MASARYKOVA UNIVERZITA
FAKULTA INFORMATIKY

Srovnání současných vývojových platforem pro mobilní zařízení založená na OS Linux

DIPLOMOVÁ PRÁCE

Martin Kolman

Brno, jaro 2013

Prohlášení

Prohlašuji, že tato diplomová práce je mým původním autorským dílem, které jsem vypracoval samostatně. Všechny zdroje, prameny a literaturu, které jsem při vypracování používal nebo z nich čerpal, v práci řádně cituji s uvedením úplného odkazu na příslušný zdroj.

Martin Kolman

Vedoucí práce: doc. RNDr. Aleš Horák, Ph.D.

Poděkování

V prvé řadě bych chtěl poděkovat vedoucímu mé diplomové práce doc. RNDr. Aleši Horákovi, Ph.D., za vstřícnost, připomínky a všechny poskytnuté zdroje. Bez něj by tato práce nemohla vzniknout. Chtěl bych také poděkovat všem členům komunity otevřených mobilních platforem za jejich rady a zpětnou vazbu.

Shrnutí

Práce se zabývá srovnáním současných (převážně linuxových) mobilních platforem a popisuje základní odlišností mobilních a „stolních“ platforem. Dále jsou formulovány zásady pro tvorbu multiplatformní mobilní aplikace. Tyto zásady jsou pak demonstrovány v podobě multiplatformního navigačního systému modRana. V závěru práce jsou popsány praktické výsledky tohoto projektu a další směry možného pokračování vývoje v návaznosti na výsledky této diplomové práce.

Klíčová slova

mobilní platformy, multiplatformní aplikace, Linux, Python, GTK, modRana, pydroid, Android, BlackBerry 10, Sailfish, Maemo, svobodný software

Obsah

1 Úvod

Mobilní zařízení ve všech možných podobách stále více zasahují do života běžných uživatelů. Ať už se jedná o chytré telefony, tablety, ultrabooky či jiná mobilní zařízení, většina z nich používá operační systém založený na linuxovém jádře. Linuxovým kernelem však často vzájemná shoda končí a tyto operační systémy jsou mezi sebou zpravidla silně nekompatibilní.

Přestože mnoho vývojářů volí cestu tvorby samostatných aplikací pro každou platformu zvlášť, umožňují dostupné nástroje vytvářet multiplatformní aplikace podporující nejen mobilní, ale i klasické „stolní“ platformy.

Druhá kapitola diplomové práce je věnována srovnání existujících mobilních platforem, které jsou založené na linuxovém jádře. Pro úplnost jsou stručně zahrnuty i takové platformy, které linuxové jádro nepoužívají. Toto srovnání slouží pro volbu prostředků vhodných pro úspěšný vývoj multiplatformní aplikace.

Třetí kapitola je zaměřená na specifika mobilních, a to nejen linuxových, platforem z pohledu vývojáře aplikací pro „stolní“ počítače. Tento přehled obsahuje informace potřebné pro to, aby výsledná aplikace využívala všech důležitých funkcí mobilních platforem a aby přesto nebylo vyloučeno její fungování na stolních počítačích.

Tématem čtvrté kapitoly je tvorba aplikace, která bude podporovat co nejvíce v současnosti rozšířených mobilních platforem. Kapitola popisuje výběr prostředků vhodných pro dosažení tohoto cíle v návaznosti na kapitoly 2 a 3.

Flexibilní navigační systém modRana je rozebrán v páté kapitole jako příklad multiplatformní aplikace vyvinuté podle principů, které jsou uvedeny v kapitole čtvrté.

Konkrétní přínosy této diplomové práce jak pro vývojáře, tak pro konkrétní uživatele multiplatformních mobilních aplikací, jsou popsány v kapitole šesté.

Závěr práce přibližuje možné další směry vývoje oblasti mobilních linuxových platforem obecně, stejně jako vývoj projektů jako jsou modRana a pydroid, vycházejících z této práce.

2 Srovnání mobilních linuxových platforem

V současnosti existuje mnoho mobilních platforem založených na linuxovém jádře a dalších komponentech *„stolních“* linuxových distribucí. Zatímco z pohledu jejich uživatele mohou jednotlivé platformy vypadat podobně – jako operační systém pro mobilní zařízení, který umožňuje instalovat aplikace třetích stran. Ve skutečnosti se jednotlivé platformy většinou vzájemně velmi liší.

Znalost těchto odlišností je velmi důležitá zejména pro vývojáře mobilních aplikací. Vývojář pracuje s platformou na výrazně nižší úrovni abstrakce než uživatel, a znalost interní struktury dané mobilní platformy je tak pro něj zásadní.

Následující srovnání zahrnuje převážně linuxové platformy, ale i několik platforem, které jsou sice POSIXově kompatibilní, ale nepoužívají linuxový kernel[1]. Pouze stručně jsou zde popsány ne-POSIXové a ne-linuxové platformy[2].

2.1 Kritéria pro srovnávání

Pro srovnání jednotlivých platforem jsem zvolil několik hlavních kritérií:

- architektura
- otevřenost
- grafika a multimédia
- balíčkování a distribuce softwaru
- vývojové prostředí
- dostupnost hardwaru

Jednotlivé platformy jsem porovnával především podle výše uvedených kritérií, která také posloužila k vytvoření srovnávacích tabulek.

2.1.1 Architektura

Architekturou je v tomto případě myšleno to, z jakých částí se daná platforma skládá a jak jsou tyto části mezi sebou provázány. Důležité může být rovněž to, do jaké míry jsou jednotlivé části nahraditelné a zda je možné se s některými z nich setkat na různých platformách[3].

Dobrá analýza architektury usnadňuje portováním aplikace, při kterém

1. BB10 a iOS
2. mobilní verze Windows
3. příkladem multiplatformní technologie jsou WebKit nebo Qt

je vhodné zaměřit se zejména na jedinečné aspekty platformy. Právě ony často vyžadují největší změny v portované aplikaci.

2.1.2 Otevřenost

Otevřenost mobilní platformy lze pojmout z několika úhlů:

- dostupnost zdrojových kódu platformy a možnost modifikace
- transparentnost vývoje platformy
- otevřenost platformy pro software třetích stran

Zdrojové kódy a možnost modifikace

Dostupnost zdrojových kódů platformy je důležitá hned z několika důvodů. Prvním je možnost modifikace platformy – pokud nejsou zdrojové kódy k dispozici, jsou možnosti modifikace velmi omezené[4].

Ani plná dostupnost zdrojových kódů však nemusí garantovat, že pozměněný operační systém půjde spustit na stávajících zařízeních. Mnoho mobilních zařízení totiž sleduje trend tzv. *tivoizace*. Termín označuje kryptografické ověřování integrity obrazu operačního systému před jeho spuštěním. Pokud je zařízení *tivoizováno*, je spuštěn pouze takový obraz operačního systému, který je podepsán privátním klíčem výrobce. Operační systém, který tímto prověřením neprojde, nebude spuštěn.

Zdrojové kódy také umožňují další pokračování vývoje platformy v případě úpadku jejího původního provozovatele. Pokud je však hardware *tivoizován* a privátní klíč pro podepisování obrazu OS není znám, nebude bohužel možné k pokračování vývoje platformy použít stávající zařízení.

Dalším hlediskem je bezpečnost, která je vzhledem k velkému množství citlivých informací obvykle uchovávaných na mobilních zařízeních velmi důležitá. Bez zdrojových kódů odpovídajících nainstalovaným binárním komponentům platformy není možné provést plně nezávislou analýzu bezpečnostních rizik.

Transparentnost vývoje platformy

Dalším důležitým kritériem pro hodnocení otevřenosti mobilní platformy je otevřenost jejího vývoje.

I v případě, že jsou zdrojové kódy mobilního operačního systému plně k dispozici pod svobodnou licencí, může celý vývoj probíhat za zavřenými dveřmi, bez zájmu o zpětnou vazbu z vývojářské a uživatelské komunity

4. v podstatě pouze dílčí změny zásahem do případných konfiguračních souborů

a bez možnosti sledovat průběh vývoje. Komunikace s komunitou uživatelů a externích vývojářů pak spočívá pouze v občasném zveřejnění balíků zdrojových kódů[5] zpravidla krátce po uvedení nové verze systému. Tímto způsobem probíhá například vývoj mobilní platformy Android[1]: vývoj probíhá za zavřenými dveřmi a je ovlivněn pouze Googlem a korporátními členy Open Handset Aliance[2].

Při uzavřeném způsobu vývoje je díky dostupnosti zdrojových kódů stále možné operační systém modifikovat. Bez možnosti ovlivnit směřování hlavní vývojové větve je však nutné změny buťto průběžně adaptovat pro použití s novější verzí systému, nebo se spokojit s používáním staré verze systému.

Vývoj však také může probíhat plně transparentně – a to s využitím otevřených repositářů zdrojového kódu a otevřených komunikačních kanálů[6]. Tímto způsobem probíhá např. vývoj mobilních distribucí Mer a Nemo Mobile, SHR či komunitní údržba Maemo 5.

Hlavní výhodou tohoto přístupu je nízká vstupní bariéra pro přispěvatele a reálná možnost ovlivnit směřování platformy vlastním přímým zapojením do vývoje. Lze také předpokládat, že transparentně vyvíjený projekt nebude mít problém se začleněním prospěšných změn, které nebude nutné odděleně udržovat a které zároveň mohou nalézt širší uplatnění v hlavní větvi vývoje.

Otevřenost platformy pro aplikace třetích stran

U všech moderních mobilních platforem existuje v určité podobě podpora instalace a spouštění aplikací třetích stran. Právě dostupnost aplikací třetích stran může být důležitým faktorem rozhodujícím o úspěchu platformy. Mobilní platforma většinou obsahuje nejméně jeden centrální aplikační repositář, který umožňuje uživatelům snadnou instalaci aplikací.

Z hlediska otevřenosti je třeba sledovat dva hlavní aspekty:

- pravidla pro přijetí aplikace do centrálního repositáře
- možnost přímé instalace aplikace

Začlenění aplikace do centrálního repositáře aplikací je velmi důležité, protože se jedná o nejkratší cestu ke koncovému uživateli. Provozovatel repositáře pro začlenění aplikace stanovuje pravidla a snaží se dohlížet na jejich dodržování – zpravidla formou manuální či automatizované kontroly aplikace před jejím zveřejněním. Pokud jsou pravidla rozumná[7], je vše

5. tzv. *code drop*
6. např. veřejné emailové konference, IRC kanály a fóra
7. zákaz virů, trojských koní, atd.

v pořádku a kontrola aplikace do repositáře znamená jen mírné zdržení jejího vydání.

Pokud však provozovatel není nestranný a brání v začlenění aplikací bez objektivního důvodu[8], není platforma opravdu otevřená. Pouze by uživatel by měl rozhodovat, která aplikace není vhodná – tím, že si ji nenainstaluje.

Termín *možnost přímé instalace aplikace* znamená, zda je možné instalovat aplikace, aniž by musely projít centrálním repositářem (a splňovat tak pravidla provozovatele repositáře). Pokud platforma ve výchozím nastavení neumožňuje přímou instalací a aplikace je možné získat pouze prostřednictvím přednastaveného centrálního repositáře s restriktivní politikou, bývá označována jako tzv. *walled garden*.

2.1.3 Grafický systém

Pro mobilní platformy je samozřejmě rovněž možné vytvářet aplikace bez grafického rozhraní, které plní funkci *démonů*, jsou řízeny vzdáleným rozhraním či jednoduše běží v emulátoru terminálu. Přesto je většinou pojmem *mobilní aplikace* míněna aplikace s grafickým rozhraním. Tento fakt je výrazně ovlivněn fyzickými možnostmi ovládání mobilních aplikací.

Fyzická velikost obrazovky značně limituje množství informací, které lze „čitelně“ zobrazit. Většina zařízení také nedisponuje hardwarovou klávesnicí. Tento fakt velmi ztěžuje využití konsolových aplikací a v zásadě vyžaduje grafické rozhraní přizpůsobené mobilnímu zařízení.

Grafické systémy mobilních platforem se podstatně liší od grafických systému používaných na „stolních“ počítačích. Jsou mimo jiné v mnoha aspektech velmi zjednodušené. I takové funkce, které se na první pohled jeví jako samozřejmé – jako např. možnost zobrazení více oken současně či podpora více oken jedné aplikace – nebývají ve většině případů k dispozici. Zatímco na stolních linuxových počítačích má X-server téměř monopolní postavení, mezi mobilními platformami je jeho zastoupení poměrně malé, což v důsledku velmi omezuje škálu použitelných grafických knihoven.

2.1.4 Multimédia

Pojmem multimédia v případě mobilních platforem rozumíme podporu pro hardwarovou akceleraci grafických operací, videa (nahrávání i přehrávání) a zvuku. Pro využití multimédií na mobilních platformách je hardwa-

8. zákaz konkurujících si aplikací, zákaz aplikací pod svobodnou licencí, zákaz interpretovaného kódu, atd.

rová akcelerace stěžejní. Tento fakt vychází ze specifické situace mobilních zařízení, které mají často relativně slabá CPU a musí vzhledem k provozu z baterie šetřit energii.

Hardwarová akcelerace tak umožňuje práci s typy multimédií, které by mobilní CPU vůbec zpracovat nezvládl [9]. Díky použití čipových bloků, vytvořených pro tuto činnost na míru, má hardwarová akcelerace mnohem menší spotřebu energie než CPU provádějící ekvivalentní výpočty.

Zatímco většina mobilních platforem podporuje hardwarovou akceleraci grafiky prostřednictvím standardního rozhraní OpenGL ES[3], možnosti akcelerace videa a zvuku se velmi liší nejen podle platformy, ale především podle konkrétního zařízení. Dalším problémem pro práci s multimédii mohou být platformně závislá navzájem nekompatibilní rozhraní. Tento problém naštěstí částečně řeší použití multimédií s pomocí abstrakce, kterou nabízí některá z multiplatformních grafických a multimediálních knihoven jako Qt, SDL či GStreamer.

2.1.5 Balíčkování

Pojem *balíčkování* označuje postup, při kterém je aplikace upravena do podoby, která umožňuje její snadnou distribuci a instalaci. Výsledkem *balíčkování* je jeden soubor[10], který obsahuje všechny soubory balíčkované aplikace, metadata balíčku a instalační skripty.

Metadata obsahují jak položky určené pro uživatele (název aplikace, stručný popis, kategorii, odkaz na webové stránky projektu, kontaktní adresu autora atd.), tak data pro instalační systém (číslo verze aplikace, kontrolní součty souborů nebo seznam dalších balíčků potřebných pro instalaci).

Důležitým aspektem balíčkovacího systému je, jestli podporuje automatické řešení závislostí mezi balíčky. To umožňuje balíčkům popsat, které další balíčky potřebují pro správné fungování. Často používanou funkcionalitu je pak možné vyčlenit do balíčku, na který další balíčky odkazují pomocí deklarace závislostí. Díky tomu lze výrazně zmenšit nejen velikost jednotlivých balíčků, ale i celkovou spotřebu úložného místa na zařízení.

Další výhodou podpory závislostí je rovněž fakt, že v případě nálezu bezpečnostní díry v jednom z balíků stačí opravit pouze tento balík. Veškeré aplikace, které využívají jeho funkcionality, jsou po aktualizaci opraveného balíčku před nalezenou bezpečnostní dírou automaticky chráněny.

Mnoho mobilních platforem však vůbec nepodporuje závislosti mezi

9. např. přehrávání full-HD videa ve formátu h264 v reálném čase
10. většinou se jedná o komprimovaný archiv

balíčky, a proto je nutné všechny potřebné knihovny buď to přibalit, nebo řešení závislostí suplovat pomocí aplikační logiky[11].

2.1.6 Vývojové prostředí

Osud mobilní platformy je vázán na dostupnost aplikací a proto se provozovatelé mobilních platforem co nejvíc snaží vývojářům vývoj aplikací zjednodušit a nabízejí tzv. SDK. SDK je sada všech nástrojů[12] potřebných pro vývoj aplikace pro danou mobilní platformu.

SDK se většinou zaměřují pouze na jedinou kombinaci programovacího jazyka a sady knihoven. SDK jednotlivých platforem bývají mezi sebou také často nekompatibilní, což ztěžuje tvorbu multiplatformních aplikací.

Ve většině případů je možné pro danou platformu vyvíjet aplikace i bez plného využití SDK: vývojář se však v tomto případě musí obejít bez podpory provozovatele a stává se pak, že narazí na části platformy, které nejsou tak stabilní a otestované jako rozhraní, která využívá oficiální SDK.

2.1.7 Hardware

Pro objektivní hodnocení mobilní platformy je podstatná dostupnost zařízení. Přestože mnoho platforem v rámci SDK obsahuje více či méně použitelný simulátor, nebývají všechny funkce a rozhraní pokaždé věrně simulovány a většina simulátorů je rovněž velmi pomalá.

Pro vývojáře je také důležité znát typy dostupného hardwaru: pro vývoj aplikací potřebuje vědět, o jaký formát zařízení (chytrý telefon, tablet, netbook,...) se jedná a jaké subsystémy (GPS, klávesnice, kamera, NFC[4][5]) bude mít k dispozici.

V rámci vývoje aplikací hraje významnou roli také to, jakým způsobem platforma odlišnosti různých zařízení abstrahuje. Je totiž důležité vědět, zda jsou všechny funkce zpřístupněny jednotným aplikačním rozhraním nebo naopak množinou specifických API. Běží na všech formátech zařízení a architekturách vůbec stejný operační systém nebo se jedná o různé implementace OS, které mají společný pouze název platformy ?

2.2 MeeGo

Mobilní platforma MeeGo[6][7] byla společným projektem firem Intel a Nokia. Cílem platformy bylo vytvořit otevřenou mobilní distribuci nejen pro

11. viz například aplikace Ministro na platformě Android
12. knihovny, kompilátory a někdy i IDE či simulátor platformy

chytré telefony, ale i pro tablety, netbooky a zábavní systémy v dopravních prostředcích. Obě firmy do projektu MeeGo přispěly svou vlastní mobilní distribucí, společnost Nokia přispěla výsledky projektu Maemo a společnost Intel nabídla distribuci pro netbooky, zvanou Moblin. Použity byly také některé komponenty distribuce Fedora a OpenSuse. Rodokmen projektu MeeGo i dalších mobilních linuxových distribucí je k dispozici v první přílozeA.

11. února 2011 však Stephen Elop, šéf společnosti Nokia, ohlásil opuštění projektu MeeGo a oznámil přechod své společnosti na mobilní platformu Windows Phone. Následkem toho byly zastaveny práce na projektu MeeGo, který byl následně opuštěn rovněž společností Intel.

Prostředky vložené do projektu MeeGo však nevyšly nazmar díky mobilní metadistribuci Mer, kterou založila skupina bývalých MeeGo vývojářů a proto ji lze do značné míry považovat za pokračovatele projektu MeeGo.

2.3 Mer

Projekt Mer[8] byl založen skupinou vývojářů, kteří se po rozpadu projektu MeeGo rozhodli na vývoji této platformy pokračovat. Usoudili však, že cesta zvolená pro projekt MeeGo – tzn. vývoj kompletní linuxové distribuce včetně grafického rozhraní a údržby kernelu – nebyl dobrý nápad.

Projekt Mer je tedy tzv. metadistribuce, neobsahuje totiž grafické rozhraní ani distribuční kernel. Zato ale Mer poskytuje veškeré nástroje pro tvorbu plnohodnotné linuxové distribuce. Vývojáři Meru si od tohoto přístupu slibují větší flexibilitu vývoje svého projektu bez nutnosti čekání na vývoj grafického rozhraní a udržování vlastní verze linuxového jádra. Další výhodou jejich přístupu je, že díky této své struktuře je Mer v podstatě neutrálním projektem, na kterém mohou spolupracovat i vzájemně si konkurující komerční subjekty, podobně jako řada konkurujících si firem spolupracuje na vývoji linuxového jádra.

Doposud se zdá, že postup, který zvolili vývojáři Meru, se osvědčil, protože již vznikla řada mobilních distribucí, které Mer používají jako svůj základ. Jedná se např. o:

- Nemo Mobile
- Sailfish
- Plasma Active
- Cordia
- Seadot

2.3.1 Otevřenost

Vývoj metadistribuce Mer probíhá zcela otevřeně. Její zdrojové kódy jsou ve všech fázích vývoje k dispozici a veškerá komunikace o vývoji Meru probíhá zcela veřejně na IRC kanále[13] a v emailové konferenci[14]. Projekt Mer používá pro průběžný audit změn ve zdrojovém kódu vlastní veřejnou instanci[15] systému Gerrit[9], která zároveň ukazuje zajímavý obraz vývoje celého projektu.

O skutečné otevřenosti a neutrálnosti projektu Mer svědčí také to, že se na jeho vývoji ruku v ruce podílejí jak vývojáři společnosti Jolla, tak členové komunity Plasma Active nebo komunitní vývojáři distribuce Nemo Mobile.

2.3.2 Architektura

Mer používá balíčkovací systém RPM. Samotná metadistribuce Meru je vytvářena pomocí systému pro tvorbu a správu balíčků OBS[10]. K dispozici jsou nejrůznější komponenty vhodné pro tvorbu mobilních distribucí, např.: X server, grafická knihovna Qt, DBUS, systemd, Python, GNU utility, Pulseaudio, GStreamer, klávesnicový framework Maliit atd. Tvorbě instalačních obrazů slouží specializovaný nástroj zvaný MIC[11].

2.3.3 Vývojové prostředí

Vývojové prostředí projektu Mer[12] tvoří dvě části: tzv. platformní SDK a aplikační SDK. Platformní SDK slouží vývoji samotné distribuce Mer, aplikační SDK je pak zaměřena na vývoj aplikací, které používají grafickou knihovnu Qt nebo prostředí HTML 5. Platformní SDK je určená také pro vývoj a portování knihoven.

2.4 Nemo Mobile

Nemo Mobile [13] je komunitní distribuce pro mobilní zařízení, založená na metadistribuci Mer. Nemo k Meru přidává vlastní uživatelské rozhraní a adaptace pro jednotlivá zařízení.

2.4.1 Architektura

Nemo tedy doplňuje do *metadistribuce* Mer to, co Mer úmyslně neposkytuje:

13. logy IRLC kanálu #mer: `http://www.merproject.org/logs/%23mer/`
14. `http://www.mail-archive.com/mer-general@lists.merproject.org/`
15. `http://review.merproject.org/#q,status:merged,n,z`

- linuxové jádro
- ovladače pro zařízení
- uživatelské rozhraní

Uživatelské rozhraní distribuce Nemo Mobile je postaveno na grafické knihovně Qt[16], pomocí které jsou vytvořeny i veškeré předinstalované aplikace. Qt prostřednictvím Qt-Mobility taktéž poskytuje data ze senzorů. X server plní funkci grafického systému, v budoucnu lze předpokládat přechod na grafický systém Wayland.

2.4.2 Otevřenost

Vývoj distribuce Nemo Mobile je plně otevřený, veškerý kód je k dipozici po celou dobu vývoje a všechny diskuze o směru vývoje distribuce probíhají na veřejných IRC kanálech a emailových konferencích.

Jedinou uzavřenou komponentou jsou některé ovladače zařízení, které jejich výrobce poskytuje pouze v binární podobě.

2.4.3 Balíčkování

Nemo plně přejímá balíčkovací systém z Meru, tzn. používá RPM a utilitu zypper. Tvorba balíků probíhá v instanci OBS, používané také distribucí Mer. V této instanci má Nemo Mobile vlastní OBS projekt, který nese název *nemo*[14]. V tomto projektu jsou k dispozici sekce, které reprezentují jednotlivé vývojové větve (stabilní, testovací, atd.) a také sekce věnované podpoře cílových zařízení.

Pro zapojení se do vývoje distribuce Nemo mobile nebo např. jen pro sestavení balíku oproti Nemu stačí:

- zaregistrovat se na Bugzillu projektu Mer[15]
- přihlásit se stejnými údaji na Mer OBS[16]
- vytvořit si vlastní projekt
- nastavit jako *target* tohoto projektu jednu z hlavních větví Nema (stable/testing)

Přestože tento postup může na první pohled působit složitě, je ve skutečnosti velmi praktický a to zejména díky vysoké úrovni automatizace práce s balíčky[17]). Jakmile vývojář zvládne do OBS přidat první balíček, ty další už mu půjdou přidat velmi snadno.

---

16. momentálně ve verzi 4.8
17. automatické „přebalení“ při změnách v distribučních balících, možnost kopírování mezi projekty, verzování balíků, atd.

Obrázek 2.1: Nemo Mobile na Nokia N950

Praktická je také možnost nabídnutí balíčku k začlenění do cizího projektu. Tato funkce se dá využít i pro začlenění vlastního balíku do distribuce. Právě tímto způsobem je distribuce vyvíjena – vývojáři nabízejí balíky do hlavních repositářů a jejich integritu si vzájemně kontrolují.

2.4.4 Vývojové prostředí

Hlavní kombinací nástrojů pro vývoj jak distribuce, tak aplikací pro distribuci určených, je C++ a Qt/QML. Použití C++ však není podmínkou, protože oficiálních repositářích obsahují Python a PySide. Díky tomu je možné pro Nemo vytvářet grafické aplikace v Pythonu. Další možností je tvorba aplikací pouze s využitím QML a Javascriptu (bez použití C++).

Vzhledem k otevřenosti distribuce nejsou vývojářům kladeny žádné překážky pro použití dalších jazyků a knihoven – a to tak dlouho, dokud je možné tyto jazyky a knihovny zkompilovat a spustit je na cílových zařízeních platformy.

2.4.5 Hardware

Nemo mobile v současnosti podporuje především chytré telefony N900, N950 a N9 společnosti Nokia. Pro tato zařízení jsou k dispozici pravidelně aktualizované instalační obrazy. Jsou to také zařízení, na kterých Nemo používá nejvíce vývojářů i uživatelů. Nemo na nich podporuje základní funkcionality, jako jsou volání, fotoaparát nebo 3D akcelerace. Další funkce jako např. GPS zatím podporovány nejsou.

Kromě *oficiálně* podporovaného hardwaru mohou vývojáři a uživatelé

Obrázek 2.2: *lockscreen* Nemo Mobile na Nokia N950 s vyklopenou klávesnicí

Nemo provozovat i na řadě dalších zařízení, je však nutné počítat s tím, že ne všechny funkce daného zařízení budou podporovány. Nejaktuálnější informace o podporovaném hardwaru jsou k dispozici na wiki projektu Nemo Mobile.

Nemo se dá spustit i na virtuálním stroji, a to díky dostupnosti obrazu pro VirtualBox. Vzhledem k tomu, že velká část hardwaru není virtuálním strojem podporována a jeho výkon neodpovídá mobilnímu zařízení, jedná se spíš o nouzovou variantu. Virtuální stroj se však může hodit pro testování různých verzí Nema před jejich instalací na reálné zařízení.

2.5 Sailfish

Sailfish[17] je komerční linuxová distribuce pro mobilní zařízení vytvořená finskou firmou Jolla[18]. Tuto společnost tvoří z většiny bývalí zaměstnanci firmy Nokia, kteří pracovali na projektech Maemo (N900), MeeGo 1.2 Harmattan (N9) a společném projektu MeeGo s firmou Intel. Sailfish lze považovat za pokračovatele těchto projektů, a to pod záštitou společnosti Jolla.

2.5.1 Architektura

Distribuce Sailfish se skládá z těchto hlavních součástí:

- linuxového jádra

- metadistribuce Mer
- prvků distribuce Nemo mobile
- grafických komponent Silica[19]
- uživatelského rozhraní pro Sailfish
- ovladačů zařízení

Sailfish se svou architekturou silně podobá klasické linuxové distribuci pro PC: většina běžně dostupných systémových utilit by měla být k dispozici, je použit DBUS, Systemd, balíčkovací systém RPM, grafické rozhraní využívá X server a knihovnu Qt. Na rozdíl od Androidu, který staví na značně omezené libc, známé pod názvem Bionic, používá Sailfish plnohodnotnou knihovnu libc.

2.5.2 Grafický systém

Současná podoba systému Sailfish používá pro správu grafiky X server a grafickou knihovnu Qt ve verzi 4.8. V budoucnu se předpokládá přechod na Wayland a Qt verze 5.

2.5.3 Otevřenost

Metadistribuce Mer a distribuce Nemo Mobile, které tvoří podstatnou část platformy Sailfish poskytují nejen svůj zdrojový kód pod svobodnou licencí, ale jejich vývoj probíhá také zcela veřejně. Na základě toho je možné se do něj přímo zapojit, a nepřímo tak ovlivnit směřování distribuce Sailfish. To samé platí samozřejmě i pro další důležitou komponentu – linuxové jádro.

V tomto ohledu se Sailfish diametrálně liší od Androidu, jehož zdrojový kód je sice volně k dispozici, ale veškerý vývoj až do vydání stabilní verze probíhá za zavřenými dveřmi, kde o směřování projektu rozhoduje výhradně společnost Google a její firemní partneři v Open Handset Aliance.

2.5.4 Distribuce softwaru

Balíčkovací systém Sailfish vychází z Meru, jedná se tedy o RPM. Vzhledem k tomu, že zatím nebyla na trh oficiálně uvedena žádná zařízení pro koncové zákazníky, nebylo zatím ani oznámeno, jakým způsobem bude probíhat distribuce softwaru od vývojářů směrem k uživatelům. Jediná doposud známá a potenciálně relevantní informace je, že Sailfish nebude podporovat DRM[20].

2.5.5 Vývojové prostředí

Sailfish SDK byla vydána koncem února 2013. Skládá se z Qt Creatoru, který tvoří výchozí IDE a dvou virtuálních strojů typu VirtulBox[21]. První virtuální stroj obsahuje kompilační prostředí postavené na Meru, druhý pak simulátor zařízení. Využití virtuálních strojů umožňuje vývojovému prostředí snadné fungování všude tam, kde jde spustit VirtualBox[18].

Běžné použití SDK spočívá v psaní zdrojového kódu v Qt Creatoru, odkud vývojář může přímo spustit kompilaci v kompilačním prostředí prvního virtuálního stroje. Výsledek kompilace je pak automaticky spuštěn v režimu ladění na druhém virtuálním stroji se Sailfish simulátorem.

2.5.6 Hardware

Podle dosavadních informací bude Sailfish nasazena jak na mobilních zařízeních, vyráběných přímo firmou Jolla, tak licencována pro zařízení dalších výrobců. Třebaže už Jolla při nejrůznějších příležitostech několikrát demonstrovala Sailfish na reálných zařízeních[19], žádný instalační obraz OS Sailfish zatím zveřejněn nebyl.

První zařízení formy Jolla s operačním systémem Sailfish, pojmenované „Jolla“, bylo oficiálně ohlášeno 20.5.2013 v Helsinkách[22] a je to o chytrý dotykový telefon s obrazovkou o úhlopříčce 4.5 palce. Zařízení dále disponuje dvou-jádrovým procesorem, 16 GB interní pamětí, slotem na micro-SD kartu, 8 Mpix fotoaparátem a vyměnitelnou baterií. Zadní kryt zařízení je odnímatelný a bude dostupný v různých barvách. Společnost Jolla oznámila, že prostřednictvím výměny krytu bude možné rozšiřovat technickou výbavu zařízení[20]. Podrobnější technické specifikace jako rozlišení obrazovky, objem RAM nebo frekvence CPU zatím nebyly zveřejněny. Chytrý telefon „Jolla“ si mohou zájemci od 20.5.2013 předobjednat[23] s předpokládaným termínem dodání do konce roku 2013.

18. VirtualBox podporuje většinu hlavních linuxových distribucí, MacOS X a Windows
19. jednalo se především o chytré telefony Nokia N950 či N9
20. byl např. silnější blesk, objektiv typu rybí oko nebo přídavná baterie

Obrázek 2.3: Chytrý telefon *Jolla*

Obrázek 2.4: *Jolla* s krytem v jiné barvě a pohled na zadní stranu

2.6 Plasma Active

Plasma Active[24] je linuxová distribuce pro mobilní zařízení, vyvíjená komunitou desktopového prostředí KDE[25].

2.6.1 Architektura

Plasma Active vychází z metadistribuce Mer, ke které přidává vlastní uživatelské rozhraní, zvané rovněž *Plasma Active*. Dále přidává také vlastní kernel a ovladače pro cílová zařízení.

2.6.2 Otevřenost

Vývoj distribuce Plasma Active probíhá zcela veřejně a může se do něj zapojit či jen sledovat aktuální dění kdokoli. Veškeré zdrojové kódy jsou k dispozici z Gitových repositářů, hostovaných na infrastruktuře projektu KDE[26].

Veškeré plánování a diskuze o vývoji probíhají veřejně – na IRC, v emailové konferenci[27] nebo v diskuzních fórech.

Otevřenosti vývoje lze vytknout přístup k vývoji hardwaru pro dedikované Plasma Active zařízení, o kterém jsou zatím známé pouze kusé informace a který probíhá tzv. „za zavřenými dveřmi".

2.6.3 Balíčkování

Balíčkování taktéž vychází z Meru: Plasma Active používá balíčkovací systém RPM. Přestože zatím nejsou známy podrobnosti o způsobu distribuce softwaru k uživatelům, vývojáři Plasma Active ohlásili, že plánují flexibilní a distribuovaný systém šíření jak softwaru, tak dalšího obsahu.

2.6.4 Vývojové prostředí

K vývoji aplikací pro Plasma Active mohou vývojáři použít SDK s názvem *Plasmate*[28]. Hlavní podporovanou grafickou knihovnou je Qt Quick. Mezi oficiálně podporované jazyky patří v Javascript, Python, Ruby a C++[29]. Lze vytvářet jak aplikace specifické pro prostředí Plasma, tzv. Plasmoidy, tak samostatné aplikace využívající Qt Quick.

Vzhledem k tomu, že základ Plasma Active tvoří téměř regulérní linuxová distribuce Meru, půjdou pravděpodobně využít i další knihovny a programovací jazyky.

2.6.5 Hardware

Příběh mobilního zařízení s Plasma Active je poměrně dlouhý a komplikovaný. Vše začalo ohlášením tabletu Spark[30] na začátku roku 2012. Tablet byl následně pro možný konflikt názvu Spark s názvem podobného výrobku v polovině roku 2012 přejmenován na *Vivaldi*[31]. Tablet Vivaldi byl založen na přeznačeném čínském tabletu, který se již běžně prodával s OS Android, ale zato se všemi ovladači, nutnými pro správný chod linuxové distribuce s Plasma Active. Uvedení tabletu na trh bylo stále oddalováno a v září 2012 bylo ohlášeno nové směřování projektu. Projekt Plasma Active zveřejnil, že na základě dosavadních zkušeností z projektu Vivaldi navrhne

vlastní tablet[32][33]. Konkrétní datum vydání tohoto zařízení dosud zveřejněno nebylo.

Třebaže zatím neexistuje dedikované zařízení s Plasma Active, je možné tuto platformu v současnosti už vyzkoušet na celé řadě existujících mobilních zařízení[34]. Jedná se zatím převážně o tablet-PC a kombinace tablet-notebook, běžící na architektuře x86. Jedinými „skutečnými" tablety v seznamu podporovaných zařízení je Nexus 7 a některé tablety firmy Archos.

2.7 Maemo 5 Fremantle

Operační systém Maemo 5[35] byl vyvinut firmou Nokia pro chytrý telefon Nokia N900. Jak již plyne z označení *Maemo 5*, jedná se již o pátou verzi systému Maemo. Předchozí verze však byly určeny pro tzv. *internetové tablety* – což jsou přenosná zařízení bez podpory připojení do mobilní sítě – ne však pro chytré telefony. Maemo 5 je poslední verzí systému Maemo, kterou Nokia vydala. MeeGo 1.2 Harmattan sice na Maemo 5 v mnoha aspektech navazuje, a to natolik, že bývá neoficiálně označován jako *Maemo 6*. Harmattan se přesto od Maemo 5 výrazně liší výchozím grafickým rozhraním, ve kterém bylo GTK[36] nahrazeno Qt.

Maemo 5 je založené na Debianu a velmi se podobá klasické desktopové distribuci, od které se liší hlavně nahrazením většiny klasických distribučních utilit busyboxem a použitím tzv. optifikace. Termín označuje instalaci všech programů, které nejsou pro start operačního systému nezbytně nutné, do adresáře `/opt`, aby bylo ušetřeno místo v kořenovém souborovém systému. Tento neobvyklý přístup je vynucen hardwarovým omezením N900, u které je kořenový systém souboru namapován na NOR flash paměť, která má pouze 256 MB. Adresář `/opt` je namapován na 2 GB oddíl na 32 GB MMC flash paměti. Přestože je ve většině případů optifikace jednoduchá a existují také poloautomatické optifikační skripty, může portování větších softwarových projektů již vyžadovat nejen úpravy balíčků ale i rozsáhlé zásahy do zdrojového kódu.

2.7.1 Otevřenost

Jelikož je Maemo 5 linuxovou distribucí, jsou všechny změny, provedené Nokií na kernelu[21] v souladu s GPL[37], všem k dispozici. Mnoho ovladačů pro N900 bylo také začleněno do hlavní větve jádra. Dostupnost zdrojových kódů samozřejmě platí i pro všechny další GPL komponenty, které

21. ve verzi 2.6.28

byly pro Maemo 5 upraveny[22]. Rovněž některé některé komponenty, které byly vytvořeny přímo pro Maemo jako např. grafické prostředí a knihovna *Hildon*[38] byly zveřejněné pod svobodnou licencí.

Další komponenty, aplikace a ovladače však zůstaly uzavřené. Komunitním iniciativám to v současnosti způsobuje problémy při údržbě a dalším vývoji platformy. Komunita proto již některé aplikace nahradila kompatibilní svobodnou alternativou. Hlavní problém však nadále tvoří uzavřený ovladač GPU, který brání užití jiného kernelu než toho současného vycházejícího z prastaré řady 2.6.28. Vzhledem ke složitosti tohoto ovladače je však nepravděpodobné, že dojde k jeho nahrazení svobodným ovladačem, a Maemo 5 tak proto pravděpodobně v dohledné budoucnosti zůstane u kernelu řady 2.6.28. Navzdory staré verzi je kernel nyní pod označením Kernel Power[39] spravován Maemo komunitou[23] a obsahuje mnoho backportů z novějších verzí kernelu.

2.7.2 Grafický systém

Jako grafický systém na Maemo 5 slouží X-server. Grafické prostředí je založeno na GTK s rozšířením zvaným Hildon a na hardwarově akcelerované animační knihovně Clutter[40]. Použití kombinace GTK + Hildon navazuje na předchozí generace systému Maemo; Clutter je pak novinkou, která na N900 v mezích možností zpřístupňuje implementaci efektů a animací, očekávaných od grafického rozhraní moderních chytrých telefonů.

V souvislosti s akvizicí společnosti Trolltech společností Nokia bylo na Maemo 5 později přidána také podpora pro grafickou knihovnu Qt. Vývojáři mohou vytvářet Qt aplikace, využívající jak klasický koncept QWidgets, tak aplikace v QML/Qt Quick. Existuje také komunitní port Harmattan Qt Components.

3D akcelerace grafiky je zpřístupněna rozhraním OpenGL ES 1.1 a 2.0, ovšem bez podpory vertikální synchronizace. Tato skutečnost se v některých případech projevuje mírným „trháním“ obrazu při překreslování obrazovky. Vzhledem k tomu, že ovladač GPU je uzavřený a šance na jeho aktualizaci je nulová, nebude vertikální synchronizace na Maeamu 5 pravděpodobně nikdy k dispozici.

22. patche pro GTK, Clutter
23. hlavním vývojářem je Pali Rohár z MFF CUNI

2.7.3 Multimédia

Maemo 5 nabízí široké možnosti práce s multimédii[41]. Dekódování videa funguje prostřednictvím multimediálního frameworku GStreamer[42] a umožňuje využití hardwarově akcelerovaného dekódování některých formátů. GStreamer také slouží pro práci s videem z kamery/fotoaparátu.

Práci s audiem slouží framework Pulseaudio[43] a dekódování audia běží převážně na procesoru. Podle údajů uvedených v dokumentaci je tomu tak proto, že použití DSP pro dekódování audia může mít na N900 v konečném důsledku vyšší režii než prosté dekódování na CPU.

K dispozici je rovněž rozhlasový subsystém N900 – který podporuje jak poslech analogových rozhlasových stanic, tak vysílaní audia na rozhlasové frekvenci s dosahem několika metrů[24].

2.7.4 Distribuce softwaru

Jelikož Maemo 5 vychází z Debianu, používá jeho mírně upravený balíčkovací systém. Proto je například možné na Maemo vkládat do `control` souboru base64 kódovanou ikonu aplikace (která slouží k zobrazení ikony ve správci aplikací) nebo také vložit adresu projektového bugrackeru. Balíčky je také nutné optifikovat.

Distribuce softwaru k uživatelům probíhá pomocí hlavního repositáře zvaného *Extras*[44], který se skládá ze tří větví:

- `Extras` – stabilní repositář předinstalovaný na všech N900
- `Extras-Testing` – testovací repositář
- `Extras-Devel` – vývojový repositář

Do Extras-Devel posílají vývojáři svoje aplikace a mohou je odtud dále posílat do Extras-Testing. V Extras-Testing probíhá komunitní testování aplikací – tzn. že uživatelé testují jednotlivé aplikace a udělují jim kladné či negativní hodnocení. Aplikační balíček, který získá celkem 6 kladných hlasů, je po uplynutí desetidenní karantény vpuštěn do stabilního repositáře Extras.

V Maemo 5 jsou všechny balíky rovnocenné, libovolný balík v systému je tedy možné odinstalovat nebo nahradit. Tato skutečnost umožnila vznik nejen vznik komunitního kernelu, ale i projektu komunitní údržby distribuce, zvaného CSSU[45]. Vedlejším efektem plné kontroly nad balíčky je však to, že nevhodnou manipulací s balíky může uživatel systém dostat do stavu, ve kterém přestanou fungovat některé důležité funkce nebo systém dokonce ani nenabootuje.

24. tzv. FM transmitter

2.7.5 Vývoj softwaru

Maemo 5 oficiálně podporuje tvorbu grafických aplikací postavených na grafické knihovně GTK s rozšířením Hildon, které usnadňuje tvorbu aplikací pro dotykovou obrazovku. Dodatečně byla také přidána podpora pro vývoj aplikací založených na grafické knihovně Qt. Na Maemo 5 je s pomocí Qt možné vytvářet jak klasické QWidget aplikace, tak aplikace v QML/Qt Quick.

Podobně jako na jiných platformách se pro vývoj GTK aplikací používá jazyk C a pro vývoj Qt aplikací jazyk C++. Oficiálně je rovněž podporována tvorba GTK i Qt aplikací v jazyce Python, čehož využívá velké množství populárních programů určených pro Maemo 5[25].

Mnoho aplikací fungujících v příkazovém řádku na Maemo 5 funguje bez úprav, stačí je pouze zkompilovat.

2.7.6 Hardware

Mobilní operační systém Maemo 5 Fremantle byl vyvinut společností Nokia pro chytrý telefon N900[48] a na jiném zařízení nebyl nikdy nasazen nasazen.

Protože značná část kritických systémových komponent Maemo 5 je uzavřená, není komunitní port na jiný hardware pravděpodobný. Vzhledem k tomu, že společnost Nokia na konci roku 2012 dokonce převedla správu platformy Maemo na uživatelskou komunitu, nelze očekávat další „oficiální“ zařízení s Maemo 5.

2.8 MeeGo 1.2 Harmattan

MeeGo 1.2 Harmattan je další operační systém pro mobilní zařízení, který vyvinula firma Nokia. Třebaže jeho název vyvolává dojem, že tato platforma je založená na projektu MeeGo, navazuje tento operační systém ve své podstatě na projekt Maemo. Neoficiálně bývá proto „Harmattan“ označován jako *Maemo 6*.

2.8.1 Architektura

Harmattan se velmi podobá operačnímu systému Maemo 5, protože používá stejný balíčkovací systém, stejný systém pro práci s grafikou, zachovává i použití DBUSu nebo dostupnost Pythonu. O databázi kontaktů se

25. gPodder[46], FMRadio, modRana[47], PyGTKEdit, Mieru a další

Obrázek 2.5: Maemo 5 Fremantle na chytrém telefonu Nokia N900

i na Harmattanu stará Tracker[49], o video Gstreamer a o zvuk Pulseaudio. Mnoho balíčků z Maemo 5 se verzi pro Harmattanu liší pouze mírně vyšším číslem verze.

Hlavní odlišností od Maemo 5 je výchozí grafická knihovna – zatímco na Maemo 5 bylo rozhraní tvořeno kombinací GTK, Hildon & Clutter, na Harmattanu je celé rozhraní založeno na grafické knihovně Qt ve verzi 4.7. Další odlišností je „bezpečnostní systém“ Aegis, který se více či méně úspěšně snaží udržet integritu platformy.

Platformy Maemo a Harmattan se od sebe liší způsobem kompilace binárních komponent:

- pro Harmattan jsou zkompilovány s podporou hardwarového zpracování čísel s plovoucí desetinnou čárkou[26]
- Maemo 5 je naopak zkompilováno pouze s podporou softwarového zpracování čísel s plovoucí desetinnou čárkou[27].

Hlavní výhodou „hardfloat“ komponent je větší rychlost při zpracovávání čísel v plovoucí desetinné čárce. Hardfloat a softfloat jsou mezi sebou nekompatibilní[28], takže míchání binárních komponent Maemo 5 a Harmattanu většina vývojářů příliš nedoporučuje. Přesto existuje komunitní projekt, který se o to snaží, a to o podporu spouštění aplikací z Harmattanu na Maemo 5 v binární podobě bez nutnosti rekompilace[50].

26. hardware floating point, čili zkráceně „hardfloat“
27. software floating point, neboli „softfloat“
28. jejich kombinování může vést k nedefinovanému chování systému

Obrázek 2.6: N900 s vysunutou klávesnicí a vyklopeným stojánkem

2.8.2 Otevřenost

Podobně jako Maemo 5 obsahuje Harmattan ke škodě věci řadu uzavřených komponent. Jedná se např. o kompozitor, některé vstupní metody a ovladače grafické karty a GPS. K dalším sysytémovým komponentám jako jsou např. grafické frameworky Qt Components a MeeGo Touch či klávesnicový framework Maliit, zdrojové kódy k dispozici jsou.

Otevřenost Harmattanu velmi negativním způsobem ovlivňuje bezpečnostní framework Aegis[51], který ve svém výchozím nastavení zabraňuje modifikaci některých částí systému, tzn. nejde např. provádět chanegroot, připojit vzdálený systém přes sshfs anebo nahradit některé systémové balíčky. Aplikace navíc pro přístup k některým rozhraním OS potřebují tokeny, přidělované systémem Aegis. Tyto tokeny mohou mít pouze ty aplikace, které byly do systému nainstalovány z balíčku. Vzhledem k tomu, že tvorba a instalace balíčku trvá nezanedbatelnou dobu, zdržuje tento požadavek systému Aegis výrazným způsobem vývoj některých aplikací.

Výchozí nastavení Aegis uživatel nemůže navíc běžnými prostředky změnit. V této oblasti se však projevila iniciativa komunity, ze které vzešel projekt INCEPTION[52], který využívá implementačního nedostatku systému Aegis k předání kontroly nad systémem zpět uživateli. Po aplikaci INCEPTION Aegis sice stále běží, uživatel jej však může libovolně konfigurovat.

Další možností jak obejít Aegis, je nahrazení výchozího linuxového jádra komunitním jádrem. Nahrazení výchozího jádra způsobí deaktivaci bezpečnostního systému Aegis a odstranění všech výše jmenovaných omezení. Zároveň však některé části systému, které s Aegis počítají, mohou přestat fungovat.

Otevřenost vývoje

Vývoj MeeGo 1.2 Harmattan byl společností Nokia po vydání chytrého telefonu N9 ukončen. Ještě několik měsíců po vydání zbývající vývojáři zpracovávali hlášení o chybách od uživatelské komunity, po vydání aktualizace PR 1.3 ale veškerý oficiální vývoj platformy definitivně ustal. Většina vývojářů, kteří se na vývoji Harmattanu podíleli, z Nokie odešla, mnoho z nich se stalo zaměstnanci společnosti Jolla. Vzhledem k tomu, že některé N9 jsou doposud v záruce, lze pravděpodobně v budoucnu dobu očekávat kritické bezpečnostní aktualizace nebo aktualizace zabudovaných aplikací[29].

Nic také nenasvědčuje tomu, že by společnost Nokia měla v úmyslu vydat zdrojový kód uzavřených komponent platformy MeeGo 1.2 Harmattan.

2.8.3 Balíčkování

Podobně jako Maemo 5 využívá také Harmattan balíčkovací systém pocházející z Debianu. Jedinou podstatnou odlišností na Harmattanu je možnost přidání manifestu pro bezpečnostní systém Aegis. Bez manifestu, který požaduje odpovídající oprávnění, nemá aplikace přístup k některým datům, rozhraním a funkcím. Bez manifestu nelze např. používat GPS, kameru či pracovat s databází kontaktů.

Výrazně odlišný je však způsob distribuce softwaru. Neexistuje přímý ekvivalent předinstalovaného komunitního repositáře Extras, většina aplikací je distribuována prostřednictvím repositáře/elektronického obchodu Nokia Store[30][53]. Nokia Store přijímá proprietární i otevřené aplikace a vývojář může rozhodnout, jestli jeho aplikace bude k dispozici za úplatu nebo zcela zadarmo. Aplikace, odeslané do Nokia Store, procházejí před zveřejněním manuální kontrolou kvality, která v závislosti na vytížení kontrolního týmu způsobuje zhruba týdenní prodlevu mezi odesláním a zveřejněním aplikace. Velkým nedostatkem Nokia Store je chybějící podpora pro přidávání závislostí, aplikace se tak musí omezit buď to pouze na knihovny

29. příkladem může být aktualizace Twitter klienta na začátku roku 2013
30. dříve známého jako Ovi Store

dostupné ve výchozích repositářích, nebo musí všechny závislosti začlenit do svého balíčku.

Komunitní alternativou Nokia Store měl být projekt Apps for MeeGo[54], který měl sloužit jako komunitní aplikační repositář nejen pro Harmattan, ale i pro další platformy, které jsou založené na příbuzném projektu MeeGo. Apps for MeeGo využívá OBS pro správu instalačních balíčků a podobně jako Maemo Extras funguje na základě komunitní kontroly kvality. Vzhledem k absenci jakýchkoli informací o dalším vývoji tohoto projektu a poměrně malému množství dostupných aplikací, je velmi pravděpodobné, že vývoj tohoto projektu byl opuštěn.

Kvůli absenci fungujícího komunitního repositáře je tak mnoho aplikací pro Harmattan distribuováno ze samostatných repositářů, které provozují jednotliví vývojáři, nebo přímo v podobě instalačních balíčků.

2.8.4 Vývojové prostředí

Oficiálním vývojovým prostředím pro MeeGo 1.2 Harmattan je Qt SDK[55]. Jako kompilační prostředí je použit systém Scratchbox[56], jako IDE slouží Qt Creator[57]. K dispozici je také softwarový simulátor zařízení Nokia N9.

Použití Qt SDK však není nutnou podmínkou pro vývoj aplikací na Harmattan. Pokud není nutné zdrojový kód aplikace kompilovat, jako je tomu u jednoduchých aplikací používajících pouze QML a Javascript nebo aplikací psaných v Pythonu, postačí pro tvorbu aplikace nástroje, dostupné v repositářích většiny linuxových distribucí.

2.8.5 Hardware

Jediným oficiálně vydaným zařízením s MeeGo 1.2 Harmattan je chytrý telefon Nokia N9[58]. Jedná se o dotykový telefon bez klávesnice s jednojádrovým procesorem, 1GB RAM a AMOLED obrazovkou.

Kromě N9 vyrobila společnost Nokia ještě jedno zařízení s Harmattanem – a to chytrý telefon Nokia N950[59][60]. Toto zařízení však bylo vyrobeno pouze v počtu několika tisíc kusů a nebylo nikdy oficiálně uvedeno na trh. Veškeré existující exempláře N950 byly přiděleny vývojářům v rámci vývojářských programů, nebo sloužily jako hlavní cena v soutěžích sponzorovaných společností Nokia. N950 pohání stejný SOC jako N9, liší se však rozklápěcím tělem vyrobeným z hliníku, které obsahuje hardwarovou klávesnici. Dale se N950 od N9 liší použitím LCD obrazovky místo AMOLED a absencí technologie NFC[4][5].

Obrázek 2.7: MeeGo 1.2 Harmattan na chytrém telefonu Nokia N9

2.9 Android

Android sice obsahuje linuxový kernel, od klasických linuxových distribucí se však výrazně odlišuje rozdílným pojetím *userspace*. Dalo by se konstatovat, že se oproti klasickému *GNU/Linux* jedná o *Linux-GNU*. V userspace nejsou totiž použity klasické GPL licencované GNU utility, které byly nahrazeny *ořezaným* BSD licencovaným ekvivalentem. Androidu však chybí i další součásti, běžné se vyskytující v klasických linuxových distribucích, jako jsou systém pro správu oken, předinstalovaný emulátor terminálu, DBUS či balíčkovací systém s podporou automatického řešení závislostí. Také samotný linuxový kernel v Androidu obsahuje řadu patchů nekompatibilních s hlavní větví jádra. Jejich počet se však v poslední době postupně daří redukovat[61].

2.9.1 Architektura

Základem platformy Androidu je linuxový kernel, další systémovou vrstvou jsou ovladače zařízení a knihovna jazyka C, známá jako Bionic[62]. Bionic není bohužel kompatibilní s GNU libc či EGLIBC používanou na běžných linuxových distribucích, což např. ztěžuje použití binárních ovladačů pro Android v jiných linuxových distribucích.

Na systémovém základě, který tvoří jádro, ovladače a Bionic, pak běží

virtuální stroj zvaný Dalvik, umožňující běh aplikací v jazyce podobném Javě. O přístup ke grafice se na Androidu stará systém Surface flinger. Dále je na Androidu použit systém pro sandboxování aplikací, systém pro meziprocesovou komunikaci, zvaný Binder a systém pro správu balíčků.

2.9.2 Otevřenost

Veškerý zdrojový kód Androidu je k dipozici pod svobodnými licencemi, úpravy jádra pod GPL a zbytek systému pod BSD licencí. Vývoj systému je naopak zcela uzavřený a plně řízený společností Google a jejími korporátními partnery z Open Handset Aliance[2]. Členové OHA mohou nejen ovlivňovat vývoj Androidu, ale mají také po celou dobu vývoje přímý přístup ke zdrojovým kódům. To jim dává určitou konkurenční výhodu, spočívající v rychlejší uvedení produktů s novou verzí Androidu na trh. Plné zveřejnění zdrojového kódu nové verze Androidu totiž zpravidla následuje s určitým zpožděním až po vydání prvních zařízení s touto verzí.

Dalším potenciálním rizikem pro otevřenost projektu Android je skutečnost, že Google a OHA jsou výhradními vlastníky licence ke zdrojovému kódu, samozřejmě s výjimkou linuxového jádra. Google tedy může kdykoli přestat zdrojové kódy distribuovat a tento postup bude přesto licenčně zcela v pořádku.

Míru otevřenosti Androidu ovlivňuje také distribuování většiny kódu pod licencí BSD. Od výrobců zařízení tak není na rozdíl od GPL vyžadováno poskytování provedených změn. Mnoho zařízení s Androidem obsahuje částečně i díky tomu řadu proprietárních komponent a změn bez zveřejněného zdrojového kódu. Tento fakt znesnadňuje pozici vývojářů a uživatelů, kteří se na takovém zařízení snaží provádět úpravy systému.

2.9.3 Balíčkování

Software pro Android je distribuován v balíčcích typu APK[63], což je balíčkovací formát vycházející z archivačního souborového formátu JAR[64]. Interně se jedná o archiv, který obsahuje aplikační soubory a instalační metadata.

Pro Android existuje řada repositářů, které mohou pro uživatele sloužit jako zdroj aplikací. Největší repositář provozuje přímo společností Google pod názvem *Google Play*[65]. Jednotlivé repositáře nabízejí aplikace všeho druhu a obsahují také systém, který umožňuje jejich hodnocení a posílaní zpětné vazby od uživatelů k vývojáři aplikace[31]. Kontrola aplikací odesla-

31. tyto systému jsou však ve většině případů pouze jednocestné: vývojář totiž většinou

ných do Google Play je plně automatická, díky tomu jsou nové aplikace dostupné ke stažení v řádu desítek minut, ne dnů, tak jak je tomu v případě manuálně prováděné kontroly kvality.

Google Play a a žádaný další alternativní repositář, se kterým jsem dosud setkal, nepodporuje závislosti mezi balíčky. Řešení závislostí tak na Androidu musí suplovat aplikace. Potřebné knihovny jsou tedy buď to přibalovány do instalačního APK souboru nebo při prvním spuštění aplikace stahovány z Internetu. Další možným řešením je nainstalování speciální aplikace, která plní úlohu správce balíků. Tato aplikace má na starost stahování, aktualizaci a poskytování knihoven dalším aplikacím. Jako příklad poslouží program Ministro, což je součást projektu Necessitas. Ministro se stará se o efektivní poskytování Qt knihoven aplikacím, které tuto grafickou knihovnu používají na Androidu.

Android ve výchozím nastavení rovněž umožňuje přímou instalaci aplikací z APK souborů. Někteří výrobci zařízení nebo mobilní operátoři však tuto funkci vypínají.

Z vývojářského hlediska je také dobré vědět, že Android po instalaci aplikace uchovává kopii instalačního balíčku. Tento přístup umožňuje vrácení aplikace do výchozího stavu bez nutnosti opětovného stažení instalačního balíku[32]. Vedlejším efektem však je, že aplikace zabírají více místa (velikost nainstalované aplikace + velikost balíku).

Hry a další aplikace, které vyžadují velké množství lokálně dostupných dat[33], proto raději tato data při prvním spuštění aplikace stahují z Internetu.

2.9.4 Vývojové prostředí

Primárním vývojovým prostředím pro platformy Android je tzv. Android SDK[66], založená na frameworku Eclipse[67] a zaměřená na vývoj aplikací v programovacím jazyce podobném Javě. Další oficiálně podporovanou možností je Android NDK, což je prostředí umožňující vývoj nativních aplikací v C/C++ a jejich kompilaci pro Android. NDK využívají zejména hry či projekty založené na knihovnách napsaných v C/C++.

Zajímavou alternativou k oficiálním vývojovým nástrojů je Necessitas SDK[68], která umožňuje tvorbu aplikací v C++/QML/Javascriptu pro Android s využitím grafické knihovny Qt. K dispozici je také IDE Qt Creator. Na Necessitas SDK pak staví nadějný projekt pydroid[69], který zpřístup-

---

nemá možnost na kritiku uživatelů reagovat

32. také díky tomu nemusí po uživateli vyžadovat instalační soubor, pokud byla aplikace manuálně nainstalována ze souboru

33. složitější hry, slovníky, navigační systémy

Obrázek 2.8: Android 4.0 na tabletu HP TouchPad (komunitní distribuce CyanogenMod 9)

ňuje tvorbu aplikací pro Android v Pythonu a Qt.

2.9.5 Hardware

Zařízení s Androidem jsou dispozici v mnoha různých formátech, převážně se však jedná a chytré telefony a tablety. Co se týče celkového počtu doposud vyrobených zařízení, oznámila společnost Google v září 2012[70], že bylo již aktivováno celkem 500 milionů zařízení s Androidem a denně je jich aktivováno kolem 1.3 milionu.

2.10 Tizen

Tizen[71] je operační systém pro mobilní zařízení, vyvíjený společně firmami Samsung a Intel. Oficiálně bývá Tizen označován jako pokračování projektu MeeGo (na kterém se firma Intel dříve podílela), z projektu MeeGo však neobsahuje téměř nic. Tizen je ve skutečnosti převážně založen na recyklaci mobilních platforem, který vyvinula firmou Samsung pod názvy LiMo a Bada.

Obrázek 2.9: Tizen na vývojářském zařízení zvaném *lunchbox*

2.10.1 Otevřenost

Třebaže je Tizen navenek prezentován jako otevřená platforma, probíhá většina vývoje kompletně za zavřenými dveřmi. Ačkoli se má oficiálně jednat o spolupráci firem Samsung a Intel, má v rozhodování o dalším směru vývoje evidentně vždy první slovo firma Samsung. Tato provádí dokonce jednostranná rozhodnutí o dalším vývoji Tizenu, která bez jakéhokoli předchozího upozornění poškozují práci vývojářů firmy Intel i dalších subjektů, snažících se na vývoji se Samsungem spolupracovat[72][73].

Tento stav má nepříznivý vliv na vývojářskou komunity. Je pravděpo-

Obrázek 2.10: Rozebrané zařízení *lunchbox*

dobné, že pouze málo vývojářů bude ochotno věnovat svůj čas platformě Tizen, když jejich snažení může přijít kdykoli nazmar kvůli další předem neohlášené zásadní změně platformy.

Podobně nepříznivá je situace v oblasti zdrojového kódu této platformy. Zdrojové kódy nejsou během vývoje vůbec k dispozici a podobně jako na Androidu jsou zveřejněny teprve po vydání nové verze Tizenu. Zdrojové kódy jsou navíc distribuovány pod spletí různých licencí, z nichž některé vůbec nejsou open source kompatibilními licencemi. Licenční zmatek tak brání využití komponent vyvinutých pro Tizen na jiných mobilních platformách.

2.10.2 Balíčkování

Balíčkovací systém Tizenu byl během vývoje platformy několikrát bez jakéhokoli předchozího upozornění změněn. Po určitou dobu byl používán debianí balíčkovací systém, v současnosti je s největší pravděpodobností používán balíčkovací systém RPM.

Vzhledem k tomu, že žádná zařízení platformy Tizen, určená pro koncové uživatele, doposud vydána nebyla, není jasné, jak bude probíhat distribuce aplikací směrem od vývojářů k uživatelům.

2.10.3 Vývojové prostředí

Po oficiálním zahájení projektu byl Tizen označován za platformu podporující pouze webové aplikacích. Společně s vydáním Tizenu 2.0 však do něj byla přidána také podpora pro tvorbu nativních aplikací[34]. Pro Tizen je k dispozici SDK, včetně simulátoru zařízení.

2.10.4 Hardware

Na trh zatím nebylo uvedeno ani jedno zařízení s Tizenem určené pro koncové zákazníky. Existuje však již několik prototypů pro vývojáře, z nichž se nejčastěji vyskytuje prototyp zvaný „lunchbox“, vyráběný firmou Samsung.

2.11 Ubuntu Touch

Ubuntu Touch je mobilní platforma vyvíjená pro chytré telefony a tablety společností Canonical. Tato platforma je poměrně nová a její vývoj teprve

34. vycházející z platformy Bada

nedávno odstartoval. Momentální stádium vývoje bylo v únoru 2013 společností Canonical označeno jako „developer preview“.

2.11.1 Architektura

Platforma Ubuntu Touch byla navržená tak, aby byla kompatibilní s binárními ovladači pro Android. Kompatibilita je zajištěná prostřednictvím libhybris[74], tj. knihovny vyvinuté Carstenem Munkem, vývojářem společnosti Jolla, který pracuje na vývoji platforem Mer a Sailfish. Kompatibilita s ovladači pro Android znamená, že je možné Ubuntu Touch triviálně portovat na většinu moderních zařízení podporujících Android.

Také zbytek Ubuntu Touch se skládá převážně z různých open source knihoven, vyvinutých pro jiné projekty. Ubuntu Touch používá grafický systém SurfaceFlinger, původně vytvořený firmou Google pro Android; pro telefonii slouží framework ofono, pro správu účtu account-sso a pro zpracování vstupu od uživatele slouží framework Maliit – všechno jsou to komponenty původně vyvinuté pro Maemo a MeeGo.

Ubuntu Touch má podle oficiálních vyjádření tu zajímavou vlastnost, že dostatečně výkonná zařízení by po „zadokování“, připojení monitoru a klávesnice měla uživateli nabídnout plnohodnotné desktopové prostředí. Zatím není jasné, nakolik tato funkce bude v praxi použitelná.

Grafické rozhraní Ubuntu Touch by mělo podporovat zobrazení více aplikací současně[35] – což je funkci, kterou většina mobilních platforem nedisponuje.

2.11.2 Otevřenost

Zdrojové kódy Ubuntu Touch jsou k dispozici pod open source licencemi. Třebaže vývoj zezačátku probíhal několik měsíců tajně, po oficiálním zveřejnění projektu se zdá, že vývoj přešel do „otevřeného“ režimu. Repositáře zdrojového kódu jsou nyní veřejně k dispozici, existují veřejné komunikační kanály a do vývoje je možné se kdykoli zapojit[75].

2.11.3 Balíčkování

Stejně jako distribuce Ubuntu pro stolní počítače používá i mobilní platforma Ubuntu Touch debianí balíčkovací systém. Jaká bude struktura repositářů a jestli budou podporovány závislosti mezi balíčky – to v této fázi vývoje ještě není jasné.

35. v kontextu Ubuntu Touch je tato funkce označována jako „side stage“

2.11.4 Vývojové prostředí

Pro vývoj aplikací je k dispozici SDK, zaměřená na vývoj aplikací v C++ a Qt Quick. Grafická knihovna Qt je dostupná ve verzi 5.0.

Vývoj aplikací v Pythonu je zatím komplikován chybějícím rozhraním, které by plně podporovalo všechny funkce Qt 5. Teoretickým řešením tohoto problému by mohla být knihovnu PyQt, která podporuje Qt 5 za předpokladu, že bude použita pouze funkcionalita na úrovni Qt 4. Přestože zatím projekty PySide ani PyQt nezveřejnily časový plán pro plnou podporu všech funkcí Qt 5, je to snad jen otázkou nepříliš vzdálené budoucnosti. Pak bude možné i na Ubuntu Touch vytvářet plnohodnotné aplikace v Pythonu.

2.11.5 Hardware

Navzdory rané fázi vývoje již Ubuntu Touch podporuje instalaci na celou řadu mobilních zařízení, určených původně pro Android[76]. Oficiálně jsou podporovány chytré telefony Galaxy Nexus a Nexus 4 a tablety Nexus 7 a Nexus 10. Mnoho dalších zařízení je dále podporováno komunitními porty.

2.12 BlackBerry 10

Jedná se o platformu pro mobilní zařízení, kterou vyvíjí kanadská firma BlackBerry[36]. Platforma BlackBerry 10[77] je založena na RTOS QNX, který je již dlouhou dobu používán v mnoha průmyslových odvětvích.

2.12.1 Architektura

BlackBerry 10 je postavena na RTOS QNX, který zajišťuje jak nízkoúrovňové funkcionalitu, tak veškeré základní systémové utility. Operační systém je sice POSIXově kompatibilní, paleta ve výchozím stavu dostupných utilit je však velmi malá, a to i ve srovnání s omezenou nabídkou utilit na Androidu. Systém poskytuje nízkoúrovňové rozhraní pro jazyk C/C++ zpřístupňující většinu funkcí: přístup ke grafice, práci s audiem, senzorová data a tak podobně. Na tomto nízkoúrovňovém rozhraní staví pravděpodobně všechny vysokoúrovňové frameworky používané na BlackBerry 10.

Vzhledem k tomu, že BlackBerry 10 je proprietární operační systém a jeho zdrojové kódy nejsou k dispozici, je zjišťování podrobnějších informací o jeho struktuře obtížné.

36. do konce ledna 2013 se tato firma jmenovala RIM – Research In Motion

2.12.2 Otevřenost

Vzhledem k tomu, že je BlackBerry 10 proprietární OS, nejsou jeho zdrojové kódy volně k dispozici a vývoj plně probíhá za zavřenými dveřmi. Některé oficiálně podporované frameworky pro vývoj aplikací jako např. grafická knihovna Qt jsou však open source a společnost BlackBerry se dokonce podílí na jejich vývoji.

Hlavní grafický framework pro vývoj aplikací zvaný Cascades staví na Qt, jeho zdrojový kód je však uzavřený.

2.12.3 Balíčkování

BlackBerry 10 využívá balíčky ve formátu BAR. Jedná se opět o klasický balíčkovací formát, tedy archiv, ve kterém jsou metadata a aplikační soubory. Tento formát také podporuje kryptografické ověření integrity balíčku.

Hlavním distribučním kanálem je softwarový repositář & obchod BlackBerry World, který je přeinstalovaný na všech zařízeních s operačním systémem BlackBerry 10. Aby byly aplikace v BlackBerry World zveřejněny, musí splňovat řadu podmínek a projít manuální kontrolou kvality. V závislosti na vytížení testovacího týmu trvá kontrola kvality zhruba týden.

Instalace aplikací z jiných zdrojů než z „obchodu“ BlackBerry World je sice možná, daná aplikace však musí být podepsaná platným vydavatelským klíčem. Aplikace tak v zásadě nemusí projít kvalitou kontroly a nemusí splňovat pravidla oficiálního repositáře, vývojář si však musí od společnosti BlackBerry vyžádat platný vydavatelský klíč. Aplikační balíčky, které nejsou podepsané platným klíčem, nedovolí systém vůbec nainstalovat. Aplikace, které nebyly nainstalovány z podepsaného balíčku, neumožní systém ani spustit.

Díky tomu, že lze nainstalovat podepsané aplikace mimo hlavní repositář, není platforma BlackBerry 10 úplná *walled garden*. Společnost BlackBerry však stále může nepřímo rozhodovat o tom, který software bude na platformě fungovat, a to prostřednictvím přidělování vydavatelských podepisovacích klíčů či revokací těch existujících.

2.12.4 Vývojové prostředí

BlackBerry 10 je oproti ostatním mobilním platformám poměrně atypická oficiální podporou několika různých SDK. Kromě primární SDK, která podporuje vývoj aplikací v C++ s použitím grafického frameworku Cascades, podporuje BlackBerry 10 rovněž:

- vývoj v C/C++ s možností využití grafické knihovny Qt

- vývoj v HTML5 s pomocí WebWorks SDK
- vývoj v Adobe Air
- aplikace napsané pro platformu Android

Kromě oficiálních vývojářských nástrojů existují také komunitní projekty, umožňující tvorbu grafických aplikací v jazyce Python. Z Pythonu je díky projektu Tart možné použít framework Cascades. Díky mému portu PySide na BlackBerry 10 byly zpřístupněny také klasické Qt knihovny[37]. Použití PySide na této platformě se podrobněji věnuje kapitola popisující portování aplikace modRana na platformu BlackBerry 10.

2.12.5 Hardware

Ještě před oficiálním globálním představením platformy BlackBerry 10 byl předchůdce tohoto operačního systému nasazen na sedmipalcový tablet, zvaný PlayBook. 30. ledna 2013 byla platforma BlackBerry 10 oficiálně uvedena a při této příležitosti byla oznámena dvě první BlackBerry 10 zařízení, zvaná Z10[78] a Q10[79].

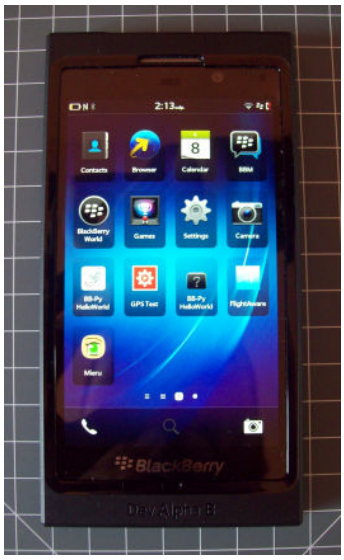
BlackBerry Z10 je dotykový telefon s HD-ready obrazovkou o úhlopříčce 4,2 palce, procesorem se dvěma jádry a 2 GB RAM. Q10 se od Z10 liší menší obrazovkou (720x720 pixelů s úhlopříčkou 3,1 palce), pod kterou se nachází čtyřřadá QWERTY klávesnice.

2.13 webOS

WebOS[80][81] byl ambiciózní operační systém pro mobilní zařízení, částečně založený na webových technologiích. S jeho vývojem začala firma Palm, která v lednu 2009 na trh uvedla první zařízení s webOS OS, chytrý telefon Palm Pre[82]. V roce 2010 koupila Palm firma Hewlett-Packard, která s vývojem webOS dál pokračovala a vydala další chytré telefony řady Pre a Veer a tablet TouchPad[83]. V srpnu 2011[84] však společnost HP náhle oznámila ukončení vývoje webOS a všech webOS zařízení. Oficiální ukončení vývoje vyvolalo ze strany vývojářů aplikací postupné opuštění platformy. V průběhu roku 2012 byl webOS uvolněn jako open source software pod MIT licencí, což vedlo ke vzniku projektu Open webOS[85]. V únoru 2013 ohlásila firma HP prodej webOS společnosti LG, která plánuje nasazení webOS na svých „chytrých“ televizorech[86].

Ukončení vývoje webOS mělo jeden zajímavý vedlejší efekt: skladové zásoby webOS zařízení byly vyprodány hluboko pod výrobní cenou, takže

37. použít lze QWidget, QML i Qt Components



Obrázek 2.11: BlackBerry 10 na vývojářském zařízení Dev Alpha B

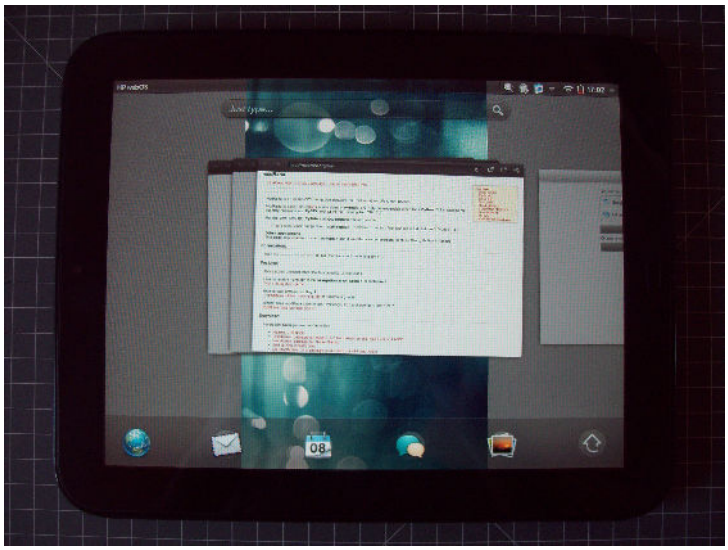
tablet TouchPad tak bylo možné sehnat za 100$ v 16 GB verzi[87] a za 150$ v 32 GB. Protože si v tomto výprodeji TouchPad pořídilo mnoho vývojářů, objevují se stále nové zajímavé projekty, používající HP TouchPad jako výchozí zařízení. Jedná se např. o projekt vývoje otevřených ovladačů pro mobilní GPU Adreno[88], dále o projekt podpory distribucí založených na Meru[89] nebo projekt na zprovoznění Ubuntu Touch[90].

Ve zmíněném výprodeji se mi podařilo si HP TouchPad za zvýhodněnou cenu opatřit. Na tablet jsem nainstaloval Android a používám jej především pro vývoj aplikací pro Android s použitím Pythonu a grafické knihovny Qt. Díky TouchPadu jsem se také podrobněji seznámil s platformou webOS[38], která v praxi funguje velmi dobře, je však viditelně poznamenaná ukončením svého vývoje a nedostatkem aplikací třetích stran.

2.13.1 Architektura

Základ webOS se velmi podobal běžné linuxové distribuci a byl vytvářen s pomocí systému Openembedded[91]. Pro práci s grafikou nebyl použit X server, ale vlastní systém, založený na knihovně DirectFB[92]. Ve verzi 3.0 byla část uživatelského rozhraní implementována s využitím grafické knihovny Qt[39], která však ke škodě věci není dostupná pro aplikace třetích

38. Android jsem nainstaloval v režimu dualboot
39. odhadem verze v intervalu od 4.6 do 4.7



Obrázek 2.12: webOS na tabletu HP TouchPad

stran. Webový prohlížeč je postaven na vykreslovacím jádře Webkit[93].

2.13.2 Otevřenost

webOS začal svůj vývoj jako proprietární operační systém, který však obsahoval mnoho částí pod otevřenou licencí jako např. linuxové jádro nebo balíčkovací systém ikpg. Vývoj distribuce probíhal tzv. za zavřenými dveřmi. V průběhu roku 2012 však byl kompletní zdrojový kód webOS zveřejněn pod svobodnou licencí. Další vývoj nyní probíhá pod taktovkou projektu Open webOS již zcela veřejně prostřednictvím projektu na Githubu[94].

2.13.3 Balíčkování

Jako balíčkovací systém slouží platformě webOS ipkg[95], používaný rovněž několika dalšími embedded distribucemi. Kromě výchozího aplikačního repositáře, zvaného App Catalog, lze na webOS zařízení doinstalovat komunitní repositář, zvaný Preware[96]. Od verze 0.9.4 podporuje Preware dokonce závislosti mezi balíky[97], což je funkcionalita málokdy dostupná na poli mobilních platforem.

2.13.4 Vývojové prostředí

Pro vývojáře aplikací je k dispozici SDK, která umožňuje tvorbu aplikací založených na webových technologiích, Javascriptu a na prostředích Mojo a Enyo.

Tzv. PDK umožňuje také vývoj nativních aplikací s využitím multimediální knihovny SDL. Většinu aplikací, vyvinutých pomocí PDK, tvořily až na několik výjimek hry. Mezi výjimky patřily např. emulátor terminálu, čtečka e-knih nebo port X serveru pro běh v aplikačním okně.

Vývojářské komunitě se rovněž na platformě webOS podařilo rozjet knihovnu PyGame[40] a několik jednoduchých her napsaných v Pythonu a založených na PyGame[98][99][100].

2.13.5 Firefox OS

Firefox OS je operační systém pro mobilní zařízení společnosti Mozilla, která je známé především vývojem internetového prohlížeče Firefox, emailového klienta Thunderbird nebo systému pro sledování chyb Bugzilla.

Základ systému tvoří velmi jednoduchá linuxová distribuce, založená částečně na Androidu. Nízkoúrovňová kompatibilita s Androidem umožňuje platformě Firefox OS využít již existující binární ovladače určené pro Android. Součástí distribučního základu je dále vrstva, poskytující abstraktní rozhraní pro přístup k hardwaru[41].

Veškeré vysokoúrovňové funkce Firefox OS, včetně systémových aplikací, jsou implementovány pomocí W3C[101] standardizovaných rozhraní v HTML 5 a v Javascriptu. V HTML5 a Javascriptu musí být napsány též aplikace třetích stran, protože vývoj nativních aplikací není v současnosti vůbec podporován. Zařízení platformy Firefox OS mají být podle zatím dostupných informací poměrně levná a k dostání by měla být nejdříve v méně vyspělých zemích. Uvedení prvních zařízení s Firefox OS na trh je předpokládáno v průběhu roku 2013.

Během konference OpenMobility 2013 v Bratislavě jsem měl možnost se s vývojovou verzí Firefox OS osobně seznámit a mohu konstatovat, že na mě tento systém dobrý dojem nezanechal. Firefox OS podporuje totiž pouze standardizovaná webová API. Žádný jiný mobilní operační systém však tato rozhraní v současnosti nepodporuje a je otázkou, jestli je někdy v budoucnu plně podporovat bude. Aplikace napsané pro Firefox OS proto v současnosti nefungují na žádném jiném mobilním OS.

---

40. rozhraní jazyka Python pro SDL
41. tzv. HAL – Hardware Abstraction Layer

Rozhodnutí vývojářů Firefox OS neumožnit vývoj nativních aplikací znamená, že při tvorbě aplikace nelze využít ani jednu z nepřeberného množství existujících nativních knihoven. Veškerá funkcionalita knihoven tak musí být pro využití v aplikaci přepsána do Javascriptu. Implementace některých funkcí běžných pro nativní aplikace nemusí být kvůli omezenému rozsahu „webových" rozhraní možná. Příkladem takové funkcionality může být správce souborů, kterého kvůli chybějícímu „webovému" rozhraní pro přímou práci se soubory na této platformě nelze vůbec implementovat.

Dalším aspektem Firefox OS, který mne osobně velmi znepokojuje, je použití bezpečnostního frameworku[102], který aplikace rozděluje do 3 kategorií podle důvěryhodnosti. Nejdůvěryhodnější, tzv. certifikované, aplikace, ke všem podporovaným hardwarovým rozhraním jako SMS, Bluetooth nebo telefonie plný přístup. Za certifikované aplikace jsou považovány pouze ty, které ověřil operátor nebo výrobce zařízení a které jsou ve výchozím stavu systému předinstalovány.

Další méně důvěryhodnou kategorií jsou tzv. privilegované aplikace, mající přístup k podmnožině méně kritických rozhraní z množiny těch, která jsou dostupná certifikovaným aplikacím. Aby se aplikace stala privilegovanou, musí být prověřena a digitálně podepsána autorizovaným aplikačním „tržištěm".

Nejméně důvěryhodné jsou tzv. „webové" aplikace, které nepotřebují ověření třetí strany. Webové aplikace mají přístup pouze k omezené množině „bezpečných" API.

V dokumentu, který toto rozdělení popisuje[102], není nikde uvedeno, jestli může uživatel s výchozím nastavením důvěryhodnosti jednotlivých aplikací manipulovat či ne. Když jsem se na to zeptal vývojáře ze společnosti Mozilla během jeho přednášky na OpenMobility 2013, nebyl mi schopen jasně odpovědět. Pokud uživatel opravdu nemůže s úrovněmi oprávnění pro aplikace manipulovat, je Firefox OS *walled garden* podobná platformě iOS či Windows Phone. Na rozdíl od nich by však absolutní kontrolu nad platformou neměl její výrobce, jak je to mu u iOS a Windows Phone, ale výrobce zařízení daného zařízení a případně mobilní operátor distribuující zařízení s Firefox OS.

Z tohoto úhlu pohledu se Firefox OS podobá platformě iOS také tím, že se snaží omezit, které technologie mohou vývojáři aplikací používat. Platforma iOS se pokouší prosadit Objective C na úkor všech ostatních technologií, Firefox OS pak vyloženě upřednostňuje pouze a jen HTML 5 + Javascript. Aplikace používající jiné technologie jsou na obou platformách v zásadě zakázány, přestože hardware jejich zařízení použití alternativních

technologií nebrání.

2.14 Další mobilní platformy

2.14.1 SHR

SHR[103][42] je mobilní linuxová distribuce pro otevřený chytrý telefon OpenMoko Neo FreeRunner (GTA02)[104]. Podporuje i další mobilní telefony, jako je např. Nokia N900[105]. Jako grafické prostředí na SHR slouží Illume 2, postavené na EFL[106]. Sestavení distribuce probíhá za použití systému Openembedded[91] a pro vytvoření jednotného API k hardwaru mobilních aplikací slouží middleware FSO[107].

Projekt SHR se v současnosti nezdá být příliš aktivní.

2.14.2 QtMoko

QtMoko[108] je linuxová distribuce určená pro otevřené chytré telefony:

- OpenMoko Neo FreeRunner (GTA02)[104]
- OpenPhoenux (GTA04)[109][110].

Distribuce QtMoko je založená na Debianu a na modifikovaném grafickém prostředí Qtopia, které vyvinula firma Trolltech[43]. QtMoko vyvíjí nezávislá vývojářská komunita. Vývoj QtMoko je velmi aktivní a nová verze distribuce vychází přibližně jednou za měsíc.

2.14.3 Cordia

Mobilní operační systém Cordia[111] byl založený na grafické knihovně Hildon a na prostředí Hildon desktop, používaném na zařízení Nokia N900 v systému Maemo. Zdrojový kód Hildonu byl pročištěn a upraven tak, aby Hildon fungoval s novými verzemi GTK a akcelerované grafické knihovny Clutter. Jako distribuční základ byl použit Mer.

Cordia byla primárně zaměřená na tablety a prvním zařízením měl být přeznačený sedmipalcový tablet čínské provenience. Tvůrcům distribuce se ale nepodařilo najít žádného čínského výrobce, který by byl ochotný při relativně malém objemu objednávaných zařízení dodat všechny potřebné ovladače či dokumentaci nutnou pro jejich vytvoření.

V současnosti je projekt Cordia neaktivní: poslední aktualizace stránek projektu je z května 2012.

---

42. Simple Hybrid Release
43. původní tvůrce grafické knihovny Qt

2.14.4 Seadot

Seadot[112] je komunitní distribuce určená pro zařízení typu tablet-pc jako např. x86 tablet ExoPC[113](prodávaný též pod názvem WeTab). Distribuce je založená na Meru, ke kterému přidává vlastní uživatelské rozhraní, zvané Enigma[114].

2.14.5 iOS

Mobilní operační systém iOS[115], který vytvořila firma Apple[116], běží na chytrých telefonech iPhone, tabletech iPad a multimediálních přehrávačích iPod Touch. Jedná se o velmi uzavřenou platformu pod absolutní kontrolou firmy Apple. iOS je ukázkovým příkladem *walled garden*, protože na zařízeních s iOS nelze instalovat aplikace mimo hlavní aplikační repositář, zvaný AppStore. Veškeré aplikace třetích stran tak musí splňovat drastické podmínky, které firma Apple stanovila pro přijetí aplikace do AppStore.

Do září roku 2010 například aplikace nesměly obsahovat žádný intepretovaný kód[117] a pravidla pro přijetí aplikace také zakazují přijetí aplikací licencovaných pod GPL[118].

Do iOS ve verzi 4.0 nepodporoval systém multitasking pro aplikace třetích stran a rovněž ve verzi 5.0 jsou možnosti běhu více aplikací současně velmi omezené.

2.14.6 Windows Phone

Windows Phone je mobilní platforma pro chytré telefony, kterou vyvinula firma Microsoft. Jedná se o plně uzavřenou platformu, která stejně jako iOS dokonale splňuje definici *walled garden*. Společnost Nokia se v únoru 2011 po opuštění projektu MeeGo rozhodla přijmout Windows Phone jakou svou hlavní mobilní platformu. Zařízení s Widnows Phone nabízí Nokia pod označením Lumia.

2.15 Tabulkové srovnání platforem

Nápad udělat tabulkové srovnání mobilních platforem vznikl na základě tabulky sestavené Thomasem Perlem v roce 2012[119]. Některá data použitá v následujících tabulkách pocházejí z této tabulky.

Platformy jsou v tabulkách uvedeny ve stejném pořadí, ve kterém jsou v této kapitole popsány. Pokud pro danou tabulku platforma není relevantní, je vynechána.

platforma	vznik	uvedení na trh	vývoj
MeeGo	2010	2010 netbooky	neaktivní
Mer	2011	2011	aktivní
Nemo Mobile	2011	2011	aktivní
Sailfish	2012	2013	aktivní
Plasma Active	2011	2011	aktivní
Maemo 5	2009	11.11.2009	komunita
MeeGo 1.2 Harmattan	2011	září 2011	neaktivní
Android	2005	2007	aktivní
Tizen	2011	2013 ?	aktivní
Ubuntu Touch	2013	2013 ukázka	aktivní
BlackBerry 10	2011	30.1.2013	aktivní
webOS	209	2009	komunitní
Firefox OS	2012	2013	aktivní
SHR	2009	2009	aktivní
QtMoko	2009	2009	aktivní
iOS	2007	2007	aktivní
Windows Phone	2010	2010	aktivní

Tabulka 2.1: Přehledová tabulka se základními údaji o popisovaných mobilních platformách

2.15.1 Architektura

platforma	grafický systém	IPC	vstup	multimedia	libc
MeeGo	X server	DBUS	Maliit	GStreamer	glibc
Mer	X server	DBUS	Maliit	GStreamer	glibc
Nemo Mobile	X server	DBUS	Maliit	GStreamer	glibc
Sailfish	X server	DBUS	Maliit	GStreamer	glibc
Plasma Active	X server	DBUS	Maliit	GStreamer	glibc
Maemo 5	X server	DBUS	Hildon	GStreamer	glibc
MeeGo 1.2 Harmattan	X server	DBUS	Maliit	GStreamer	glibc
Android	SurfaceFlinger	binder	ano	ano	Bionic
Tizen	X server	?	?	?	glibc
Ubuntu Touch	SurfaceFlinger	?	Maliit	?	libhybris
BlackBerry 10	?	QNX	?	?	
WebOS	DirectFB	Luna	Enyo	GStreamer	glibc
SHR	X server	DBUS	EFL	ne	glibc
QtMoko	framebuffer	?	?	ne	glibc
Cordia	X server	DBUS	Hildon	GStreamer	glibc

Tabulka 2.2: Tabulka popisující typ hlavních systémových komponent jednotlivých mobilních platforem

2.15.2 Otevřenost

platforma	otevřený kód	otevřený vývoj
MeeGo	ano	částečně
Mer	ano	ano
Nemo Mobile	ano	ano
Sailfish	částečně	částečně
Plasma Active	ano	ano
Maemo 5	částečně	ne
MeeGo 1.2 Harmattan	částečně	ne
Android	ano	ne
Tizen	ano	ne
Ubuntu Touch	ano	částečně
BlackBerry 10	ne	ne
WebOS	ano	ano
SHR	ano	ano
QtMoko	ano	ano
Cordia	ano	ano

Tabulka 2.3: Tabulka otevřenosti jednotlivých mobilních platforem

2.15.3 Balíčkování

platforma	formát	podpora závislostí	tvorba balíků
MeeGo	RPM	-	OBS
Mer	RPM	-	OBS
Nemo Mobile	RPM	ano	OBS
Sailfish	RPM	?	OBS
Plasma Active	RPM	?	OBS
Maemo 5	deb	ano	autobuilder
MeeGo 1.2 Harmattan	deb	částečně	OBS
Android	APK	ne	-
Tizen	RPM	?	OBS
Ubuntu Touch	„click“	ne	-
BlackBerry 10	bar	ne	-
WebOS	ipkg	ne/Preware ano	Openembedded
SHR	ipkg	ano	Openembedded
QtMoko	deb	ano	-
Cordia	RPM	ano	OBS

Tabulka 2.4: Tabulka balíčkovacích systémů jednotlivých platforem

2.15.4 Distribuce softwaru

platforma	hlavní repositář	kontrola kvality	GPL	přímá instalace
Mer	Mer OBS	komunitní	ano	ano
Nemo Mobile	Nemo OBS	komunitní	ano	ano
Plasma Active	Plasma OBS	komunitní	ano	?
Maemo 5	Extras	komunitní	ano	ano
MeeGo 1.2 Harmattan	Ovi Store	ano	ano	ano
Android	Google Play	automatická	ano	ano
BlackBerry 10	World	ano	ano	ano
WebOS	Preware	komunitní	ano	ano
SHR	ano	komunitní	ano	ano
QtMoko	ano	komunitní	ano	ano
iOS	App Store	ano	ne	ne
Windows Phone	Marketplace	ano	ano	ne

Tabulka 2.5: Tabulka popisující způsob distribuce aplikací třetích stran ke koncovým uživatelům. Sloupeček *GPL* značí, jestli hlavní repositář dané platformy přijímá aplikace, jejichž zdrojový kód je licencován pod GPL. *Přímá instalace* vyjadřuje, jestli je na dané platformě možné instalovat aplikace i mimo hlavní repositář.

2.15.5 Vývojové prostředí

platforma	Qt	GTK	SDL	HTML5	Python
MeeGo	ano	ne	ano	ano	ano
Mer	ano	ne	ano	ano	ano
Nemo Mobile	ano	komunitní	ano	ano	ano
Sailfish	ano	ne	?	ano	ano
Plasma Active	ano	ne	?	?	?
Maemo 5	ano	ano	ano	ano	ano
MeeGo 1.2 Harmattan	ano	komunitní	ano	ano	ano
Android	ano	ne	ano	ano	ano
Tizen	ne	ne	ne	ano	ne
Ubuntu Touch	ano	ne	ne	ne	ne
BlackBerry 10	ano	ne	ano	ano	komunitní
WebOS	ne	ne	ano	ano	komunitní
SHR	ne	ano	ne	ne	ano
QtMoko	ano	ano	ne	ne	ano
Cordia	?	ano	?	?	ano

Tabulka 2.6: Tabulka dostupnosti hlavních knihoven a programovacích jazyků na jednotlivých mobilních platformách

2.15.6 Hardware

platforma	telefony	tablety	další zařízení
MeeGo	-	WeTab	netbooky
Mer	-	-	-
Nemo Mobile	N900, N950, N9	Nexus 7, WeTab	VirtualBox
Sailfish	zatím ne	-	simulátor na PC
Plasma Active	-	Tablet PC + Archos	netbooky
Maemo 5	N900	-	simulátor
MeeGo 1.2 Harmattan	N9, N950	-	simulátor na PC
Android	mnoho	mnoho	různé
Tizen	prototyp	-	simulátor na PC
Ubuntu Touch	řada Nexus	řada Nexus	-
BlackBerry 10	Z10, Q10	PlayBook	simulátor na PC
WebOS	Pre & Veer	TouchPad	simulátor na PC
SHR	GTA02 a jiné	-	-
QtMoko	GTA02, GTA04	-	-
Cordia	-	prototyp	-

Tabulka 2.7: Tabulka dostupnosti hardwaru pro jednotlivé mobilní platformy

3 Specifika mobilních linuxových platforem

Mobilní linuxové platformy se od klasických linuxových distribucí, používaných na osobních počítačích a serverech, v mnohém liší. Aby byl vývoj multiplatformní aplikace efektivní, musí vývojář tyto odlišnosti znát a během vývoje je zohledňovat.

3.1 Grafické rozhraní

3.1.1 Vysoké rozlišení

Moderní mobilní zařízení dosahují rozlišení *HD-ready* či dokonce *full-HD*, čímž dohánějí, případně překonávají, klasické stolní počítače. Přesto je fyzická velikost obrazovky mobilních zařízení ve srovnání s monitorem stolního počítače stále velmi omezená. Obrazovky chytrých telefonů mají většinou úhlopříčku od 3 do 4 palců, u tabletů je to 7 až 10 palců. To je mnohem méně, než je velikost obrazovky u většiny Notebooků a je to pouze zlomek velikosti běžně používaných stolních monitorů.

Vysoké rozlišení obrazovky v kombinaci s její malou úhlopříčkou má za následek velké rozlišení na jednotku plochy, běžně udávané pomocí jednotky DPI[1][2]. Takto vysoké a navíc proměnlivé DPI (takto vysoké DPI nemají všechna mobilní zařízení) při návrhu grafického rozhraní prakticky znemožňuje použití absolutních hodnot v pixelech. Velikost prvků grafického rozhraní pro dosažení konzistentního výsledku na různých zařízeních musí proto být přímo či nepřímo závislá na rozlišení obrazovky a DPI.

Přímou závislost lze implementovat tak, že velikostí prvků bude v poměrné závislosti na aktuálním rozlišení, s přihlédnutím k DPI fyzické obrazovky.

Nepřímá závislost naopak využívá několika rozvržení s absolutními hodnotami v pixelech, která byla vytvořena pro konkrétní kombinace rozlišení a DPI. Z těchto kombinací je vždy vybrána ta nejvhodnější pro konkrétní zařízení.

První metoda je flexibilní v tom, že by měla fungovat na zařízeních s libovolnou kombinací rozlišení a DPI. Druhá metoda zase umožňuje úpravu velikosti všech prvků grafického rozhraní přímo na míru konkrétním zařízením. Nejlepších výsledků lze dosáhnout kombinací těchto dvou metod, tedy návrhem rozvržení na míru pro nejpoužívanější cílová zařízení. Pro

1. Dots Per Inch
2. např. obrazovka zařízení BlackBerry Z10 dosahuje 356 DPI (1280 x 768 na 4.2’)

ostatní zařízení je pak použito flexibilní rozvržení s poměrnými hodnotami.

3.1.2 Proměnlivý poměr stran

Dalším aspektem, který je třeba vzít v úvahu, je proměnlivý poměr stran obrazovky. Mobilní zařízení umožňuje na rozdíl od notebooků a většiny stolních monitorů snadné otáčení obrazovky na výšku a na šířku. Na tuto skutečnost je třeba dbát při návrhu grafického rozhraní aplikace.

Implementace této funkce se také na různých zařízeních a operačních systémech může vzájemně lišit. V některých případech zastřešuje tuto činnost operační systém, jinde je rotace obrazovky plně v režii aplikace, anebo platí kombinace předchozího. Proměnlivý poměr stran lze jednoduše vyřešit uzamčením aplikace pouze do jediné orientace. Toho využívá mnoho aplikací, zejména her. Je-li to však možné, měla by aplikace podporovat variabilní poměr stran, zejména pokud je tato funkce široce podporována ostatními aplikacemi na cílové platformě.

3.2 Uživatelské rozhraní

Při tvorbě uživatelského rozhraní určeného pro mobilní zařízení je důležité vzít v úvahu specifika interakce uživatele s mobilním zařízením. Při tvorbě aplikací pro stolní počítače lze předpokládat určitou poměrně jednotnou základní výbavu. Uživatel mívá ve většině případů k dispozici monitor větší než 15 palců a s rozlišením alespoň 1024x768. Aplikaci ovládá pomocí klávesnice s přibližně 100 tlačítky a pomocí přesného polohování prostřednictvím počítačové myši nebo ekvivalentního polohovacího zařízení[3].

Společným činitelem většiny současných mobilních zařízení bude naopak nepřesná kapacitní dotyková obrazovka s poměrně nízkým rozlišením a malými fyzickými rozměry. Většina mobilních zařízení nemá klávesnici a v některých případech u nich dokonce zcela chybí jakákoli hardwarová tlačítka, kterých by aplikace mohla využít.

Specifika mobilních zařízení znamenají pro vývoj mobilní aplikace především to, že ovládací prvky musí být větší, aby se na ně uživatel pohodlně trefil prstem. Vhodné je také použití většího písma, aby byly nápisy na malé obrazovce dobře čitelné. Většinou také na mobilních zařízeních nelze použít: klávesové zkratky, akce spouštěné pravým tlačítkem myši[4] nebo nápo-

3. trackball, touchpad, trackpoint, atd.
4. tyto akce je možné spouštět pomocí tap-and-hold

vědu aktivovanou, když je kurzor delší dobu na jednom místě[5].

Na rozdíl od stolních počítačů však vývojář může využít exkluzivních funkcí, poskytovaných mobilními zařízeními jako jsou multitouch gesta, kamery, senzory zrychlení, gravitace, úrovně světla, atd.

3.3 Knihovny a utility

Na mobilních zařízeních se vyskytuje mnoho knihoven, běžně známých ze stolních linuxových distribucí. Pokud knihovna potřebná pro vývoj aplikace k dispozici není a pokud platforma podporuje vývoj nativních aplikací, bývá zpravidla možné danou knihovnu s větším či menším úsilím zkompilovat a použít.

Složitější situace je na poli grafických knihoven; zatímco u stolních distribucí bývá téměř všude použit X server, používá mnoho mobilních platforem svůj vlastní nekompatibilní systém pro správu grafiky. Důsledkem je tak paleta grafických knihoven, omezená pouze na takové knihovny, které dokážou pracovat bez X serveru a jsou současně kompatibilní s grafickým systémem dané platformy. Příkladem takových grafických knihoven jsou např. Qt a SDL.

3.4 Balíčkování

Balíčkovací formáty používané na mobilních platformách se velmi podobají těm, které používají stolními linuxové distribuce. Odlišují se hlavně tím, že většina mobilních platforem vůbec nepodporuje závislosti mezi balíčky, a každá aplikace si tak sama musí zařídit dostupnost všech dependencí.

Nejobvyklejším řešením tohoto problému je „přibalení“ všech dependencí do instalačního balíčku. „Přibalování“ však nejen zbytečně zvyšuje nároky na úložný prostor díky tomu, že aplikace používající stejnou knihovnu ji nemohou sdílet, ale zastaralé knihovny „přibalené“ k aplikaci mohou rovněž znamenat bezpečnostní riziko.

Další možností jak chybějící podporu závislostí obejít, je nainstalování speciální aplikace, která supluje činnost správce balíků. Příkladem této aplikace je Ministro pro Android, vyvinutá projektem Necessitas.

5. tzv. hover

3.5 Multitasking

Multitaskingu na mobilních platformách omezuje především chybějící podpora pro současné zobrazení více aplikačních oken. Většina platforem podporuje pouze zobrazení jediného aplikačního okna v režimu plné obrazovky.

Některé platformy manipulují s během aplikací, jejichž okno není v daném okamžiku viditelné: systém například pozastaví jejich běh nebo aplikaci automaticky ukončí za účelem uvolnění systémových prostředků.

Systém chovající se tímto způsobem, je pak v zásadním rozporu se situací na linuxovém desktopu, kde systém do fungování aplikačních procesů v zásadě vůbec nezasahuje. Výjimku tvoří uživatelem vynucené ukončení procesu a mezní situace jako je aktivace omkiller mechanismu v linuxovém jádře.

Automatické ukončování aplikací částečně ospravedlňuje skutečnost, že většina mobilních zařízení nemá swapovací oddíl, na který stolní počítače běžně „odklízejí“ po delší dobu neaktivní aplikace. Přesto se však některé mobilní platformy bez automatického ukončování aplikací dovedou obejít[6]

Možnost automatického ukončení aplikace vyžaduje od vývojáře dobré zvládnutí práce se stavem aplikace. Aplikace musí být schopná kdykoli uložit svůj stav a při startu jej co nejrychleji obnovit. Pokud práce s aplikačním stavem není dobře zvládnutá, může dojít jak ke ztrátě dat, tak k nekonzistentnímu chování aplikace, která se z uživatelského pohledu po přepnutí na pozadí a zpět chová nekonzistentně.

3.6 Hardware

Kromě již výše zmíněných hardwarových omezení, co se týče velikosti obrazovky a omezených vstupních metod, je důležité si při vývoji aplikace uvědomit, že mobilní zařízení jsou většinu času napájena z baterie.

Na mobilním zařízení spotřebuje nejvíc energie obrazovka. Na zařízeních s LCD displayem nemůže aplikace energii spotřebovávanou displayem příliš ovlivnit, naopak na zařízeních s OLED displayem, kde je spotřeba závislá na počtu jasných pixelů, ovlivňuje vzhled aplikace do určité míry spotřebu. Čím méně světlých ploch v aplikaci bude, tím menší bude spotřeba.

Velké množství energie také spotřebovávají bezdrátové datové přenosy – ať už se jedná o přenos dat přes WiFi či mobilní síť. Aplikace by tedy měla

6. Maemo 5 (má swap) & Harmattan (1GB RAM)

přenášet data jen, když je to nezbytně nutné. Pokud aplikace potřebuje přes síť periodicky kontrolovat stav určitého on-line zdroje (emailová schránka, RSS, nové IM zprávy), neměla by být tato perioda příliš krátká. Pokud operační systém podporuje sdružování periodických požadavků, tzv. systém PUSH, měla by aplikace být schopná využít jej pro úsporu energie. Jednotlivé technologie bezdrátového přenosů dat mají různou energetickou spotřebu závislou zpravidla na vzdálenosti, kterou musí data překonat. Tuto skutečnost je možné při návrhu aplikace vzít v potaz a odložit například velké datové přenosy prostřednictvím mobilní síť, dokud se zařízení nepřipojí na WiFi.

Velkou roli hraje při spotřebě energie rovněž využití CPU. Aplikace by se měla primárně vyvarovat tzv. aktivního čekání[7], tzn. periodického provádění výpočtu ve smyčce s krátkou periodou. Na stolních počítačích, připojených na externí zdroj napájení, nepředstavuje aktivní čekání problém. Naopak na mobilních zařízeních vyvolává zbytečné probouzení CPU z úsporného režimu, což vede k razantnímu zkrácení výdrže zařízení při provozu z baterie.

Aplikace by tedy měly reagovat pouze na události, jako jsou vstupy od uživatele nebo změna stavu zařízení (např. změna aktuálních souřadnic, příchod SMS, odemknutí obrazovky, atd.). Pokud okno aplikace není vidět, neměla by aplikace provádět jeho překreslování. Veškeré náročnější výpočty na pozadí by měly probíhat pouze na pokyn uživatele.

Nezanedbatelnou spotřebu energie má rovněž zjišťování polohy s využitím systému GPS. Výdrž většiny mobilních zařízení se po aktivací GPS modulu zkracuje na pouhých několik hodin. Pokud tedy aplikace nevyžaduje GPS po celou dobu svého běhu (jako je tomu v např. v případě navigačního systému nebo GPS loggeru), měla by aplikace vždy zapnout GPS pouze na nezbytně dlouhou dobu (např. pro označení fotografie geografickými souřadnicemi). Moderní mobilní zařízení zpravidla používají technologii AGPS, která umožňuje zjištění aktuálních souřadnic v řádu sekund od aktivace GPS přijímače, takže zapnutí/vypnutí GPS příliš nezdržuje.

7. anglicky busy waiting

4 Tvorba multiplatformní aplikace

Kapitola se zabývá plánováním vývoje multiplatformní aplikace. Nejprve je popsán výběr vhodných nástrojů a zvolena vhodná struktura projektu. Další kapitola pak popisuje konkrétní aplikaci, která vznikla na základě těchto poznatků.

4.1 Výběr programovacího jazyka

Aby aplikace fungovala na mnoha různých platformách, je třeba vybrat vhodný programovací jazyk. Pokud se jedná o jazyk kompilovaný, je nutné, aby kompilátory tohoto jazyka byly dostupné pro co největší počet platforem. Pokud půjde o jazyk interpretovaný, je zásadní zajistit přenositelnost interpretru. Hlavní výhodou interpretovaných jazyků je, že program není nutné pro každou platformu zvlášť kompilovat. Postačí, když pro každou platformu bude k dispozici intrepretr.

Pokud intrepretr ve výchozím stavu k dispozici není, je nutné zkompilovat jej pro danou platformu. Kompilace intrepretru může sice na první pohled působit jako krok zpět, ale je třeba si uvědomit, že stačí pro danou platformu interpretr zprovoznit pouze jednou. Jakmile se jej podaří zkompilovat a zprovoznit, mohou jej využívat veškeré programy, které používající daný intepretovaný jazyk. Široce používané interpretry[1] bývají poměrně složité, ale také poměrně robustní, a to díky široké uživatelské komunitě pokrývající mnoho různých platforem.

Pro implementaci multiplatformní aplikace jsem vybral intepretovaný jazyk Python.

4.2 Výběr grafické knihovny

Tvorba aplikací, které nevyžadují uživatelské rozhraní, se v případě mobilních platforem příliš neliší od tvorby podobných aplikací pro PC. Mnoho klasických utilit příkazové řádky lze na mobilních zařízeních používat bez modifikace zdrojového kódu, stačí je pouze zkompilovat pro architekturu daného zařízení.

Tvorba multiplatformních mobilních aplikací s grafickým rozhraním je výrazně složitější. Zatímco linuxové distribuce na PC používají téměř univerzálně pro práci s grafikou dlouhodobě prověřený X server, u mobil-

1. jako je např. interpretr jazyka Python

ních platforem tento jednotící prvek chybí. X server je sice použit na několika mobilních linuxových platformách, většina mobilních OS však používá různé, vzájemně nekompatibilní metody práce s grafikou.

Většina existujících grafických knihoven podporuje X server, ale pouze málokterá podporuje grafické systémy používané mobilními platformami. Jediné dvě grafické knihovny, které jsou široce dostupné na většině mobilních platforem, jsou vlastně Qt a SDL. Další multiplatformní alternativou ke klasickým grafickým knihovnám může být webové rozhraní.

4.2.1 Qt

Qt[120][121] je grafická knihovna napsaná v programovacím jazyce C++ a dostupná pod LGPL[122]. Byla původně vytvořena pro použití na PC norskou firmou Trolltech. V roce 2008 koupila Trolltech firma Nokia[123] a od té doby začala být knihovna Qt více směřována pro použití na mobilních zařízeních. V létě roku 2012 koupila Qt od Nokie společnost Digia[124]. Díky tomu, že Digia na rozdíl od Nokie nevyrábí vlastní mobilní zařízení, byl odstraněn střet zájmů, který bránil oficiální podpoře Qt pro mobilní platformy nespadající pod Nokii.

Qt podporuje četné platformy jako např.:

- Linux (PC)
- Windows
- Mac OS
- Maemo & MeeGo 1.2 Harmattan
- Symbian[125]
- BlackBerry 10
- Android – oficiální pokračování komunitního portu firmou Digia
- iOS – oficiální port firmou Digia
- Windows RT – oficiální port firmou Digia
- Tizen – komunitní portu firmou Digia[126]

Z hlediska využití Qt pro tvorbu aplikací určených na mobilní zařízení je zajímavá technologie Qt Quick, která díky svým vlastnostem umožňuje snadnou a rychlou tvorbu mobilních aplikací. Jedná se především o tyto vlastnosti:

- deklarativní paradigma
- podpora automatického provázání proměnných (tzv. property binding)
- základní grafické stavební prvky pro tvorbu dotykového rozhraní
- možnost využít Javascriptu
- podpora rozšíření napsaných v jiném programovacím jazyce (např.

v C++ či Pythonu)
- podpora grafické akcelerace s možnosti přepnutí na softwarové vykreslování

Pokud jde o mou osobní zkušenost, jedná se o dosud nejlepší technologii pro tvorbu grafických aplikací určených pro mobilní zařízení.

Co se týče podpory programovacích jazyků grafickou knihovnu Qt, primárně je podporován programovací jazyk C++, je však možné použít i další programovací jazyky. Python je dokonce podporován dvěma projekty, které se v zásadě liší pouze svou licencí, pod kterou jsou distribuovány: PyQt pod GPL a PySide pod LGPL. Žádný z nich však zatím plně nepodporuje Qt 5.

Je nutné si také uvědomit, že Qt není pouhou grafickou knihovnou, poskytuje totiž vývojářům mnoho praktických modulů pro zpracování dat, paralelní programování, práci se systémovou sběrnicí, získávání dat ze senzorů, lokalizaci, zobrazování webových stránek, přehrávání videa či práci se zvukem.

Pro implementaci grafického rozhraní multiplatformní aplikace jsem na základě těchto výhodných vlastností zvolil grafickou knihovnu Qt.

4.2.2 SDL

SDL[127] je multiplatformní multimediální knihovna, která zpřístupňuje grafiku, audio, vstupní zařízení a 3D akceleraci. Je napsaná v programovacím jazyce C a dostupná pod licencí LGPL. Ve srovnání s Qt operuje SDL na mnohem nižší úrovni abstrakce, která však zato umožňuje plnou kontrolu nad všemi detaily fungování aplikace. Nízkoúrovňovost SDL znamená, že neobsahuje žádné složitější stavební bloky[2] pro tvorbu aplikací. Z tohoto důvodu bývá knihovna SDL používána zejména pro tvorbu her.

S ohledem na nízkou úroveň abstrakce a chybějící základní stavební prvky jsem se rozhodl knihovnu SDL nepoužít k implementaci aplikace. To však neznamená, že by SDL použít nešlo, jen by bylo potřeba znovu implementovat řadu prvků, které jsou v Qt k dispozici již ve výchozím stavu. Použitelnost SDL pro vývoj navigačního systému demonstruje například projekt Cloud GPS[128].

4.2.3 Webové rozhraní

Přestože se nejedná o grafickou knihovnu v klasickém slova smyslu, může rozhraní aplikace napsané v HTML představovat zajímavou alternativu k rozhraní vytvořenému s pomocí nativní grafické knihovny. Je to také

2. tlačítka, vstupní pole, seznamy, layout, webview, atd.

o zajímavý doplňkový způsob prezentace některých informací z aplikace. Webové rozhraní aplikace funguje tak, že aplikace provozuje webový server na lokálním rozhraní, který nabízí webové stránky, tvořící grafické rozhraní aplikace. Uživatel pouze otevře libovolný webový prohlížeč, nasměruje jej na lokální rozhraní a port webového serveru aplikace. Poté uživatel s rozhraním interaguje jako s každou jinou webovou stránkou. Mnoho mobilních zařízení podporuje vytváření ikon odkazujících na webové stránky, takže lze aplikace s webovým rozhraním poměrně dobře do systému začlenit.

Síťová transparentnost je další zajímavou vlastností konceptu webového rozhraní. Při správném nastavení sítě lze na webové rozhraní přistupovat z jiných zařízení, nacházejících se na lokální sítí; pokud má zařízení veřejnou IP adresu, je možné na webové zařízení přistupovat i přes Internet.

Hlavní nevýhodou webového rozhraní však je to, že se vývojář při jeho tvorbě musí podřídit limitům webových technologií. Vytvořit jednoduchou textovou stránku s několika tlačítky by nemělo být složité. Tvorba graficky bohatého interaktivního rozhraní, které by bylo ekvivalentní nativnímu rozhraní, bude s použitím webových technologií na mobilním zařízení většinou komplikovaná či dokonce nerealizovatelná.

4.3 Struktura aplikace

I když je vybrán vhodný programovací jazyk a grafická knihovna, které dokáží společně eliminovat většinu platformně závislého kódu, budou jednotlivé platformy stále obsahovat odlišnosti, na které musí aplikace zvládnout.

Je také možné, že na některých platformách vybraná grafická knihovna nebude fungovat. Vývojář také musí vzít v úvahu alternativní přístupy k tvorbě grafického rozhraní, jako např. textové či webové rozhraní.

Aplikaci je proto vhodné pojmout jako modulární celek, se kterým lze pracovat jako s kombinací tří částí, propojených jasně definovaným rozhraním:

- **jádro** – obsahuje většinu kódu a musí být plně nezávislé jak na grafických knihovnách, tak na platformně závislých funkcích
- **platformní modul** – obsahuje všechna nastavení a kód potřebný pro fungování aplikace na dané platformě
- **grafický modul** – na základě datových modelů *jádra* vytváří uživatelské rozhraní s pomocí konkrétní grafické knihovny

Takto navržená aplikace bude obsahovat jedno jádro, alespoň jeden grafický modul a jeden platformní modul pro každou podporovanou platformu.

Při startu aplikace dojde nejdříve ke spuštění jádra, které na základě autodetekce či spouštěcích parametrů vybere a načte platformní a grafický modul. Platformní modul je načten jako první, protože většinou poskytuje důležitá platformně specifická nastavení[3], bez kterých nelze grafický modul korektně spustit.

4.3.1 Výběr modulů pro danou platformu

Nejjednodušším řešením je přímé určení, které platformní a grafické moduly mají být použity. Toho lze dosáhnout např. nastavením parametrů příkazové řádky ze spouštěcího skriptu nebo vložením konfiguračního souboru, který určuje vhodné moduly.

Další možností je automatická detekce platformy a načtení odpovídajících modulů podle zjištěných parametrů. Automatická detekce je vhodná zejména během testování, kdy stačí aplikaci jednoduše stáhnout a spustit bez nutnosti provádět všechny kroky balíčkovacího procesu.

Přímý stejně jako automatický výběr modulů lze zkombinovat tak, že když aplikace při startu neobdrží informace o tom, které moduly má načíst, tak provede automatickou detekci platformy.

Další výhodou modulární aplikace je možnost distribuce pouze těch modulů, které jsou pro danou platformu k dispozici. Tato výhodná vlastnost může především v případě grafických modulů přinést výrazné úspory co do velikosti instalačních balíčků.

4.3.2 Platformní modul

Jedná se o samostatný modul, načítaný jádrem aplikace při startu, který obsahuje všechna nastavení a kód nutný pro fungování aplikace na dané platformě. Platformní modul nesmí obsahovat závislosti na jiných platformních modulech a musí být schopen fungovat i bez načteného grafického modulu, který je načten až jako další v pořadí.

Jedním z úkolů platformního modulu je nastavení vhodných cest v systému souborů dané platformy. Může se jednat například o:

- cestu k profilovému adresáři
- cestu pro ukládání dočasných dat
- cestu pro ukládání větších objemů dat

3. rozlišení obrazovky, důležité cesty, dostupné senzory, atd.

- cesty pro ukládání specifických druhů dat
- výchozí cestu dialogu pro práci se soubory

Cestu pro profil a dočasná data jde naštěstí většinou získat z proměnných prostředí zvaných `$HOME` a `$TMP`. Na těch platformách, kde tyto proměnné definované nejsou, či jsou špatně nastaveny, je nutné tyto cesty nastavit v platformním modulu.

Správné nastavení cesty pro ukládání velkého objemu dat je velmi důležité zejména na mobilních platformách. Zde totiž ve většině případů[4] bývá úložný prostor rozdělen na oddíl pro aplikace (kam bývá uložen rovněž profil) a oddíl pro skladování velkých objemů (zejména multimediálních) dat. Úložiště pro velké objemy dat v některých případech podléhá zvláštním požadavkům, protože se může nacházet např. na paměťové kartě či není dostupné v případě, že je zařízení připojeno přes USB v režimu *mass storage*. Aplikace by měla na tento zvláštní režim umět reagovat.

Neméně důležitý je dále výběr správné výchozí cesty pro dialog práce se soubory. Zde je třeba se vyhnout často se vyskytující chybě, kdy aplikace při prvním spuštění dialogu práce se soubory otevře aktuální adresář. Vývojář by měl naopak zvolit vhodnou cestu podle toho, s jakými soubory bude uživatel aplikace s největší pravděpodobností pracovat. Tato cesta se mezi jednotlivými platformami velmi liší [5] a je tedy v kompetenci platformního modulu.

4. Maemo, Android, BB10
5. například `/sdcard` na Androidu, `/home/user/MyDocs` na Maemo či `share/downloads` v BB10

5 modRana – příklad multiplatformní aplikace

ModRana[129] je flexibilní GPS navigační systém pro mobilní zařízení, postavený na principech tvorby multiplatformní aplikace, popsaných v předchozí kapitole.

5.1 Implementace jádra

Jádro modRany bylo navrženo tak, aby obsahovalo všechen zdrojový kód, který není závislý na použité grafické knihovně a konkrétní platformě. Jedinou závislostí jádra je Python a jeho standardní knihovna. Velkou výhodou tohoto přístupu je, že veškeré funkce jádra jsou dostupné na všech platformách a všem uživatelským rozhraním. Když jsem přidával nové funkce, snažil jsem se, aby co největší část implementačního kódu byla v jádře (získávání a zpracování dat, datový model) a grafický modul obsahoval pouze kód nutný ke správné prezentaci dat.

Vzhledem k tomu, že modRana dokáže fungovat i v režimu „utilita“, ve kterém vrací data na základě argumentů z příkazové řádky, musí jádro modRany fungovat i na takovém operačním systému, který nemá žádnou z podporovaných grafických knihoven k dispozici.

Dalším aspektem jádra je podpora široké škály verzí programovacího jazyka Python. Ne všechny platformy, které modRana podporuje, mají stejnou verzi Pythonu. Viz následující přehled:

- Python 2.5 – Maemo 5 (nejstarší podporovaná verze Pythonu)
- Python 2.6 – MeeGo 1.2 Harmattan, SHR
- Python 2.7 – desktopové distribuce, Android[1]
- Python 3.2 – BlackBerry 10

ModRana dokáže fungovat na všech těchto verzích Pythonu bez nutnosti provádět modifikace zdrojového kódu.

5.1.1 Struktura jádra

Jádro modRany tvoří tří části:

- hlavní skript *modrana.py*
- jednoduché moduly v adresáři *core*
- „funkční“ moduly v adresáři *modules*

1. pro přenos implementace na Android byl využit projekt pydroid

Při startu modRany dojde ke spuštění hlavního skriptu *modrana.py*, ten zvolí[2] a načte platformní a poté grafický modul. Následně dojde k postupnému načtení všech „funkčních“ modulů.

Jednoduché moduly obsahují základní funkcionalitu potřebnou pro běh modRany a složitějších modulů. Jedná se např. o modul pro načítání konfiguračních souborů, pro práci s vlákny, pro geografické výpočty nebo pro zpracování argumentů příkazové řádky.

Funkční moduly jsou složitější a nejsou pro běh modRany nezbytné. Pokud se některý funkční modul nepodaří načíst, modRana by přesto měla dál fungovat, a to samozřejmě bez funkcionality poskytované daným modulem. Každý funkční modul je zaměřený na jednu konkrétní činnost, jsou to např. routování, správa mapových dílců, práce s body zájmu nebo správa záznamů trasy. Moduly mezi sebou mohou komunikovat bud'to přímým přístupem a voláním metod, posíláním zpráv nebo s pomocí signálů.

Velká modularita umožňuje modRaně fungovat jak v režimu aplikace s grafickým rozhraním, tak v režimu utility, která vyřizující dotazy na základě argumentů příkazové řádky. V režimu utility načte modRana pouze ty moduly, které jsou pro vyřízení konkrétního dotazu nezbytně nutné. Zodpovězení dotazu díky tomu probíhá velmi rychle.

5.2 Implementace platformního modulu

Platformní modul je v modRaně z historických důvodů označován označován jako *device module*, všechny moduly mají jednotné rozhraní dané rodičovskou třídou, `DeviceModule`. Moduly pro jednotlivé platformy pak toto rozhraní implementují – ale pouze ty metody, kde se chování platformy liší od výchozího chování specifikovaného třídou `DeviceModule`.

Platformní moduly mají na starost např. přístup k GPS a k ovládání obrazovky, nastavují také důležité cesty a poskytují modRaně informace o vlastnostech dané platformy[3]. Platformní modul je modRanou načten jako první funkční modul.

5.3 Implementace grafického modulu

Grafický modul je více či méně tenkou prezentační vrstvou nad datovými modely v jádře modRany. Kromě prezentace informací má grafický modul

2. automaticky nebo podle argumentů příkazové řádky
3. informace o tom, zda má být aplikace spuštěna v režimu plné obrazovky, zda je podporována minimalizace aplikačního okna, zda jsou k dipozici hardwarová tlačítka, atd.

také za úkol korektní inicializaci grafické knihovny. Jedná se také o jediný modul, který může importovat grafickou funkcionalitu. Dostupnost grafických knihoven se mezi platformami liší, což však díky načtení grafického modulu až po detekci platformy není problém. Pokud by však spoléhaly i další moduly na dostupnost jedné z grafických knihoven, mohlo jejich načtení selhat.

Podobně jako u platformního modulu vycházejí jednotlivé grafické moduly z rozhraní rodičovské třídy, která pro grafické moduly nese název `GUIModule`.

Grafický modul je modRanou načten jako druhý modul v pořadí, tedy hned po platformním modulu. Poté následuje postupné načtení všech ostatních funkčních modulů.

ModRana obsahuje dva moduly, z nich jeden využívá grafickou knihovnu GTK a druhý používá kombinaci Qt/QML.

5.4 Podporované platformy

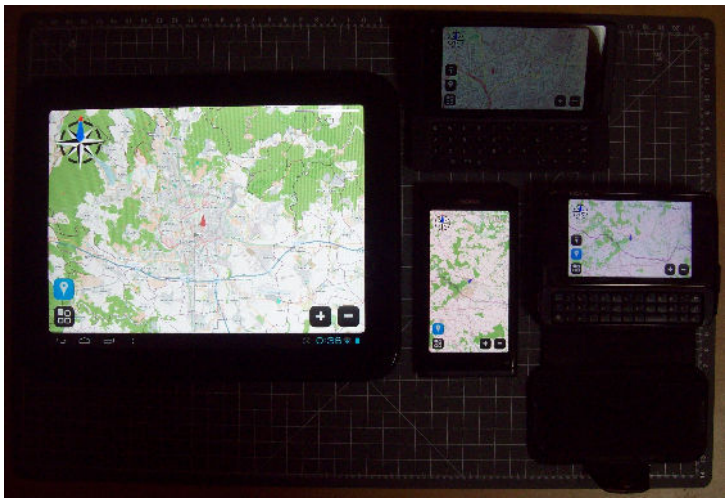
ModRana podporuje mnoho různých platforem.

platforma	rozhraní
Maemo 5	GTK & QML rozhraní
MeeGo 1.2 Harmattan	QML rozhraní
stolní počítače	GTK & QML rozhraní
Android	QML rozhraní
BlackBerry 10	QML rozhraní
Nemo Mobile	QML rozhraní
SHR na OpenMoko Neo FreeRunner	GTK rozhraní
OpenPandora	GTK rozhraní

Tabulka 5.1: Tabulka platforem podporovaných modRanou

Všechny tyto platformy podporuje modRana jednotnou sadou zdrojových kódů, což přináší tu výhodu, že jakákoli nová funkce je na všech platformách ihned k dispozici. Tvorba instalačních balíků pro jednotlivé platformy je do značné míry zautomatizována. Není tedy problém vydat každou novou verzi modRany současně pro všechny podporované operační systémy a zařízení.

ModRana tak názorně demonstruje, jak je možné s minimem prostředků vyvíjet multiplatformní aplikace, které podporují většinu současných nejpopulárnějších mobilních platforem.



Obrázek 5.1: modRana běžící na chytrých telefonech Nokia N900, N950, N9 a tabletu HP TouchPad

5.5 Popis přenosu implementace na novou platformu

Tato sekce přibližuje, jak probíhal přenos implementace modRany na několik nových mobilních platforem.

Tento postup je založený na jednoduchém principu: nejprve je nutné seznámit se s cílovou platformou a zjistit, které komponenty nezbytně nutné pro fungování modRany jsou k dispozici. Komponenty, které k dispozici nejsou, je poté třeba na danou platformu přenést. Přenos komponent nemusí být jednoduchý, ale stačí jek provést pro každou kombinaci komponenty a platformy pouze jednou. Komponentu pak na dané platformě mohou využít i další projekty.

Jakmile jsou všechny závislosti modRany pro danou platformu k dispozici, je nezbytné ošetření případných nekompatibilit modRany s danou platformou. V zásadě se jedná o napsání nového platformního modulu. Pokud je potřeba provést změny ve zdrojovém kódu, který se nachází se mimo platformní modul, je důležité zajistit, aby změny nezpůsobily regrese na jiných platformách.

Posledním krokem při portování modRany je vytvoření instalačního balíčku a jeho odeslání do aplikačních repositářů (pokud je daná platforma má). Je vhodné balíčkování co nejvíce zautomatizovat a zapojit do stávajících multiplatformních balíčkovacích skriptů.

Před popisem několik nedávných přenosů implementace či také tzv.

„portů“ modRany na nové mobilní platformy, je třeba upozornit, že pojmem „portovat“ je v tomto kontextu znamená úpravu zdrojového kódu modRany takovým způsobem, aby fungoval na nové platformě – a to při zachování funkčnosti na všech ostatních podporovaných platformách ! Nejedná se o jednorázově upravenou a samostatně udržovanou vývojovou větev. Stejný zdrojový kód modRany, který např. funguje na Androidu, funguje rovněž na BlackBerry 10 nebo na PC.

5.5.1 Android

Platformu Android byla pro port modRany vybrána především proto, že se v současnosti jedná o vůbec nejrozšířenější mobilní platformu. Android jinak ničím nevyniká a obsahuje řadu nedostatků. Přenos implementace modRany na Android některé z nich odhalil a bylo nutné je vyřešit nebo obejít.

Android ve výchozím stavu neobsahuje Python ani žádnou z grafických knihoven, které modRana momentálně podporuje(GTK a Qt). Existují však komunitní porty Pythonu[130][131] a také port grafické knihovny Qt[132]. Již v roce 2011 Thomas Perl experimentoval[133] s použitím Pythonu a Qt na Androidu prostřednictvím projektu Pyside, který umožňuje používání grafické knihovny Qt v Pythonu.

Musel jsme tedy „pouze“ sjednotit jednotlivé komponenty a využít je pro běh modRany. Největší problémy byly s kompilací PySide vůči moderní verzi Pythonu a Necessitas[4]. I ty se mi však podařilo překonat, částečně díky tomu, že jsem v té době paralelně pracoval na portu modRany na BlackBerry 10, kde bylo rovněž nutné PySide kompilovat.

Zbyla poslední překážka: vytvoření samostatného balíčku, který bude obsahovat modRanu, Python a všechny další závislosti. Zjistil jsem však, že tento problém se již v průběhu roku 2012 podařilo vyřešit projektu android-python27[130]. Řešení spočívá ve vytvoření APK balíku, který obsahuje dva zip archívy. Jeden archív obsahuje závislosti: v případě modRany je to Python, PySide a Qt Components, druhý archiv obsahuje aplikaci, což je v tomto případě modRana. Balík obsahuje upravený javový kód, běžně používaný pro inicializaci aplikačního okna pro aplikace využívající knihoven Qt na Androidu. Úprava spočívá v tom, že kód při prvním spuštění rozbalí dva dříve uvedené archívy do instalačního adresáře.

Výsledkem je jediný soubor typu APK, který po běžné instalaci na zařízení připraví vše tak, že modRanu lze spustit stejným způsobem jako běžné

4. Necessitas je název portu Qt pro Android

aplikace pro Android. Dokonce rychlost spuštění i běhu modRany na Androidu je v podstatě stejná jako u nativních aplikací pro Android.

ModRana je na Androidu zatím ve stádiu testování, v plánu je její začlenění do Google Play a dalších aplikačních repositářů pro Android.

5.5.2 MeeGo 1.2 Harmattan

Port na platformu MeeGo 1.2 Harmattan byl poměrně jednoduchý. Platforma se od operačního systémů Maemo, který modRana podporuje již velmi dlouhou dobou, příliš neliší. Přímo ve výchozích aplikačních repositářích byla k dispozici hlavní závislost modRany – Python. Dostupná byla také grafická knihovna Qt a její rozhraní pro Python zvané PySide.

Hlavním těžištěm tohoto přenosu implementace modRany tak bylo vylepšování QML rozhraní a přidání podpory pro novou metodu přístupu k GPS. Na Maemo 5 používá modRana *liblocation* a na ostatních platformách dosud pracovala s *gpsd*. Pro Harmattan jsem přidal podporu pro použití knihovny Qt Mobility.

Také jsem mírně upravil instalační balíček tak, aby obsahoval manifest pro bezpečnostní systém Aegis. Bez platného Aegis-manifestu by totiž modRana neměla přístup k GPS a také by nemohla řídit stmívání obrazovky.

Bylo také třeba, abych abych vyrobil novou, větší ikonu.

5.5.3 BlackBerry 10

Mobilní platforma BlackBerry 10 obsahuje poměrně mnoho prerekvizit pro běh modRany: přímo na zařízeních je k dispozici jak Python, tak grafická knihovna Qt. Nejnáročnější částí portace tak byla kompilace knihovny PySide a adaptace modRany pro Python ve verzi 3.2, ve které je Python na zařízeních s BlackBerry 10 k dispozici.

Třebaže již v minulosti proběhl přenos implementace PySide na BlackBerry PlayBook[134], nebylo portování PySide na operační systém BlackBerry 10 právě jednoduché. Během kompilace se objevilo mnoho esoterických chyb, vyskytly se také pády pokusných aplikací s nejasnými chybovými hláškami. Vytrval jsem však ve svém snažení a s vítanou pomocí z řad vývojářů projektu BlackBerry-Py[135] jsem celou věc dostal do použitelného stavu. Ve výsledku tak Pyside nyní funguje na BlackBerry 10, což otevřelo dveře této platformy nejen modRaně, ale i dalším aplikacím napsaným v Pythonu a používajícím grafickou knihovnu Qt.

Dalším netriviálním úkolem bylo přizpůsobení modRany tak, aby fun-

govala s Pythonem trojkové řady. ModRana pro BlackBerry 10 je současné době ve stádiu testování a plánuji její začlenění do hlavního repositáře platformy, známého pod názvem (BlackBerry)World.

5.6 Kompatibilita s Pythonem verze 2.5 až 3.2

Všechny dosud modRanou podporované platformy nabízely pouze Python dvojkové řady. Python 3 obsahuje řadu nekompatibilních změn a navíc celou snahu dále komplikuje potřeba zachovat podporu pro Python 2.5 na platformě Maemo 5. Celou proceduru se mi nakonec podařilo provést za jeden jediný den, rád bych však dodal, že jsem se celý den nevěnoval ničemu jinému. V pozdních večerních hodinách jsem však již měl hotovou verzi modRany, která funguje jak s Pythonem 2.5 na Maemo 5, tak s Pythonem 2.7 a 3.2. Přitom mi velmi pomohl kompatibilitní modul *six*[136] (2x3=6), který umožňuje velmi zjednodušit úpravy stávajících aplikací tak, aby fungovaly jak s Pythonem dvojkové, tak trojkové řady.

Přestože modRana používá Python 3 zatím pouze na platformě BlackBerry 10, podpora pro Python 3 se bude jistě v budoucnu hodit, protože je jen otázkou času, kdy na Python trojkové řady začne přecházet také většina dalších stolních a mobilních platforem.

O úpravách zdrojového kódu tak, aby byl kompatibilní jak s Pythonem 2.5, tak 3.2 i všemi mezilehlými verzemi, jsem rovněž napsal článek[137] na wiki projektu modRana. Věřím, že se tyto informace budou při nadcházející migraci na Python 3 mnoha projektům hodit.

6 Vyhodnocení a ověření výsledků diplomové práce

6.1 modRana

6.1.1 Uživatelská komunita

Nejvíc uživatelů má modRana zatím stále ještě na první podporované platformě, kterou je Maemo 5 Fremantle. GTK GUI je na Maemo 5 výchozím uživatelským rozhraním a podporuje v zásadě veškerou funkcionalitu jádra modRany. QML GUI je rovněž možné použít, po instalaci balíčku je k dispozici paralelně s GTK GUI. Hlavním kanálem pro komunikaci s uživateli modRany na Maemo 5 je diskuzní vlákno[47] na talk.maemo.org. Toto diskuzní vlákno mělo začátkem května 2013 přes **230 tisíc shlédnutí** a celkem **1371 příspěvků** na **138 stranách**.

Vzhledem k tomu, že podpora některé většinu dalších dalších platforem byla do modRany přidána relativně nedávno a vzhledem k tomu, že QML GUI zatím nepodporuje všechny funkce jádra, je uživatelská komunita na nově podporovaných platformách[1] zatím poměrně malá. Lze však předpokládat, že s postupným vylepšováním QML GUI a hlavně po zveřejnění modRany v centrálních aplikačních repositářích dojde k nárůstu počtu uživatelů modRany na těchto platformách.

6.1.2 Statistiky stažení

U otevřeného softwaru je odhad počtu uživatelů velmi složitý. Kdokoli si může stáhnou zdrojové kódy a dále je distribuovat. Určitý obraz o množství uživatelů lze získat z počtu stažení instalačních balíčků. Je však třeba si uvědomit, že zde přímá úměra – jedno stažení, jeden uživatel – neplatí. Povýšení balíčku se totiž ve statistice projeví jako jeho další stažení, uživatelé také mohou aplikaci provozovat na více zařízeních. Stažené balíky lze dále distribuovat, takže jedno stažení balíčku může znamenat i >1 reálných uživatelů.

Jako nejlepší se jeví sledování špičky v počtu stažených balíčků po vydání nové verze. Tyto špičky trvají obvykle několik dnů – do té doby než většina uživatelů provede povýšení aplikace. Počet stažení za tento časový úsek je možné sečíst a od výsledku odečíst průměrné množství stažení krát daný počet dnů špičky. Výsledkem pak bude hrubý odhad počtu aktivních instalací aplikace.

---

1. Nemo Mobile, Android, BlackBerry 10

Další možností, kterou využívá řada mobilních aplikací, by bylo přidání sledovací funkcionality, která by na centrální server hlásila každé spuštění aplikace a případně i další data o jejím běhu. Tento přístup by jistě umožnil získat velmi přesná data o uživatelích modRany, podle mého názoru by se však jednalo o hrubé narušení jejich soukromí. Sledování uživatelské aktivity jsem tedy do modRany neimplementoval a nikdy je také implementovat nehodlám.

6.2 Mieru

Mieru je flexibilní čtečka mangy a komiksu, která byla vytvořena podle stejných zásad pro tvorbu multiplatformní aplikace jako modRana. Tvoří ji tedy rovněž jádro a platformní a grafické moduly. Vzhledem k tomu, že se jedná o podstatně jednodušší aplikaci než jakou je modRana, byla Mieru použita pro prvotní přenos implementace komponent na nové platformy. Mieru již byla také zveřejněna v několika centrálních repositářích různých mobilních platforem. Díky tomu si tak lze udělat představu o tom, jaká data o uživatelích aplikace je na těchto platformách možné získat.

platforma	rozhraní
Maemo 5	Clutter & QML rozhraní
MeeGo 1.2 Harmattan	QML rozhraní
stolní počítače	Clutter & QML rozhraní
Android	QML rozhraní
BlackBerry 10	QML rozhraní
Nemo Mobile	QML rozhraní

Tabulka 6.1: Tabulka platforem podporovaných aplikací Mieru

6.2.1 Mieru v Ovi Store

Mieru byla v Ovi Store zveřejněna začátkem roku 2012. K datu 8.5.2012 měla aplikace Mieru z Ovi Store celkem 15806 stažení. Nejvíc stažení se vyskytlo ihned po vydání (bylo jich několik set denně), poté se počet stažení rychle snížil, až se stabilizoval na současných přibližně 10 staženích denně.

Co do celkového počtu stažení vede bezkonkurenčně Čína, na druhém místě je Vietnam a na třetím Irák. Třetí místo poněkud vzbuzuje pochyby o přesnosti geografických statistik v Ovi store.

Obrázek 6.1: Mieru běžící na PC, chytrých telefonech Nokia N900, N9, BlackBerry Dev Alpha a tabletu HP TouchPad

6.2.2 Mieru v BlackBerry World

Mieru byla v BlackBerry World zveřejněna v únoru 2013. Od té doby byla aplikace stažena pouze celkem 269 krát a průměrný počet stažení za den je 4 až 5. Co tomuto repositáři chybí na objemu stažení, to dohání svou geografickou pestrostí. Nejvíc stažení proběhlo pochopitelně z Kanady[2], ale také např. z Indie, Jihoafrické republiky, Singapuru, Indonésie, Nigérie nebo Ekvádoru. Podobně jako v Ovi Store nastává otázka, jestli opravdu někdo čte mangu či komiks s pomocí Mieru v těsné blízkosti rovníku, nebo jestli má firma BlackBerry rovněž problémy se správnou lokalizací původu jednotlivých stažení.

6.2.3 Mieru v Maemo Extras

Podobně jako modRana je i Mieru k uživatelům ke stažení k dispozici z repositáře Maemo Extras. Během migrace platformy Maemo 5 do rukou uživatelské komunity na konci roku 2012 však přestaly fungovat statistiky stažení pro aplikace v Maemo Extras. V současnosti tedy nelze odhadnou, kolik uživatelů si Mieru z Maemo Extras stáhlo.

2. BlackBerry je kanadská firma a její zařízení jsou v Kanadě velmi populární

6.3 Python & Qt na Androidu

Když jsem začal pracovat na tom, aby bylo možné vytvářet grafické aplikace v Pythonu pro Android, stanovil jsem si dva cíle:

- umožnit běh modRany a Mieru na Androidu
- celý proces co nejvíce zjednodušit a zdokumentovat a otevřít tak Android pro aplikace v Pythonu

Jsem toho názoru, že oba cíle se mi podařilo splnit.

6.3.1 Pyside for Android

Po otestování, že vše funguje jak má, jsem vytvořil podrobný návod, popisující jak kompilaci PySide pro Android, tak vnitřní fungování celého projektu a ukázkové aplikace. Návod jsem nejdříve zveřejnil na wiki projektu modRana a následně – na popud PySide komunity – na wiki Qt-Projectu[138].

6.3.2 Projekt pydroid

Po zveřejnění mého návod na vývoj aplikací v Pythonu a Qt pro platformu Android a oznámení tohoto návodu na emailovou konferenci projektu PySide, mne kontaktoval Aaron Richiger ze Švýcarska. Můj pokrok s vývojem aplikací v Pythonu pro Android ho zaujal, a proto se rozhodl celý proces vylepšit a ještě víc pro vývojáře aplikací zjednodušit. Výsledkem jeho iniciativy je projekt pydroid[69], vyvinutý primárně Aaronem Richigerem. Z mé strany pocházely rady a pomáhal jsem mu při řešení některých nízkoúrovňových záležitostí.

Projekt pydroid je silným vývojářským nástrojem, umožňujícím velmi jednoduchou tvorbu aplikací pro Android v Pythonu. Umožňuje snadné vytvoření kostry projektu nebo přejmenování již existujícího projektu[3]. Vývoj aplikace umožňuje pydroid jak s pomocí Qt Creatoru, tak i přímo z příkazové řádky. Archívy s knihovnami a aplikačními daty projekt pydroid vytváří zcela automaticky a aplikace během instalace knihoven a aplikačních dat zobrazuje názorný procentuální průběh.

Pydroid také podporuje tzv. rychlé spuštění aplikace, které je možné díky tomu, že jak Python, tak QML není nutné před spuštěním kompilovat. Pydroid v tomto režimu jednoduše zkopíruje pouze aplikační data na zařízení a spustí aplikační aktivitu. Projekt pydroid má však i mnoho dalších užitečných funkcí jako jsou např. podpora pro logování do systémového

3. jméno aplikace pro Android musí být jedinečné

logu na Androidu, instalaci balíčků z PIP, zabudované ukázkové aplikace, podpora pro automatické doplňování příkazů atd.. Nejlepší přehled o dostupných funkcích lze získat po spuštění pydroidu s parametrem `-help`.

6.4 BlackBerry 10

6.4.1 Využití Qt Components aplikací AeroTests

Mieru a modRana používají grafický framework Qt Components. Při portování Mieru a modRany na platformu BlackBerry 10 bylo nutné tento framework upravit pro fungování na zařízeních s vysokým DPI. Takto upravený framework již od té doby našel využití v aplikaci Aerotests[139], která slouží jako pomůcka pro výuku pilotů ultralehkých letadel.

7 Závěr

Vývoj v oblasti mobilních platforem je velmi rychlý. Stále se objevují nové skutečnosti, a proto bylo nutné text této diplomové práce průběžně aktualizovat. Jakmile však práce půjde do tisku, nebude již možné ji dále aktualizovat. Je pravděpodobné, že část informací, které byly aktuální v době, kdy diplomová práce vznikala, již po jejím dokončení aktuální být nemusí.

ModRana a její příbuzné projekty sice v současnosti podporují velké množství platforem, přesto však existují některé zajímavé platformy, které modRanou podporovány nejsou. Jedná se například o platformu Ubuntu Touch, která začala být pro modRanu potenciálně dostupnou teprve začátkem května 2012, a to díky zveřejnění PyQt5. Dále se jedná o platformu Sailfish, která je díky podpoře pro Nemo Mobile již částečně modRanou podporována. Vzhledem k tomu, že první zařízení s operačním systémem Sailfish již bylo oficiálně oznámeno, bylo by vhodné přidat plnohodnotnou podporu i pro tuto platformu.

Hlavním cílem dalšího rozvoje modRany je vylepšování QML GUI takovým způsobem, aby bylo dosaženo funkční parity s GTK GUI. Pak bude možné zveřejnění modRany s QML GUI do hlavních aplikačních repositářů platforem Android a BlackBerry 10.

Z dalších funkci, které by bylo vhodné do modRany přidat, patří např. podpora pro vektorové vykreslování mapových podkladů, podpora pro překlady navigačních pokynů, vylepšené offline routování nebo možnost kreslení do mapy.

Ačkoli projekt pydroid vznikl velmi nedávno, jedná se v současné podobě již o velmi funkční nástroj, jak dokazuje využití projektu pydroid při přenosu implementace modRany na platformu Android. K dokonalosti scházejí pydroidu dvě věci: možnost přímého využívání API platformy Android a dokumentace. Vyřešení prvního problému je na dobré cestě, a to díky nástroji pro přímé volání JNI API, který nese název Pyjnius. Pokud jde o dokumentaci, obsahuje pydroid poměrně podrobnou nápovědu, která chybějící dokumentaci do značné míry nahrazuje.

Vše tak naznačuje, že jakmile budou tyto dva drobné nedostatky odstraněny, zaznamená projekt pydroid velké úspěchy.

Literatura

[1] Ed Burnette. *Hello, Android: introducing Google's mobile development platform*. Pragmatic Bookshelf, 2009.

[2] Open Headset Alliance [online]., [citováno 19. 4. 2013]. Dostupný z WWW:
`http://www.openhandsetalliance.com`.

[3] Aaftab Munshi, Dan Ginsburg, and Dave Shreiner. *OpenGL ES 2.0 programming guide*. Addison-Wesley Professional, 2008.

[4] C Patauner, H Witschnig, D Rinner, E Maier, A and Merlin, and E Leitgeb. High Speed RFID/NFC at the Frequency of 13.56 MHz. In *The First International EURASIP Workshop on RFID Technology, RFID*, 2007.

[5] NFC Forum : About NFC [online]., [citováno 1. 4. 2013]. Dostupný z WWW:
`http://www.nfc-forum.org/aboutnfc/`.

[6] Ibrahim Haddad. Introduction to the MeeGo software platform. *Linux Journal*, 2010(198):5, 2010.

[7] Shannon Schroeder. Introduction to MeeGo. *Pervasive Computing, IEEE*, 9(4):4–7, 2010.

[8] Mer Project [online]., [citováno 28. 4. 2013]. Dostupný z WWW:
`http://merproject.org/`.

[9] gerrit - Gerrit Code Review - Google Project Hosting [online]., [citováno 28. 4. 2013]. Dostupný z WWW:
`http://code.google.com/p/gerrit/`.

[10] Open Build Service [online]., [citováno 22. 5. 2013]. Dostupný z WWW:
`http://openbuildservice.org/`.

[11] Image Creation - Mer Wiki [online]., [citováno 28. 4. 2013]. Dostupný z WWW:
`https://wiki.merproject.org/wiki/Image_Creation`.

[12] Platform SDK - Mer Wiki [online]., [citováno 28. 4. 2013]. Dostupný z WWW:
https://wiki.merproject.org/wiki/Platform_SDK.

[13] Nemo Mobile [online]., [citováno 28. 1. 2013]. Dostupný z WWW:
https://wiki.merproject.org/wiki/Nemo.

[14] přehled projektů distribuce Nemo Mobile na Mer OBS [online]., [citováno 12. 4. 2013]. Dostupný z WWW:
https://build.merproject.org/project/subprojects?project=nemo.

[15] Mer Bugzilla [online]., [citováno 12. 4. 2013]. Dostupný z WWW:
https://bugs.merproject.org/.

[16] Welcome - Mer Project Open Build Service [online]., [citováno 12. 4. 2013]. Dostupný z WWW:
https://build.merproject.org/.

[17] SailfishOS.org [online]., [citováno 10. 3. 2013]. Dostupný z WWW:
https://sailfishos.org/.

[18] We are Jolla. We are Unlike. [online]., [citováno 10. 3. 2013]. Dostupný z WWW:
http://jolla.com/.

[19] Sailfish Silica 1.0: Creating applications with Sailfish Silica [online]., [citováno 10. 3. 2013]. Dostupný z WWW:
https://sailfishos.org/sailfish-silica/.

[20] QA - SailfishOS [online]., [citováno 10. 3. 2013]. Dostupný z WWW:
https://sailfishos.org/wiki/QA.

[21] Oracle VM VirtualBox [online]., [citováno 10. 3. 2013]. Dostupný z WWW:
https://www.virtualbox.org/.

[22] The first Jolla phone: 4.5-inch display, Android app compliant, 399 euros [online]., [citováno 21. 5. 2013]. Dostupný z WWW:
http://www.engadget.com/2013/05/20/jolla-phone/.

[23] The Movement - Jolla [online]., [citováno 21. 5. 2013]. Dostupný z WWW:
https://join.jolla.com/.

[24] Plasma Active [online]., [citováno 13. 4. 2013]. Dostupný z WWW: http://plasma-active.org/.

[25] KDE - Experience Freedom! [online]., [citováno 13. 4. 2013]. Dostupný z WWW: http://www.kde.org/.

[26] Plasma Active Core Development[online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://community.kde.org/Plasma/Active/Development#Plasma_Active_Core_Development.

[27] emailová konference projektu Plasma Active [online]., [citováno 18. 4. 2013]. Dostupný z WWW: https://mail.kde.org/mailman/listinfo/active.

[28] Plasma Active SDK [online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://community.kde.org/Plasma/Active/Development#Plasma_SDK:_Plasmate.

[29] Developing Active Apps [online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://community.kde.org/Plasma/Active/Development#Developing_Active_Apps.

[30] Spark: Tablet s Linuxem a bohatou konektorovou výbavou [online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://smartmania.cz/bleskovky/spark-tablet-s-linuxem-a-bohatou-konektorovou-vybavou-1949.

[31] Vivaldi tablet s KDE za 200 EUR [online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://www.root.cz/zpravicky/vivaldi-tablet-s-kde-za-200-eur/.

[32] Plasma/Active/Devices [online]., [citováno 18. 4. 2013]. Dostupný z WWW: http://community.kde.org/Plasma/Active/Devices.

[33] Vivaldi KDE open source Linux tablet gets new hardware, could launch this spring [online]., [citováno 18. 4. 2013]. Dostupný z WWW:

http://liliputing.com/2013/02/vivaldi-kde-open-source-linux-tablet-gets-new-hardware-could-launch-this-spring.html.

[34] Plasma/Active/Devices [online]., [citováno 18. 4. 2013]. Dostupný z WWW:
http://community.kde.org/Plasma/Active/Devices.

[35] maemo.org:Home of the Maemo community [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://maemo.org/.

[36] The GTK+ Project [online]., [citováno 22. 5. 2013]. Dostupný z WWW:
http://www.gtk.org/.

[37] Gnu general public license v3.0 - gnu project - free software foundation (fsf) [online]., [citováno 8. 3. 2013]. Dostupný z WWW:
http://www.gnu.org/licenses/gpl.html.

[38] Hildon Reference Manual [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://maemo.org/api_refs/5.0/beta/hildon/.

[39] Kernel Power [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://wiki.maemo.org/Kernel_Power.

[40] Clutter Project | Have fun! [online]., [citováno 22. 5. 2013]. Dostupný z WWW:
http://blogs.gnome.org/clutter/.

[41] Documentation/Maemo 5 Developer Guide/Architecture/Multimedia Domain - maemo.org wiki [online]., [citováno 19. 4. 2013]. Dostupný z WWW:
http://wiki.maemo.org/Documentation/Maemo_5_Developer_Guide/Architecture/Multimedia_Domain.

[42] GStreamer: open source multimedia framework [online]., [citováno 19. 4. 2013]. Dostupný z WWW:
http://gstreamer.freedesktop.org/.

[43] Software/PulseAudio [online]., [citováno 19. 4. 2013]. Dostupný z WWW:
http://www.freedesktop.org/wiki/Software/PulseAudio.

[44] Extras [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://wiki.maemo.org/Extras.

[45] Community SSU [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://wiki.maemo.org/Community_SSU.

[46] gpodder, a free podcast aggregator for linux, mac os x, windows, freebsd and meego [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://gpodder.org/.

[47] Announce] modRana: a flexible GPS navigation system[online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://talk.maemo.org/showthread.php?t=58861.

[48] Nokia N900 - maemo.org wiki [online]., [citováno 19. 4. 2013]. Dostupný z WWW: http://wiki.maemo.org/Nokia_N900.

[49] Tracker [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://projects.gnome.org/tracker/features.html.

[50] MeeCoLay - maemo.org wiki [online]., [citováno 20. 4. 2013]. Dostupný z WWW: http://wiki.maemo.org/MeeCoLay.

[51] Elena Reshetova and Casey Schaufler. Mobile simplified security framework overview. In *MeeGo Conference*, 2010.

[52] INCEPTION 0.2.5: Assume direct control with PR1.3! - maemo.org - Talk [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://talk.maemo.org/showthread.php?t=85238.

[53] Nokia Store [online]., [citováno 19. 4. 2013]. Dostupný z WWW: http://store.ovi.com/.

[54] Apps for MeeGo [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://apps.formeego.org/applications/.

[55] QtSDK 1.2.1 [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://www.developer.nokia.com/info/sw.nokia.com/id/da8df288-e615-443d-be5c-00c8a72435f8/Qt_SDK.html.

[56] Veli Mankinen and Valtteri Rahkonen. Cross-Compiling tutorial with Scratchbox, 2005.

[57] Qt Creator 2.7 | Documentation | Qt Project [online]., [citováno 22. 5. 2013]. Dostupný z WWW:
http://qt-project.org/doc/qtcreator-2.7/.

[58] Nokia N9-00 Specifications - Nokia [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://www.nokia.com/global/products/phone/n9/specifications/.

[59] Martin Kolman. N950 development kit. *Openmobility.cz*.

[60] Maemo Wiki, Nokia N950 [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://wiki.maemo.org/Nokia_N950.

[61] Android Mainlining Project [online]., [citováno 30. 1. 2013]. Dostupný z WWW:
http://elinux.org/Android_Mainlining_Project.

[62] Edward J Naughton. The Bionic Library: Did Google Work Around the GPL. *Brown Rudnick Advisory*, 2011.

[63] [Learn to Fish] - General Structure of an APK [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://forum.sdx-developers.com/index.php?topic=3472.0.

[64] JAR File Specification [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://docs.oracle.com/javase/6/docs/technotes/guides/jar/jar.html.

[65] Google Play [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
https://play.google.com/store.

[66] Android SDK | Android Developers [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://developer.android.com/sdk/index.html.

[67] Eclipse - The Eclipse Foundation open source community website [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://www.eclipse.org/.

[68] Necessitas/InstallSDK [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://community.kde.org/Necessitas/InstallSDK.

[69] raaron/pydroid · github[online]., [citováno 21. 4. 2013]. Dostupný z WWW:
https://github.com/raaron/pydroid.

[70] Google: 500 million Android devices activated [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://news.cnet.com/8301-1035_3-57510994-94/google-500-million-android-devices-activated/.

[71] Tizen | an open source, standards-based software platform for multiple device categories [online]., [citováno 5. 4. 2013]. Dostupný z WWW:
https://www.tizen.org/.

[72] [Tizen General] Samsung clobbering all of Intel/others work?? [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
https://lists.tizen.org/pipermail/general/2012-October/001074.html.

[73] [Phoronix] Intel: Samsung Clobbering Others With Tizen [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.phoronix.com/scan.php?page=news_item&px=MTIwMDU.

[74] libhybris/libhybris · GitHub [online]., [citováno 5. 5. 2013]. Dostupný z WWW:
https://github.com/libhybris/libhybris.

[75] Touch/Contribute - Ubuntu Wiki [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
https://wiki.ubuntu.com/Touch/Contribute.

[76] Touch/Devices - Ubuntu Wiki [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
https://wiki.ubuntu.com/Touch/Devices/.

[77] BlackBerry 10 - See The New BlackBerry Z10 & BlackBerry Q10 - Global [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://global.blackberry.com/blackberry-10.html.

[78] BlackBerry Z10 - The New BlackBerry Z10 is Here - Global [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://global.blackberry.com/smartphones/blackberry-z10/en.html.

[79] BlackBerry Q10 Smartphone With QWERTY Keyboard - BlackBerry Q10 Release - Global [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://global.blackberry.com/smartphones/blackberry-q10.html.

[80] Aaron Weiss. WebOS: say goodbye to desktop applications. *Networker*, 9(4):18–26, 2005.

[81] Mitch Allen. *Palm webOS*. O'Reilly Media, 2009.

[82] Palm Pre Specifications [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://www.weboshelp.net/webos-mojo-development-resources/palmprespecifications.

[83] HP TouchPad Wi-Fi 16 GB 9.7-inch Review and Specifications | TechCrunch [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://tablets.techcrunch.com/1/58/HP-TouchPad-Wi-Fi-16-GB-9-7-inch.

[84] BREAKING: HP shutting down webOS device operations, will "continue to explore options"| webOS Nation [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://www.webosnation.com/breaking-hp-shutting-down-webos-device-operations-will-continue-explore-options.

[85] Open webOS [online]., [citováno 10. 3. 2013]. Dostupný z WWW: http://www.openwebosproject.org/.

[86] HP News - LG Electronics Acquires webOS from HP to Enhance Smart TV [online]., [citováno 4. 5. 2013]. Dostupný z WWW: http://www8.hp.com/us/en/hp-news/press-release.html?id=1375489.

[87] Darren McCarra. Defunct HP TouchPad now an Amazon bestseller. 2011.

[88] freedreno open source graphics drivers [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://freedreno.github.io/.

[89] Adaptation/Touchpad - Mer Wiki [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
https://wiki.merproject.org/wiki/Adaptation/Touchpad.

[90] HP Touchpad running Ubuntu Touch - xda-developers [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://forum.xda-developers.com/showthread.php?t=2175277.

[91] Openembedded.org [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.openembedded.org/.

[92] directfb.org | Main [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://directfb.org/.

[93] The WebKit Open Source Project [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.webkit.org/.

[94] openwebos (Open webOS) [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
https://github.com/openwebos.

[95] Pierluigi Frullani, Carl Worth, and Steve Ayer. ipkg–Itsy Package Management System http://handhelds. org/moin/moin. cgi.

[96] Preware [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://preware.org/#/index/.

[97] Twitter / webosinternals: Preware 0.9.4, with full package ... [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
https://twitter.com/webosinternals/status/4779885909.

[98] webOS hacks, Python on webOS and the python-webos module (thp.io) [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://thp.io/2011/webos/.

[99] pygame_touchpad – gps navigace [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://modrana.org/trac/wiki/pygame_touchpad.

[100] Pygame on the hp touchpad [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.modrana.org/touchpad_pygame/.

[101] World Wide Web Consortium (W3C) [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.w3.org/.

[102] Firefox OS security overview - Mozilla | MDN [online]., [citováno 19. 4. 2013]. Dostupný z WWW:
https://developer.mozilla.org/en-US/docs/Mozilla/Firefox_OS/Security/Security_model#Trusted_and_Untrusted_Apps.

[103] SHR [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://shr-project.org.

[104] Neo FreeRunner [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://wiki.openmoko.org/wiki/Neo_FreeRunner.

[105] Devices/NokiaN900 – SHR [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.shr-project.org/trac/wiki/Devices/NokiaN900.

[106] Enlightenment Foundation Libraries [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.enlightenment.org/?p=about/efl.

[107] Freesmartphone.org [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.freesmartphone.org/.

[108] QtMoko [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://qtmoko.sourceforge.net/.

[109] OpenPhoenux - Open Pho(n)e (w. Li)nux - The independent Open Mobile Tool community [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://projects.goldelico.com/p/openphoenux/.

[110] GTA04 - Master page for the next generation Openmoko motherboard [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://projects.goldelico.com/p/gta04-main/.

[111] Cordia Hildon-Desktop [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://cordiahd.org/.

[112] Seadot - Home [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.seadot.org/.

[113] ExoPC [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://www.exopc.com/.

[114] Enigma [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://enigma-project.org/.

[115] Apple - iOS [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.apple.com/ios/.

[116] Apple [online]., [citováno 26. 4. 2013]. Dostupný z WWW:
http://www.apple.com/.

[117] Daring Fireball: A Taste of What's New in the Updated App Store License Agreement and New Review Guidelines [online]., [citováno 3. 5. 2013]. Dostupný z WWW:
http://daringfireball.net/2010/09/app_store_guidelines.

[118] No GPL Apps for Apple's App Store | ZDNet [online]., [citováno 3. 5. 2013]. Dostupný z WWW:
http://www.zdnet.com/blog/open-source/no-gpl-apps-for-apples-app-store/8046.

[119] Mobile Matrix (thp.io) [online]., [citováno 3. 5. 2013]. Dostupný z WWW:
http://thp.io/2012/mobile-matrix/.

[120] Qt Project [online]., [citováno 8. 3. 2013]. Dostupný z WWW:
http://qt-project.org/.

[121] Jozef Mlich, Pavel Zemčík, and Aleš Láník. Vývoj aplikací v Qt pro mobilní zařízení. *Sborník přízpěvků*, (41):93–104, 2012.

[122] GNU Lesser General Public License v3.0 - GNU Project - Free Software Foundation (FSF) [online]., [citováno 8. 3. 2013]. Dostupný z WWW:
http://www.gnu.org/copyleft/lesser.html.

[123] Mobilní Telefony a Dotykové Mobily - Nokia - Česká republika [online]., [citováno 8. 3. 2013]. Dostupný z WWW:
http://www.nokia.com/.

[124] Digia Plc [online]., [citováno 8. 3. 2013]. Dostupný z WWW:
http://www.digia.com/.

[125] Ben Morris. *The Symbian Os Architecture Sourcebook: Design and Solution of a Mobile Phone Os*. John Wiley & Sons, 2007.

[126] Qt for Tizen Blog: Qt Launches on Tizen with Standard Look and Feel [online]., [citováno 21. 5. 2013]. Dostupný z WWW:
http://qtfortizen.blogspot.cz/2013/05/1.0alpha1.html.

[127] Simple DirectMedia Layer [online]., [citováno 9. 3. 2013]. Dostupný z WWW:
http://www.libsdl.org/.

[128] Cloud GPS [online]., [citováno 4. 5. 2013]. Dostupný z WWW:
http://store.ovi.com/content/222534.

[129] GPS Navigace [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://www.modrana.org.

[130] android-python27 [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://code.google.com/p/android-python27/.

[131] python-for-android [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://github.com/kivy/python-for-android.

[132] Welcome to KDE Necessitas project [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://necessitas.kde.org/.

[133] PySide for Android [online]., [citováno 21. 4. 2013]. Dostupný z WWW:
http://thp.io/2011/pyside-android/.

[134] Building PySide [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://hg.microcode.ca/blackberry-py/wiki/Building%20PySide.

[135] BlackBerry-Py Project [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://blackberry-py.microcode.ca/.

[136] Six: Python 2 and 3 Compatibility Library [online]., [citováno 21. 4. 2013]. Dostupný z WWW: http://pythonhosted.org/six/.

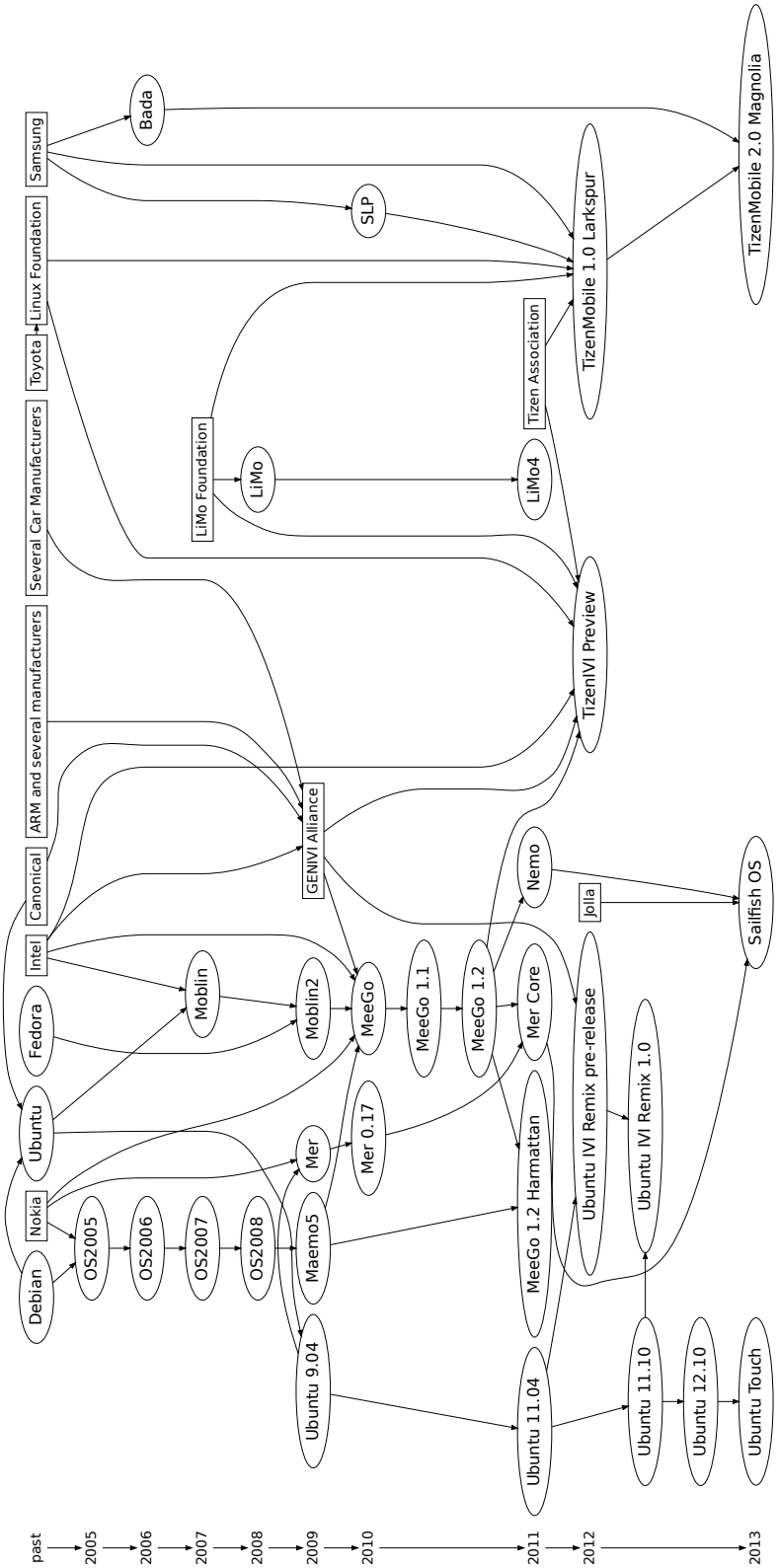
[137] Making an application compatible with Python 2.5 to 3.2 at the same time [online]., [citováno 9. 5. 2013]. Dostupný z WWW: http://modrana.org/trac/wiki/python2and3CompatiblityEN.

[138] PySide for Android [online]., [citováno 20. 5. 2013]. Dostupný z WWW: http://qt-project.org/wiki/PySide_for_Android_guide.

[139] BlackBerry World - Aerotests [online]., [citováno 20. 5. 2013]. Dostupný z WWW: /http://appworld.blackberry.com/webstore/content/24422931/.

[140] kumadasu/tizen-history · github [online]., [citováno 20. 5. 2013]. Dostupný z WWW: https://github.com/kumadasu/tizen-history/.

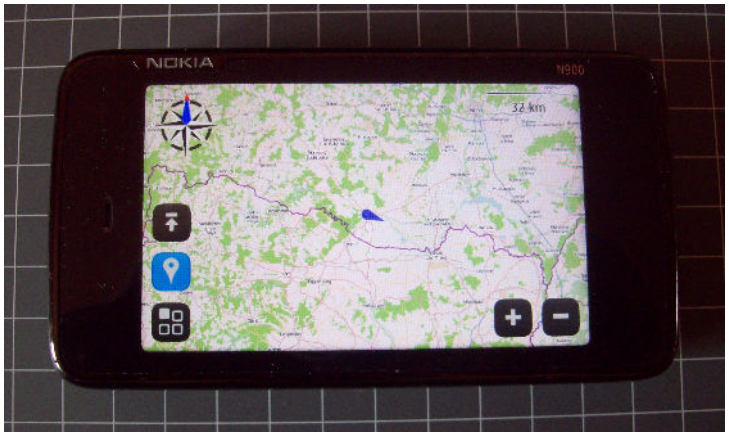
A Rodokmen mobilních linuxových platforem



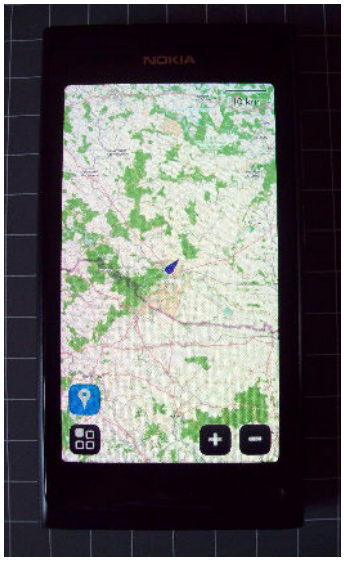
Obrázek A.1: Rodokmen mobilních linuxových platforem[140]

B Galerie navigačního systému modRana na různých mobilních platformách

Obrázek B.1: modRana s GTK rozhraním na chytrém telefonu Nokia N900 s Maemo 5 Fremantle



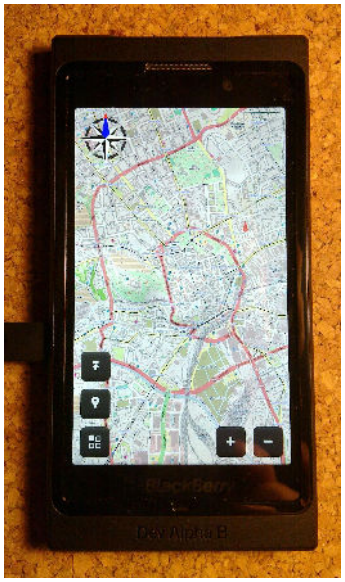
Obrázek B.2: modRana s QML rozhraním na stejném zařízení a platformě



Obrázek B.3: modRana s QML rozhraním na chytrém telefonu Nokia N9 s MeeGo 1.2 Harmattan

Obrázek B.4: modRana s GTK rozhraním na chytrém telefonu Nokia N950 s Nemo Mobile

Obrázek B.5: modRana s QML rozhraním na tabletu HP TouchPad s Androidem 4.0



Obrázek B.6: modRana s QML rozhraním na vývojářském zařízení BlackBerry Dev Alpha s platformou BlackBerry 10

C Obsah DVD

Přiložené DVD obsahuje:

- text diplomové práce v elektronické podobě
- zdrojové kódy diplomové práce
- zdrojové kódy multiplatformní aplikace modRana
- instalační balíčky modRany pro různé platformy
- prezentace a ukázkové materiály z konferencí OpenMobility 2012 a 2013