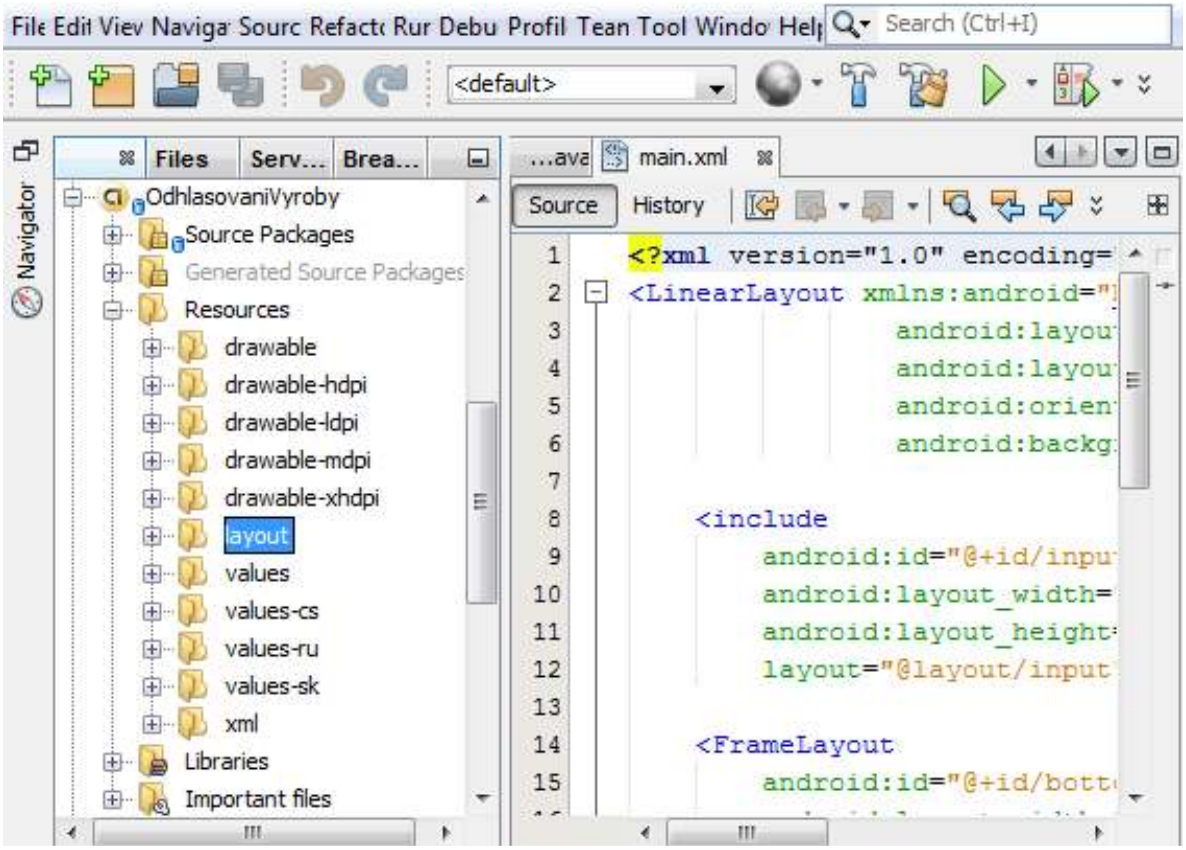
6.3.3 Ostatní třídy

Aplikace obsahuje ještě další 3 třídy s odlišným účelem. Mezi ně patří:

- *CustomDialog* – Umožňuje vlastní vzhled dialogového okna. Upozornění pomocí dialogových oken jsou uplatněny ve třídách *OdhlasovaniVyrobyActivity*, *ServiceClient* a *EditPreferencesFragment*.
- *O0Hash* – Pro zpracování a šifrování uživatelského hesla. Třída byla poskytnuta z jiných projektů organizace OR-CZ spol. s r.o.
- *AppStatus* – Umožňuje kontrolu sítě, tedy zdali je datové připojení k dispozici.

6.3.4 Zdroje, knihovny a využití třetích stran

Rozdělení zdrojů ve struktuře projektu je zobrazeno na Obr. 18 (adresář Resources).



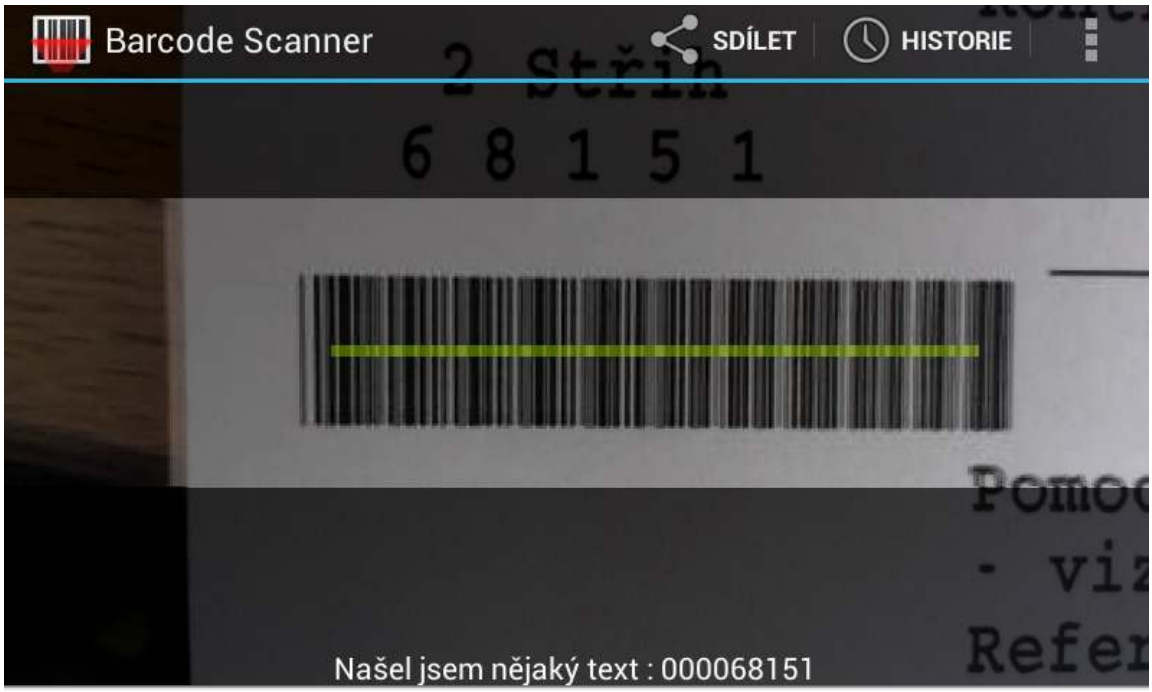
Obr. 18: Struktura zdrojů aplikace Odhlašování výroby
Zdroj: vlastní

Jednotlivé zdroje jsou rozděleny do několika adresářů:

- *Drawable* – Obsahuje soubory ve formátu XML definující vzhled textových polí.
- *Drawable s příponou* – Obsahují ikonu aplikace a ikonu nastavení v odlišných velikostech (pro vysvětlení přípon viz kapitolu 4.1).
- *Layout* – Uchovává masky (layouty), které určují vzhledovou stránku pro třídy *OdhlasovaniVyrobyActivity*, *CaptureActivity*, *OutputFragment* a *CustomDialog*.
- *Values* – Zde jsou uloženy XML soubory obsahující definici barev, textových řetězců a řetězcových polí v anglickém jazyce. Navíc obsahuje soubor *theme.xml* určující téma aplikace, tedy její základní vzhled (to zahrnuje horní menu, titulek, základní dialogová okna aj.).

- *Values s příponou* – Kromě souboru *theme.xml* obsahují to samé jako adresář *values*, orientované jsou však na jiné jazyky (dle rozdělení v kapitole 4.1).
- *Xml* – Uchovává soubor určující položky v konfiguraci aplikace (v nastavení), který je spojený s fragmentem *EditPreferencesFragment*.

V aplikaci Odhlašování výroby jsou používány i volně dostupné knihovny třetích stran. Pro podporu webových služeb se využívá knihovny *ksoap2* nabízející řešení komunikace klientské a serverové části pomocí SOAP protokolu. Skenování čárových kódů kamerou mobilního zařízení je zase umožněno díky bezplatné aplikaci *Barcode Scanner*, která se v Odhlašování výroby spouští ihned po kliknutí na tlačítko *Skenuj*.[12] Pro využití této volně dostupné aplikace tvůrci vytvořili knihovnu *android-zxinglib*, díky čemuž lze její funkci skenování využít i ve vlastní aplikaci (viz následující obrázek).



Obr. 19: Obrazovka skenování spouštěná z aplikace Odhlašování výroby
Zdroj: vlastní

[12] Pokud není aplikace Barcode Scanner na zařízení nainstalována, zobrazí se okno nabízející její instalaci.

6.4 Serverová část

Serverová část zahrnuje webovou službu aplikačního serveru IS OR-SYSTEM Open a její dvě metody (pro obecný popis webových služeb a webových metod viz kapitolu 5).

Webová služba na straně serveru komunikuje s databází IS, ze které získává informace o operacích, přístupových údajích a o pracovnících. Stejně jako pro klientskou část byl pro její vývoj použit jazyk Java (projekt Web Application). Má za úkol reagovat na podněty zadávané uživatelem a odesílané prostřednictvím zařízení. Podle nich získává data z databáze díky komunikaci s daným IS (popřípadě je dán příkaz k zápisu dat, aktualizaci apod.). Výsledná data se zpětnou vazbou odešle zpět na konkrétní mobilní zařízení a to se zachová podle nabytých údajů (např. zobrazí informace o operaci, dialogové okno s upozorněním). Toho dosahuje pomocí pěti tříd a dvou webových metod.

6.4.1 Třídy pro podporu komunikace mezi mobilním klientem a IS

Komunikace mobilního zařízením (klienta IS) s IS je na serverové části umožněna pomocí pěti tříd:

- *OrAndroidSrv* – Třída nesoucí označení webová služba, ve které jsou definovány 2 webové metody přebírající z klientské části objekt třídy *OperaceVdiVstup* a navracejí objekt třídy *CharakteristikaVyrobku*. První webová metoda s názvem *getOperaceVdi* vrací informace o operaci a pracovníkovi, hledané v databázi IS podle čísla operace (pracovníka). Druhá metoda *provedHlaseniVdi* zapisuje nově provedené operace a množství v jakém byly provedeny, a navrací aktualizované informace o operaci a pracovníkovi. Obě metody také provádí logické kontroly a v případě konfliktu vrací návratový kód s chybovým hlášením. Kontrolováno je heslo, login, čísla operací (pracovníků) a vyrobené množství s údaji v databázi.
- *OperaceVdiVstup* – Odpovídá stejnojmenné vstupní třídě z klientské části programu, jejíž objekt je přenášen na webovou službu. Nevyužívá však knihovny *ksoap2*, a proto neobsahuje některé její metody.

- *CharakteristikaVyrobku* – Výstupní třída. Její objekt se přenáší z webové služby zpět do klienta a s sebou nese jeden objekt typu *RadekZakazky* a několik objektů typu *VlastnostHodnota*.
- *RadekZakazky* – Instanci této třídy se ve webových metodách přiřazují informace o zakázce a jejím stavu. Obsahuje také návratový kód (jako hodnotu proměnné typu String), díky kterému se může na mobilním zařízení zobrazit zpětná vazba uživateli prostřednictvím dialogového okna (potvrzení o nahlášení výkonu, ukončení operace, neexistující zakázka, k operaci nevydán materiál apod.).
- *VlastnostHodnota* – Třída jejíž objekty jsou využívány pro přenos údajů o operaci (pracovníkovi), které jsou později zobrazeny na displeji mobilního zařízení (název operace, zakázka, plánované množství, jméno pracovníka atd.).

6.5 Motorola TC55

Pro co nejefektivnější využití aplikace Odhlašování výroby v praxi bylo vybráno mobilní zařízení Motorola TC55. Jedná se o průmyslové zařízení s možností lineárního skenování prostřednictvím laseru, které je dodáváno i s OS Android.[13] Oproti skenování pomocí kamery telefonu je lineární skenování spolehlivější, rychlejší a funguje i za snížené viditelnosti. Mimo to má zařízení dostatečné parametry pro používání ve výrobních dílnách. Jak uvádí zdroj českého dodavatele [26], displej telefonu lze ovládat prstem v rukavicích a funguje i za mokra. Zařízení obsahuje velkokapacitní baterii (možnost odkoupit i její rozšířenou verzi s ještě delší životností) a je dostatečně odolné proti nárazu či pádu (navíc lze dokoupit gumový návlek pro ještě lepší ochranu).

Kvůli možnosti lineárního skenování však telefon potřebuje pro aplikaci Odhlašování výroby speciální nastavení. Toho docílíme pomocí DataWedge, což je aplikace dodávaná v základním vybavení telefonu, prostřednictvím které lze nastavit lineární skenování tak, aby skenované údaje po spuštění obdržela právě naše aplikace.

[13] Mezi průmyslová zařízení dodávaná s OS Android dále patří např. Motorola MC32N0 a Honeywell Dolphin 7800. Většina ostatních průmyslových zařízení není zatím dodávána se systémem Android.

6.5.1 Nastavení DataWedge

Aby se skenované údaje správně převáděly po spuštění aplikace do jejích textových polí, musí se v DataWedge vytvořit nový profil, který je nutno nastavit podle příručky [27]:

1. Nejdříve je potřeba zapnout aplikaci DataWedge a vytvořit nový profil (*New profile*), obvykle pomocí hardwarového tlačítka menu, a po té profil pojmenovat jako „orcz“.
2. Profil se musí kliknutím upravit (či pomocí dlouhého podržení a *Edit profile*).
3. Dále je nutno otevřít položku *Associated apps*, kliknout na tlačítko menu a vybrat *New app/activity*.
4. Ze zobrazených aplikací se musí vybrat ta, u které je uvedeno *cz.orcz.ov* a poté aktivita *cz.orcz.ov.OdhlasovaniVyrobyActivity*.
5. Nyní je potřeba se vrátit zpět do editace profilu a zrušit *KEYSTROKE OUTPUT*. Položka *Enabled* se u něj tedy musí nastavit tak, aby *nebyla* zatržená.
6. Po té se musí zatrhnout *Enabled* pod nadpisem *INTENT OUTPUT* a u něj do položky *Intent action* vepsat „cz.orcz.ov.RECVR“ a do položky *Intent category* „android.intent.category.DEFAULT“.
7. Nakonec je potřeba zapnout aplikaci Odhlašování výroby a v nastavení zatrhnout položku *Lineární skener*.

6.6 Zhodnocení přínosu řešení

Původní proces hlášení výkonů (odhlašování operací) ve výrobě byl neefektivní a zdlouhavý (viz kapitolu 6.1.1). Výkony se zaznamenávaly ručně pomocí průběžných poznámek a do IS se zapisovaly nepravidelně, přibližně dvakrát do měsíce. Byla potřeba zlepšit a zrychlit komunikaci s IS.

6.6.1 Stav po zavedení mobilního řešení

S vytvořením zakázky se u zákazníka vytvoří i operace v IS OR-SYSTEM Open. Aplikace Odhlašování výroby umožňuje, že po načtení čárového kódu operace se informace o ní zobrazí na obrazovce mobilního zařízení. Čárový kód lze načíst pomocí lineárního skeneru, pomocí kamery nebo ho lze zadat ručně. Po doplnění kódu pracovníka a množství lze operaci potvrdit. Pokud tak uživatel učiní a logické kontroly vstupních údajů proběhnou v pořádku, vznikne v IS záznam o hlášení výkonů a provedou se případné automatické skladové transakce. Na obrazovce telefonu se ukáže potvrzení v podobě dialogového okna a aktualizuje se skutečné množství oproti plánovanému.

6.6.2 Přínosy pro zákazníka a řešitele

U zákazníka došlo zavedením mobilní aplikace ke zrychlení a zjednodušení činnosti hlášení výkonů, které lze provést přímo ve výrobních halách, okamžitě po provedení určité operace. Díky tomu není potřeba dělat si žádné poznámky o výkonech pracovníků a nosit listiny do kanceláře. Navíc si mohou zaměstnanci v případě potřeby vzít seznamy k odhlašování domů a pracovat odtud. Konstrukce aplikace umožňuje mistrovi procházení jednotlivých operací a tím průběžnou kontrolu stavu zakázek. Další výhodou je získání okamžité zpětné vazby z výroby a zpřesnění údajů v IS, který díky rychlému procesu hlášení obsahuje aktuální data o provedených operacích, což mu dovoluje ihned automaticky měnit příjem a výdej ze skladu. Komunikace s IS prostřednictvím sítě mobilního operátora O2 je navíc mnohem méně nákladnou variantou než by bylo zavádění Wi-Fi pokrytí ve výrobních dílnách plných kovu (odhadem pověřené firmy v řádech desetitisíců).

Přínosem pro řešitelskou firmu OR-CZ spol. s r.o. je do budoucna možnost implementace mobilního řešení hlášení výkonů u dalších zákazníků. Odhlašování výroby je také první mobilní aplikací řešitele, ve které byla uplatněna možnost dvojího výběru mezi skenováním, možnost okamžité změny jazyka a písma v konfiguraci, obnovení nastavení, přizpůsobení vlastního vzhledu dialogového okna a textových polí, využití vlastního rozhraní pro možnost zobrazení detailu fragmentu s informacemi o operaci, filtrování vstupní sekvence znaků z klávesnice, inkrementální zvyšování nebo snižování množství pomocí tlačítek, a mnoho dalších vymožeností, které lze využít při vývoji budoucích firemních aplikací.

6.6.3 Ekonomické zhodnocení[14]

Měsíční náklady zákazníka za datový tarif od O2 jsou 65 Kč, normálně by však činily cca 1500 Kč. Zákaznická firma je ale členem Svazu českých a moravských výrobních družstev sdružující skupinu výrobních družstev, které mají podle zákona 435/2004 Sb. o zaměstnanosti u zaměstnavatelů s vyšším jak 50% podílem spolupracovníků se zdravotním postižením nárok na speciální podmínky a slevy.

IS OR-SYSTEM Open v zákaznické firmě už zavedený byl, náklady na pořízení telefonu Motorola TC55, aplikace a vlastní zavedení řešení tak byly řádově 80 tisíc Kč. Kalkulace návratnosti investic prováděna nebyla, zákazník s ní nepočítá. Řešení odhlašování výroby prostřednictvím mobilní aplikace bylo provedeno pro zjednodušení procesu hlášení tak aby bylo možné zaměstnávání handicapovaných pracovníků a kvůli získání okamžité zpětné vazby z výroby.

[14] Náklady a návratnost investic řešitele nejsou v kapitole uvedeny s ohledem na politiku firmy.

Závěr

Hlavní cíl bakalářské práce, vývoj aplikace komunikující s IS podniku, byl splněn, aplikace byla úspěšně vytvořena. Mezi další stanovené úkoly práce se řadí popis trendů a rizik souvisejících s užíváním mobilních zařízení, popis a přizpůsobení NetBeans IDE pro možnost vývoje pro OS Android s postupem tvorby jednoduché aplikace, specifikace vývoje aplikací pro Android, popis analýzy požadavku zákaznické firmy a na jejím základě vytvoření aplikace pro OS Android komunikující s IS podniku, a zhodnocení přínosu řešení. Všechny tyto úkoly byly také provedeny, lze tedy říci, že byly cíle práce splněny.

Zavedení aplikace Odhlašování výroby sloužící pro podporu ve výrobě umožnilo zrychlení a zjednodušení procesu hlášení výkonů pracovníků, což má pozitivní dopady na zákaznickou firmu i její zaměstnance. Při vývoji aplikace došlo k jedné odchylce od původního plánu, nebylo možné zařídit funkci automatického připojování k zákaznické VPN po spuštění. Pokud by se v budoucnu zpřístupnily třídy umožňující konfiguraci VPN, šlo by funkci doplnit. Zpřístupnění těchto tříd by ocenilo i mnoho dalších programátorů. Je obecně známo, že se některým podařilo obejít opatření a dokázali využít interních tříd v osobní prospěch, jedná se však o porušení práv.

V zákaznické firmě byla aplikace zavedena do provozu v lednu tohoto roku a jejím prostřednictvím bylo od začátku ledna do konce března odhlášeno 17082 operací. Uživatelé jsou s výsledkem spokojeni, další změny nejsou zatím požadovány.

Zdrojový kód aplikace Odhlašování výroby využil autor textu i u jiných mobilních aplikací řešitelské firmy a vycházet z něj budou jistě i další projekty. Okamžitá změna jazyka, obnovení nastavení či jiné prvky byly například využity při vývoji aplikace Virtuální docházkový snímač, umožňující evidenci docházky u externích pracovníků, či u aplikace Notes, která je reklamním poznámkovým blokem určeným pro zaměstnance i širokou veřejnost.

Všechny znalosti a zkušenosti získané během tvorby aplikace jsou jistě významným aktivem pro osobnostní i budoucí kariérní rozvoj. Hlavním přínosem pro autora tak zůstává schopnost vytvářet moderní aplikace pro OS Android s propojením se složitějšími podnikovými systémy.

Citace

[1] Android, the world's most popular mobile platform. *Android Developers* [online]. 2012 [vid. 2015-01-15]. Dostupné z: http://developer.android.com/about/index.html

[2] ELIAS, Paul. Samsung, Apple Both Infringed on Smartphone Patents. *Claims Journal - Insurance news and resources for the claims industry* [online]. 2014 [vid. 2015-01-15]. Dostupné z: http://www.claimsjournal.com/news/national/2014/05/05/248430.htm

[3] ALLEN, Grant. *Android 4: průvodce programováním mobilních aplikací.* Překlad Jakub Mužík. Brno: Computer Press, 2013, 656 s. ISBN 978-80-251-3782-6.

[4] BEAL, Vangie. What is Application Program Interface (API)? Webopedia. In: *Webopedia: Online Tech Dictionary for IT Professionals* [online]. © 2015 [vid. 2015-04-14]. Dostupné z: http://www.webopedia.com/TERM/A/API.html

[5] <uses-sdk>: What is API Level?. *Android Developers* [online]. 2014 [vid. 2015-01-21]. Dostupné z: http://developer.android.com/guide/topics/manifest/uses-sdk-element.html#ApiLevels

[6] MEIER, Reto. *Professional Android 4 application development.* Updated for Android 4. Indianapolis: John Wiley, 2012, xlii, 817 p. ISBN 978-111-8262-153.

[7] RYAN, Cohen a Tao WANG. *GUI Design for Android Apps.* New York: Apress, 2014. ISBN 978-1-4842-0383-5.

[8] Android - History. *Android* [online]. 2014 [vid. 2015-01-21]. Dostupné z: http://www.android.com/history/

[9] HILDENBRAND, Jerry. Inside the different Android Versions. In: *Android Central* [online]. 2013 [vid. 2015-01-21]. Dostupné z: http://www.androidcentral.com/android-versions

[10] Android Lollipop. *Android Developers* [online]. 2014 [vid. 2015-01-21]. Dostupné z: http://developer.android.com/about/versions/lollipop.html

[11] TEJASJASANI. 5 Trends That Will Dominate Mobile App Development In 2015. In: *SAP Blogs: The Best-Run Businesses Run SAP* [online]. 2014 [vid. 2015-04-14].

Dostupné z: http://blogs.sap.com/innovation/mobile-applications/5-trends-that-will-dominate-mobile-app-development-in-2015-01838179

[12] MOWAT, Jon. 7 Predictions On Mobile Marketing Trends In 2015 You Need To Know. In: *Business 2 Community - Top Trends, News & Expert Analysis* [online]. 2015 [vid. 2015-04-14]. Dostupné z: http://www.business2community.com/mobile-apps/7-predictions-mobile-marketing-trends-2015-need-know-01166049

[13] REDAKCE2. Mobilní zařízení v byznysu: Zásadní příležitosti a trendy. In: *BusinessIT: Informační technologie pro profesionály* [online]. 2014 [vid. 2015-01-21]. Dostupné z: http://www.businessit.cz/cz/mobilni-zarizeni-v-byznysu-zasadni-prilezitosti-a-trendy.php

[14] DREDGE, Stuart. 10 app trends for 2014: privacy, wearables, education and more. In: *The Guardian: Latest news, sport and comment from the Guardian* [online]. 2014 [vid. 2015-01-21]. Dostupné z: http://www.theguardian.com/technology/2014/jan/10/app-trends-2014-privacy-wearables-education

[15] SIVANPILLAI, Mathumuniasamy. Mobile Business Trends in 2015 - All You Need to Know. In: *RS Web Solutions - Web Resource Library* [online]. 2015 [vid. 2015-04-14]. Dostupné z: http://www.rswebsols.com/tutorials/internet/mobile-business-trends-2015

[16] NetBeans IDE Features. *NetBeans* [online]. © 2015 [vid. 2015-01-22]. Dostupné z: https://netbeans.org/features/

[17] NBAndroid - NetBeans Plugin detail. RKUBACKI. *NetBeans* [online]. © 2012 [vid. 2015-01-22]. Dostupné z: http://plugins.netbeans.org/plugin/19545/nbandroid

[18] BEAL, Vangie. What is Android SDK? Webopedia. In: *Webopedia: Online Tech Dictionary for IT Professionals* [online]. © 2015 [vid. 2015-01-22]. Dostupné z: http://www.webopedia.com/TERM/A/Android_SDK.html

[19] MOTL, Petr. OR-CZ SPOL. S R.O. *OR-SYSTEM – metodika vývoje Android aplikace.* Moravská Třebová, 2013.

[20] Providing Resources. *Android Developers* [online]. 2015 [vid. 2015-04-15]. Dostupné z: http://developer.android.com/guide/topics/resources/providing-resources.html

[21] Fragments. *Android Developers* [online]. 2015 [vid. 2015-01-23]. Dostupné z: http://developer.android.com/guide/components/fragments.html

[22] Accessing Resources. *Android Developers* [online]. 2015 [vid. 2015-04-16]. Dostupné z: http://developer.android.com/guide/topics/resources/accessing-resources.html

[23] Input Events. *Android Developers* [online]. 2015 [vid. 2015-01-23]. Dostupné z: http://developer.android.com/guide/topics/ui/ui-events.html

[24] TRAVIS, Brian E. *XML a SOAP: progamování serverů BizTalk.* Vyd. 1. Praha: Computer Press, 2000, 418 s. ISBN 80-722-6303-X.

[25] ROUSE, Margaret. SOAP (Simple Object Access Protocol) definition. In: *TechTarget* [online]. 2014 [vid. 2015-02-03]. Dostupné z: http://searchsoa.techtarget.com/definition/SOAP

[26] Motorola TC55 - Kodys. KODYS. *Kodys* [online]. © 2009 [vid. 2015-03-19]. Dostupné z: http://www.kodys.cz/produkty/mobilni-terminaly/enterprise-digital-assistant/motorola-tc55.html

[27] DWDEMO SAMPLE. MOTOROLA SOLUTIONS. *Zebra Technologies LaunchPad* [online]. © 2013 [vid. 2015-03-19]. Dostupné z: https://launchpad.motorolasolutions.com/documents/dwdemo_sample.html

Bibliografie

Android Developers [online]. 2015 [vid. 2015-01-26]. Dostupné z: http://developer.android.com/

ARIFF, M. H. a I. ISMAIL. Livestock information system using Android Smartphone. *2013 IEEE Conference on Systems, Process.* IEEE, 2013, s. 154-158. DOI: 10.1109/SPC.2013.6735123.

ATWOOD, Jeff a Joel SPOLSKY. *Stack Overflow* [online]. © 2015 [vid. 2015-01-26]. Dostupné z: http://stackoverflow.com/

BARCÍK, Jan. *Vývoj e-commerce platformy pro mobilní zařízení s podporou webového backendu*. Praha, 2011. Bakalářská práce. Unicorn College.

BERGER, Ondřej. Analýza systému Android ve vztahu ke klientské části informačních systémů. *Systémová integrace 4/2011*. 2011, č. 4. ISSN 1210-9479.

DOSTÁL, Leoš. *Synchronizace databáze mobilního zařízení OS Android se serverovou databází.* Liberec, 2014. Diplomová práce. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií.

HOZÁK, Martin. *Pokladní systém pro restaurace a bary.* Liberec, 2013. Diplomová práce. Technická univerzita v Liberci, Fakulta mechatroniky a mezioborových inženýrských studií.

NetBeans [online]. © 2015 [vid. 2015-01-26]. Dostupné z: https://netbeans.org/

NOVÁK, Lukáš. *Aplikace na objednávání jídel pro platformu Android*. Olomouc, 2012. Bakalářská práce. Univerzita Palackého, Přírodovědecká fakulta.

OPEN HANDSET ALLIANCE. *Open Handset Alliance* [online]. 2007 [vid. 2015-01-26]. Dostupné z: http://www.openhandsetalliance.com/

SOJKA, Jiří. *Klient server aplikace pro Activiti workflow.* Liberec, 2013. Diplomová práce. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií.

ŠESTÁK, Martin. *Klient pro studijní informační systém KOS na platformě Android.* Praha, 2013. Bakalářská práce. České vysoké učení technické v Praze, Fakulta elektrotechnická.

ŠOBÁŇ, Vojtěch. *Informační systém pro řízení skladu a návrh mobilní aplikace pro Android.* Brno, 2013. Diplomová práce. Masarykova univerzita, Fakulta informatiky.

UTPATADEVI, Ni L. P. Pravina, A. A. K. Oka SUDANA a A. A. Kt. Agung CAHYAWAN. Implementation of MVC (Model-View-Controller) Architectural to Academic Management Information System with Android Platform Base. *International Journal of Computer Applications.* 2012, No.8. DOI: 10.5120/9131-3313. Dostupné z: http://research.ijcaonline.org/volume57/number8/pxc3883313.pdf

VACEK, Ondřej. *Mobilní aplikace pro práci s univerzitním elearningovým portálem na platformě Android.* Liberec, 2013. Diplomová práce. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií.

VORÁLEK, Tomáš. Android. In: *WikiKnihovna: Knihovníci sobě* [online]. 2012 [vid. 2015-01-21]. Dostupné z: http://wiki.knihovna.cz/index.php/Android

YUDAKHIN, Mikhail. *Systém řízení chytrého domu s inteligentní elektroinstalací na základě OS Android.* Liberec, 2012. Kvalifikační práce. Technická univerzita v Liberci, Fakulta strojní.

Seznam příloh