ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	MobChar - balíček pro Dračí doupě 3.0
Student:	Šárka Weberová
Vedoucí:	Ing. Zdeněk Rybola
Studijní program:	Informatika
Studijní obor:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2017/18

Pokyny pro vypracování

Cílem bakalářské práce je rozšířit funkcionalitu stávajícího balíčku pro Dračí doupě, který je součástí aplikace MobChar.

Mezi hlavní rozšiřující funkce patří:
- komunikace s balíčkem pro Pána jeskyně v Dračím doupěti - nahlížení na postavu, předávání předmětů, soukromé zprávy,
- odvozování hodnot pro dovednosti, kouzla, atd.
- simulace hodů kostkou,
- příp. další funkce po domluvě s vedoucím práce.

V souladu s postupy softwarového inženýrství:
- analyzujte stávající funkčnost balíčku pro Dračí doupě a jádra aplikace,
- analyzujte nové požadavky na rozšíření funkčnosti,
- popište architekturu balíčku a navrhněte realizaci nových požadavků v této architektuře,
- implementujte nové funkce v souladu s provedeným návrhem,
- implementaci patřičně zdokumentujte a otestujte,
- vytvořte uživatelskou příručku pro funkčnost celého balíčku.

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 30. ledna 2017

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Bakalářská práce

MobChar – balíček pro Dračí doupě 3.0

Šárka Weberová

Vedoucí práce: Ing. Zdeněk Rybola

15. května 2017

Poděkování

Chtěla bych poděkovat vedoucímu bakalářské práce, Ing. Zdeňku Rybolovi, za jeho odborné rady a pomoc při tvorbě této práce.

Dále bych chtěla poděkovat Matěji Shánělovi za rady a tipy v oblasti Javy pro Android. Také děkuji za podporu své rodině a svým přátelům.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 15. května 2017 .....

Odkaz na tuto práci

Weberová, Šárka. *MobChar – balíček pro Dračí doupě 3.0*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Cílem této bakalářské práce je rozšíření balíčku Dračí doupě pro aplikaci MobChar. MobChar umožňuje hráčům her na hrdiny spravovat svoje postavy. Jedná se o aplikaci pro mobilní operační systém Android. Balíček Dračí doupě se zaměřuje na českou hru Dračí doupě. V rámci této bakalářské práce se balíček rozšíří o simulaci hodu kostkou, odvozování hodnot atributů dovedností, kouzel a dalších objektů. Dále bude implementována komunikace s nově vznikajícím balíčkem pro Pána jeskyně.

Klíčová slova MobChar, Dračí doupě, Android, Java, hra na hrdiny, mobilní aplikace

Abstract

The goal of this bachelor thesis is to extend the package Dračí doupě for MobChar aplication. MobChar is the Android mobile application which allows players of role playing games to manage their characters. The package Dračí doupě is focused on Czech game named Dračí doupě. The package will be extended by simulation of a dice roll, the derivation of values of abilities, spells and other objects attributes. The other extension is communication with a newly developed package for the Lord of the Cave.

Keywords MobChar, Dračí doupě, Android, Java, role playing game, mobile application

Obsah

Seznam obrázků

Seznam zdrojových kódů

Úvod

Dračí doupě je jedna z českých her na hrdiny. V této hře si každý hráč vymyslí svojí postavu a s ní prožívá příběh vyprávěný Pánem jeskyně. Postavy ve hře, stejně jako v jiných hrách na hrdiny, ztrácejí životy, nabývají zkušeností, nalézají, či kupují předměty atd. Průběh těchto aktivit je detailně popsán v rozsáhlých pravidlech této hry. Pravidla obsahují mnoho tabulek a postupů, podle kterých se vyhodnocují různé situace, které mohou při hře nastat. Tyto postupy často obsahují složité výpočty, které musí hráč provádět vždy, když se do nějaké z těchto situace dostane[1]. Dále si hráč musí někam zapisovat aktuální stav postavy (počet životů, zkušeností...), který se často mění. Tato evidence se nazývá deník postavy.

Hráči v dnešní době mají mnoho možností, v jaké formě deník postavy vést (od papírového sešitu, přes excelovské tabulky až po aplikace pro deník postavy). Žádná z těchto variant není přizpůsobena přímo hře Dračí doupě. Proto u všech současných variant musí hráči stále nahlížet do pravidel a „ručně“ počítat změny atributů jejich postav.

Cílem práce je rozšíření aplikace, která většinu těchto výpočtů provádí sama. Aplikace v současné době umí pracovat s atributy postavy, s inventářem a předměty, dále pak lze v aplikaci evidovat efekty, kouzly a schopnostmi. V aplikaci je také možné si zobrazit všechny změny na postavě, které byly provedeny od jejího vytvoření.

Obsahem bakalářské práce bude analýza stávající aplikace a analýza, návrh a implementace nových funkcí, jako je hod kostkou, odvození hodnot pro schopnosti a kouzla, a dále bude aplikace schopna komunikace s nově vznikající aplikací pro Pána jeskyně. Dalším z výstupů bakalářské práce bude uživatelská příručka pro celý balíček.

Cíl práce

Cílem bakalářské práce je rozšíření balíčku Dračí doupě aplikaci MobChar. Výsledkem bakalářské práce bude funkční mobilní aplikace pro Android (verze 4.1 a vyšší) schopná nasazení.

V první, analytické, části bude analyzována stávající funkcionalita balíčku Dračí doupě a jádra aplikace. Také budou v této části analyzovány nové funkce. Druhá část bude zaměřena na návrh nových funkcí a jejich zakomponování do současné aplikace. V realizační čísti budou popsány použité nástroje, implementace nových funkcí, jejich testování a vytvoření uživatelské příručky pro celý balíček.

Mezi nové funkce patří simulace hodu kostkou, odvozování hodnot pro schopnosti, kouzla a další. Další novou funkčností je komunikace s balíčkem pro Pána jeskyně. Součástí této práce je návrh a implementace komunikace pouze ze strany balíčku Dračí doupě. Cílem práce není návrh spojení mezi zařízeními ani návrh a implementace komunikace ze strany balíčku pro Pána jeskyně. To je součástí bakalářské práce Matěje Sháněla[2].

Kapitola 1

Analýza

V analytické části bakalářské práce bude popsán stávající stav aplikace. Bude řečeno, jak je aplikace navržena a co všechno už je implementováno. Největší důraz bude kladen na balíček Dračí doupě a na nové funkčnosti, které budou implementovány jako součást práce. U těchto funkčností bude naznačena jejich potřeba v aplikaci a budou rozebrány případy užití.

1.1 Aplikace MobChar

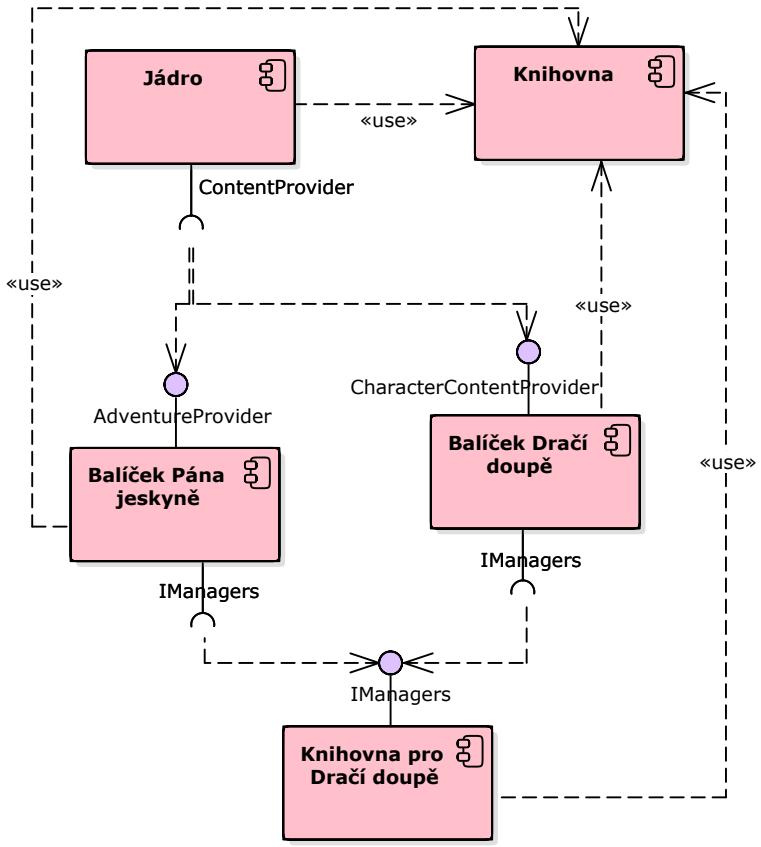
Aplikace MobChar slouží jako elektronická forma deníku postavy. Hráči her na hrdiny si do ní mohou ukládat informace o své postavě.

MobChar je aplikace napsaná v Javě pro Android a je podporována na mobilních zařízeních a tabletech s operačním systémem Andorid verze Jelly Bean (4.1) nebo vyšší. Základ aplikace se skládá z jádra a knihovny. Tyto dvě části slouží k instalaci a implementaci balíčků pro jednotlivé hry. Díky tomuto návrhu je možné mít v aplikaci vytvořené postavy z více balíčků (her) najednou. Vztahy mezi jednotlivými částmi aplikace vyjadřuje diagramu komponent, který je zachycen na obrázku 1.1. V aplikaci je jazyková podpora pro češtinu a angličtinu. Data aplikace jsou ukládána do databáze typu SQLite.

Tato aplikace je již několik let vyvíjena na Fakultě informačních technologií ČVUT týmy studentů jako projekt v předmětech BI-SP1 a BI-SP2 a byly na tuto aplikaci napsány již dvě bakalářské práce. Jedna z bakalářských prací byla na jádro a knihovnu aplikace[3], druhá na balíček Dungeons and dragons[4]. Také současně s touto prací vznikají i dvě další bakalářské práce, které jsou zaměřeny na vytvoření specializovaných aplikací pro Pána jeskyně.

1.1.1 Jádro aplikace

Jádro aplikace slouží jako společné prostředí pro všechny balíčky. V běžící aplikaci je reprezentováno úvodní obrazovkou, na které uživatel vidí seznam všech svých vytvořených postav nehledě na to, z jakého balíčku jsou vytvořeny.



Obrázek 1.1: Diagram komponent aplikace

Také umožňuje vytvoření postavy z uživatelem nainstalovaného a vybraného balíčku. Dále implementuje vyhledávání postav podle jména.

Jádro také zprostředkovává funkci importu a exportu postav, která musí být implementována v jednotlivých balíčcích.

1.1.2 Knihovna

Aplikace programátorovi poskytuje také knihovnu, která slouží k zjednodušení implementace jednotlivých balíčků.

Knihovna obsahuje předpisy tříd pro některé hlavní struktury, jako je například postava nebo předmět. Také obsahuje základní třídy pro uživatelské rozhraní a základ logické a datové vrstvy pro práci s inventářem. Tyto třídy je možné použít či rozšířit při implementaci balíčku. Díky těmto třídám může jádro pracovat s některými objekty v jednotlivých balíčcích bez ohledu na to, ze kterého jsou.

1.1.3 Balíček Dračí doupě

Balíček Dračí doupě umožňuje spravovat vytvořenou postavu. Vnitřní logika a uspořádání simuluje některé z pravidel hry Dračí doupě. Jelikož byl balíček implementován již několika týmy, disponuje mnoha funkcemi. Mezi implementované funkce patří například:

Informace o postavě Uživateli je umožněno spravovat základní údaje o postavě a její atributy.

Předměty Dále umožňuje evidovat různé předměty, které postava vlastní. Předměty je možné vkládat do batohů. Uživateli je umožněno předměty mezi batohy přesouvat či kopírovat, čímž je zajištěna přehlednost. Z předmětů se automaticky dopočítává naložení postavy.

Schopnosti a kouzla Aplikace také umožňuje evidovat schopnosti a kouzla, které daná postava ovládá.

Efekty Další funkčností je evidence efektů. Ke každému efektu je možné přidat libovolný počet modifikátorů, které ovlivňují atributy postavy nebo předmětu. Po aktivaci efektu s modifikátory jsou změny promítnuty do daných atributů.

Bojové statistiky V balíčku lze nalézt také bojové statistiky, kde si může uživatel vytvořit ze svých předmětů set. Tyto sety je možné aktivovat a ze zbraní v setu si poté vybrat tu, se kterou bude postava útočit. Této zbrani jsou automaticky dopočítány atributy potřebné pro boj.

Logování Další funkcí je záznam všech změn, které byly uživatelem na dané postavě provedeny. Každá změna je uložena i s časem jejího provedení. Uživatel si může jednoduše podle data dohledat, kdy co na postavě měnil.

Šablony Balíček také obsahuje již připravené šablony pro předměty, kouzla, schopnosti a efekty. Tyto šablony jsou uloženy v souborech formátu XML a jsou ve složce MobChar/templates, která je vytvořena aplikací. Šablony jsou automaticky překládány do aplikace, kde jsou uživateli k dispozici při tvorbě nového předmětu, kouzla atd. Uživatel si může vytvořit vlastní šablony, a to tak, že je ve správném formátu připíše do některých z aplikací vygenerovaných souborů, nebo do nového souboru v téže složce. Takto uživatelem vytvořené šablony jsou také automaticky překládány a jsou přidány do seznamu šablon.

Import a export Další funkcí, kterou současný balíček Dračí doupě implementuje, je import a export postavy. Tato funkce využívá možnosti překladu jednotlivých položek (předměty, kouzla atd.) do souborů formátu XML. Při exportu existující postavy je vytvořen XML soubor ve složce

`MobChar/character`, který obsahuje kompletní informace o postavě. Při importu je takovýto soubor přeložen zpět a uložen do databáze jako nová postava.

1.2 Nové požadavky na balíček

I když je balíček Dračí doupě již funkční a obsahuje mnoho implementovaných funkcí, zjistily se v průběhu jeho používání nedostatky. Řešení těchto nedostatků se stalo základem této bakalářské práce. Dále pak byl, z důvodu vzniku nového balíčku jako bakalářské práce kolegy Matěje Sháněla[2], definován požadavek na komunikaci s tímto balíčkem. Komunikace ze strany balíčku Dračí doupě je také součástí této bakalářské práce. Nyní budou podrobněji popsány nové požadavky.

1.2.1 Hod kostkou

Ve hře Dračí doupě je hod kostkou častý. Nejedená se vždy o hod jednou nebo dvěma kostkami, může nastat i situace, kdy hráč hází vícekrát a všechny hody je potřeba například sečíst. To už není pro hráče příjemný úkol a je pohodlnější nechat hody spočítat aplikací.

Také jsou ve hře speciální typy hodů, jako je například hod na procenta, které nejsou ve většině simulátorů hodů kostkou k dispozici.

Z těchto důvodů byl definován požadavek na simulaci hodu kostkou přímo v balíčku Dračí doupě. Tato simulace bude umožňovat rychlý přístup k hodům šestistěnnou a desetistěnnou kostkou, které se ve hře používají nejčastěji, dále pak simulaci pro speciální typy hodů a také hod kostkou s libovolným počtem stěn. Bude umožněno hody libovolně opakovat a sčítat jejich výsledky.

1.2.2 Odvozování hodnot pro schopnosti

V zadání bakalářské práce je uveden požadavek na odvozování hodnot atributů pro různé objekty. Po prostudování pravidel hry Dračí doupě bylo usouzeno, že jediným objektem, u kterého má dopočítávání hodnot atributů na základě hodnot atributů postavy smysl, jsou schopnosti[1].

V současné verzi aplikace jsou již schopnosti implementovány, ale pouze základním způsobem. Je možné u nich evidovat jméno, popis a šanci na úspěšné použití. Některé schopnosti mají ale dle pravidel navíc ještě úroveň a jejich šance může být ovlivněna některými atributy postavy.

Návrh a implementace těchto typů schopností se staly součástí této bakalářské práce. Aplikace tedy bude umožňovat evidování schopností, u kterých bude možné uvést úroveň a jejich šance bude automaticky propočítávána dle uživatelem daných závislostí.

1.2.3 Komunikace s balíčkem pro Pána jeskyně

Paralelně s touto bakalářskou prací je vytvářena ještě bakalářská práce[2], jejíž tématem je vytvoření nového balíčku pro Pána jeskyně.

Pán jeskyně je ve hře vypravěčem, který vymýšlí příběh pro ostatní hrdiny. Jeho role je tedy ve hře jiná, než role ostatních hráčů. Balíček pro Pána jeskyně bude navržen k zjednodušení tvorby dobrodružství. Během hry dochází k interakci mezi hráčem a Pánem jeskyně. Některou komunikaci mezi nimi je možné pomocí aplikace zjednodušit.

1.2.3.1 Nahlížení na postavu

Pán jeskyně občas potřebuje získat informace o hrdinech. Aby nemusel o informace žádat hráče, bude balíček Dračí doupě umožňovat poslat balíčku Pána jeskyně celou postavu aktivního hrdiny.

1.2.3.2 Posílání objektů

Pán jeskyně si bude moci v aplikaci předem vytvořit předměty, kouzla, schopnosti a další objekty, které budou moci poté hráči pro své hrdiny najít, naučit se, koupit atd. Aby bylo předání informací o těchto objektech jednodušší, budou balíčky umožňovat jejich posílání.

Jako součást této práce bude řešen požadavek na balíček Dračí doupě. Tímto požadavkem je umožnění přijetí objektu posílaného balíčkem pro Pána jeskyně a uložení tohoto objektu do příslušné sekce v aplikaci (předmět – inventář – zem, kouzlo – kouzla, schopnost – schopnosti, efekt – efekty). Požadavky na balíček pro Pána jeskyně v oblasti posílání zpráv nejsou součástí této práce. Budou řešeny v bakalářské práci Matěje Sháněla[2].

1.2.3.3 Tajné zprávy

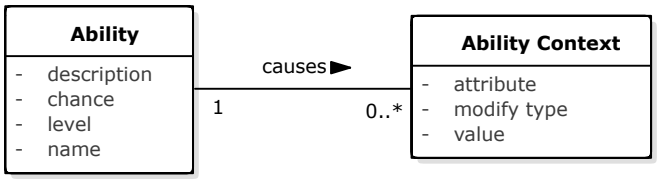
Balíček bude také umožňovat posílání tajných zpráv mezi hrdinou a Pánem jeskyně. Tyto tajné zprávy budou zobrazeny jako konverzace, aby bylo možné zprávy zpětně dohledávat. Příchod nové zprávy bude hráči oznámen upozorněním.

1.2.4 Vrhací zbraně

Vrhací zbraně jsou speciálním typem zbraně na dálku. Odlišují se tím, že mohou být v souboji použity k boji na dálku, ale i na blízko.

V současné době je v aplikaci možné u zbraně na dálku určit typ (vrhací, střelná). Je to pouze atribut, se kterým se dále nepracuje.

Novým požadavkem na balíček je, že bude umožňovat vytvoření vrhací zbraně, které bude možné nastavit hodnotu obrany na blízko a bude ji možné v bojových statistikách uvádět jako zbraň na dálku i jako zbraň na blízko.



Obrázek 1.2: Doménový model nové struktury schopností

1.2.5 Efekty na postavě modifikující předměty

V současné verzi balíčku je možné vytvářet efekty u předmětu nebo u postavy. Efekty předmětů mohou upravovat atributy předmětu i postavy. Efekty postavy v současné verzi mohou modifikovat jen atributy postavy.

Balíček by měl tedy umožnit vytvoření efektů postavy, které budou ovlivňovat atributy všech předmětů. Tyto efekty bude možné vytvořit tak, aby modifikovaly jen některou ze skupin typů předmětů.

1.3 Doménový model

Z důvodu nových funkcí je potřeba upravit současný doménový model aplikace. V doménovém modelu se projeví pouze požadavky na vrhací zbraně, odvozování hodnot pro schopnosti a na komunikaci.

1.3.1 Odvozování hodnot pro schopnosti

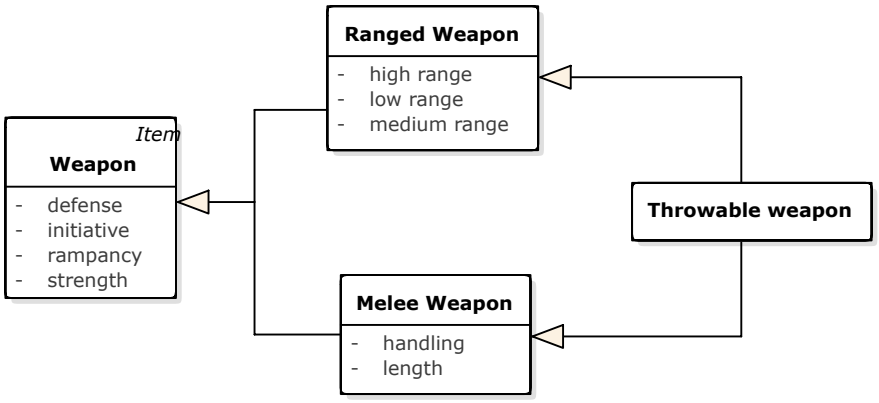
Protože schopnosti, jejichž šance je závislá na atributech postavy, mají stejné atributy, jako již implementované obyčejné schopnosti, bylo navrženo tyto schopnosti pouze rozšířit o kontexty.

Kontext bude představovat ovlivnění šance jedním atributem. Bude udávat, jakým způsobem, jakým atributem a jakou hodnotou bude šance ovlivněna. Každá schopnost bude moci mít neomezený počet těchto kontextů, což umožní hráči vytvořit i schopnosti ovlivněné více atributy. Výsledný objekt představující kontext je zobrazen na obrázku 1.2.

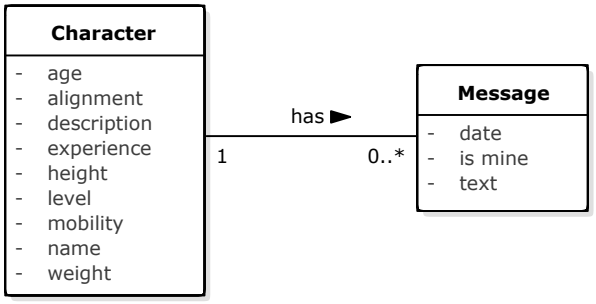
Dále byl všem schopnostem přidán atribut udávající jejich úroveň. Tento atribut nebude povinný, a tak bude moct hráč evidovat i schopnosti bez explicitně zadané úrovně.

1.3.2 Vrhací zbraně

Vrhací zbraň má všechny atributy a funkce jako zbraň na dálku i jako zbraň na blízko. Je proto zbytečné ji vytvářet jako nový typ zbraně. Zároveň nemá smysl ji udávat jako objekt, který dědí pouze od jednoho z typů. Vrhací zbraň byla tedy navržena jako objekt, který dědí od obou původních typů zbraní,



Obrázek 1.3: Doménový model nové struktury vrhacích zbraní

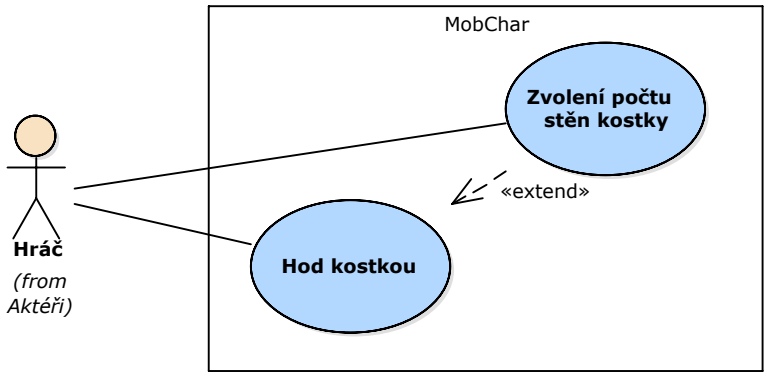


Obrázek 1.4: Doménový model nové struktury pro zprávy

jak je vidět na obrázku 1.3. Tak bude možné používat vrhací zbraň v obou situacích.

1.3.3 Posílání zpráv

V aplikaci bude možné komunikovat s Pánem jeskyně pomocí zasílání zpráv. Aby bylo možné zprávy zpětně dohledat a prohlížet, bude nutné z nich udělat objekt. Každá postava tedy bude moci mít libovolné množství zpráv. Každá zpráva bude obsahovat text, datum jejího odeslání/přijetí a bude u ní nutné rozlišit, jestli je vytvořena hráčem, nebo Pánem jeskyně. Navrhovaná struktura zprávy je znázorněna na obrázku 1.4.



Obrázek 1.5: Diagram případů užití spojenými s hodem kostkou

1.4 Případy užití

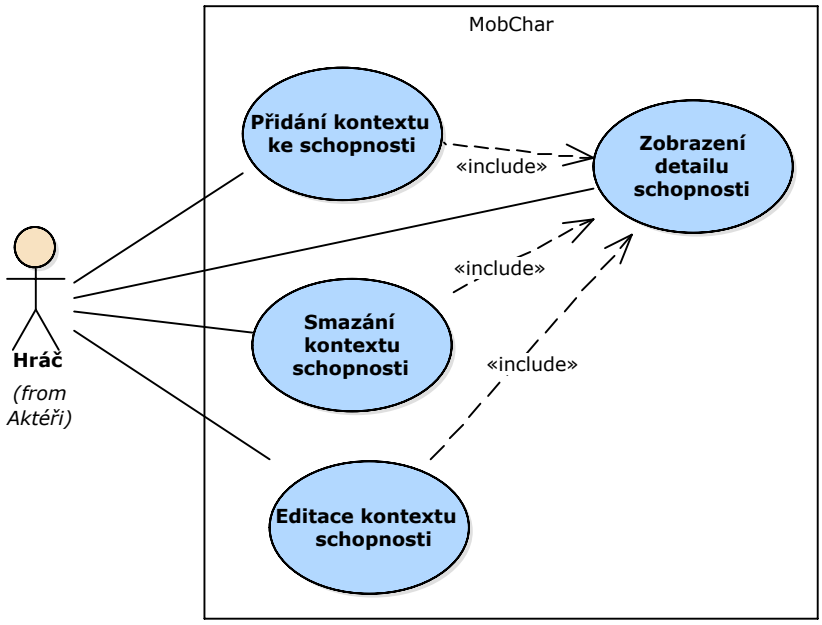
Případy užití popisují, jakým způsobem bude řešena interakce mezi aplikací a uživatelem v nových funkcích. Diagram je rozdělen do více případů užití. Každý z diagramů zachycuje případy užití spojené s danou funkcí.

1.4.1 Hod kostkou

Jedním z diagramů je diagram případů užití pro hod kostkou, který je zobrazen na obrázku 1.5. Součástí tohoto diagramu je případ užití hod kostkou a zvolení počtu stěn kostky.

Hod kostkou Tento případ užití popisuje, jak musí hráč v aplikace postupovat při hodu kostkou. Hráč zvolí na úvodní obrazovce své postavy ikonu kostky. Aplikace mu zobrazí obrazovku pro hod kostkou. Na této obrazovce si hráč může zvolit jakým typem kostky bude házet, počet hodů, které se provedou. Po stisknutí tlačítka „Hoď“ aplikace provede požadovaný výpočet a zobrazí hráči výsledek. Také lze nový hod přičíst či odečíst od předchozího výsledku. Potom je výsledek hodu zapsán ve formě matematického vzorce, aby bylo zřejmé, jaké hodnoty měli jednotlivé sčítané/odečítané hody.

Zvolení počtu stěn kostky Pokud chce hráč hodit kostkou, která má jiný počet stěn, než jsou předvolené hodnoty, bude postupovat podle tohoto případu užití. Hráč dlouze klikne na typ hodu s nastavitelným počtem stěn kostky. Aplikace hráči umožní zadat požadovaný počet stěn. Potom, co hráč potvrdí svou volbu, zobrazí aplikace počet stěn ve volbě místo původního.



Obrázek 1.6: Diagram případů užití spojenými s prácemi se schopnostmi

1.4.2 Odvozování hodnot pro schopnosti

Pro splnění požadavku na odvozování hodnot u schopností je třeba vytvořit nové případy užití pro práci s nově vzniklými kontexty. Jejich seznam je na obrázku 1.6.

Zobrazení detailu schopnosti Součástí všech případů užití, které jsou spojené s kontexty, je zobrazení detailu schopnosti. Aby se hráč dostal na detail schopnosti, musí nejdříve přejít do záložky „Schopnosti“. V této záložce se zobrazí seznam schopností, které postava ovládá. Hráč klikne na schopnost ze seznamu, jejíž detail chce zobrazit. Aplikace zobrazí obrazovku s názvem, popisem, šancí a úrovní schopnosti. Na této obrazovce je také seznam všech kontextů, které schopnost ovlivňují.

Přidání kontextu ke schopnosti Chce-li hráč vytvořit nový kontext schopnosti, je potřeba se dostat na detail schopnosti, kterou chceme upravit. V tomto detailu je v horní menu obrazovky symbol „+“ pro přidání nového kontextu. Když uživatel klikne na tento symbol, zobrazí se obrazovka s možností zadat atribut, typ změny a hodnotu změny. Atributem kontextu může být kterýkoliv z atributů postavy a typ změny může být buď přičtení násobku hodnoty atributu, nebo přičtení daných procent ze základu či z aktuální hodnoty atributu. Po zadání těchto hodnot klikne hráč na tlačítko pro uložení kontextu. Aplikace zkontro-

luje, zda jsou zadaná data validní, a pokud ano, tak nový kontext uloží do databáze a zobrazí hráči zpět detail schopnosti i s nově vytvořeným kontextem.

Smazání kontextu schopnosti Pro smazání kontextu se hráč musí dostat na detail schopnosti, u které chce kontext/kontexty smazat. Dlouhým kliknutím se kontext označí a v menu přibude ikona pro smazání. Hráč může v této chvíli označit i další kontexty ke smazání. Má-li hráč označené všechny kontexty, které chce smazat, klikne na ikonu pro smazání. Aplikace vymaže dané kontexty a zobrazí detail schopnosti již bez nich.

Editace kontextu schopnosti Chce-li hráč upravit nějaký z již existujících kontextů, klikne na něj. Aplikace mu zobrazí obrazovku shodnou s obrazovkou pro vytváření kontextu, ve které budou předvyplněné aktuální hodnoty daného kontextu. Hráč může na této obrazovce kontext upravit a když je s úpravami hotov, klikne na ikonu pro uložení. Aplikace uloží úpravy daného kontextu a zobrazí hráči detail schopnosti s aktualizovaným kontextem.

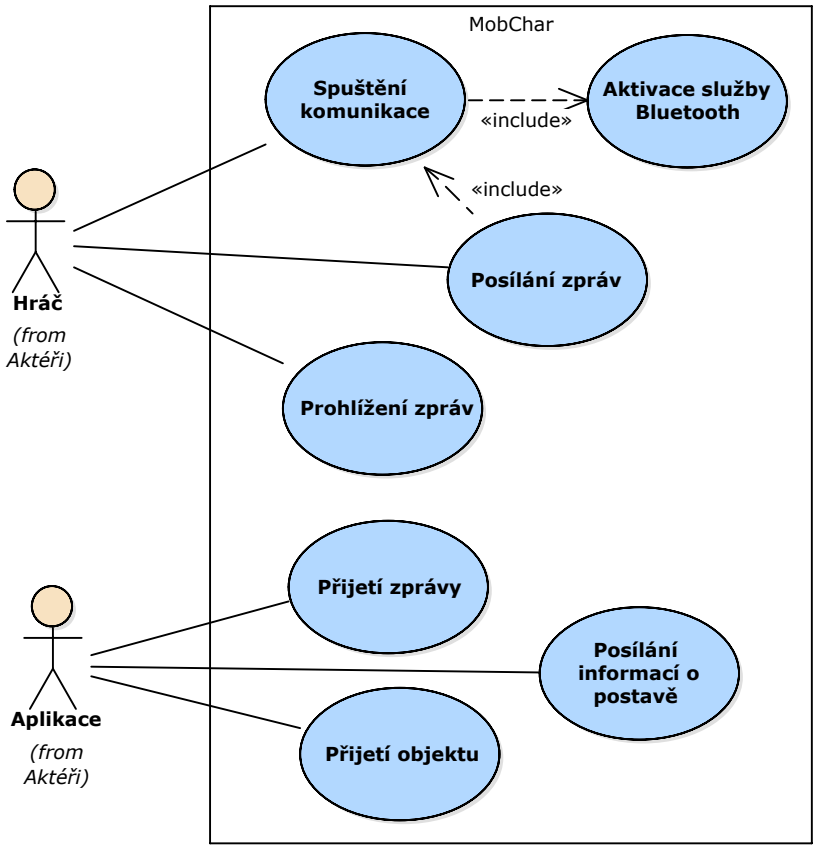
1.4.3 Komunikace s balíčkem pro Pána jeskyně

Požadavek na komunikaci s balíčkem Pána jeskyně se rozpadl na několik případů užití. Je potřeba řešit posílání zpráv, posílání informací o postavě, příjem objektu a příjem zprávy. Jednotlivé případy užití jsou zachyceny na obrázku 1.7.

Spuštění komunikace Před tím, než začne aplikace komunikovat s Pánem jeskyně, je potřeba tuto komunikaci spustit. Hráč tak může učinit kliknutím na ikonu v menu v záložce „Atributy“ nebo „Zprávy“. Po kliknutí nejdříve aplikace zkontroluje, jestli je dostupná služba Bluetooth na zařízení. Pokud služba není dostupná, proces se ukončí a hráči se zobrazí dialog s informací o ukončení. Pokud služba Bluetooth je dostupná, zkontroluje se, jestli je aktivována. Je-li neaktivní, je potřeba provést proces aktivace služby Bluetooth. Je-li služba aktivní, otevře aplikace spojení a bude čekat na navázání komunikace se zařízením Pána jeskyně.

Aktivace služby Bluetooth Je-li potřeba aktivovat službu Bluetooth, zobrazí aplikace dialog s možností spuštění této služby. Klikne-li hráč na tlačítko „OK“, spustí aplikace službu Bluetooth a pokračuje procesem, který spustil tento proces. Nepotvrdí-li hráč aktivaci Bluetooth, ukončí se tento proces i proces nadřazený.

Posílání zpráv Jako součást komunikace s balíčkem Pána jeskyně bude hráči umožněno poslat Pánu jeskyně zprávu. Hráč se v aplikaci přesune do záložky „Zprávy“. Aplikace v této záložce zobrazí dosavadní konverzaci mezi aktuálním hrdinou a Pánem jeskyně. Naspodu této záložky



Obrázek 1.7: Diagram případů užití spojenými s komunikací s balíčkem pro Pána jeskyně

klikne hráč na okénko pro napsání zprávy. Když zprávu napíše, klikne na tlačítko pro její odeslání. Aplikace zkontroluje dostupnost spojení. Pokud spojení není aktivní, musí hráč provést proces spuštění komunikace. Pokud je spojení aktivní, pokusí se aplikace zprávu odeslat na zařízení Pána jeskyně. O úspěchu či neúspěchu odeslání bude aplikace hráče informovat. Pokud proběhne odeslání v pořádku, zobrazí aplikace konverzaci zpráv i s novou zprávou, která bude umístěna nejníže.

Prohlížení zpráv Hráč si může prohlížet všechny doposud napsané zprávy. Tyto zprávy nalezne v záložce „Zprávy“. U každé zprávy je zobrazen její text a datum jejího vzniku. Zprávy jsou uspořádány podle data, nejnovější zpráva je umístěna nejníže. Chce-li hráč vidět starší zprávy, než jsou zobrazené, může skrolovat ve zprávách směrem na horu.

Přijetí zprávy Hráč může také obdržet zprávu od Pána jeskyně. Pokud je

hráč při jejím přijetí v záložce „Zprávy“, zobrazí se přijatá zpráva jako poslední zpráva v konverzaci. Je-li hráč jinde, obdrží systémové upozornění o přijetí nové zprávy.

Posílání informací o postavě Tento případ užití je pouze systémový, děje se bez hráčovy vědomosti. Obdrží-li aplikace požadavek na odeslání postavy, vytvoří z dané postavy soubor formátu XML a tento soubor odešle na zařízení, ze kterého přišel požadavek.

Přijetí objektu Pán jeskyně může hráči poslat nějaký objekt (například předmět, kouzlo...). Tyto objekty jsou mezi zařízeními sdíleny v souborech formátu XML. Při přijetí objektu se aplikace pokusí ze souboru vytvořit daný objekt podle předepsaných struktur. Pokud se to podaří, uloží aplikace nový objekt do deníku postavy. Objekty jsou přidány mezi seznamy příslušného typu (kouzlo mezi kouzla, schopnost mezi schopnosti...), je-li objektem předmět, je uložen do inventáře na zem. O obdržení objektu aplikace informuje hráče pomocí systémové notifikace.

1.4.4 Vrhací zbraně

Případy užití pro vrhací zbraně jsou stejné jako pro ostatní předměty. Podrobný popis těchto případů užití je na přiloženém CD.

Kapitola 2

Návrh

V této části práce bude blíže rozebrán návrh aplikace. Bude zde popsán model architektury balíčku Dračí doupě, změny, které je nutné provést kvůli novým funkcím v databázovém modelu a v návrhovém modelu tříd. Bude popsáno několik modelů komunikace nových funkcí.

2.1 Model architektury

Kvůli nově vznikajícímu balíčku pro Pána jeskyně bylo nutné vyčlenit z balíčku Dračí doupě společné struktury. Tato potřeba byla vyřešena vznikem nové knihovny společné pro balíček Dračí doupě a balíček Pána jeskyně. Bude řešena zvlášť architektura knihovny a balíčku samotného.

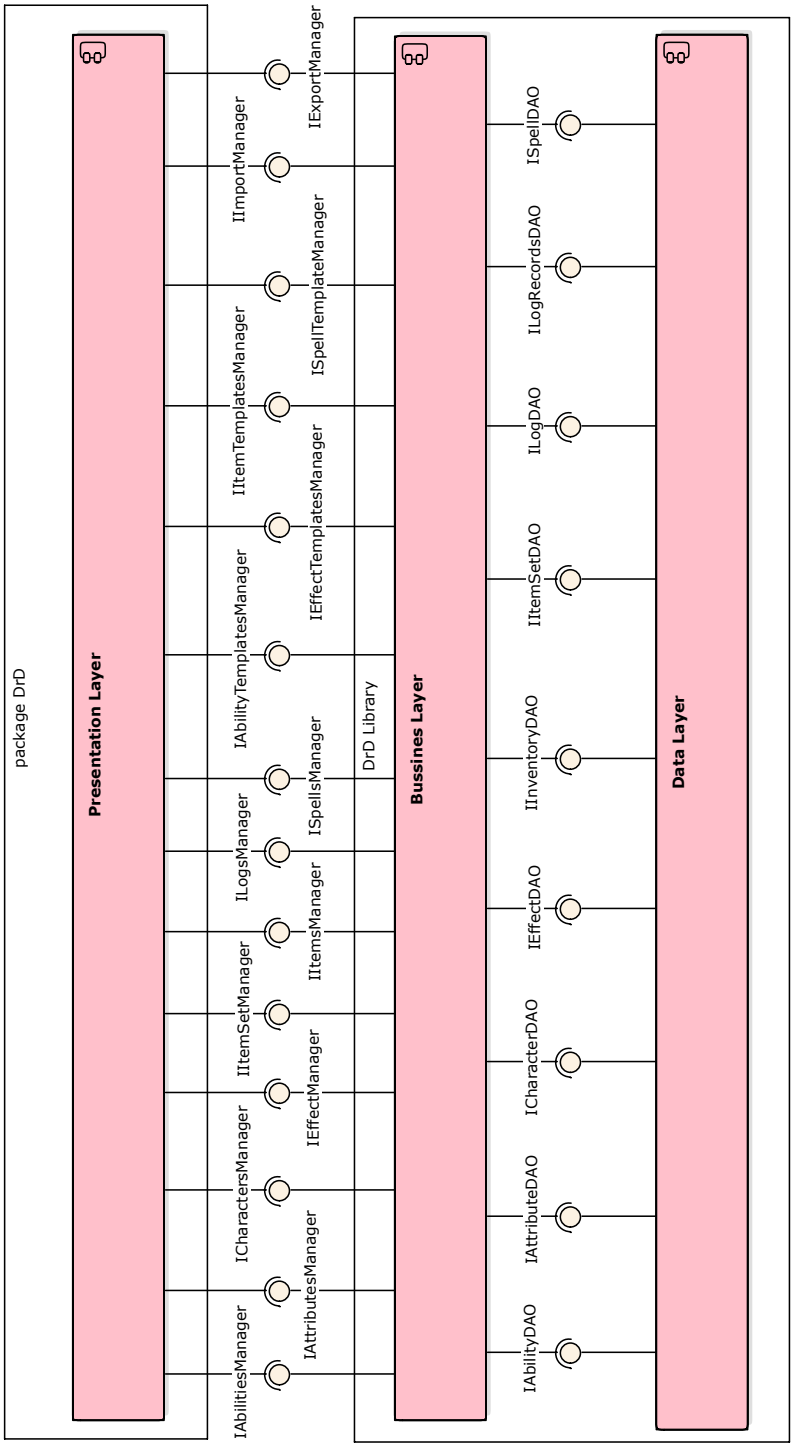
2.1.1 Balíček Dračí doupě

Balíček Dračí doupě je navržen v třívrstvé architektuře. Tato architektura se skládá z vrstvy prezentační, logické a datové[5]. Na logické a datové vrstvě je použit návrhový vzor továrna. Mezi vrstvami je definováno několik rozhraní, kterým továrna přiřazuje konkrétní implementace. Uvnitř okolních vrstev se poté pracuje pouze s rozhraními a vnitřní implementace vrstvy tak může být jednoduše změněna bez ohledu na další vrstvy[6]. Jednotlivá rozhraní jsou zobrazena na diagramu 2.1.

V závislosti na vzniku nového balíčku pro Pána jeskyně byla logická a datová vrstva vyčleněna do knihovny pro Dračí doupě.

Prezentační vrstva Prezentační vrstva je tvořena především fragmenty a aktivitami. Slouží ke komunikaci mezi uživatelem a aplikací. Je v ní obsaženo minimum logiky a práce s daty.

Logická vrstva Logická vrstva má za úkol zprostředkování dat mezi datovou a prezentační vrstvou. Jsou v ní implementovány veškeré logické operace s objekty. Jsou v ní také zahrnuty konstanty.



Obrázek 2.1: Model architektury balíčku Dračí doupě

Datová vrstva Datová vrstva má za úkol komunikaci s databází. Ukládá data z definovaných struktur objektů do databáze a naopak. Pomocí těchto definovaných struktur se přenáší data mezi logickou a datovou vrstvou.

2.1.2 Knihovna Dračí doupě

V nově vzniklé knihovně pro Dračí doupě jsou obsaženy společné struktury pro objekty používané v balíčcích a kompletní logická a datová vrstva původního balíčku Dračí doupě.

2.1.3 Jádro a knihovna

Jádro je navrhnuto jako třívrstvá aplikace stejně jako balíček Dračí doupě. Knihovna není nijak členěna, obsahuje pouze jednotlivé předpisy pro třídy. Podrobnější popis architektury naleznete v příloze a případně v bakalářské práci Jakuba Nižaradze[3].

2.2 Relační datový model

V balíčku Dračí doupě je používána databáze typu SQLite. V této části bude rozebrána pouze změna nutná pro implementaci odvoditelných atributů pro schopnosti a pro ukládání konverzace mezi postavou a Pánem jeskyně. Pro splnění ostatních požadavků není potřeba datový model upravovat. Relační datový model celého balíčku je na přiloženém CD.

2.2.1 Odvozování hodnot pro schopnosti

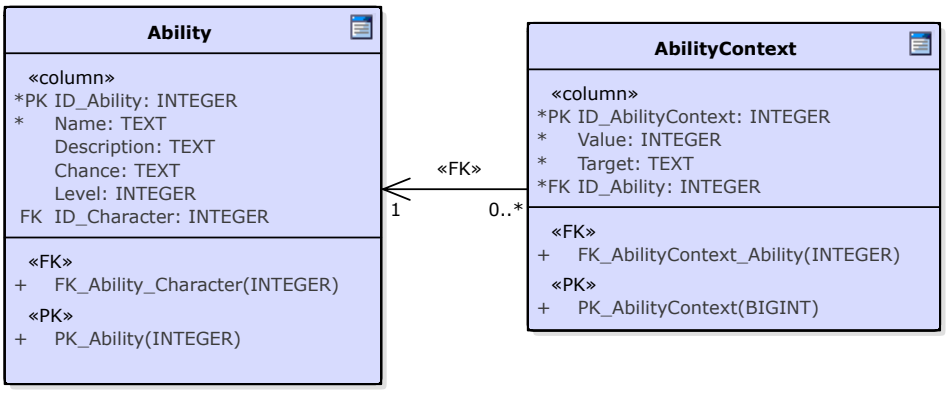
Pro implementaci odvoditelných hodnot pro schopnosti je třeba do databáze přidat novou tabulku pro kontexty schopností (obrázek 2.2). Tato tabulka bude obsahovat identifikační číslo kontextu, které bude použito jako primární klíč. Dále pak bude obsahovat hodnotu, typ úpravy a atribut, kterým je cílová schopnost upravována. Kvůli spárování s tabulkou schopností bude obsahovat jako cizí klíč identifikační číslo schopnosti, ke které patří.

V tabulce schopností je nutné přidat atribut pro úroveň schopnosti.

Vztah mezi tabulkami bude takový, že každá schopnost může mít libovolný počet kontextů, včetně možnosti, že nebude mít žádný, a každému kontextu musí být přiřazena právě jedna schopnost.

2.2.2 Komunikace s balíčkem pro Pána jeskyně

U komunikace s Pánem jeskyně bude potřeba ukládat celou konverzaci mezi postavou a Pánem jeskyně. Je potřeba vytvořit novou tabulku v databázi, která bude představovat zprávy. U každé zprávy je nutné evidovat její text,



Obrázek 2.2: Relační datový model pro schopnosti

datum jejího vzniku a zda je zpráva vytvořena postavou, nebo Pánem jeskyně. Zprávy musí být přiřazeny konkrétní postavě v balíčku a proto bude tabulka obsahovat identifikační číslo postavy, ke které patří.

2.3 Návrhový model tříd

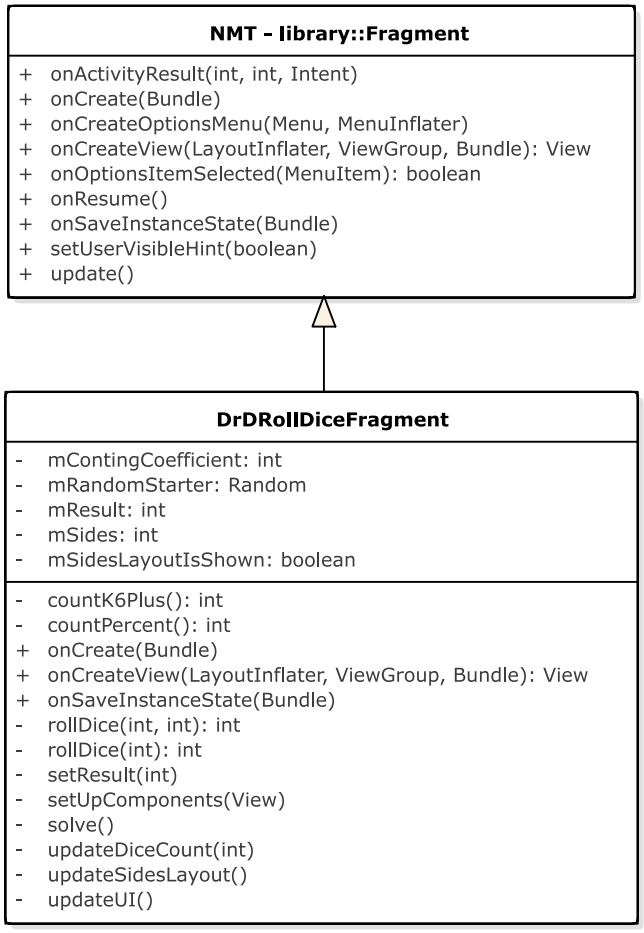
V této části budou popsány změny, které je potřeba provést v návrhovém modelu tříd původní verze balíčku Dračí doupě, za účelem splnění požadavků na nové funkce. Kompletní návrhový model celého balíčku je na přiloženém CD.

2.3.1 Hod kostkou

Hod kostkou bude probíhat pouze na prezentační vrstvě. Logika této funkcionality je tak malá, že je zbytečné ji extrahovat do logické vrstvy. Funkcionalita je tedy implementována jako aktivita hostující fragment.

DrDRollDiceActivity Třída představující aktivitu pro hod kostkou je potomkem třídy `McSingleFragmentActivity`, která je součástí knihovny aplikace. Tato třída je předpisem pro aktivity, které hostí fragment. `DrDRollDiceActivity` bude implementovat pouze poděděné metody. Těmito metodami jsou `createFragment` a `onCreateInit`.

DrDRollDiceFragment Celou logiku simulace hodu kostkou obsahuje třída `DrDRollDiceFragment`. Její struktura je znázorněna na obrázku 2.3. Tento fragment dědí od třídy Fragment, která je součástí knihovny aplikace. `DrDRollDiceFragment` umožňuje generování náhodných čísel z daného rozmezí a jejich kombinování.



Obrázek 2.3: Návrh fragmentu pro hod kostkou

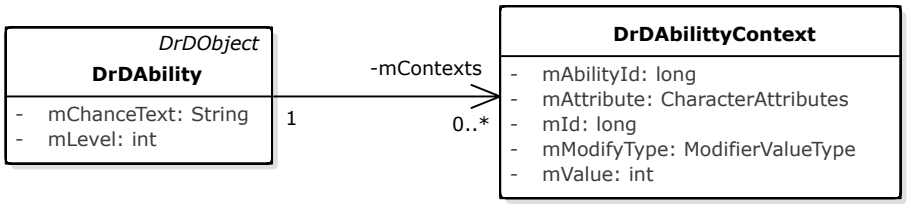
2.3.2 Odvozování hodnot pro schopnosti

Pro tuto funkcionalitu je nutné vytvořit třídy ve všech vrstvách balíčku Dračí doupě i v knihovně pro Dračí doupě. Tyto třídy zajišťují možnost vytvoření, smazání a editaci kontextů schopností a započítání jejich efektu do hodnoty šance schopnosti.

2.3.2.1 Struktury objektů

Do knihovny pro Dračí doupě je potřeba přidat strukturu kontextu schopnosti a upravit strukturu schopnosti samotné. Vztah mezi třídou reprezentující schopnost a jejím kontextem je znázorněn na obrázku 2.4.

Také se musí do knihovny přidat třída pro předpis XML formátu a šablony kontextu a upravit třídu pro předpis XML formátu a šablony schopnosti. Tyto třídy budou mít atributy shodné s ekvivalentními třídami obsahující strukturu



Obrázek 2.4: Návrh struktury pro schopnost a její kontexty

daných objektů (pouze u šablon nebude identifikační číslo). Navíc budou u tříd funkce pro převod mezi jednotlivými formami. Diagram těchto tříd je součástí kompletní dokumentace na přiloženém CD.

DrDAbility Třída `DrDAbility` reprezentuje strukturu pro schopnost. Od třídy `DrDObject` tato třída dědí atributy jméno, popis a identifikační číslo. Navíc má třída vlastní atribut představující šanci na úspěšné použití schopnosti a nově i úroveň schopnosti. Dále třída uchovává seznam všech kontextů přiřazených dané schopnosti.

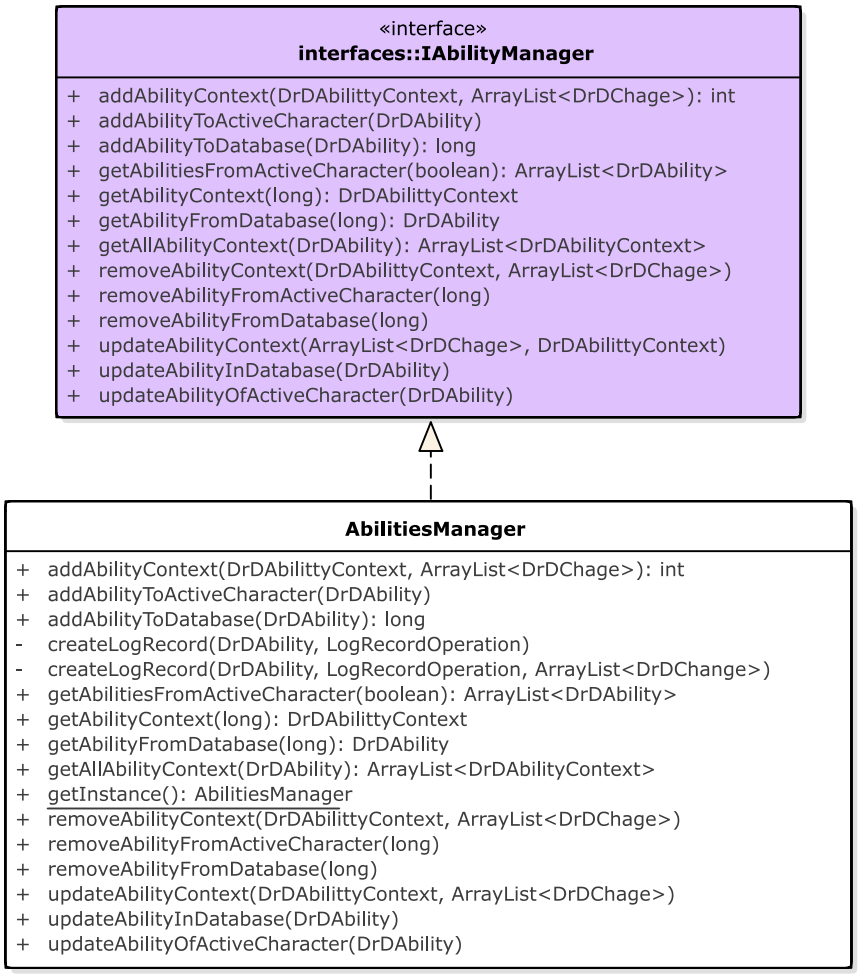
DrDAbilityContext Tato třída reprezentuje jeden kontext. Každý kontext má přiřazen vlastní identifikátor a identifikátor schopnosti, které náleží. Dále obsahuje atribut pro hodnotu, typ úpravy a atribut, kterým je šance schopnosti ovlivněna.

2.3.2.2 Prezentační vrstva

V prezentační vrstvě je třeba vytvořit nový fragment pro vytváření a úpravu kontextu schopnosti. Bude potřeba upravit fragment pro detail schopnosti, aby odpovídal aktuálním potřebám, a do fragmentu pro editaci bude potřeba přidat pole pro zadání úrovně schopnosti.

DrDAbilityContextModifyFragment Pro vytvoření nového kontextu a úpravu již existujícího kontextu bude využíván stejný fragment. Tento fragment bude umožňovat zadání všech potřebných informací o kontextu a uložení těchto hodnot. Rozdíl ve vytváření a editaci kontextu bude pouze v hodnotách předvyplněných pro uživatele. Při vytváření budou předvyplněny základní hodnoty a při editaci budou předvyplněny původní hodnoty upravovaného kontextu.

DrDAbilityDetailFragment Fragment pro detail schopnosti bude potřeba upravit tak, aby zobrazoval seznam jemu přiřazených kontextů a nový atribut pro úroveň. Atribut pro úroveň bude nepovinný a bude v detailu schopnosti zobrazen jen pokud jeho hodnota bude kladná. Na každý kontext ze seznamu bude možné kliknout krátce pro jeho editaci, nebo dlouze pro jeho smazání.



Obrázek 2.5: Návrh manageru schopnosti

2.3.2.3 Logická vrstva

V logické vrstvě bude třeba přidat funkce do rozhraní `IAbilityManager` (obrázek 2.5) a vytvořit novou třídu `DrDAbilityBO`.

IAbilityManager V rozhraní `IAbilityManager` bude potřeba implementovat metody pro vytvoření nového kontextu, upravení již existujícího kontextu, smazání kontextu a získání kontextu s daným identifikátorem z databáze, nebo získání všech kontextů náležících dané schopnosti. V závislosti na změně rozhraní se bude muset změnit i třída `AbilityManager`, která toto rozhraní implementuje.

DrDAbilityBO Třída `DrDAbilityBO` slouží jako objekt, který uchovává danou schopnost a umí dopočítat aktuální hodnotu šance pro úspěšné po-

užití schopnosti z kontextů přiřazeným ke schopnosti a základní šance.

2.3.2.4 Datová vrstva

V datové vrstvě se musí upravit rozhraní `IAbilityDAO` pro práci se schopnostmi, stejně jako `IAbilityManager` v logické vrstvě, tak, aby bylo schopné provádět základní operace s kontexty schopností. Stejně tak se bude muset upravit i třída `AbilityDAO`, která toto rozhraní implementuje.

2.3.3 Vrhací zbraně

K vytvoření vrhacích zbraní je potřeba upravit jen prezentační vrstvu a vytvořit strukturu v knihovně pro Dračí doupě. Logická a datová vrstva přijímá všechny typy zbraní pomocí abstraktní třídy reprezentující zbraň obecně. V těchto vrstvách dojde tedy jen k implementačním změnám.

2.3.3.1 Struktury objektů

Protože vrhací zbraň kombinuje vlastnosti zbraně na dálku a zbraně na blízko, bylo navrženo vytvoření dvou rozhraní. Jedno rozhraní bude definovat funkce, které musí implementovat zbraň použitelná na blízko, a druhé funkce pro zbraň na dálku. Zároveň budou všechny typy zbraní dědit od abstraktní třídy `DrDWeapon`. Závislosti mezi těmito objekty jsou zobrazeny na obrázeku 2.6.

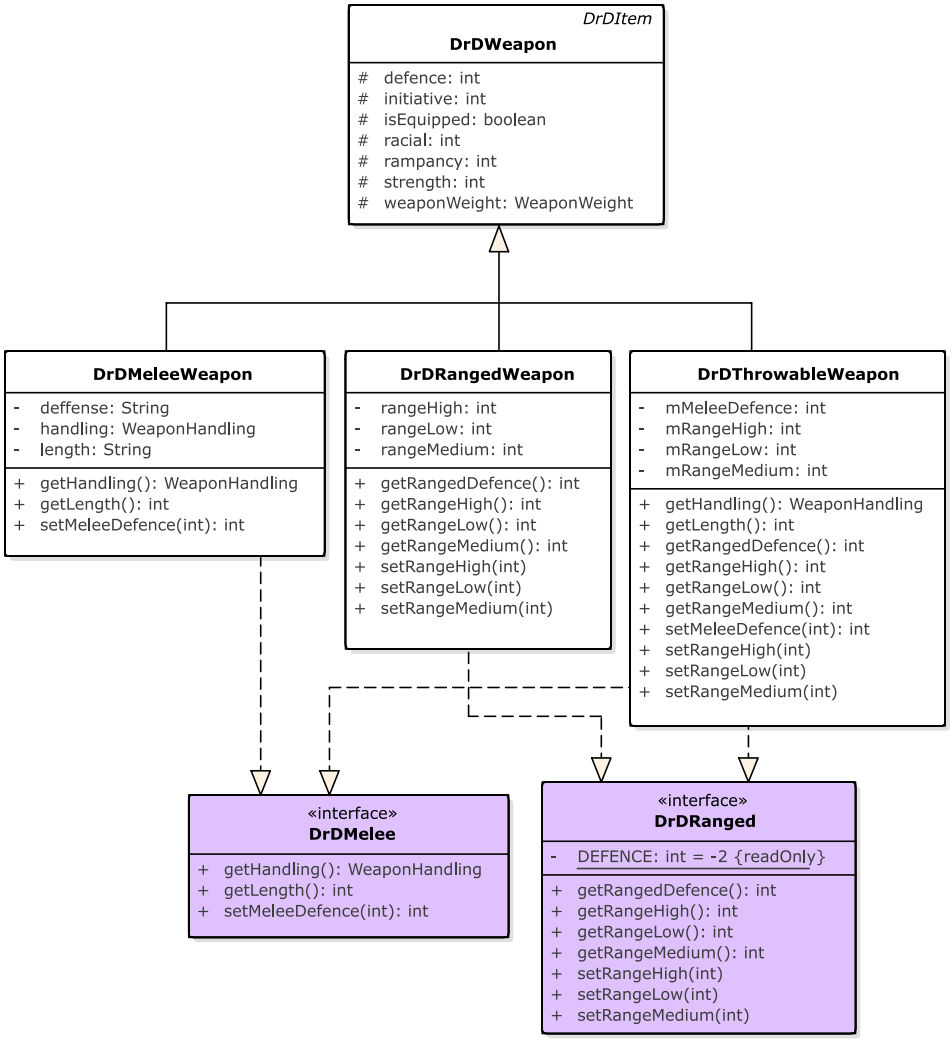
DrDMelee Třída `DrDMelee` je rozhraní předepisující funkce pro zbraně, které mohou být použité na blízko. Tyto zbraně musejí být schopny definovat své držení a svoji délku.

DrDRanged Třída `DrDRanged` je rozhraní předepisující funkce pro zbraně, které mohou být použité na dálku. Tyto zbraně musejí být schopny definovat svou obranu na blízko, svůj dosah na blízko, na dálku a střední dosah. Pro všechny zbraně implementující toto rozhraní je také společná konstanta pro obranu na blízko.

DrDMeleeWeapon Tato třída reprezentuje zbraň na blízko. Implementuje rozhraní `DrDMelee` a dědí od třídy `DrDWeapon` společné atributy pro všechny zbraně. Má nastavitelné hodnoty pro délku zbraně, obranu a pro její držení.

DrDRangedWeapon Tato třída reprezentuje zbraň na dálku. Implementuje rozhraní `DrDRanged` a dědí od třídy `DrDWeapon` společné atributy pro všechny zbraně. Má nastavitelné hodnoty pro nízký, střední a vysoký dosah zbraně.

DrDThrowableWeapon Tato třída reprezentuje vrhací zbraň. Implementuje rozhraní `DrDMelee` i `DrDRanged` a dědí od třídy `DrDWeapon` společné



Obrázek 2.6: Návrh struktury zbraní

atributy pro všechny zbraně. Má nastavitelné hodnoty pro nízký, střední a vysoký dosah zbraně a pro obranu na blízko. Délku zbraně má nastavenou vždy na 1 a je vždy jednoruční.

2.3.3.2 Prezentační vrstva

V prezentační vrstvě je potřeba vytvořit nové třídy, aby bylo umožněno uživateli vrhací zbraň vytvořit, editovat a získávat o ní informace. Proto vzniknou dva nové fragmenty.

DrDThrowableWeaponDetailFragment Tento fragment zobrazuje informace o již existující vrhací zbrani. Fragment je potomkem abstraktní třídy `DrDWeaponDetailFragment`, která předdefinovává funkce společné pro všechny zbraně. Obrazovka fragmentu obsahuje všechny atributy, které je možné u vrhací zbraně definovat.

DrDThrowableWeaponModifyFragment Tento fragment slouží k vytvoření nové, nebo k editaci již existující vrhací zbraně. Umožňuje hráči zadat všechny požadované atributy, které je možně u vrhací zbraně nastavit. Podle toho, zda je fragment spuštěn za účelem vytvoření, nebo editace vrhací zbraně, jsou předvyplněny buď základní hodnoty, nebo původní hodnoty editované zbraně.

2.3.4 Efekty na postavě modifikující objekty

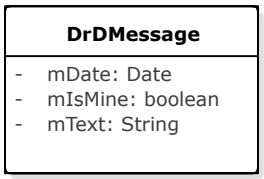
V případě tohoto požadavku nebude nutné provádět změny v návrhovém modelu tříd. Bude se měnit pouze implementace metod některých tříd. Změna nastane ve fragmentu `DrDModifierModifyFragment`, kde je potřeba uživateli umožnit výběr cíle modifikátoru. Další změna bude ve třídě `EffectDAO`, ve které přibudou dvě metody. Jedna bude umožňovat získání všech číselných modifikátorů efektů postavy pro daný typ zbraně a druhá bude fungovat ekvivalentně pro nečíselné modifikátory, jako je například držení zbraně. Tyto modifikátory se poté budou započítávat do aktuálních hodnot předmětů v příslušných business objektech.

2.3.5 Evidence zpráv mezi postavou a Pánem jeskyně

Z požadavku na komunikaci s balíčkem pro Pána jeskyně vyplynula nutnost vytvoření a ukládání komunikace v podobě zpráv mezi nimi. Evidence zpráv se jako jediná z částí nutnou pro splnění celého požadavku promítne do všech vrstev aplikace.

2.3.5.1 Struktury objektů

V knihovně pro Dračí doupě je nutné vytvořit novou strukturu představující jednu zprávu (obrázek 2.7). Tato struktura bude obsahovat identifikační číslo



Obrázek 2.7: Návrh struktury zprávy

zprávy, její text, datum vzniku a informaci o tom, kým byla vytvořena (jestli Pánem jeskyně, nebo hráčem).

2.3.5.2 Prezentační vrstva

Uživateli je nutné umožnit napsat novou zprávu a prohlédnout si historii všech zpráv, které byly danou postavou napsány nebo byly obdrženy od Pána jeskyně. Bylo tedy navrženo vytvořit v aplikaci novou záložku, která bude hráči tyto funkce umožňovat.

K vytvoření záložky stačí přidat pouze jeden nový fragment. Tímto fragmentem bude `DrDMessageFragment` (obrázek 2.8) a bude muset obsahovat třídu pro zachycení broadcastu vytvořeného ve chvíli přijetí nové zprávy. Tato třída bude muset zajistit, aby se nevytvářela systémová notifikace, ale pouze se na obrazovce zobrazila přijatá zpráva.

2.3.5.3 Logická vrstva

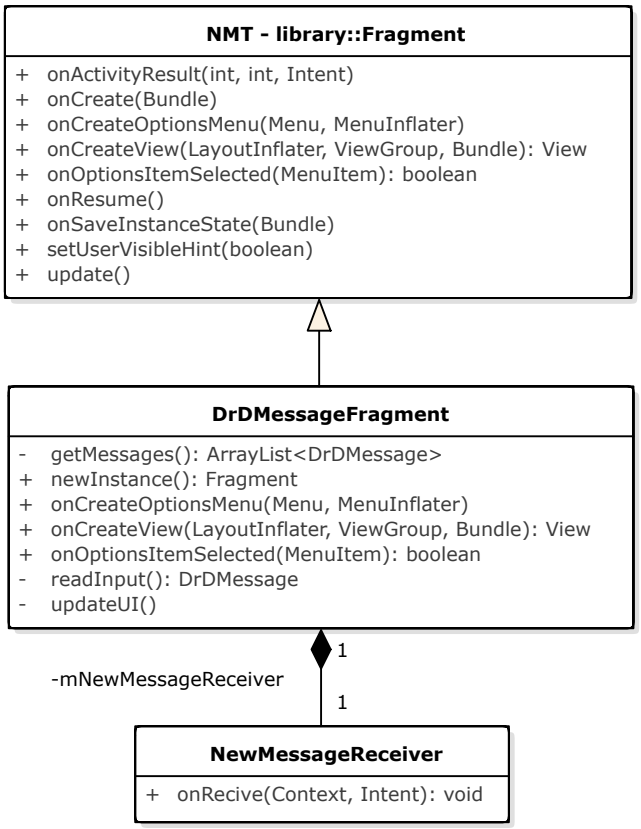
V logické vrstvě je potřeba vytvořit nové rozhraní `IMessageManager` a třídu `MessageManager`, která toto rozhraní bude implementovat. Oba tyto objekty můžete vidět na obrázku 2.9. Rozhraní bude obsahovat metody pro vytvoření nové zprávy u dané či aktivní postavy, získání jedné zprávy podle jejího identifikačního čísla a získání všech zpráv od dané nebo aktivní postavy.

2.3.5.4 Datová vrstva

Do datové vrstvy se musí přidat stejně jako u logické vrstvy nové rozhraní pro práci se zprávami a třída implementující toto rozhraní. Rozhraní datové vrstvy bude implementovat stejné metody jako logická vrstva, kromě možnosti přiřazení zprávy k aktivní postavě a získání všech zpráv od aktivní postavy. Pro tyto dvě operace bude datová vrstvy vždy vyžadovat identifikátor postavy.

2.3.6 Komunikace s balíčkem pro Pána jeskyně

Kromě evidence zpráv je nutné zajistit komunikaci ze strany balíčku Dračí doupě přes službu Bluetooth s balíčkem pro Pána jeskyně. Také je potřeba

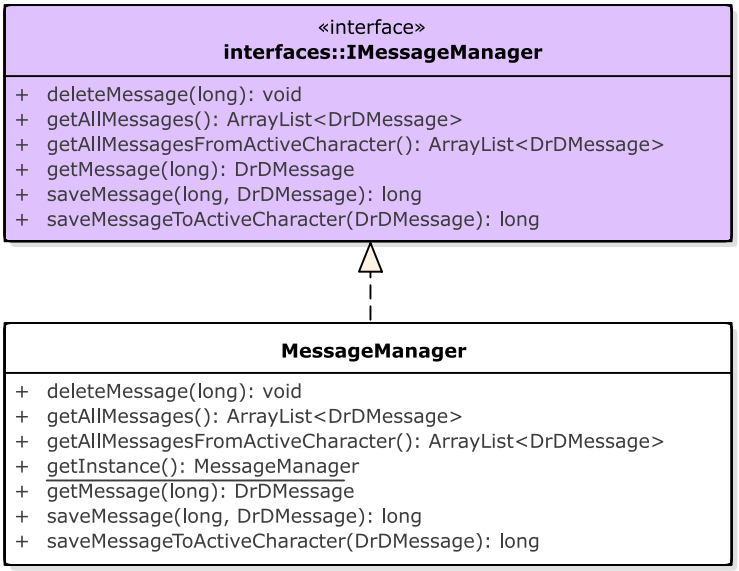


Obrázek 2.8: Návrh fragmentu pro zprávy

vytvořit notifikaci při přijetí nové zprávy. Obě funkcionality budou implementovány pouze v prezentační vrstvě. Třídy, které je nutné vytvořit, jsou zobrazeny na diagramu 2.10.

2.3.6.1 Bluetooth

O spojení pomocí služby Bluetooth se bude starat třída `BluetoothService`. Bude možné vytvořit pouze jednu instanci této třídy. Tato instance se bude starat o spojení mezi jednou postavou na zařízení uživatele a zařízením Pána jeskyně. Třída bude umožňovat vytvoření spojení, posílání zpráv, přijímání různých objektů a ukončení spojení. Pro tyto funkcionality bude nutné vytvořit ve třídě `BluetoothService` třídy `AcceptThread` a `ConnectedThread`. Tyto dvě třídy budou vytvořeny na podle dokumentace Android [7].



Obrázek 2.9: Návrh manageru pro zprávy

2.3.6.2 Notifikace

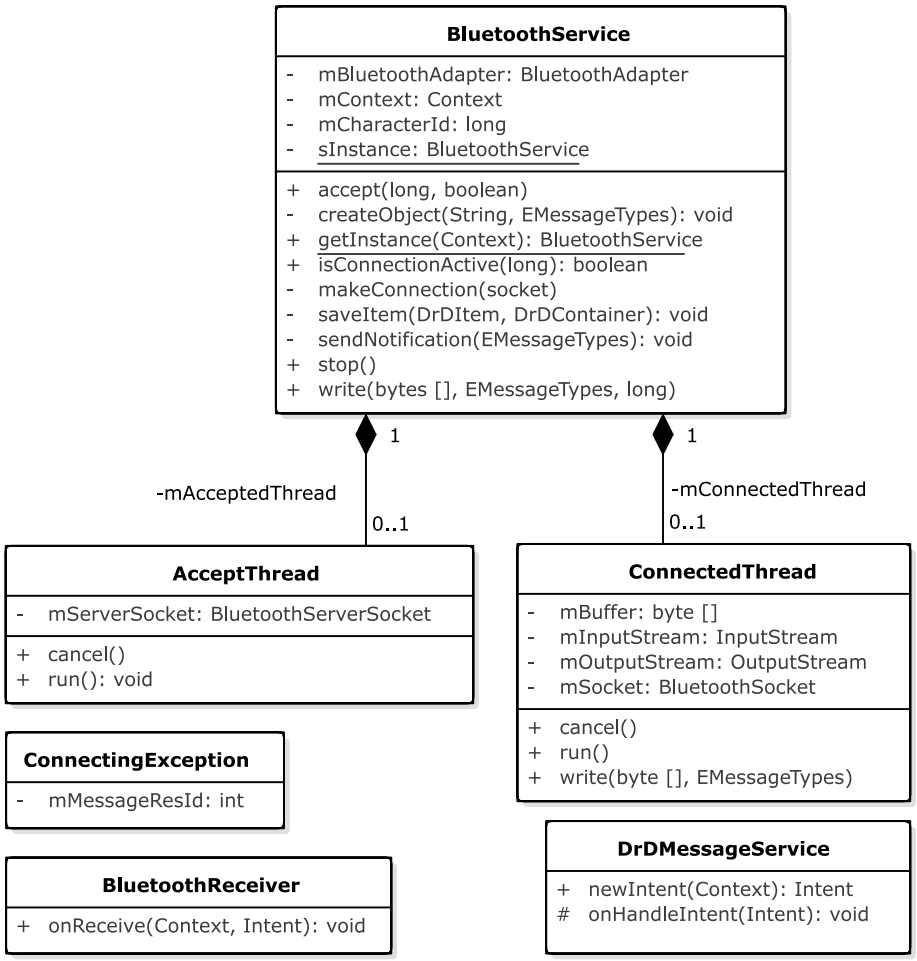
Jak je uvedeno v požadavku na komunikaci, při přijetí nového objektu od Pána jeskyně bude vytvořena systémová notifikace. Tato notifikace se bude zobrazovat pouze pokud nebude na zařízení aktivní obrazovka, na které by se měl daný objekt zobrazit. Pro vytvoření této notifikace bylo navrženo vytvoření třídy `DrDMessageService`. Tato třída bude dědit od třídy `IntentService` a při jejím vytvoření bude také vytvořena notifikace.

Systémová notifikace vytvořená balíčkem Dračí doupě nebude umožňovat spuštění aplikace při kliknutí na ní. Je to z toho důvodu, že balíček Dračí doupě sám o sobě není spustitelnou aplikací. Pro spuštění balíčku musí být nejdříve spuštěno jádro aplikace.

Pro propagaci informace o přijetí zprávy budou použity broadcasty. Pro zachycení tohoto broadcastu bude vytvořena třída `BluetoothReceiver`, která v případě přijetí nové zprávy zajistí vytvoření notifikace vytvořením třídy `DrDMessageService`. Různé způsoby využití broadcastů, včetně tohoto, jsou obsaženy v knize Android Programming: The Big Nerd Ranch Guide[8].

2.4 Model komunikace

V této části bude popsána komunikace mezi navrhovanými třídami. Pro ilustraci komunikace byly vybrány dvě situace, které mohou nastat při komunikaci s balíčkem pro Pána jeskyně. Několik dalších diagramů komunikace je na přiloženém CD, tyto diagramy nebudou v bakalářské práci více rozepsány.

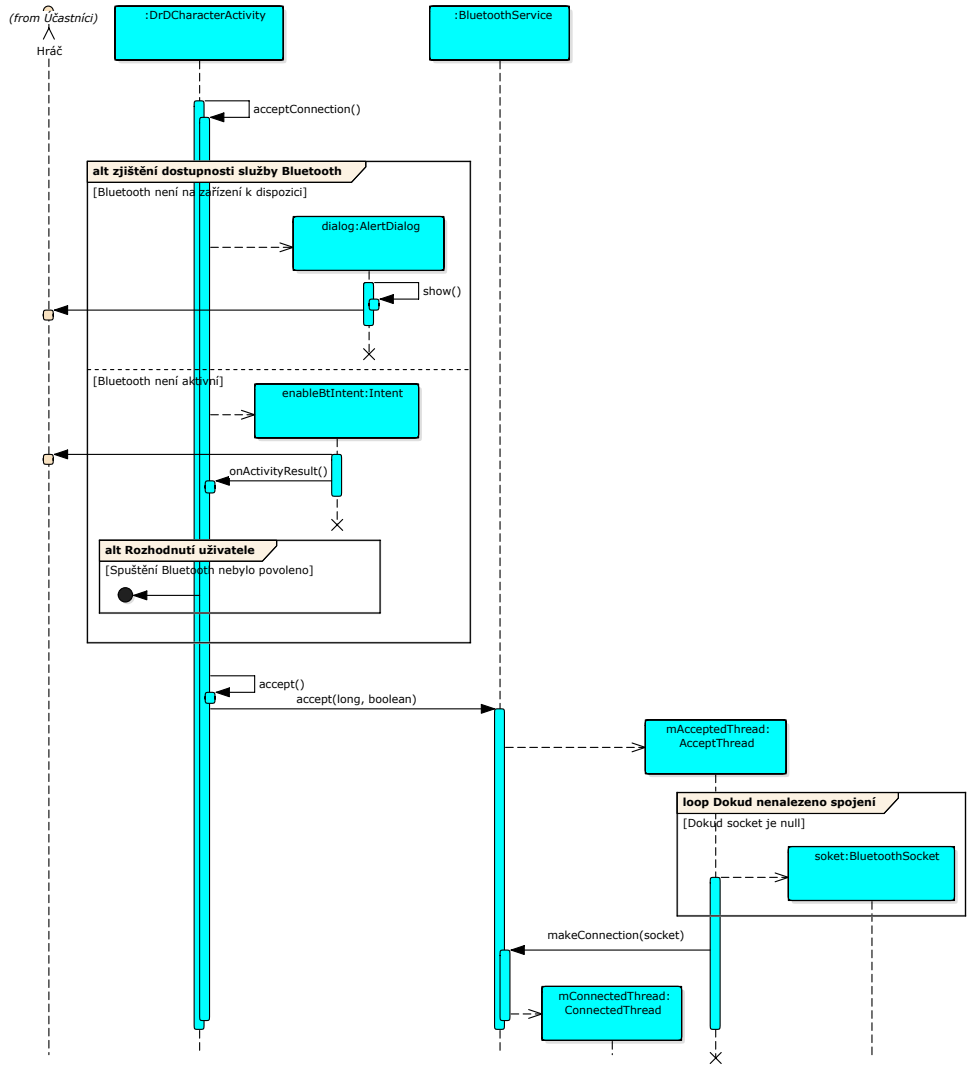


Obrázek 2.10: Návrh komunikace pomocí Bluetooth

2.4.1 Vytvoření spojení s balíčkem Pána jeskyně

První z diagramů zachycuje komunikaci mezi třídami potřebnou pro vytvoření spojení. Jednotlivé kroky jsou zobrazeny na diagramu 2.11.

Tato situace je vyvolána kliknutím uživatele na ikonu v menu hlavní aktivity pro aktivaci spojení. Tuto akci zaznamená třída `DrDCharacterActivity` a zavolá svou metodu `acceptConnection`. V této metodě se provede kontrola služby Bluetooth. Kontrola služby může dopadnout třemi způsoby. První možností je, že na zařízení služba Bluetooth není nalezena, v takovémto případě je uživateli zobrazen dialog s informací o chybě a celý proces je ukončen jako neúspěšný. Druhou možností je, že služba Bluetooth je dostupná, ale není aktivní. V tom případě aplikace zobrazí dialog, který umožní službu aktivovat, či proces ukončit. Aktivuje-li uživatel službu Bluetooth, je volána metoda `accept` třídy `DrDCharacterActivity`. Tato metoda je volána i v případě, že



Obrázek 2.11: Diagram zobrazující průchod programem při vytváření spojení

služba Bluetooth již byla aktivní před spuštěním tohoto procesu.

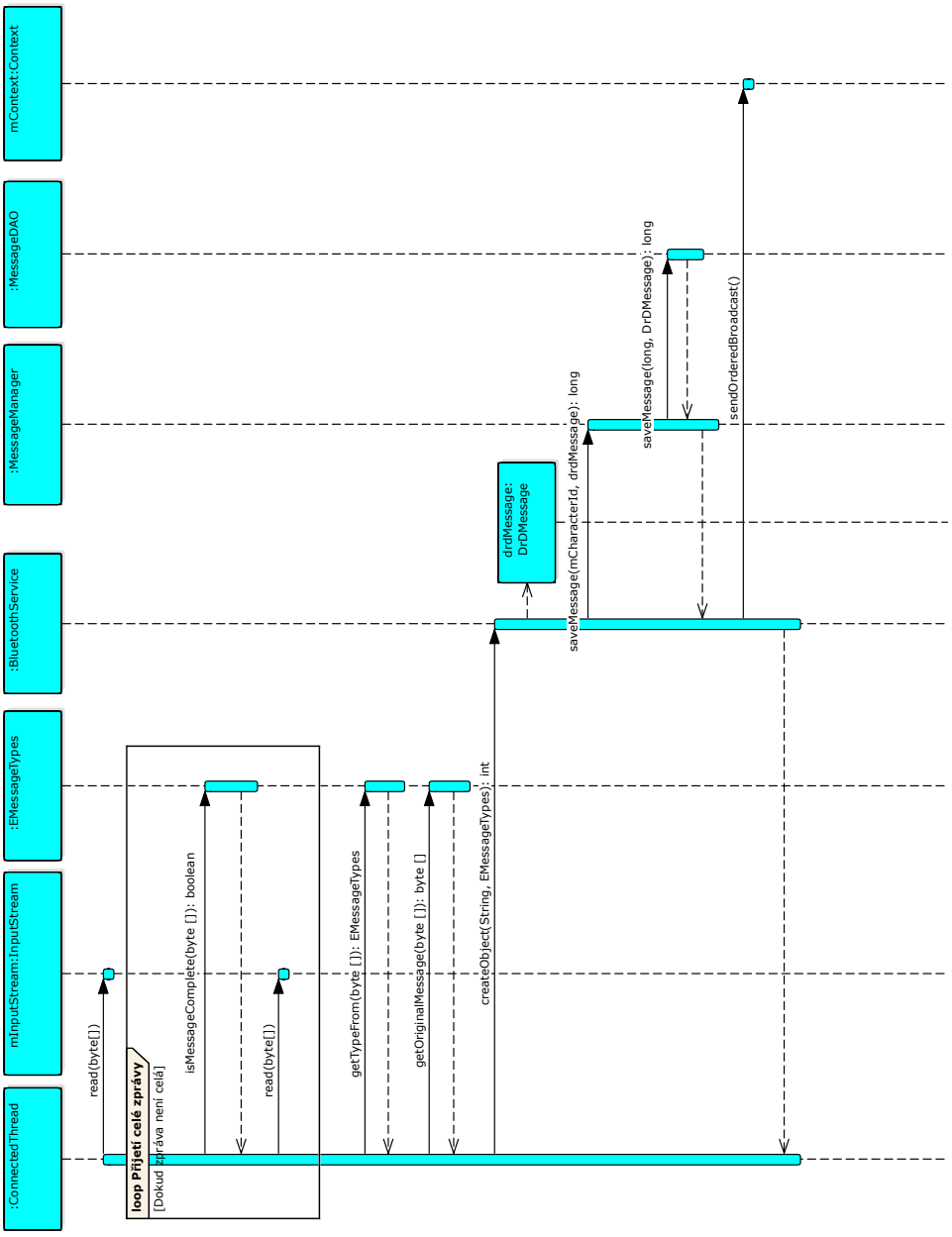
Metoda `accept` volá metodu `accept` třídy `BluetoothManager`. V této metodě je vytvořeno a spuštěno vlákno `AcceptThread`. Úkolem tohoto vlákna je vytvoření serverového socketu a následné čekání na příchozí spojení. Jakmile je navázáno nové spojení, volá se metoda `makeConnection`, ve které je vytvořeno a spuštěno vlákno `ConnectedThread`, které následně udržuje spojení se zařízením Pána jeskyně. Tímto je proces vytváření spojení úspěšně ukončen.

2.4.2 Přijetí zprávy

Druhý diagram komunikace 2.12 zachycuje komunikaci mezi třídami při přijetí nové zprávy od Pána jeskyně. Na přijetí jakékoliv informace od Pána jeskyně čeká vlákno `ConnectedThread` spuštěné na konci předchozího procesu 2.11. Na začátku tohoto procesu se tedy předpokládá, že již existuje aktivní spojení mezi hráčem a Pánem jeskyně.

Dojde-li k přijetí nových dat, je potřeba nejprve zjistit, zda jsou data kompletní. K tomu slouží statická metoda třídy `EMessageTypes`. Pokud tato metoda vrátí `false`, značí to, že data nejsou kompletní a pokračuje se v cyklu opět čtením dat od Pána jeskyně a následnou kontrolou kompletnosti dat. Jakmile jsou data kompletní, cyklus se ukončuje. Poté jsou volány statické metody `getTypeFrom` a `getOriginalMessage` třídy `EMessageTypes` k určení typu zprávy a obsahu zprávy bez hlavičky a ukončovacího znaku.

Jsou-li data přijata a potřebné informace získány, jsou poslány do metody `createObject` třídy `BluetoothService`. V této metodě se provede vytvoření objektu podle zaslaného typu. Diagram 2.12 zachycuje pouze případ, kdy je přijatým objektem zpráva. V takovém případě je vytvořen nový objekt `DrDMessage` a ten je uložen skrze logickou a datovou vrstvu do databáze. Po uložení je navíc, na rozdíl od ostatních objektů, ještě vyslán broadcast informující o přijetí nové zprávy.



Obrázek 2.12: Diagram zobrazující průchod programem při přijetí zprávy

Kapitola 3

Realizace

Tato část se bude zabývat implementační částí bakalářské práce. Budou zde popsány použité nástroje a nějaké části implementace. Dále zde bude rozebráno testování aplikace a uživatelská příručka.

3.1 Nástroje

Jako první budou vyjmenovány nástroje, které byly použity za účelem vypracování této bakalářské práce. Jedná se o verzovací systémy, vývojové prostředí a další. U každého nástroje bude stručný popis.

Texmaker 4.5 Tento text bakalářské práce je napsán v jazyce LaTeX. Byl využit editor Texmaker. Tento editor umožňuje překlad z jazyka LaTeX do formy PDF a následné zobrazení vygenerovaného souboru.

Enterprise architect (verze 12) Program Enterprise architect není nástrojem použitým pro implementaci, ale pro návrh a analýzu. V tomto programu byly vytvořeny všechny grafy od business procesů a případů užití, přes doménový model, relační datový model až po návrhový model tříd. Všechny tyto diagramy jsou vytvořeny v jazyce UML.

Android studio 2.3.1 Android studio je vývojové prostředí pro vývoj aplikací pro operační systém Android. Poskytuje mnoho možností práce s vyvíjenou aplikací, jako je například napojení na verzovací systém nebo spuštění aplikace v emulátoru. Emulátor je virtuální zařízení, na kterém je spuštěn operační systém Android a je tak možné na něm spouštět a testovat vyvíjené aplikace.

Verzovací systémy Při práci na této bakalářské práci byly používány dva verzovací systémy. Je to Git a SVN. Git je využíván pro verzování všech kódů a na SVN jsou ukládány verze diagramů analýzy a návrhu.

3.2 Implementace

Jak již bylo dříve řečeno, MobChar je aplikací pro operační systém Android s minimální verzí API 16. Aplikace je podporována na mobilních telefonech a tabletech.

V této části bude několik ukázek kódu, které bylo potřeba implementovat pro splnění požadavků. Bude se jednat o části požadavku na komunikaci s balíčkem pro Pána jeskyně, protože pro splnění ostatních požadavků bylo nutné vytvoření základních aktivit a fragmentů, průchod skrz logickou vrstvu a případně zápis do databáze nebo čtení z ní. Kompletní kódy celé aplikace jsou na přiloženém CD.

3.2.1 Notifikace

Jedním ze zajímavých prvků v implementaci je vytvoření systémové notifikace při přijetí objektu od Pána jeskyně. Celá logika vytváření notifikací včetně dále vysvětlených broadcastů byla vytvořena na základě ukázkové implementace z knížky Android Programming: The Big Nerd Ranch Guide[8]. Pro vytvoření notifikace byla implementována třída `DrDMessageService`. Tato třída je potomkem třídy `IntentService` a při jejím vytvoření se automaticky volá metoda `onHandleIntent` (zdrojový kód 3.1), která vytvoří notifikaci s daným textem. Text notifikace se liší podle typu přijatého objektu. Protože balíček Dračí doupě není sám o sobě spustitelnou aplikací, není notifikaci nastavena žádná aktivita a při kliknutí na notifikaci tedy nebude provedena žádná akce.

Zdrojový kód 3.1: Ukázka implementace metody onHandleIntent

```
public class DrDMessageService extends IntentService {
    ...
    @Override
    protected void onHandleIntent(Intent intent) {
        Resources r = getResources();

        int resId = getResId(intent.getStringExtra(
            ↪ BluetoothService.ACTION_TYPE));

        Notification notification
                = new NotificationCompat.Builder(this)
                .setTicker(r.getString(R.string.app_name))
                .setSmallIcon(R.drawable.drd_icon)
                .setContentTitle(r.getString(R.string.
                    ↪ app_name))
                .setContentText(r.getString(resId))
                .setAutoCancel(true).build();

        NotificationManagerCompat notificationManager =
            ↪ NotificationManagerCompat.from(this);
        notificationManager.notify(0, notification);
```

```
    }
    ...
}
```

Součástí požadavku na notifikaci bylo také to, že pokud se uživatel nachází v aplikaci v místě, kam má být objekt nahrán, nebude notifikace vytvořena a přijatý objekt bude zobrazen v seznamu objektů. K vyřešení tohoto problému byla použita komunikace skrz aplikaci pomocí broadcastů. Při přijetí nového objektu se vyšle broadcast zohledňující prioritu. K zachycení tohoto broadcastu slouží dva přijímače.

Jedním z nich je třída `BluetoothReceiver`, která při přijetí tohoto broadcastu spustí servis `DrDMessageService` a vytvoří tím notifikaci. Tento přijímač má nastavenou prioritu na nejnižší a proto broadcast přijímá jako poslední. Před ním broadcast může přijmout jedna ze tříd `NewMessageReceiver`. Tyto třídy jsou součástí fragmentů, které přijaté objekty zobrazují, díky čemuž jsou aktivní jen tehdy, je-li fragment zobrazován.

Jako příklad je uveden přijímač ve třídě `DrDMessageFragment` (zdrojový kód 3.2), všechny ostatní fungují na stejném principu. Tato třída dědí od třídy `BroadcastReceiver` a implementuje její metodu `onReceive`, která je volána při přijetá broadcastu. V této metodě se kontroluje, jaký broadcast přišel a zda typ objektu odpovídá zprávě. Pokud ano, je nastaven výsledek na zamítnuto a fragment je aktualizován. Tím, že se výsledek nastavil na zamítnuto, se docílí toho, že když broadcast dojde do třídy `BluetoothReceiver`, nebude notifikace vytvořena.

Zdrojový kód 3.2: Ukázka z implementace třídy NewMessageReceiver

```
public class NewMessageReceiver extends BroadcastReceiver {

    @Override
    public void onReceive(Context context, Intent intent) {
        String action = intent.getAction();
        switch (action){
            case BluetoothService.ACTION_OBJECT_ACCEPTED:
                String type = intent.getStringExtra(
                    ↪ BluetoothService.ACTION_TYPE);
                if (type.equals(EMessageTypes.MESSAGE.
                    ↪ toString())) {
                    if (isMenuVisible()) {
                        setResultCode(Activity.
                            ↪ RESULT_CANCELED);
                    }
                    updateUI();
                }
        }
    }
}
```

3.2.2 Práce s Bluetooth

Další částí, která bude v této práci více rozebrána, je implementace vytvoření spojení mezi zařízením hráče a Pána jeskyně pomocí služby Bluetooth. Tato komunikace byla implementována podle dokumentace pro vývoj pro operační systém Android[7]. Veškerou funkčnost ohledně spojení se zařízením Pána jeskyně ovládá třída `BluetoothService`. Tato třída je schopna otevřít port pro navázání spojení, navázat spojení a poté pomocí tohoto spojení komunikovat se zařízením Pána jeskyně.

Zařízení hráče bylo navrženo jako serverová strana. K vytvoření serverového socketu byla podle dokumentace pro operační systém Android[7] vytvořena třída `AcceptThread` (zdrojový kód 3.3), která je implementována jako vlákno. Tato třída je součástí třídy `BluetoothService`. Při vytvoření tohoto vlákna je vytvořen serverový socket s předem definovanými parametry a při jeho spuštění vlákno čeká na příchozí spojení.

Zdrojový kód 3.3: Ukázka z implementace třídy AcceptThread

```
private class AcceptThread extends Thread {
    private final BluetoothServerSocket mServerSocket;

    public AcceptThread() {...}
    public void run() {
        BluetoothSocket socket;
        while (true) {
            try {
                socket = mServerSocket.accept();
            } catch (IOException e) {...}
            if (socket != null) {
                makeConnection(socket);
                mContext.sendBroadcast(new Intent(
                    ↪ ACTION_CONNECTION_CREATED));
                try {
                    mServerSocket.close();
                } catch (IOException e) {...}
                break;
            }
        }
    }
    public void cancel() {...}
}
```

Je-li spojení navázáno, je vytvořeno nové vlákno `ConnectedThread` (zdrojový kód 3.4), které vytvoří vstupní a výstupní stream pro komunikaci. Při spuštění vlákna `ConnectedThread` se provádí opakované čtení ze vstupního streamu. Je-li potřeba odeslat data na zařízení Pána jeskyně, je čtení pozastaveno a data tímto vláknem odeslána.

Zdrojový kód 3.4: Ukázka z implementace třídy ConnectedThread

```
private class ConnectedThread extends Thread {
    ...
    public void run() {
        ...
        while (true) {
            try {
                numBytes = mInputStream.read(mBuffer);
                byte[] messageWhole = new byte[numBytes];
                System.arraycopy(mBuffer, 0, messageWhole, 0,
                    ↪ numBytes);
                while(!EMessageTypes.isMessageComplete(
                    ↪ messageWhole)){
                    ...
                }

                EMessageTypes messageType = EMessageTypes.
                    ↪ getTypeFrom(messageWhole);
                byte[] message = EMessageTypes.
                    ↪ getOriginalMessageFrom(messageWhole);

                createObject(new String(message, Charset.
                    ↪ defaultCharset()), messageType);
            } catch (IOException e) {...}
        }
    }
    public void write(byte[] bytes, EMessageTypes messageType
        ↪ ) {
        try {
            mOutputStream.write(EMessageTypes.prepareMessage(
                ↪ bytes, messageType));
        } catch (IOException e) {...}
    }
    ...
}
```

3.3 Testování

V další části této kapitoly bude popsáno, jakým způsobem byla aplikace testována. Byly testovány pouze nově přidané funkce. Použité testy se dají rozdělit do několika skupin.

3.3.1 Jednotkové testy

Prvním typem testů použitých při testování nových funkcí jsou jednotkové testy. Jedná se o automatizované testy testující každou třídu zvlášť. Třídy, které testovaná třída používá, jsou nahrazeny jinou, jednodušší implementací.

Tento typ testů byl zvolen pouze pro třídu `DrDAbilityBO`. Tato třída, jako jediná z nově implementovaných, obsahuje výpočetní logiku, která se dá testovat nezávisle na ostatních třídách.

DrDAbilityBO Pro testování této třídy bylo potřeba vytvořit novou testovací třídu `AbilityBOTest`. Protože třída `DrDAbilityBO` komunikuje s třídou `DrDCharacterBO`, bylo potřeba implementovat navíc třídu `TestCharacterBO`, která slouží jako náhradní při testování.

Test této třídy je cílen na kontrolu správnosti výpočtu šance schopnosti. Testovací třída obsahuje tři testovací metody. Každá z metod nejdříve volá iniciační metodu, ve které se vytvoří instance třídy `TestCharacterBO`, několik kontextů a jedna schopnost pro testování.

První z metod je `testCountNumericChance`. V té se testuje, zda jsou výpočty pro jednotlivé typy kontextů správné. Testuje se zde také možnost kontextu s `null` hodnotami. Další testovací metodou je `testCountNonNumericChance`. Tato metoda kontroluje chování v situaci, kdy šance schopnosti není číselná. V takovém případě není výsledná šance kontexty nijak ovlivněna. Poslední testovací metodou ve třídě `AbilityBOTest` je `testNoContexts`, která testuje stav, kdy schopnost nemá žádné kontexty.

3.3.2 Integrační testy

Dalším typem implementovaných testů jsou testy integrační. Tyto testy jsou automatizované a kontrolují komunikaci mezi jednotlivými třídami. Tento typ testů byl použit na kontrolu správnosti vytváření, úpravy a mazání kontextů schopností a zpráv. Byly vytvořeny testovací třídy `AbilitiesManagerTest` a `MessageManagerTest`.

AbilitiesManagerTest Tato třída testuje vytváření, úpravu a mazání schopností a jejich kontextů. Testuje komunikaci mezi logickou a datovou vrstvou a práci s databází. Třída obsahuje tři testovací metody. Na začátku každé z nich je volána inicializační metoda, ve které je vytvořena instance třídy `AbilitiesManager` a několik testovacích kontextů a schopností.

První metodou je `testContexts`. Ta testuje operace prováděné s kontexty schopností. Nejdříve kontext v databázi vytvoří, poté kontroluje, zda se vytvořený objekt schoduje s původním, dále kontext upraví a smaže. Další testovací metodou je `testAbilityNoContext`. Ta testuje vytváření, úpravu a smazání schopnosti, které nejsou přiřazeny žádné schopnosti. Poslední testovací metodou této třídy je `testAbilityWithContexts`. Tato metoda nejdříve vytvoří schopnost s několika kontexty a kontroluje, zda byla, včetně kontextů, správně vytvořena. Poté testuje úpravu jednoho z jejích kontextů a poté jeho smazání. Obě tyto operace se musí promítnout na schopnosti získané z databáze. Nakonec se v této metodě kontroluje mazání schopnosti. Schopnost musí být smazána i s jejími kontexty.

MessageManagerTest Tato třída testuje vytváření zpráv a jejich získávání z databáze. Stejně jako třída `AbilitiesManagerTest` testuje komunikaci mezi logickou a datovou vrstvou a práci s databází. Třída obsahuje dvě testovací metody. Na začátku každé z nich je volána inicializační metoda, ve které je vytvořena instance třídy `MessageManager` a několik testovacích zpráv.

První testovací metoda `testOneMessage` testuje vytvoření a získání jedné zprávy. Na konci této metody je zpráva vymazána. Druhou testovací metodou je `testMessages`. Ta testuje vytvoření více zpráv a následné získání zpráv podle identifikátoru postavy. Na konci metody jsou zprávy z databáze taktéž odstraněny.

3.3.3 Uživatelské testy

Dalším typem testů, kterými byla aplikace testována, jsou uživatelské testy. Cílem těchto testů je zaručit funkčnost uživatelského prostředí a celkově chodu aplikace. Tyto testy nejsou automatizované a musejí se tedy provádět ručně. Bylo vytvořeno několik scénářů, které by měly pokrýt testování většiny z nově implementovaných funkcí. Jedná se o uživatelské testy na hod kostkou, správnou funkčnost vrhacích zbraní, propagaci efektů na postavě do správných předmětů, možnost vytvoření, úpravy a smazání kontextu a také zprovoznění a používání komunikace s balíčkem pro Pána jeskyně. Kompletní seznam a kroky jednotlivých scénářů jsou na přiloženém CD, v této části budou blíže popsány jen některé z nich.

Každý uživatelský test obsahuje jednoduchý popis, který shrnuje co je hlavním předmětem testu. Dále je u každého testu uvedeno potřebné testovací prostředí a předpoklady, které musejí být před testem splněny. Hlavní částí všech testů je scénář. Ten se skládá z jednotlivých úkonů testera a očekávané reakce systému na každý úkon. Po každém kroku scénáře je tedy potřeba zkontrolovat, zda aplikace reaguje předpokládaným způsobem. Součástí uživatelských testů na komunikaci s balíčkem pro Pána jeskyně je i práce s tímto balíčkem. Kroky, které je nutné provést v tomto balíčku, nejsou v testech podrobně popsány a jejich výsledek není nutné kontrolovat (je počítáno s jeho funkčností). Testování těchto kroků je součástí bakalářské práce Matěje Sháněla[2].

Aplikace byla uživatelsky testována na několika zařízeních (Samsung, Lenovo, Huawei). Během testování bylo zjištěno několik chyb. Jednou z nich je, že aplikaci nelze nainstalovat na zařízeních od výrobce Huawei. Příčinu tohoto problému se nepodařilo nalézt. Další chybou, která byla při testování nalezena, bylo to, že při vytvoření nového setu nebo efektu nebyl tento objekt přidán do seznamu objektů. K jeho zobrazení bylo potřeba přejít o několik záložek dále, aby došlo k opětovnému vytvoření celého fragmentu. Tato chyba byla opravena a výsledná aplikace by tedy měla být již bez ní. Jiné chyby nebyly během testování nalezeny.

1. Vrhací zbraně	
Popis	Testování použití vrhacích zbraní v souboji jako zbraň na dálku i na blízko.
Výsledek	Vrhací zbraň je v bojových statistikách možné zvolit jako zbraň na blízko i na dálku.
Prostředí	Zařízení Android s minimální verzí API 16, nainstalovanou aplikací MobChar a balíčkem DrD verze 3.0.
Předpoklady	Existující postava z balíčku Dračí doupě.

Krok	Úkon	Očekávaný stav
1	Zapnout aplikaci MobChar. Vybrat postavu z balíčku Dračí doupě.	Zobrazí se obrazovka přehledu postavy.
2	Kliknout na záložku "Inventář".	Zobrazí se záložka "Inventář" obsahující dva kontejnery (zem a inventář).
3	Kliknout na kontejner "Inventář".	Zobrazí se obsah kontejneru "Inventář".
4	Kliknout na ikonu "+" v horním menu.	Zobrazí se dialog se seznamem typů předmětů.
5	Kliknout na "Vrhací zbraň".	Zobrazí se seznam načtených šablon vrhacích zbraní.
6	Kliknout na jednu z šablon.	Zobrazí se obrazovka s možností editace šablony.
7	Kliknout na ikonu "Uložit" v horním menu.	Zobrazí se obsah kontejneru "Inventář" s nově vytvořenou vrhací zbraní.
8	Kliknout na záložku "Vybavení".	Zobrazí se obrazovka s informacemi pro souboj a seznamem setů.
9	Kliknout na ikonu "+" v horním menu.	Zobrazí se obrazovka s možností editace vytvářeného setu.
10	Vyplnit jméno setu.	Bude vyplněná odpovídající hodnota.
11	Kliknout na ikonu "Uložit" v horním menu.	Zobrazí se původní obrazovka záložky vybavení s nově vytvořeným setem.
12	Kliknout na vytvořený set.	Zobrazí se detail setu.
13	Kliknout na ikonu "+" v horním menu.	Zobrazí se seznam základních kontejnerů (zem a inventář).
14	Kliknout na kontejner "Inventář".	Zobrazí se obsah inventáře (včetně vytvořené vrhací zbraně).
15	Kliknout na dříve vytvořenou vrhací zbraň.	Řádek označené vrhací zbraně zezelená.
16	Kliknout na ikonu "Uložit" v horním menu.	Zobrazí se detail setu, kde obsahem setu bude zvolená vrhací zbraň.
17	Kliknout na ikonu zpět v horním menu.	Zobrazí se původní obrazovka záložky vybavení.
18	Kliknout na checkbox u vytvořeného setu.	Checkbox je zaškrtnutý.
19	Rozkliknout seznam zbraní použitelných na blízko.	Vytvořená vrhací zbraň je součástí seznamu.
20	Vybrat vytvořenou vrhací zbraň.	Napočtené hodnoty pro souboj odpovídají hodnotám zbraně.
21	Rozkliknout seznam zbraní použitelných na dálku.	Vytvořená vrhací zbraň je součástí seznamu.
22	Vybrat vytvořenou vrhací zbraň.	Napočtené hodnoty pro souboj odpovídají hodnotám zbraně.

Obrázek 3.1: Scénář uživatelského testu na vrhací zbraně

Vrhací zbraně Jedním z testů, které zde budou více rozebrány, je test na vrhací zbraně (obrázek 3.1). Cílem tohoto testu je zjistit, jestli je možné vytvořit vrhací zbraň a použít ji v bojových statistikách jako zbraň na dálku i jako zbraň na blízko. Test je možné provádět na mobilním zařízení nebo tabletu s operačním systémem Android minimální verze API 16. Na tomto zařízení

je potřeba mít nainstalovanou aplikaci MobChar a balíček Dračí doupě verze 3.0. Zároveň je potřeba mít v aplikaci vytvořenou postavu z balíčku Dračí doupě.

Test začíná zapnutím aplikace a vybráním postavy z balíčku Dračí doupě. Dále je potřeba vytvořit v inventáři předmět typu vrhací zbraň. Poté se ve vybavení musí vytvořit nový set, do kterého je nutné vytvořenou vrhací zbraň přidat. Po aktivaci tohoto setu se v bojových statistikách musí v možnostech výběru zbraně na dálku i na blízko tato vrhací zbraň objevit. Následně je nutné zkontrolovat, že pokud je daná vrhací zbraň v jednotlivých kolonkách zvolena, napočítané hodnoty souhlasí s jejími atributy.

Odeslání zprávy Dalším uživatelským testem je test na navázání spojení a odeslání zprávy na zařízení Pána jeskyně (obrázek 3.2). Cílem tohoto testu je navázat spojení pomocí služby Bluetooth s balíčkem pro Pána jeskyně a poslat na toto zařízení zprávu. Po odeslání by se měla zpráva objevit v konverzaci postavy z Dračího doupěte a na zařízení Pána jeskyně by se měla zobrazit notifikace. K provedení testu je potřeba mít dvě zařízení s operačním systémem Android minimální verze API 16, na kterých je nainstalována aplikace MobChar. Na jednom ze zařízení navíc musí být nainstalován balíček Dračí doupě verze 3.0 a na druhém balíček pro Pána jeskyně verze 1.0. Na zařízení s balíčkem Dračí doupě musí být vytvořená postava. Na zařízení s balíčkem Pána jeskyně musí být vytvořeno dobrodružství. Obě zařízení musí podporovat komunikaci pomocí služby Bluetooth, která bude na obou zařízeních vypnuta.

Scénář začíná několika kroky, které je nutné provést na zařízení s balíčkem Dračí doupě. Na tomto zařízení je potřeba aplikaci zapnout a dostat se do detailu postavy. V detailu postavy je nutné spustit vyhledávání Pána jeskyně kliknutím na ikonu „Hledat družinu“ v horním menu obrazovky. Po této akci aplikace zobrazí dialog, ve kterém uživateli nabízí zapnutí služby Bluetooth. Po potvrzení dialogu je služba Bluetooth zapnuta.

Dále je potřeba provést kroky na zařízení s nainstalovaným balíčkem Pána jeskyně. Na tomto zařízení je potřeba najít předchozí testovací zařízení a přidat ho jako postavu do družiny. Otestování této funkcionality není součástí této práce. Podrobnější postup a otestování tohoto postupu naleznete v bakalářské práci Matěje Sháněla[2]. Informace o úspěšném navázání je zobrazena na zařízení s balíčkem Dračí doupě v podobě toastu.

Pro odeslání zprávy je potřeba se na zařízení s balíčkem Dračí doupě přesunout do složky „Zprávy“, kde je v dolní části obrazovky kolonka pro zadání textu nové zprávy. Po vyplnění této kolonky stačí už jen kliknout na ikonu v horním menu pro odeslání zprávy a nově napsaná zpráva by se měla zobrazit na obrazovce vpravo dole i s aktuálním časem. Na zařízení Pána jeskyně se zobrazí systémová notifikace o příchodu zprávy.

2. Navázání spojení a odeslání zprávy	
Popis	Testování navázání spojení s balíčkem pro Pána jeskyně pomocí služby Bluetooth. Součástí testu není testování balíčku pro Pána jeskyně.
Výsledek	Je odeslána zpráva ze zařízení s balíčkem DrD na zařízení s balíčkem PJ a lze ji zobrazit na obou zařízeních.
Prostředí	Dvě zařízení Android s minimální verzí API 16 a nainstalovanou aplikací MobChar. První zařízení s nainstalovaným balíčkem DrD verze 3.0 (dále jen Z1). Druhé zařízení s nainstalovaným balíčkem PJ verze 1.0 (dále jen Z2).
Předpoklady	Existující postava z balíčku Dračí doupě na Z1. Vypnutá služba Bluetooth na Z1.

Krok	Úkon	Očekávaný stav
1	Na Z1 zapnout aplikaci MobChar a vybrat postavu z balíčku Dračí doupě.	Na Z1 se zobrazí obrazovka přehledu postavy.
2	Na Z1 kliknout na ikonu v horním menu "Hledej družinu".	Na Z1 se zobrazí dialog s možností zapnutí služby Bluetooth a zamítnutí celého procesu.
3	Na Z1 kliknout na tlačítko "ANO".	Na Z1 se zobrazí dialog "Zapínám Bluetooth", zapne se služba Bluetooth a zobrazí se toast "hledání spojení".
4	Na Z2 zapnout aplikaci MobChar vybrat dobrodružství z balíčku Pána jeskyně.	Na Z2 se zobrazí obrazovka s přehledem družiny.
5	Na Z2 přidat postavu do družiny.	Na Z1 se zobrazí toast "Spojení navázáno".
6	Na Z1 přejít do záložky "Zprávy".	Na Z1 se zobrazí seznam starých zpráv.
7	Na Z1 vyplnit text nové zprávy do spodní kolonky.	Text zprávy se zobrazí v kolonce, do které byl vyplněn.
8	Na Z1 kliknout na ikonu "Poslat zprávu" v horním menu.	Text zprávy zmizí z kolonky pro novou zprávu. V seznamu zpráv na Z1 se vpravo dole objeví napsaná zpráva s aktuálním časem. Na Z2 bude přijata nová zpráva od dané postavy.

Obrázek 3.2: Scénář uživatelského testu na odeslání zprávy

3.4 Uživatelská příručka

Poslední část zadání je vytvoření uživatelské příručky. Jelikož pro původní balíček MobChar byla již uživatelská příručka vytvořena, bylo stanoveno jako zbytečné vytvářet příručku úplně novou. Byla použita původní příručka[9], která byla vytvořena jako součást práce v předmětu BI-SP2. Tato příručka byla rozšířena o návod na používání nově vytvořených funkcí. Do příručky tedy přibyly hlavně informace o možnosti komunikace s balíčkem Pána jeskyně a možnost využít simulátor hodu kostkou. Také se v příručce objeví informace o vrhacích zbraních a efektech na postavě umožňující modifikaci atributů předmětů. Dále byla během této práce změněna struktura některých šablon. To bude v uživatelské příručce také promítnuto. Kompletní uživatelská příručka je na přiloženém CD.

Závěr

V rámci této bakalářské práce byla analyzována původní verze balíčku Dračí doupě, jádra a knihovny aplikace MobChar. Dále byly analyzovány nové požadavky na balíček Dračí doupě. Tyto požadavky byly upřesněny a byla analyzována jejich možnost začlenění do již existujícího balíčku Dračí doupě.

Pomocí vytvořené analýzy byl vypracován návrh implementace nových požadavků. Na základě návrhu byly všechny požadavky implementovány. Nově vytvořené části byly řádně zdokumentovány a otestovány jednotkovými, integračními a uživatelskými testy. Na závěr byla rozšířena uživatelská příručka o návody na použití nových funkcionalit.

Byly implementovány a otestovány všechny požadavky a tak bylo zadání práce splněno. Během implementace byly, dle požadavků od vedoucího práce, provedeny změny v implementaci některých částí balíčku předchozí verze. Šlo hlavně o opravu nevhodného použití konstant a některých tříd, které byly nahrazeny výčtem.

Do budoucna by bylo možné do balíčku přidat několik funkcí. Jednou nedokonalostí aplikace je vytváření nových šablon pro objekty. V současné době musí uživatel připsat šablonu ve správném formátu do souboru k ostatním šablonám, což nemusí být pro neznalého uživatele jednoduché. Bylo by tedy vhodnější, kdyby bylo možné šablony generovat z objektů, které uživatel vytvoří v aplikaci.

Další možností, jak balíček rozšířit, je přidat funkci na exportování postavy do formátu PDF. Tím by bylo uživateli umožněno informace o postavě vytisknout a použít v papírové podobě.

Literatura

[1] Benda, M.: *Dračí doupě: pravidla pro začátečníky : verze 1.6*. ALTAR, 2001, ISBN 80-85979-34-9.

[2] Sháněl, M.: *MobChar - balíček pro Pána jeskyně v Dračím doupěti : bakalářská práce*. Praha:České vysoké učení technické, Fakulta informačních technologií, 2017.

[3] Nižaradze, J.: *MobChar - jádro aplikace : bakalářská práce*. Praha:České vysoké učení technické, Fakulta informačních technologií, 2016.

[4] Pastorek, T.: *MobChar - balíček pro Dungeons and Dragons : bakalářská práce*. Praha:České vysoké učení technické, Fakulta informačních technologií, 2016.

[5] Dresler, R.: Vícevrstvé architektury aplikací [online]. [cit. 2. 5. 2017]. Dostupné z: `http://www.robertdresler.cz/2011/04/vicevrstve-architektury-aplikaci.html`

[6] Object oriented design: *Abstract Factory [online]*. [cit. 4. 5. 2017]. Dostupné z: `http://www.oodesign.com/abstract-factory-pattern.html`

[7] Android developers: *Bluetooth [online]*. [cit. 04. 04. 2017]. Dostupné z: `https://developer.android.com/guide/topics/connectivity/bluetooth.html`

[8] Phillips, B.; Stewart, C.; Hardy, B.; aj.: *Android Programming: The Big Nerd Ranch Guide*. Pearson Technology Group, Indianapolis, USA, Únor 2016, ISBN 13 978-0134171500.

[9] Zvěřina, J.: *MobChar - Uživatelská příručka pro balíček Dračí doupě*. Praha:České vysoké učení technické, Fakulta informačních technologií, 2016, [cit. 3. 5. 2017]. Dostupné z: `https://svn.project.fit.cvut.cz/Mobchar/documentation/drd/SP2-2.iterace-files/uzivatelska_prirucka.pdf`

Příloha **A**

Seznam použitých zkratek

BI-SP1 Softwarový týmový projekt 1

BI-SP2 Softwarový týmový projekt 2

ČVUT České vysoké učení technické v Praze

XML Extensible markup language

DrD Dračí doupě

BO Business object

DAO Data access object

PDF Portable document format

UML Unified modeling language

SVN Apache subversion

API Application programming interface

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `apk` ..... adresář se spustitelnými soubory aplikací
- `src`
  - `impl.zip` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `documentation`
  - `diagrams` ..... dokumentace analýzy a návrhu
  - `javadoc` ..... dokumentace zdrojových kódů
- `attachments`
  - `user_tests.xlsx` ..... uživatelské testy aplikace
- `manual` ..... uživatelská příručka
- `text`
  - `BP_Weberova_Sarka_2017.pdf` ..... text práce ve formátu PDF