Sem vložte zadání Vaší práce.

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

Bakalářská práce

MobChar - desktopová aplikace pro Pána jeskyně pro Dračí doupě

Jan Horáček

Vedoucí práce: Ing. Zdeněk Rybola

30. března 2017

Poděkování

Poděkování

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 30. března 2017 .....

Odkaz na tuto práci

Horáček, Jan. *MobChar - desktopová aplikace pro Pána jeskyně pro Dračí doupě*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2017.

Abstrakt

Cílem této práce je hráčům Dračího doupěte, a zejména Pánu jeskyně, usnadnit vytváření dobrodružství, postav a předmětů pomocí jednoduché desktopové aplikace. Hlavní výhodou aplikace je, možnost exportování dobrodružství pro mobilní aplikaci MobChar pro Pána jeskyně a exportování do přehledného pdf vhodné pro tisk a hraní Dračího doupěte bez mobilů a jiných zařízení. V práci jsem vytvořil aplikaci, ve které je možné pomocí grafického rozhraní vytvářet mapy dobrodružství propojené s příšerami a předměty, které se dají na mapě najít. Veškeré výtvory se ukládají pro případné další použití. Můžete si vytvářet sbírku příšer a předmětů, které můžete i sdílet s ostatními Pány jeskyně, a tak se nechat inspirovat ostatními. Hlavní přínos této aplikace je pro hráče Dračího doupěte, kteří by rádi využili moderní technologie pro tvorbu svého dobrodružství. Někteří hráči jistě ocení možnost sdílení dobrodružství s ostatními nadšenci, anebo naopak se rádi nechají inspirovat ostatními hráči. Pomocí databáze již dříve vytvořených příšer a předmětů bude velice snadné vytvářet nové a ještě zábavnější dobrodružství.

Klíčová slova desktopová aplikace, hra na hrdiny, MobChar, Dračí doupě, rozšíření, herní scéna, tvorba dobrodružství, Python, SQLite, XML

Abstract

Sem doplňte ekvivalent abstraktu Vaší práce v angličtině.

Keywords desktop application, heroes games, MobChar, Dračí doupě, extension, desktop games, story creation, Python, SQLite, XML

Obsah

Seznam obrázků

Úvod

Dračí doupě je populární hra na hrdiny, která vznikla jako česká verze hry Dungeons & dragons. V České Republice si získala velkou oblibu zvláště mezi hráči stolních her a počítačových RPG her.

Hráči hrají za své hrdiny, které si na začátku dobrodružství vytvořili, putují světem a zažívají rozličná dobrodružství, která pro ně pán jeskyně připraví. Každý hráč si může vybrat z rozličného výběru ras, které se ve světě vyskytují, a zvolit si, jakému povolání se bude věnovat. Ale ať si vybere mocného orkského válečníka, rychlého a úskočného hobitího zloděje nebo moudrého a mocného elfského válečníka, budou ho čekat složitá rozhodnutí, která určí, jakým směrem se dobrodružství bude dále ubírat.

Dračí doupě se lehce liší od klasických stolních her, kde plán hry nebo dobrodružství máte jeden a toho se musíte držet. Zde příběh vymýšlí další hráč, takzvaný pán jeskyně, který při hře celé dobrodružství řídí a popisuje hrdinům. Veškerá rozhodnutí, která neučiní samotní hráči, rozhodne pán jeskyně. Příprava propracovaných a zábavných dobrodružství je velice složitá a zabere mnoho hodin času. Je zapotřebí vytvořit přehledné poznámky a mapy, které při hře umožní všem hráčům dobře pochopit situaci a oblast, ve které se právě nachází. Cílem této bakalářské práce je tento proces usnadnit a zkrátit dobu přípravy, aby se pán jeskyně mohl věnovat převážně hraní s ostatními spoluhráči.

Aplikace vznikala souběžně s mobilní aplikací MobChar pro pány jeskyně, ve které je možné veškeré informace o dobrodružství přehledně zobrazit a prezentovat hráčům. V balíčku aplikací již existuje mobilní aplikace MobChar pro hráče, která slouží pro zaznamenávání osobního deníku hráče.

Struktura práce

V první kapitole je popsán úvod do problematiky. Jsou zde popsána základní pravidla Dračího doupěte a pravidla vytváření dobrodružství. Důležitou částí je také analýza již existujících aplikací, ať už se jedná o aplikace, které by tato aplikace měla nahradit a nebo aplikace MobChar, se kterými bude aplikace spolupracovat.

Druhá kapitola je věnována analýze problematiky. Jsou zde popsány veškeré požadavky na aplikaci a zadefinované činnosti, které uživatelé s aplikací budou provozovat.

Na základě druhé druhé kapitoly je ve třetí kapitole vytvořen návrh a struktura tříd a architektury. Jsou zde detailně popsány vrstvy architektury a komunikace mezi nimi.

Ve čtvrté kapitole je popsaná problematika, týkající se samotné implementace. Jsou zde popsány problémy, které vznikly na základě návrhu architektury a jejich řešení.

Kapitola 1

Úvod do problematiky

1.1 Cíl práce

Cílem této práce je vytvořit desktopovou aplikaci, které rozšíří již existují balíček mobilních aplikací MobChar o nové funkcionality. Hlavním účelem aplikace je vytváření dobrodružství pro Dračí doupě. Jednou z hlavních funkcionalit je tvorba xml souborů, se kterými umí pracovat mobilní aplikace MobChar.

1.2 Dračí doupě

TODO: napsat o DrD

Kapitola 2

Současný stav

2.1 Projekt MobChar

Aplikace MobChar vznikla jako samostatná aplikace, která měla sloužit jako osobní deník hráče pro hru Dračí doupě. Postupem času se rozšiřovala a měnila základní struktura. Nynější stav již umožňuje vytváření rozšíření pro další hry na hrdiny jako jsou například Dungeons & Dragons. V této chvíli jsou již plně funkční rozšíření pro Dračí doupě a Dungeons & Dragons a vznikají další.

V rámci aplikace MobChar vzniká další rozšíření, ne však pro osobní deník hráče, ale pro pána jeskyně. Jednoduchá aplikace, ve které budou k dispozici údaje o všech hráčích a právě probíhajícím dobrodružství. Protože tvorba dobrodružství je poměrně náročná činnost, která vyžaduje napsání mnoha textu, připravení velkého množství šablon a další časové náročné činnosti, bylo by velice nepraktické toto dobrodružství vytvářet na mobilním zařízení. Proto vzniká desktopová aplikace DeskChar, jejíž hlavním cílem je vytváření dobrodružství a šablon pro mobilní aplikace.

2.2 Existující aplikace

Díky rozsáhlé komunitě okolo her na hrdiny, jako jsou Dračí doupě a Dungeons & Dragons, vzniklo již mnoho aplikací, které mají hraní či přípravu dobrodružství usnadnit. Je možné najít desítky různých editorů map, které jsou více či méně povedené a propracované. Mezi nejzajímavější patří Dungeon Painter Studio, které se převážně zaměřuje na tvorbu rozsáhlých map, nebo například Donjon, který se zaměřuje na generování náhodných map a další objektů, jako jsou například obchody, truhly, počasí a další. Všechny tyto aplikace vygenerují PDF soubor, který je možné následně vytisknout a vzít s sebou na herní sezení. Žádná z těchto aplikací nenabízí mobilní prohlížeč dobrodružství, nebo export do formátu, který by bylo možné na mobilních zařízeních jednoduše prohlížet.Bohužel PDF soubor formátu A4 s mnoho údaji

není příliš přehledný, obzvlášť na malých obrazovkách.

Dungeon Painter Studion je aplikace, která se zaměřuje na tvorbu map a následné generování PDF souborů. Existuje online verze, dostupná je zdarma, která však nenabízí veškeré funkcionality plné verze. Ale i v této zjednodušené verzi dokážete vytvořit velmi propracované mapy. Plná verze je již desktopová aplikace, která je placená (stojí přibližně 400 korun). Je možné zde vytvořit opravdu propracovaná a poutavá dobrodružství včetně stínování, světelných efektů a mnoha dalšího. Je však až zbytečně složitý a pro většinu pánů jeskyně tvorba mapy bude příliš zdlouhavá. Vytvoření jedné mapy zabere mnoho času a ve většině případů takto propracované mapy nejsou pro hraní důležité. Bohužel v základní verzi není možné s mapou propojit předměty a příšery z dobrodružství, což považuji za jednu z nejdůležitějších funkcionalit.

Donjon je online aplikace, která se zaměřuje na generování náhodných lokací. Můžeme si zde navolit základní nastavení, jako je velikost mapy, velikost místností a další. Finální verzi mapy již upravit však nemůžeme. K mapě dostaneme popis jednotlivých místností v poměrně nepřehledné tabulce. Tento formát bez dodatečných úprav v obrázkovém editoru nelze rozumně vytisknout. Aplikace nabízí generování další prvků, jako jsou například poklady, obchody a další, bohužel však tyto prvky se jíž nedají propojit s vygenerovanou mapou.

Kapitola 3

Analýza

3.1 Požadavky

Sběr požadavků je jedna z prvních činností, které se musí provést na začátku analýzy projektu. Je zapotřebí sepsat veškeré funkční a nefunkční požadavky, na základě kterých bude vznikat návrh a výsledný program. Na základě požadavků je možné vytvořit první odhady rozsahu projektu a jeho ceny. Tento odhad je velice nepřesný, ale zákazníci vyžadují předběžnou cenu, již na základě jejich požadavků na program.

3.1.1 Funkční požadavky

Funkční požadavky jsou rozděleny do dvou částí, funkcionalita pro hráče a funkcionalita pro pána jeskyně. Tyto sekce jsou rozdělené, protože popisují stejný program, ale z pohledu jiného uživatele.

3.1.1.1 Funkcionalita pro hráče

Funkční požadavky, které program bude poskytovat zejména pro hráče pro práci se šablonami.

Tvorba a editace šablon pro MobChar: program bude umožňovat vytvářet nové šablony pro mobilní aplikaci MobChar. Bude také umožňovat editovat stávající šablony. Jednotlivé šablony lze přehledně uspořádat do stromové struktury.

Šablony se kterými bude možné pracovat:

- Kouzla
- Předměty
- Schopnosti

- Efekty

Export šablon pro mobilní aplikaci MobChar: program bude umožňovat jednoduchý a hromadný export vybraných šablon pro mobilní aplikaci MobChar ve formátu xml.

Import šablon z mobilní aplikace MobChar: program bude umožňovat import xml šablon, které se dají vytvořit v mobilní aplikaci MobChar. Šablony se přidají do databáze a následně je bude možné editovat.

Ukládání šablon do souboru: program bude umožňovat ukládat veškeré vytvořené šablony do společného xml souboru, který bude možné opět do programu načíst a nahrát veškeré šablony. Soubor bude sloužit pro jednoduché ukládání práce a sdílení šablon.

3.1.1.2 Funkcionalita pro pána jeskyně

Funkční požadavky, které program bude poskytovat zejména pro pána jeskyně pro vytváření nových dobrodružství. **Tvorba a editace dobrodružství:** program bude umožňovat vytváření nových a editaci stávajících dobrodružství. K dobrodružství se dají přiřadit veškeré potřebné objekty, které se v aplikaci dají vytvořit. Celé dobrodružství lze členit do přehledné stromové struktury.

Tvorba a editace mapy pro dobrodružství: Program bude umožňovat vytvářet a editovat mapy pro dobrodružství. Mapa půjde vytvářet ve třech různých módech

- Základní mód - k dispozici budou pouze jednoduché prvky a kreslící nástroj pro čáry
- Grafický mód - mapa se bude skládat z jednotlivých grafických dílků
- Obrázkový mód - jako podklad mapy bude sloužit nahraný obrázek, který bude doplněn o jednoduché objekty

Na mapu bude možné umisťovat odkazy na vytvoření postavy, příšery, předměty a další mapové prvky. Mapy bude možné sdružovat do stromové struktury pro větší přehlednost

Tvorba a editace příšer pro dobrodružství: program bude umožňovat vytváření nových a editaci již existujících příšer pro dobrodružství. Příšery půjdou seskupovat do stromové struktury pro větší přehlednost.

Tvorba a editace postav pro dobrodružství: program bude umožňovat vytváření nových postav a editaci již existujících. Postavy půjdou seskupovat do stromové struktury pro větší přehlednost.

Exportování dobrodružství pro mobilní aplikaci MobChar: celé dobrodružství půjde exportovat ve formátu xml se strukturou, kterou podporuje mobilní aplikace MobChar pro pána jeskyně.

Importování a exportování dobrodružství pro sdílení: celé dobrodružství včetně všech přiřazených objektů půjde exportovat do jednoho xml souboru, který následně bude možné do aplikace znovu nahrát. Soubor bude sloužit pro ukládání práce a případné sdílení s ostatními pány jeskyně.

Export dobrodružství pro tisk: Aplikace bude umožňovat exportování celého dobrodružství do přehledného formátu, který je vhodný pro tisk.

3.1.2 Nefunkční požadavky

Podpora operačních systémů Windows a Linux: program půjde spustit pod operačnímy systémy Windows a Linux. U operačního systému Windows budou podporovány všechny verze, které jsou novější než Windows XP. Program bude spustitelný pomocí exe souboru. U operačního systému Linux budou podporovány standardní linuxové distribuce. Program bude spustitelný pomocí bash souboru.

Jazyková podpora: program bude umožňovat přepínání jazyků a případnou lokalizaci pro další jazyky. V základu bude podporován český jazyk a anglický jazyk.

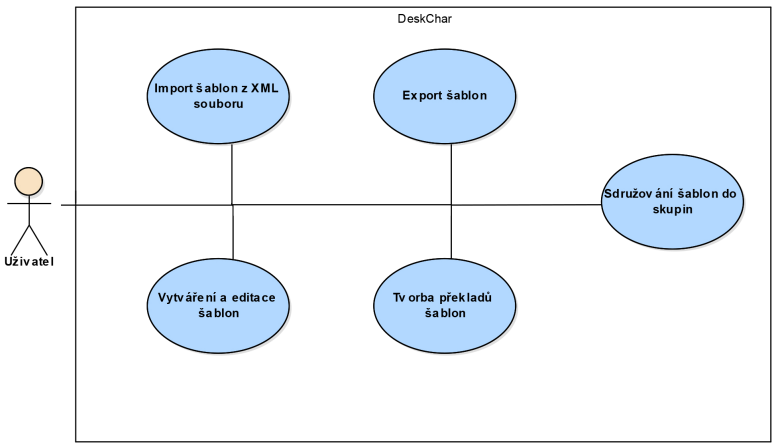
3.2 Případy užití

Případy užití popisují jednotlivé činnosti, které uživatel provádí, při práci s aplikací. Diagram byl rozdělen na tři hlavní části pro větší přehlednost.

3.2.1 Práce se šablonami

Případ užití týkající se práci se šablonami namodelovaný na obrázku 3.1, se týká veškeré práce se šablonami. Dobrodružství a postavy jsou složené z jednotlivých šablon, které se dají využít i samostatně. Samotné dobrodružství a postavy mají stejný formát jako šablony a navíc můžou obsahovat základní šablony (například postavy můžou mít u sebe zbraně).

Vytváření a editace šablon: uživatel si bude moci vytvořit nové šablony nebo editovat stávající. Veškeré parametry, které je možné do šablony zapsat, lze jednoduše upravovat v přehledném formuláři. Uživatel si může vytvořit libovolný počet šablon, které se na základě návazností sdružují do větších celků (dobrodružství, postava).



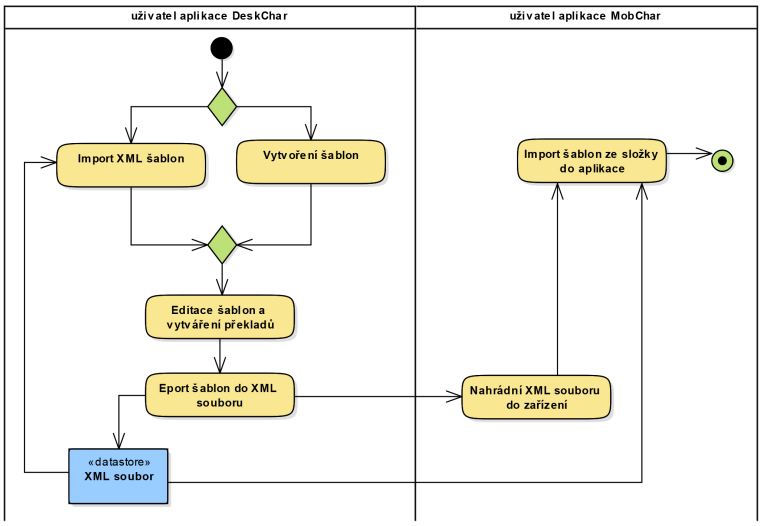
Obrázek 3.1: Případy užití pro práci se šablonami

Import šablon z XML souboru: program bude umožňovat import šablon ze strukturovaných xml souborů, které vytváří mobilní aplikace MobChar a také samotný program. Importované šablony se přidají do databáze a lze s nimi následně provádět stejné činnosti jako s nově vytvořenými. Import je možný buď hromadný, který se pokusí ze souboru dostat veškeré dostupné informace a přidat je do aplikace na příslušná místa, nebo lokální import, který ze souboru vytáhne pouze šablony o daném typu.

Sdružování šablon do skupin: veškeré šablony půjde rozřazovat do stromové struktury pro větší přehlednost a jednoduší práci s nimi. Pomocí vytváření složek a jednoduchého drag and drop systému bude rozřazování velice jednoduché a intuitivní. Díky rozřazení do složek je následná práce se šablonami velice usnadněná, ať už se jedná o export nebo přiřazování do rodičovských šablon.

Export šablon: veškeré vytvořené šablony lze z aplikace exportovat. K dispozici jsou různé druhy exportu. První možnost je export pro mobilní aplikace MobChar. Aplikace vytvoří strukturované XML soubory, které odpovídají formátu, který používá mobilní aplikace MobChar. Další možností exportu je PDF formát, který slouží pro tisknutí šablon do přehledného almanachu, který umožní využít šablony i hráčům, kteří mobilní aplikaci nemají nebo ji nechtějí použít. Poslední možnost exportu, je klasické uložení celého stavu šablon do jednoho xml souboru, který bude využívat stejnou strukturu šablon.

Tvorba překladů šablon: ke každé šabloně vytvořené nebo importované do aplikace lze vytvořit neomezený počet překladů. Překlady se vytváří v rámci jedné šablony, pomocí přehledného přepínaní mezi jazyky. Veškeré hodnoty,



Obrázek 3.2: Model bussines procesů vytváření šablon

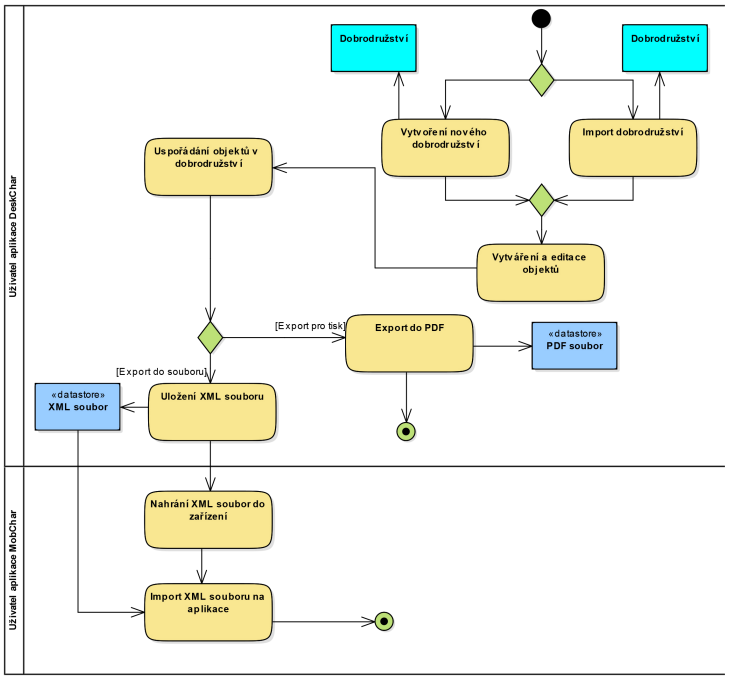
pro které překlad nemá smysl (číselné hodnoty, povolání, atd.) budou synchronizování napříč všemi jazyky.

3.3 Business procesy

Business proces (někdy též nazývaný podnikový nebo obchodní proces) je tok práce nebo činnosti. Dají se zaznamenávat pomocí textu nebo přehledných modelů. Modely business procesů se snaží přehledně do diagramu zanést jednotlivé procesy, které bude uživatel s danou aplikací nebo doménou provádět. Já jsem zvolil UML diagram aktivit pro zachycení business procesů v mé práci.

3.3.1 Vytváření šablon

Diagram na obrázku 3.2 popisuje základní proces práce se šablonami. Proces popisuje vytvoření šablon a následné nahrání do mobilní aplikace. Šablony vytvoříme v programu nebo případně importujeme z xml souboru vytvořeném buď aplikací nebo například aplikací MobChar. Samozřejmě import a vytvoření nových šablon můžeme kombinovat. Při tomto procesu nevzniká pouze jedna šablona, ale celý soubor šablon, obvykle jednoho typu. Provedeme veškeré potřebné úpravy a vytvoříme překlady (pokud jsou nějaké zapotřebí). Danou šablonu následně exportujeme do strukturovaného souboru XML ve formátu, který podporuje aplikace MobChar. Uživatel mobilní apli-

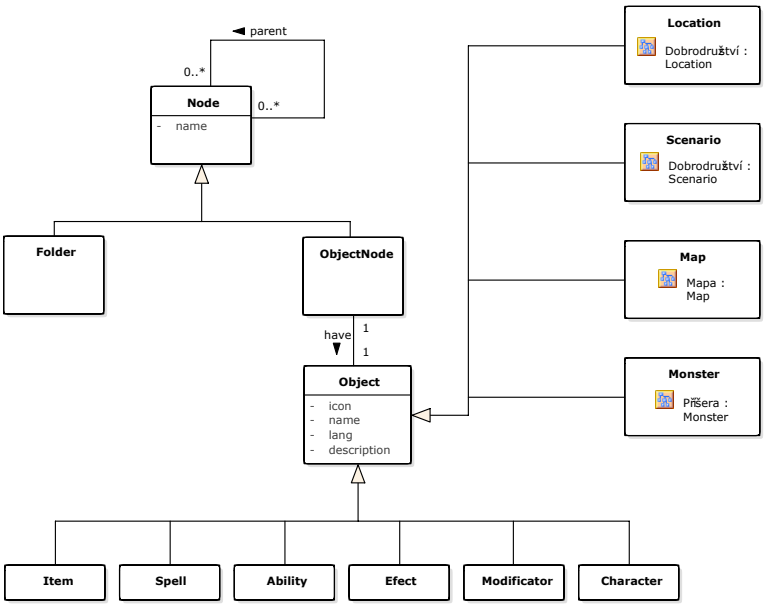


Obrázek 3.3: Model bussines procesů vytváření dobrodružství

kace si vytvořený soubor nahraje do zařízení a následně provede import do aplikace. Nyní již může s novou šablonou pracovat.

3.3.2 Vytváření dobrodružství

Diagram 3.3 popisuje proces, při kterém vzniká nové dobrodružství. Jak už jsem dříve zmiňoval, tak dobrodružství je také šablona. Protože se jedná o hlavní šablonu, která v sobě obsahuje větší počet šablon různých druhů, rozhodl jsem se vytváření dobrodružství popsat více do podrobna. Na začátku procesu vytvoříme nové dobrodružství nebo importujeme již existující a dále budeme upravovat již stávající hodnoty. V bodě **Vytváření a editace objektů** se jedná o businnes proces popsaný v kapitole 3.3.1,ve kterém vytvoříme veškeré potřebné objekty, které budou součástí dobrodružství. Do dobrodružství veškeré šablony přidáme a rozřadíme podle potřeby (podle částí dobrodružství, podle typu šablony atd.). Když máme veškeré úpravy hotové, můžeme výsledné dobrodružství exportovat. Zde si můžeme vybrat zda budeme exportovat dobrodružství pro tisk do formátu PDF nebo pro MobChar ve formátu XML.
Formát pro tisk: Vybereme části, které chceme exportovat (nemusíme exportovat celé dobrodružství naráz) a provedeme export. Vytvoří se nám přehledný PDF soubor který následně můžeme vytisknout nebo nahrát do zařízení, které



Obrázek 3.4: Analytický doménový model stromové struktury

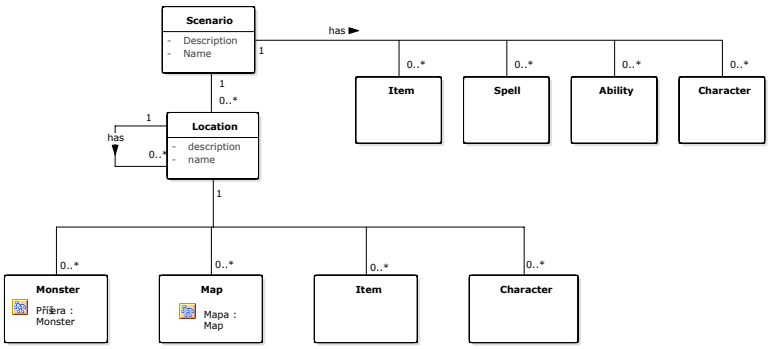
umí pracovat s formátem PDF.
Formát pro MobChar: Při exportu do formátu XML se vždy exportuje celé dobrodružství. Nedají se vybrat pouze některé části. Aplikace vytvoří XML soubor. Uživatel následně získaný soubor nahraje do zařízení a importuje ho do aplikace. Takto vytvořené dobrodružství je určené pro MobChar rozšíření pro pána jeskyně.

3.4 Doménový model

Doménový model má za úkol popsat strukturu tříd v aplikaci a jejich vazby mezi nimi. Pro zaznamenání se využívá UML diagram. Diagram nepopisuje jednotlivé funkce tříd ani proměnné, které slouží k implementaci tříd, ale zachycuje pouze základní strukturu.

3.4.1 Stromová struktura

Diagram na obrázku 3.4 popisuje strukturu objektů, které slouží pro uchování všech objektů (šablon) a jejich rozřazení do stromové struktury. Veškeré šablony využívají tuto stromovou strukturu. Třída Folder (složka) slouží pouze k rozřazování jednotlivých objektů do skupin. Složka má pouze jméno a může obsahovat libovolný počet složek nebo objektů. ObjektNode pak slouží k uchování šablony samotné. Dolní část objektů na obrázku (Item, Spell, atd.) jsou objekty, které jsou totožné s aplikací MobChar a slouží k uchování šablon pro



Obrázek 3.5: Analytický doménový model dobrodružství

základní aplikaci a využívají se i v aplikaci pro pána jeskyně. Pokud vás zajímá podrobnější popis těchto objektů, nahlédněte do bakalářské práce týkající se MobCharu. O proti tomu, objekty na pravé straně diagramu (Map item, Scenarion, atd.) jsou objekty, které slouží převážně k vytvoření dobrodružství. Jejich podrobnějšímu popisu se věnuji dále.

Veškeré objekty můžou mít potomky. Systém složek slouží pouze pro přehledné uspořádání. Každý objekt má definované objekty, které může mít jako potomky. Tento systém slouží pro ukládání závislostí, například dobrodružství může obsahovat mnoho lokací, příšer, map. Některé objekty, jako například kouzla a schopnosti, žádné potomky mít nesmějí. Veškeré vazby mezi objekty jsou zachyceny pouze ve stromové struktuře, aby bylo zabráněno duplikování informací.

3.4.2 Dobrodružství

Diagram na obrázku 3.5 popisuje strukturu pro ukládání a práci s dobrodružstvím. Hlavní třída scenario je rozdělená do jednotlivých lokací. Pro celé dobrodružství jsou však společné předměty, kouzla, schopnosti a postavy. Předměty, kouzla a schopnosti jsou pro pána jeskyně, který je nadále poskytuje hráčům, například pokud se hráč má možnost naučit nové kouzlo, nebo získal důležitý předmět pro dobrodružství (elixír, mapa, atd.). Třída Character zde zastupuje postavy hráčů, kteří dané dobrodružství hrají. Jedná se o stejné třídy jako v původní aplikaci MobChar. Pro přesnější popis těchto tříd, nahlédněte do bakalářské práce aplikace MobChar.

Celé dobrodružství je členěné do lokací, které se navíc můžou do sebe zanořovat (lokace může mít pod sebou další lokace). Lokace obsahují příšery, mapy, předměty a postavy. Příšery mají velice podobnou strukturu jako postavy, můžou se naučit schopnosti a kouzla a vlastnit předměty, mají však

odlišné atributy, proto jsou od postav odděleny. Jednotlivé mapy mají složitější strukturu, která je popsána níže. Předměty zde znamenají, předměty důležité pro tuto lokaci (klíče, věci v bednách a další). Poslední částí jsou postavy, které zde znázorňují postavy, za které nehrají hráči, ale samotný PJ. Jsou to důležité postavy pro dobrodružství, které se nacházejí v dané lokaci. Objekt je stejný jako klasické postavy, ale význam je trochu odlišný.

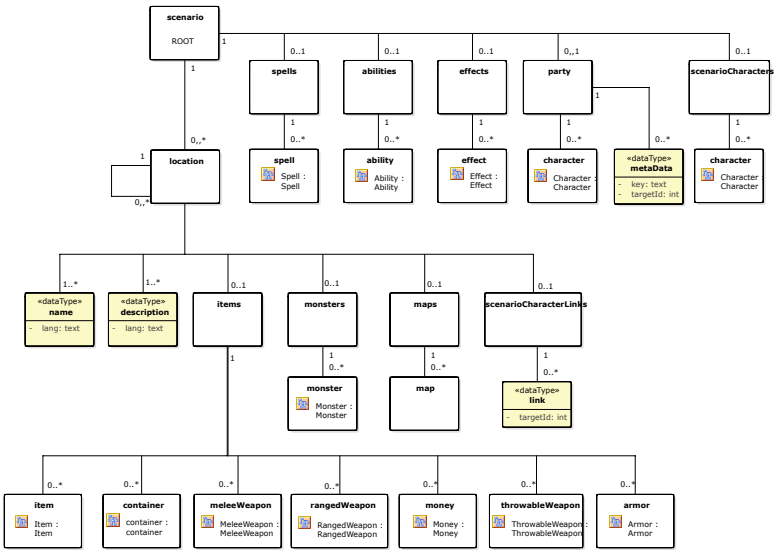
3.5 Model XML souborů

S doménovým modelem úzce souvisí návrh struktury xml souborů. Struktura těchto souborů je velmi důležitá. Na základě této definované struktury bude probíhat komunikace s mobilní aplikací MobChar. Formát základních šablon předmětů, kouzel, schopností, efektů a postav, byl převzat z původní aplikace MobChar pro hráče, aby byla zachována konzistence mezi aplikacemi. Dále se zaměřuji pouze na nové části, což jsou příšery a dobrodružství. Pokud vás zajímá struktura základních šablon, nahlédněte do bakalářky týkající se aplikace MobChar nebo do přiložené kompletní dokumentace.

Strukturu xml jsem namodeloval diagramem který se normálně využívá pro modelování tříd. Pro tento případ vypovídající hodnota diagramu je dostačující a jasně zadefinovává strukturu xml. Entity v diagramu které jsou znázorněné žlutou barvou znázorňují již konkrétní xml tag, který uvnitř obsahuje pouze hodnotu a žádné další vnořené tagy. Oproti tomu bílé entity znázorňují pouze obalující tag, který uvnitř obsahuje strukturu další tagů. Každá entita v diagramu znázorňuje jeden tag ve výsledném xml dokumentu. Parcialita a kardinalita v diagramu znázorňuje povinnost případně množství tagů, které se na daném místě v diagramu můžou objevit. Kořenový tag je vždy označen textem „ROOT“.

3.5.1 Dobrodružství

Struktura xml souboru pro dobrodružství je velice podobná doménovému modelu dobrodružství jak můžeme vidět na obrázku 3.6. První z odlišností si můžeme všimnout tagu „metaData“. Tyto data využívá pouze mobilní aplikace MobChar a slouží pro ukládání metadat ohledně připojených uživatelů. Tyto údaje v aplikaci nejdou nijak upravovat a pouze se ukládají pro udržení informace při importu a exportu.
Další odlišnost se nachází u „scenarioCharacter“. Z důvodu zachování historie cizích postav napříč lokacemi, se v lokacích nachází pouze link na postavu, která je definovaná pro celé dobrodružství. Jedná se o seznam tagů se jménem link, který mají obsah pouze identifikátor konkrétní cizí postavy.
Poslední významný rozdíl se týká rozdělení předmětů. Předměty jsou rozděleny na 7 kategorii. V aplikaci je dělení stejné, nicméně zde každý předmět má vlastní kořenový tag, čímž je jasně definován. Pokud vás zajímá přesný popis



Obrázek 3.6: Model XML souhoru dobrodružství

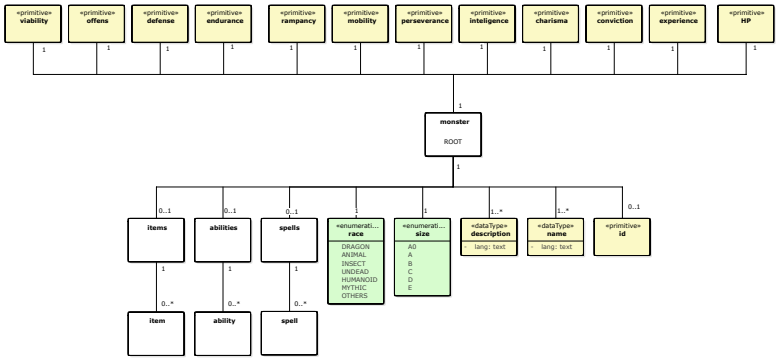
jednotlivých předmětů, můžete nahlédnout do přiložené dokumentace.

3.5.2 Příšery

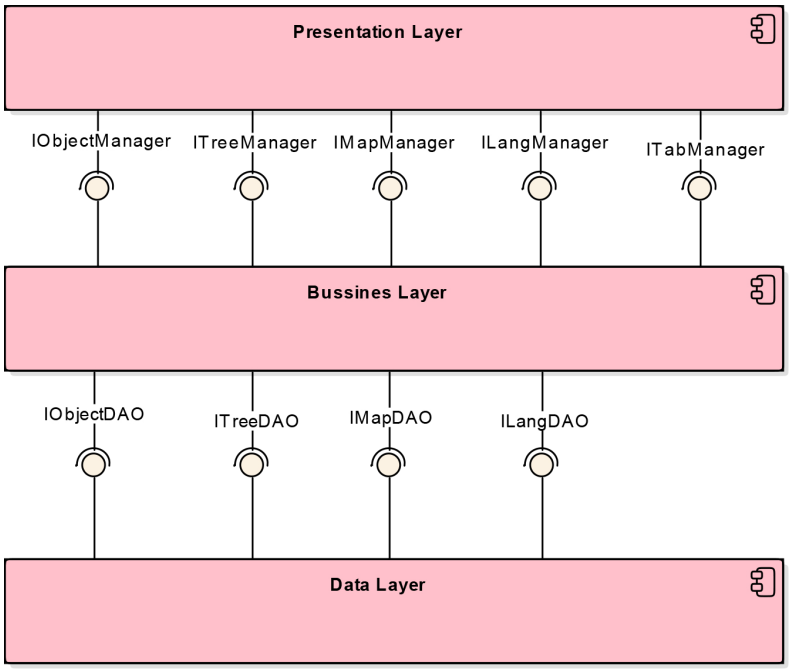
Druhý nově zadefinovaný formát xml souborů se týká příšer. Příšery jsou samozřejmě součástí dobrodružství, jak jste mohli vidět na obrázku 3.6. Na obrázku 3.7 můžeme vidět detailní strukturu souboru. Struktura je podobná postavám, také obsahuje seznam předmětů, schopností a kouzel. Také obsahuje tag race, který určuje rasu příšery, ale jedná se o odlišné rasy než pro postavy. Pro určení rasy příšer, jsme hledali kompromis, mezi dostatečnou vypovídající hodnotou a konzistencí oproti velkému množství ras, které by se velice rychle stalo nepřehledné. Uživatel si samozřejmě může příšery rozřadit podle libosti na základě stromové struktury nebo případně v popisku příšery. Proto jsme se rozhodli rasy příšer omezit pouze na základních sedm, které můžete vidět na obrázku. Hlavní důvod rozdělení příšer od klasických postav byl v rozdílných atributech. Pro počítání výsledku soubojů se používají jiné atributy než u postav.

3.6 Model architektury

Model architektury popisuje formální popis systému případně jeho detailní plán na úrovni komponent vedoucí k jeho implementaci.



Obrázek 3.7: Model XML souhoru příšery



Obrázek 3.8: Model architektury

Zvolil jsem třívrstvou architekturu z důvodu dobré přehlednosti a jednoduchém nahrazení jedné z části, bez zásahu do ostatních vrstev. Veškeré vrstvy mezi sebou komunikují na základe definovaného rozhraní.

3.6.1 Vrstvy

Prezentační vrstva zobrazuje informace pro uživatele. Jedná se o grafickou část celé aplikace. Kontroluje zadané vstupy, nijak však získané data nezpracovává.
Business vrstva je základní logická část aplikace. Leží zde jádro aplikace, její logika a funkce pro zpracování dat.
Datová vrstva je vrstva, která se stará o práci s daty. Zajišťuje komunikaci s databází a práci s XML soubory. Jejím základním úkolem je získat data z databáze nebo XML a převést je na objekty.

3.6.2 Rozhraní

Mezi vrstvami jsou definovaná rozhraní, které je nutné dodržet. Rozhraní začínají velkým písmenem I, konkrétní implementace daného rozhraní obvykle má stejný název, pouze bez počátečního písmena I.

IObjectDAO je skupina několik DAO tříd, které se starají o přístup k datům z databáze a z XML. Na diagramu 3.8 je zobrazen pouze zástupný IObjectDAO, který v implementaci neexistuje a je nahrazen všemi DAO třídami (ISpellDAO, IEffectDAO atd.).
ITreeDAO se stará o veškeré ukládání stromové struktury do databáze.
IMapDAO se stará o ukládání veškerých dat, které se týkají map. Rozhraní je společné pro všechny druhy map (grafické, jednoduché, obrázkové)
ILangDAO se stará o ukládání jazyků, které jsou v aplikaci použity. Nejedné se o překlady prezentační vrstvy, ale o jazyky vytvořené pro překlady šablon. Rozhraní se nestará o ukládání samotných překladů, pouze o jejich jazyky.

IObjectManager navazuje na IObjectDAO. Jedná se opět o skupinu tříd pro veškeré základní objekty (ISpellManager, IAbilityManager, atd.). Rozhraní definuje jak přijímá data z prezentační vrstvy. Základní funkcionalita spočívá ve zpracování dat z prezentační vrstvy a vytvoření kompletního objektu, se kterým se dále pracuje nebo pošle na datovou vrstvu pro uložení.
ITreeManager se stará u připravení stromové struktury pro zobrazení. Vytváří z objektů stromovou strukturu a přidává do struktury metadata, které slouží pro zobrazení a práci na prezentační vrstvě.
IMapManager se stará o zpracování dat ohledně mapy. Převádí zobrazenou mapu pro uživatele na formu, ve které se dá snadno uložit do databáze. Pro-

pojuje veškeré šablony s mapou.
ILangManager se stará o zpracování dat z prezentační vrstvy ohledně jazyků.
ITabManager se stará o vytváření překladových záložek pro prezentační vrstvu. Toto rozhraní nemá DAO rozhraní, protože veškeré informace které potřebuje, získá přímo z konkrétních IObjectDAO tříd. Jejím hlavním úkolem je zpracování všech překladů jedné šablony a připravení objektů pro uložení.

Kapitola 4

Návrh

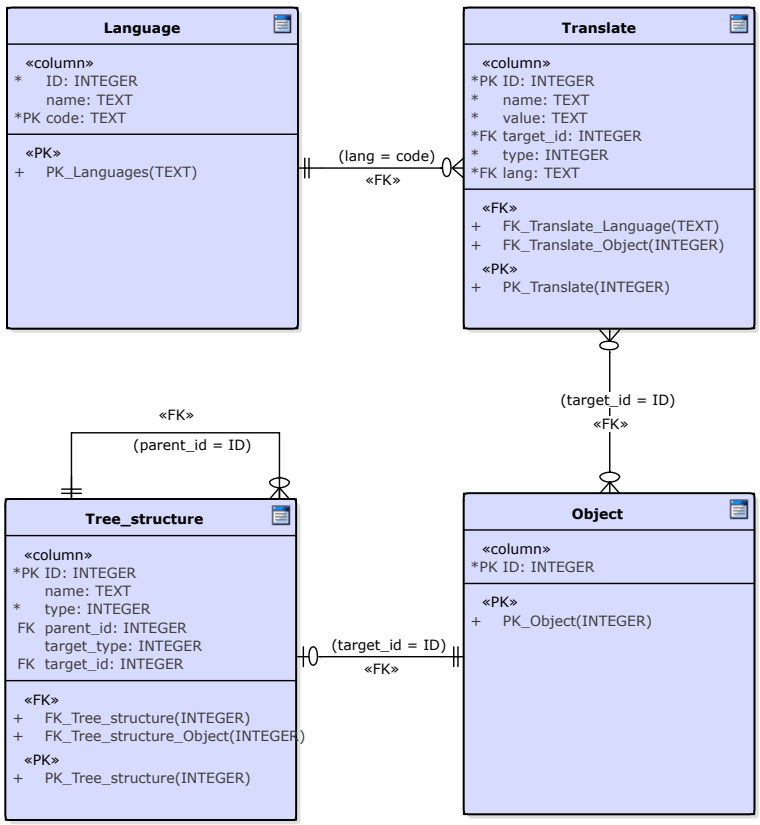
4.1 Model databáze

Pro realizaci aplikace DeskChar jsem vybral databázi sqlite, která je jednoduchá a nepotřebuje žádný složitý externí software pro běh. Na druhou stranu zvládá veškeré potřebné operace, jako jsou cizí klíče a kaskádové mazání záznamů. Pro vytváření a práci se stromovou strukturou, je kaskádové mazání nedocenitelný pomocník.

Diagram na obrázku 4.1 je zjednodušený pro větší přehlednost. Na diagramu je zachycena základní struktura a princip závislostí v databázi. Kompletní model databáze se nachází v příloze.

Z důvodu vícejazyčnosti šablon, bylo zapotřebí navrhnout strukturu databáze, která dokáže uložit neomezený počet jazykových překladů. Na obrázku 4.1 můžeme vidět základní strukturu databáze. Tabulka Languages obsahuje záznamy o jazycích. Údaj o jazyku není pouze textový v tabulce Translate, abychom docílili konzistence mezi záznamy pro stejný jazyk. Primárním klíče každého jazyka je textový kód, který je unikátní. V tabulce Translate pak nalezneme veškeré textové řetězce, které se nacházejí v objektech. Jeden záznam tabulky translate je přiřazen ke konkrétnímu objektu pomocí dvojice klíčů target_type, který určuje o jaký objekt (tabulku v databázi) se jedná, a target_id, který určuje konkrétní záznam v tabulce (ukazuje na primární klíč ID v tabulce). Informace o překladu jsou uloženy ve dvojici klíčů name a vulue, které slouží jako slovník, jejich jazyk určuje záznam lang. Takto zvolený návrh databáze umožňuje neomezený počet jazyků a jednoduchou rozšířitelnost.

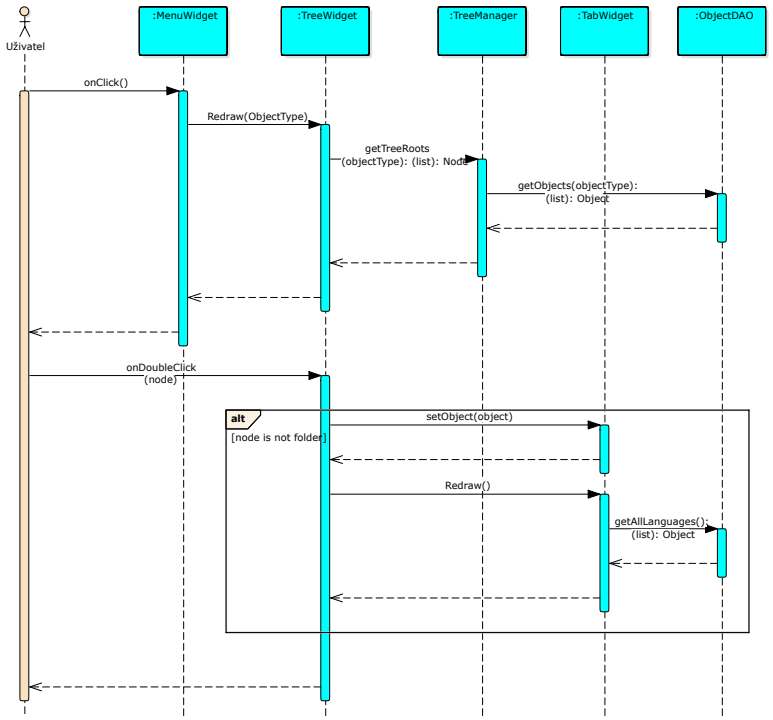
Druhá část diagramu se týká stromové struktury, která se používá pro všechny objekty v aplikaci. Tabulka Tree_structure slouží pro zaznamenávání stromové struktury pro veškeré objekty. Do tabulky se ukládají dva druhy uzlů, složky a objekty. Atribut type určuje, o který druh uzlu se jedná. Pomocí



Obrázek 4.1: Základní mode databáze

cizího klíče parent_id, vzniká stromová struktura. Zde se využívá kaskádové mazání, pokud se smaže záznam, který má pod sebou navázané další záznamy pomocí cizího klíče, smažou se tyto záznamy také automaticky v databázi. Tento princip udržuje tabulku konzistentní a nevznikají žádné záznamy, které se již nepoužívají. Každý uzel který je typu object, ukazuje pomocí dvojice klíčů target_type, který určuje tabulku v databázi, a target_id na záznam v ostatních tabulkách. V diagramu je to naznačené zjednodušeně, kde tabulka Object zastupuje tabulky všech objektů (Spell, Item, atd.).

Poslední problém návrhu databáze se skrýval v návrhu ukládání vazeb mezi jednotlivými objekty. V úvahu připadali dvě možnosti. První možnost byla zachytit veškeré vazby mezi objekty pomocí tabulek „relations“, které by obsahovaly pouze dvojici klíčů z obou tabulek vazby jako cizí klíče. Toto řešení by bylo rychlé a jednoduché na používání, bohužel se u některých vazeb nedalo použít samotné. Například u dobrodružství se podřazené objekty můžou sdružovat dále do složek, čehož by se pouze pomocí těchto vazeb nedalo docílit. Musela se zde navíc využít stromová struktura použitá pro všechny šablony

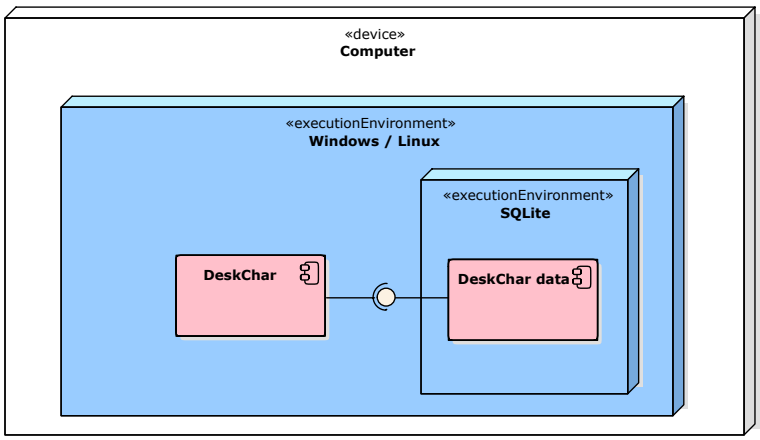


Obrázek 4.2: Model komunikace pro hlavní obrazovku

a jejich základní rozřazení, čímž by vznikali duplicitní data. Duplicitní data sebou nesou dva základní problémy. První z nich je samozřejmě větší množství dat, které je potřebné uložit. V tomto případě by se však nejednalo o drastický nárůst, který by dělal problém. Druhý a závažnější problém se týká aktuálnosti dat. Bylo by zapotřebí udržovat oboje údaje aktuální a stejné, což by mělo velké nároky na složitost ukládání a zpomalovalo by to některé operace, které je zapotřebí aby aplikace prováděla okamžitě (například drag and drop rozřazování). Z těchto důvodů jsem se rozhodl zvolit druhou možnost, kde veškeré vazby mezi objekty jsou uloženy pouze ve stromové struktuře popsané výše. Problém bude vznikat při operacích exportu, kde bude vazbu mezi objekty dohledávat ze stromové struktury. Tyto operace se však neprovádí tak často a není zde velký důraz na okamžitou odezvu.

4.2 Model komunikace

Nejčastější operaci prezentační vrstvy je práce se stromovou strukturou a vytváření překladů. Pro znázornění komunikace mezi vrstvami jsem vytvořil diagram komunikace, který znázorňuje komunikaci mezi jednotlivými třídami prezentační vrstvy. Na diagramu 4.2 můžeme vidět základní dvě činnosti. Vy-



Obrázek 4.3: Model nasazení

brání konkrétního typu šablon , které zobrazí stromovou strukturu a zobrazení hlavního widgetu pro vytváření šablon po vybrání konkrétní položky ve stromové struktuře.

Uživatel v menu klikne na požadovaný druh šablon. Tím se zavolá funkce Redraw, která má na starosti vykreslení stromové struktury ve widgetu. Potřebná data získá z TreeManageru po zavolání funkce getTreeRoots, která má na starosti, vytvoření kompletního stromu uzlů, přičemž vrací seznam kořenových uzlů. TreeWidget z tohoto listu pomocí rekurze vykreslí celý strom. TreeManager získává data z konkrétní DAO třída (v diagramu zjednodušena jako ObjectDAO). Objekty navěšené na uzly stromu můžou být různé, takže třída TreeManager volá více DAO tříd.

Druhá část diagramu se týká vykreslení hlavní části obrazovky, kde se vytváří samotné šablony a jejich překlady. Uživatel v TreeWidgetu dvakrát klikne na nějaký uzel. Pokud se jedná o složku, funkce zavolá pouze rodičovskou funkci TreeWidgetu, což má za následek rozbalení obsahu složky. Pokud se jedná o objekt zavolá se nad třídou TabWidget funkce setObject s parametrem object. Hned poté se zavolá funkce, které vykreslí všechna data. Data získá na základě nastaveného objektu, který si pamatuje vlastní DAO třídu. Třída vrátí seznam objektů, přičemž se jedná o jednu šablonu, ale ve všech jazycích, ve kterých byla vytvořena. TabWidget následně pro každý jazyk vykreslí jednu záložku, kde se dají hodnoty šablony upravovat nebo případně vytvořit nový překlad pro nový jazyk.

4.3 Model nasazení

Diagram nasazení zobrazuje způsob rozdělení systému na samostatné části a komunikační vazby mezi nimi, čímž definuje architekturu systému. Nalezneme zde veškeré komponenty systému a všechny potřebné části pro běh našeho systému.

Na diagramu 3.8 můžeme vidět, že software je určený pro počítače a funkční pod operačními systémy Windows a Linux (verze operačního systému jsou uvedeny v sekci nefunkční požadavky). Aplikace je napsaná v pythonu a zabalená do spustitelného balíčku, který nevyžaduje žádné speciální požadavky na systém. Veškeré potřebné knihovny a prostředí má uložené v balíčku. Databáze sqlite běží v rámci tohoto balíčku také a nepotřebuje žádné dodatečné prostředí.

Kapitola 5

Realizace

5.1 Použitý programovací jazyk

5.2 Využité knihovny

5.2.1 Knihovna pro práci s XML

Velká část programu se týká práce s xml šablonami. Python v základu neumí s xml šablonami pracovat. Proto bylo zapotřebí vybrat vhodnou knihovnu, která by veškerou práci co nejvíce usnadnila. Hledal jsem knihovnu, která by dokázala namapovat třídu a její atributy přímo na xml šablony. Bohužel taková knihovna pro Python neexistuje. Proto jsem využil knihovnu lxml, která umí základní parsování xml souborů a napsal jsem pro ní rozšíření, které umožňuje mapovat objekty přímo na strukturu xml souboru. V ukázce kódu 5.1 můžeme vidět namapování objektů Effect a Modifier a jejich vzájemné provázání.
Rozšíření obsahuje tři základní třídy, XElement, XAttribElement a XInstance. Každá z těchto tříd reprezentuje jiný typ atributu v objektu. Třída XElement reprezentuje základní atribut, který nemá překlady. Jako parametr, kromě jména tagu, přijímá volitelný atribut enum, kterým se dá nastavit, že hodnota se má mapovat na konkrétním enum.
Třída XAttribElement reprezentuje tagy, které mohou navíc obsahovat atribute. Nejčastěji se to využívá u atributů, které jsou vícejazyčné. Atributem lang se zde nastavuje jazyk překladu.
Poslední třída XInstance reprezentuje vazbu na další mapovací třídu. Jako parametr komě jména, přijímá instanci cílové mapovací třídy.

Třída XMLTemplate, ze které všechny mapovací třídy dědí, obsahuje dvě základní funkce, import_xml a create_xml. Funkce import_xml načte xml soubor a vrátí seznam objektů, které se dále načítají do databáze. O proti tomu třída create_xml přijímá seznam objektů, ze kterých vytvoří strukturovaný xml soubor.

Listing 5.1: Mapování objektů na xml strukturu

```
class XMLModifier(XMLTemplate):
    ROOT_NAME = 'modifier'
    OBJECT_TYPE = ObjectType.MODIFIER


    def __init__(self):
        self.id = XElement('id')
        self.valueType = XElement('valueType', ModifierValueTypes)
        self.value = XElement('value')
        self.targetType = XElement('targetType')
        self.valueTargetAttribute = XElement('valueTargetAttribute')


class XMLEffect(XMLTemplate):
    ROOT_NAME = 'effect'
    OBJECT_TYPE = ObjectType.EFFECT


    def __init__(self):
        self.id = XElement('id')
        self.name = XAttribElement('name', 'lang')
        self.description = XAttribElement('description', 'lang')
        self.targetType = XElement('targetType')
        self.modifiers = XInstance('modifiers', XMLModifier)
```

5.2.2 Grafická knihovna

Pro python existuje velké množství grafických knihoven. Některé jsou zaměřené na mobilní aplikace, některé hlavně pro webové rozhraní. Vybrat mezi takovým množstvím knihoven není lehké. Mezi nejznámější patří knihovny Kivy, PyGame, TkInter a PyQt. Knihoven existují desítky další, ale v této bakalářce se věnuji pouze nejzajímavějším z nich.
Knihovna kivy je převážně zaměřené na dotykové displeje. Výsledný vzhled a veškeré widgety jsou pro to přizpůsobené a ovládání pomocí myši a klávesnice není tak intuitivní jako u ostatních knihoven.
Dalším důležitým kritériem pro výběr knihovny byla aktuálnost a vydávání nových aktualizací. Pro knihovna TkInter již dlouho nevychází nové aktualizace. Knihovna je velice jednoduchá a intuitivní, bohužel již několik let není aktuální.
Knihovna PyGame, jak již napovídá její název, se specializuje převážně na tvorbu počítačových her. Má rozsáhlé možnosti animací a herních prvků, které

jsou důležité převážně ve hrách. Knihovna neumí vytvářet přehledné grafické rozhraní aplikace, proto jsem ji ze seznamu také vyřadil.
Poslední z čtveřice knihoven je PyQt. Knihovna se zaměřuje na tvorbu přehledných grafických rozhraní pro programy, což je přesně to co je zapotřebí. Navíc má velice moderní a přehledný vzhled, což bylo jedno z hlavních kritérii pří výběru. Aktualizace k této knihovně jsou vydávány pravidelně a nejnovější verze PyQt5 je určena pro Python 3. Proto jsem se rozhodl zvolit tuto knihovnu pro můj program.

5.2.3 Knihovna pro tvorbu PDF almanachů

Závěr

Příloha **A**

Seznam použitých zkratek

HnH Hra na hrdiny

DrD Dračí doupě

RPG Role-playing game

PJ Pán jeskyně

DAO Data access object

MobChar Mobile character

DeskChar Desktop character

Příloha **B**

Obsah přiloženého CD/USB

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `thesis.ps` ..... text práce ve formátu PS