FAKULTA INFORMAČNÍCH TECHNOLOGIÍ ČVUT V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Vytvoření počítačové verze deskové hry v jazyce Java
Student:	Igor Tsaregorodtsev
Vedoucí:	Ing. Josef Pavlíček, Ph.D.
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2019/20

Pokyny pro vypracování

Vytvořte softwarovou verzi deskové hry (dle výběru - tanky, letadla, rybolov, pouštní rallye - vše dle návrhu školitele a grafika Ing. Petra Netíka).
- Navrhněte základní kostru hry formou diagramu tříd (UML)
- Navrhněte způsoby hry (sám proti počítači, dva u počítače, více hráčů na serveru)
- Hru naprogramujte dle pravidel JEE pro běh na aplikačním serveru a s využitím prezentační vrstvy formou tenkého klienta (Vaadin framework , či jiný dle výběru a konzultace se školitelem)
- Hru jako funkční blok umístěte na aplikační server běžící v prostředí firmy USPIN.cz

Seznam odborné literatury

Dodá vedoucí práce.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

doc. RNDr. Ing. Marcel Jiřina, Ph.D.
děkan

V Praze dne 29. prosince 2018

FAKULTA
INFORMAČNÍCH
TECHNOLOGIÍ
ČVUT V PRAZE

Bakalářská práce

Vytvoření počítačové verze deskové hry v jazyce Java

Igor Tsaregorodtsev

Katedra softwarového inženýrství
Vedoucí práce: Ing. Josef Pavlíček, Ph.D.

25. června 2019

Poděkování

Srdečně děkuji své rodině za podporu během celého studia. Děkuji také panu Ing. Josefu Pavlíčkovi, Ph.D. za cenné rady a vedení této práce. Také bych rád poděkoval ČVUT a zejména Fakultě informačních technologií za možnost zde studovat a za nádherné učební osnovy a učitele.

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, avšak pouze k nevýdělečným účelům. Toto oprávnění je časově, teritoriálně i množstevně neomezené.

V Praze dne 25. června 2019 .....

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2019 Igor Tsaregorodtsev. Všechna práva vyhrazena.
Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí a nad rámec oprávnění uvedených v Prohlášení na předchozí straně, je nezbytný souhlas autora.

Odkaz na tuto práci

Tsaregorodtsev, Igor. *Vytvoření počítačové verze deskové hry v jazyce Java.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2019.

Abstrakt

Během této práce se autor zabývá vývojem herní webové aplikace, která je klonem deskové hry. Předtím než začít vývoj, autor analyzuje technologie sloužící k vytváření webových her a historii jejich vzniku. Samotný vývoj probíhá v jazyce Java s využitím technologií Java EE (Java Enterprise Edition) a podle principů unifikovaného procesu. Každý krok tohoto procesu je popsán a aplikován ke stávajícímu projektu.

Klíčová slova Java, Java EE, webová hra, desková hra, historie webových technologií, unifikovaný proces

Abstract

During this work the author develops a web game application, which is a board game clone. Before starting the development, the author analyzes technologies used to create web games and the history of their creation. The development itself takes place in Java using Java EE (Java Enterprise Edition) technologies and according to the principles of a unified process. Each step of the process is described and applied to the existing project.

Keywords Java, Java EE, web game, board game, web technologies history, unified process

Obsah

Seznam obrázků

Úvod

V nynější době si nelze představit život dnešního člověka bez internetu. Internet pronikl do většiny životních sfér a se na tom neplánuje zastavit. Díky tomu se objevuje množství technologií, které pomáhají ho měnit a zpříjemňovat. Webové zaměření se dnes považuje za jedné z nejvyžádanějších a nejpopulárnějších zaměření mezi obory IT (Informační technologie). Avšak autora láká nejen rychlé uplatnění svých znalostí, ale možnost být na hranici rozvoje webových technologií: orientovat se v jejich aplikaci, znát silné a slabé stránky a možnost vymyslet něco vlastního.

Jedním z možných způsobů aplikace posledních novinek technologií je vývoj her, a právě tím je zajímavé téma této práce. Ačkoli webové hry zabírají jenom malou část herního trhu, jsou skvělým příkladem neobvyklého použití technologií, které původně byly určeny pouze k textovému předávání informace. V důsledku takové hry postrčily webový prostor na rozšíření svých možností.

Nicméně nestačí jenom vědět, jak se technologie aplikují – ve větších projektech je potřeba dodržovat správnou architekturu, plán vývoje a dokumentaci, protože je počet tříd a závislostí mezi třídami v takových projektech poměrně velký. Při špatném přístupu je velice těžko se orientovat, a nové vývojáře jsou nuceni ztrácet hodně času, aby se vyznali v logice napsaného kódu. Aby se takovým situacím předešlo, je velmi důležité mít celivost a jedinou strukturu práce. V dnešní době existuje množství metodik vývoje softwaru, ale všechny jsou založeny na jediných základních principech unifikovaného procesu.

Kapitola 1

Cíl práce

Zásadním cílem této práce je vývoj webové herní aplikace pomocí získaných během studia zručností, které odpovídají oboru autora. Pro zaměření „webové a softwarové inženýrství“ hlavními takovými zručnostmi jsou: dovednost analyzovat a vyvíjet software v rámci unifikovaného procesu, chápání principů a fází tohoto procesu a ovládání příslušnou sadou technologií pro vyřešení zadaných úkolů.

Unifikovaný proces odpovídá na otazky „kdo“, „co“, „kdy“ a „jak“ ve všech fázích vývoje jakéhokoli softwaru.[1, s. 23] Celý proces lze rozložit na tyto body:

1. specifikace požadavků,
2. analýza,
3. návrh,
4. implementace,
5. testování. [1, s. 30]

Ve stejném pořadí budou sestavené i kapitoly praktické části této práce, ve kterých každý jednotlivý bod bude podrobněji rozepsán a aplikován ke stávajícímu projektu. Implementačním projektem byla zvolena desková hra „Poušťní Rallye“, popis a pravidla které budou popsány v příslušné kapitole. Během vývoje této hry autor prokáže schopnosti aplikace unifikovaného procesu v reálném projektu, kde popíše a argumentuje své kroky.

Předtím než začít vyvíjet aplikaci, autor analyzuje existující webové technologie, pomocí kterých se vyvíjejí webové hry. Bude také podrobně popsána jejich historie vzniku a zániku, důvody k tomu a principy fungování.

Kapitola 2

Rozvoj webových her a technologií

2.1 Pojem a důvody vzniku

Historie počítačových her jde skoro spolu s historii digitálních počítačů. Důvodem k tomu slouží nejen zábava – hry se dokonale hodí k testování nových technologií a jejich hraničních možností, což častokrát přivádí na nové nápady zdokonalení. Podobně se stalo s WWW (World Wide Web neboli „celosvětová síť“) a webovými prohlížeči, které poprvé veřejně objevil Tim Bernes-Lee v roce 1991.[2] Během několika následujících let celosvětová síť opravdu získala celosvětovou známost. Toto soustředilo úsilí mnohých lidí a firem na vytvoření svých prohlížečů a rozvíjení technologií v této oblasti, což zanedlouho přivedlo i k vzniku prohlížečových her.

Prohlížečové neboli webové hry nevyžadují od uživatele instalaci plnohodnotné hry, jak to bývá s obyčejnými počítačovými hrami, mnohé věci se stahují již během herního procesu nebo před jeho začátkem. Stačí navštívit příslušnou webovou stránku pomocí jakéhokoli prohlížeče a případně mít nainstalovaný potřebný plug-in. Takové hry nemají vysokou kvalitu grafiky a rychlý výkon na rozdíl od klasických her, ale mají lehkou dostupnost a nízký práh vstupu pro vývojáře. Webové hry získaly všeobecnou uznání svojí prostotou a rozmanitostí.

2.2 První technologie

2.2.1 HTML a CSS

HTML neboli HyperText Markup Language je základní technologií webu, pomocí které prohlížeče zobrazují webové stránky. Webové stránky jsou představené HTML soubory, které prohlížeče jsou schopné interpretovat a správně

zobrazit podle pravidel standardů. HTML také založil Tim Bernes-Lee spolu s WWW, ale první oficiální standard byl vydán až v roce 1995.[3]

Všechny další webové technologie souvisejí s HTML nebo ji rozšiřují. Příkladem může sloužit technologie CSS (Cascading Style Sheets), která rozdělila obsah HTML dokumentu od popisu jeho designu. První verze standardu CSS byla vydána v roce 1996.[4]

2.2.2 Java Applet

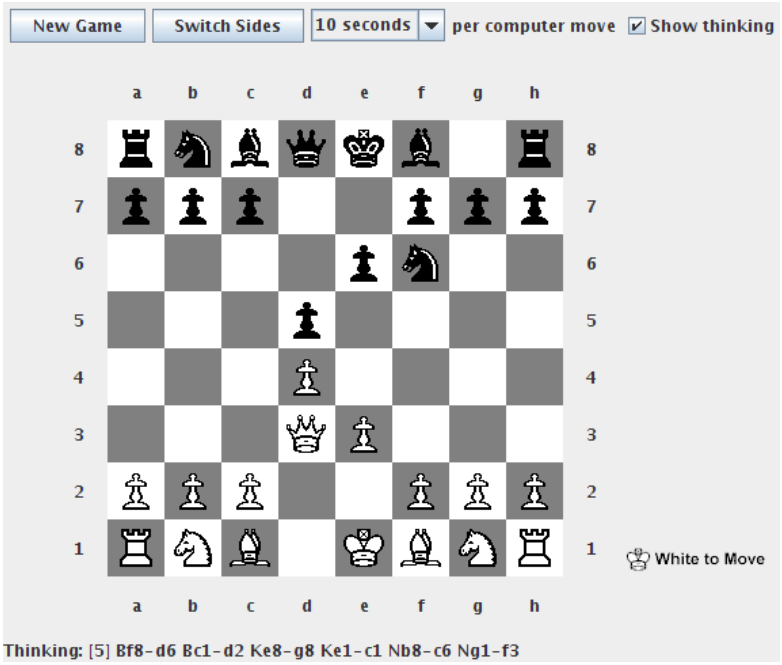
První veřejnou technologií mimo HTML pro rozvoj webového průmyslu se stal programovací jazyk Java. V květnu roku 1995 se objevila jeho debutová alfa verze spolu s technologií Java Applet, která umožňovala spouštět Java kód v jednotlivém okně prohlížeču, v případě že uživatel má nainstalované JVM (Java Virtual Machine) a příslušný plug-in.[5]

Nová technologie dovolila vývojářům vytvářet hry v „appletech“, ale takový způsob nebyl zbaven nedostatků – kreslení grafiky se mohlo provádět jenom pomocí knihovny AWT (Abstract Window Toolkit). Knihovna AWT je standardní knihovna jazyka Java a především je určená k vytváření uživatelských rozhraní, avšak mimo to poskytuje základní funkčnost pro práci s 2D grafikou: kreslení teček, přímek, obdélníků, kruhů, výběr barvy prvků atd.[6] Vyvíjet hry takovým způsobem bylo velmi obtížné – potenciálním vývojářům bylo potřeba již mít zkušenosti v programování, naučit se nový jazyk, orientovat se v teorii počítačové grafiky a mít dost času, aby se podařilo napsát něco zajímavého. Podobné podmínky nemohli dovolit této technologii okamžitě získat úspěch ve sféře her. I když v budoucnu po zlepšení a vydání nových knihoven na internetu začly vznikat hry napsané s využitím Java Applet, k tomu času všebecné uznání již získal úplně jiný způsob vývoje...

2.2.3 JavaScript

Zanedlouho po Javě se firma Netscape rozhodla vytvořit vlastní technologii pro zlepšení interaktivity webových stránek, která by nevyžadovala od vývojářů hlubokých znalostí programování na rozdíl od Javy. Takovou novou technologií se stal jazyk JavaScript. Oficiální první verze jazyka byla vydána firmou Netscape ve svém vlastním prohlížeči Netscape Navigator 2.0 beta 3 v prosinci roku 1995, ale první fungující koncept byl představen ještě před pár měsíci téhož roku.[8]

JavaScript dovolil přidat rozmanitost do statických stránek. Spolu s HTML dokumentem se stahoval speciální skript, který mohl animovat elementy, reagovat na uživatelský vstup, dinamicky měnit barvu elementů a mnohé jiné.[8] Jeho jednoduchost umožňovala skoro každému, kdo se alespoň trochu orientoval v programování, dělat z obyčejných stránek dinamické webové aplikace a hry.



Obrázek 2.1: Příklad hry v Java Applet „ChessApp“.[7]

2.2.4 Flash

V prosinci roku 1996 firma Macromedia vykoupila technologii FutureSplash od firmy FutureWave Software, dodělala to a vydala jako Macromedia Flash 1.0. Flash se skládal ze dvou částí: nástroj na vytváření grafiky a animace Macromedia Flash a nástroj na přehrávání této animace Macromedia Flash Player, který byl natolik lehký, že z něho byl vytvořen volně dostupný prohlížečový plug-in.[9]

Po instalaci tohoto plug-inu prohlížeč uživaele mohl přehrávat Flash soubory, ve kterých se později dalo kódovat nejen Flash animaci, ale i audio a video.

2.3 První hry a epocha Flash

Webový herní průmysl šel skoro spolu s technologiemi, první populární webové hry se objevily v létě roku 1996 a byly napsány v čistém HTML. To byly „Club a Seal“ a „Assassin“ vytvořené americkým studentem Tomem Fulpem.[10]

Popularita svých her postrčila Toma k dalšímu vývoji v této sféře. Po několika letech Toma zaujala technologie Flash, která již k tomu času byla vylepšena: pracovníci Macromedia přidali Actions, pomocí nichž se dalo zadat pořadí animovaných scén v závislosti od stisknutého na obrazovce tlačítka, zjednodušili vytváření animace a doplnili nástroje pro grafiku.[11]

Obrázek 2.2: Příklad Flash hry „Gold Miner“.[13]

Všechno to přivedlo Toma na vytvoření jedné z prvních herních webových stránek Newgrounds, kde lidé mohli zahrát, vyměňovat si zprávy, publikovat své animace, obrázky, a dokonce své vlastní hry.[10] Tento okamžik se fakticky stal počátečním bodem epochy Flash her. Prostota vytváření obrázků a animace dovolila doslovně každému vyzkoušet sebe v webovém herním průmyslu: stačilo nakreslit jednoduché obrázky, animovat je a přidat k tomu Actions. V budoucnu počet herních stránek všude vzrůstal, rovněž jako i množství hráčů.

Macromedia se svou technologií také nezůstali na místě: vývojáře vyvíjeli nejen nástroje na kresbu a animaci, ale i na hry. Z Actions se vyvinul plnohodnotný skriptový jazyk ActionScript: přidalo se čtení uživatelského vstupu a možnost psání scénářů animací objektů.[11] Tímto způsobem se dalo uvést do života skoro jakoukoli ideji. Paralelně s tím Flash Player se stal jediným standardem přehrávání audio a video na webových stránkách a tudíž nezadatelným plug-inem každého uživatele. V roce 2005 Flash Player byl nainstalován na 98 % připojených k internetu počítačích.[12]

2.4 Situace v dnešní době a další cesty rozvoje

Od roku 2013 většina prohlížečů začly aktivně podporovat všeobecný nový standard HTML5, který pak později byl oficiálně vydán v roce 2014. S pomocí HTML5 v prohlížečích vznikla možnost přehrávání různých audio a video formátů, významně se zlepšila efektivita JavaScriptu a byly představeny nejdůležitější věci pro herní průmysl: element Canvas a podpora protokolu WebSocket. Díky Canvasu se uskutečnila nativní podpora vektorové a 3D

Obrázek 2.3: Hra napsaná pomocí HTML5 a JavaScript „BrowserQuest“.[16]

grafiky, a pomocí WebSocketu se objevila možnost vytvářet „real-time“ hry pro více hráčů.[14]

Všechno to znamenalo konec technologie Flash, jelikož nové standardy kompletně pokrývají všechno, co umí Flash Player, dělají to lépe a mají větší podporu ze strany mobilních zařízení. Na mobilním trhu Flash Player vůbec nebyl oceněn, protože spotřeboval hodně zdrojů pro svoji práci. Firma Apple byla první, kdo odmítla tuto technologii ku prospěchu otevřeným standardům.[15] V dnešní době HTML5 spolu s JavaScript jsou dominantními technologiemi ve webové sféře. Flash se považuje za mizející a zastralý, jeho podpora skončí v roce 2020.[17] Stejně to tak i u technologie Java Applet, která byla přiznaná zastralou ve verzi Java 9 a odstraněna ve vezi Java SE 11.[18, 19] Většina projektů napsaných na Flash je již přeložená do HTML5, na to existuje množství nástrojů.[9] Převážné procento moderních webových her se teď píše na JavaScript. Vytváření podobných her není už takové lehké, avšak existuje obrovský počet doplňujících knihoven a frameworků, které usnadňují život programátorům. Vzrostl také i jejich potenciál: objevily se celé herní engine pro vývoj velkých a víceúčelových projektů. Takové nástroje slouží ke kreslení složité 3D grafiky pomocí technologie WebGL, podpora které byla také přidána v HTML5. WebGL umožňuje nejen práci s 3D objekty, ale ještě využití zdrojů grafické karty pro zrychlení výpočtů a zlepšení výkonu, což velmi rozšiřuje hranice možností prohlížečů.[20]

Podle autora hlavním cílem dalšího rozvoje webových technologií bude optimizace a vylepšení kvality 3D grafiky v prohlížečích. S příchodem WebGL již více a více webových stránek začínají integraci s 3D, to dává příležitost vyvolat unikátní dojem a nechat přání navštívit stránku ještě jednou. Určitě

takový rozvoj bude mít dopad také i na herní průmysl: dobrý výkon a kvalita dovolí mnohým firmám, které se zabývají vývojem videoher, znovu vydat své staré hry již na webu. Takovým způsobem staré hry mohou získat novou popularitu, a nostalgující hráči uvidí své oblíbené produkty v novém světle.

Kapitola 3

Hra „Pouštní Rallye“

„Pouštní Rallye“ je jednoduchá desková hra, autoří které jsou Ing. Petr Netík a Ing. Josef Pavlíček, Ph.D. Hru maximálně mohou zahrát čtyři hráči. Cílem každého hráče je dostat se prvním na finální pozici. Na začátku jsou všechny figurky umístěné na startovacích pozicích. Tahy se dělají popořádku hazením kostkou a přemístěním figurky na další pozici. Hráč má na výběr dva typy kostek „riskantní“ a „standardní“, ze kterých může na svém tahu hodit jenom jednu: na první může vypadnout čísla 4 až 7, na druhé může vypadnout 2 až 10. Čísla na kostce označují počet kroků uživatele na daném tahu. Jakmile hráč zjistí počet kroků, musí se přemístit na právě stejný počet pozic. Přemístění se uskutečňuje po dlaždicích směrem dopředu nebo na sousední dlaždici. Během jednoho tahu figurka může projít jednu dlaždici právě jednou.

3.1 Herní mapa

Herní mapa se skládá ze čtyř typů dlaždic, po kterých se hráče pohybují. Celá mapa je na obrázku 3.1.

Jednoduchá dlaždice Název mluví sam za sebe. Z těchto dlaždic se skládá větší část mapy. Nic se neděje, když hráč zaujme tuto pozici. Označují se krémovou barvou.

Zkratka Když hráč zaujme vchod na zkratku, může v dalším tahu jet kratší cestou, avšak nemusí. Označují se růžovou barvou.

Dlouhá cesta Když hráč zaujme vchod na dlouhou cestu, v dalším tahu musí jet pouze dlouhou cestou. Označují se černou barvou.

Havarijní dlaždice Když hráč zaujme takovou pozici, v dalším tahu nehází kostkou a ostatní hráče nemohou ho předjet. Označují se červenou barvou.

Obrázek 3.1: Originální mapa hry „Pouštní Rallye“.

Kapitola 4

Specifikace požadavků

Specifikace požadavků je prvním a nejdůležitějším krokem při vytváření jakéhokoli softwarového produktu. Předtím než začít modelovat objekty v systému, je třeba pochopit, k čemu ten systém bude sloužit a jaké bude plnit funkce. Dodavatel softwaru se přesně musí dohodnout se zákazníkem na správné funkčnosti a omezeních systému, aby se nevznikly rozporné požadavky a v důsledku i zbytečná práce.

4.1 Funkční a nefunkční požadavky

V dnešní době již tradičním způsobem zachycení požadavků je nalezení funkčních a nefunkčních požadavků.

- **Funkční** požadavky určují, co aplikace musí a nemusí umět a jaké musí mít chování.[1, s. 49]
- **Nefunkční** požadavky určují, jaké vlastnosti musí mít aplikace a jaká omezení musí splňovat (výkonnost, stabilita, škalovatelnost, bezpečnost atd.).[1, s. 49]

Takové požadavky definují hranice systému a musejí splňovat kriteria jednoznačnosti, splnitelnosti a ověřitelnosti.[21]

4.1.1 Požadavky hry „Pouštní Rallye“

Po zkoumání zadání práce a diskuzi s vedoucím práce byly definovány následující požadavky.

4.1.1.1 Funkční požadavky

F1. Identifikace uživatele Hra musí identifikovat uživatele podle jména.

F2. Podoba deskové hře Hra musí být podobná deskové hře: hráči dělají tahy popořádku, jeden tah se dělá hazením kostkou a výběrem nejdalší pozice, do které je možné se dostat směrem dopředu v závislosti od čísla, které vypadlo na kostce.

F3. Dva typy kostek Hra musí mít dvě možnosti hodit kostkou: standardní a riskantní. Při standardním hodu mohou vypadnout čísla 4 až 7, při riskatním hodu může vypadnout 2 až 10.

F4. Dlouhé cesty Hra musí mít dlouhé cesty označené černou barvou. Když se hráč dostane na začátek dlouhé cesty, musí jet pozue dlouhou cestou.

F5. Krátké cesty Hra musí mít krátké cesty označené růžovou barvou. Když se hráč dostane na zkratku, může jet kratší cestou.

F6. Blokující pozice Hra musí mít blokující pozice označené červenou barvou. Když se hráč dostane na takovou pozici, jedno kolo nehází kostkou a blokuje trať. Hráči za ním nemohou ho předjet.

F7. Počet figurek na místě Hra musí mít pravidlo, že na jednem místě může stát jenom jeden hráč.

F8. Podpora hry více hráčů Hra musí podporovat možnost zahrát s jinými uživateli. Jeden hráč zakládá hru a sděluje společný klíč, ostatní se pomocí známého klíče připojují.

F9. Podpora hry jednoho hráče Hra musí mít možnost samostatné hry s umělou inteligenci, v takovém prípadě hráč musí mít možnost vybrat počet botů, které budou kontrolované počítačem.

4.1.1.2 Nefunkční požadavky

NF1. Implementační jazyk Hra musí být napsaná v programovacím jazyce Java.

NF2. Prezentační vrstva Prezentační vrstva hry musí být napsaná formou tenkého klienta s využitím frameworku Vaadin.

NF3. Nasazení Výsledný produkt musí být v takovém tvaru, aby se dalo nasadit na aplikační server Jave EE.

NF4. Dostupnost přes web Aplikace musí být umistěná na serveru, který je dostupný přes web.

NF5. Podpora českého jazyka Všechny texty musejí být orientované na českého uživatele. Dovoluje se podpora jiných jazyků pouze s možností přepínání do češtiny.

NF6. Stabilita Aplikace musí fungovat stabilně, bez chyb, které by mohly vadit hernímu procesu. V případě objevení nějaké interní chyby, chyba musí být nahlašená uživateli s podrobným popisem.

4.2 Případy užití

Případy užití neboli „use cases“ jsou doplňujícím způsobem zachycení požadavků kladených na systém, které pomáhají detailněji popsat, co chce zákazník a jak by měl ten systém reagovat na jednotlivé kroky každého aktéru v systému. Dřív to byl jediný způsob zachycení požadavků, ale ten nezahrnuje nefunkční část, což nepopisuje produkt kompletně.

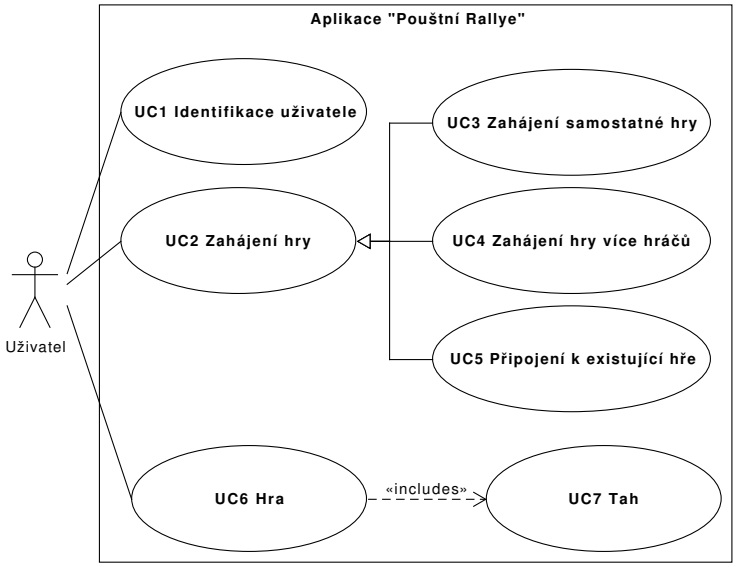
Pomocí případů užití se popisují scénáře akcí uživatele a systému pro dosažení určitého výsledku. Tabulka popisu případu užití se skládá z názvu, ID, aktérů, předběžných podmínek, scénáře nebo toku událostí a důsledku. Případy užití si pomáhají představit prvky budoucího systému.[1, s. 57, 64]

4.2.1 Případy užití hry „Pouštní Rallye“

Prvním důležitým krokem při modelování případů užití je nalezení aktérů, které se zúčastní procesů budoucího systému a provádějí v něm nějaké akce. V daném případě uživatel aplikace neboli hráč je jediným takovým aktérem. V druhém kroce je potřeba definovat a popsat samotné procesy na základě již známých funkčních požadavků. Všechny tabulky byly vytvořeny pomocí nástroje UMLet.[22]

4.2.1.1 Diagram případů užití

Po analýze existujících funkčních požadavků hry byly definovány následující případy užití, které jsou zobrazeny na obrázku 4.1. Takový diagram zobrazuje vztahy mezi aktéry a jednotlivými případy užití.



Obrázek 4.1: Diagram případů užití hry „Pouštní Rallye“.

4.2.1.2 Detailní popis případů užití

UC1. Identifikace uživatele Tento případ užití plní funkční požadavek „F1“ a modeluje způsob přihlášení do aplikace. Detailní popis je na obrázku 4.2.

Případ užití: Indentifikace uživatele
ID: UC1
Aktéry: Uživatel
Předběžné podmínky:
Tok událostí: 1. Případ užití začína, když uživatel poprvé navštíví stránku aplikace nebo odhlásí se ze systému. 2. Aplikace zobrazí stránku s přihašovacím oknem. 3. Uživatel zadá jméno, pod kterým bude v systému identifikován a klikne na tlačítko "Přihlásit se". 4. Aplikace zapamatuje uživatele a zobrazí svoje hlavní menu.
Důsledek: Uživatel je identifikován v systému.

Obrázek 4.2: Detailní popis případu užití UC1.

UC2. Zahájení hry Tento případ užití je abstraktní a najednou plní funkční požadavky „F8“ a „F9“. Pomocí něj si můžeme představit obecný princip připojení ke hře. Detailní popis je na obrázku 4.3.

Případ užití: Zahájení hry
ID: UC2
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být identifikovaný.
Tok událostí: 1. Případ užití začíná, když uživatel klikne na tlačítko "Začít samostatnou hru" nebo "Začít hru s kamarády" v hlavním menu. 2. Aplikace zobrazí okno s herními nastaveními. 3. Uživatel zadá příslušné parametry hry a klikne na tlačítko "Začít". 4. Aplikace připojí uživatele ke hře a zobrazí herní stránku.
Důsledek: Uživatel je připojen ke hře.

Obrázek 4.3: Detailní popis případu užití UC2.

UC3. Zahájení samostatné hry Tento případ užití dědí od „UC2“ a detailně modeluje způsob zahájení samostatné hry. Plní se pomocí něj funkční požadavek „F9“. Detailní popis je na obrázku 4.4.

UC4. Zahájení hry více hráčů Tento případ užití dědí od „UC2“ a detailně modeluje způsob zahájení hry s více hráči. Plní se pomocí něj funkční požadavek „F8“. Detailní popis je na obrázku 4.5.

UC5. Připojení k existující hře Tento případ užití dědí od „UC2“ a detailně popisuje způsob zahájení hry pomocí připojení k již existující hře. Plní se pomocí něj funkční požadavek „F8“. Detailní popis je na obrázku 4.6.

UC6. Hra Tento případ užití plní funkční požadavek „F2“ a popisuje základní princip herního procesu. Detailní popis je na obrázku 4.7.

UC7. Tah Tento případ užití plní funkční požadavky „F2“, „F3“, „F4“, „F5“, „F6“, „F7“. Popisuje princip provedení jednoho tahu. Detailní popis je na obrázku 4.8. Popisy alternativních scénářů jsou na obrázcích 4.9 a 4.10.

Případ užití: Zahájení samostatné hry
ID: UC3
Rodičovský případ užití: UC2
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být identifikovaný.
Tok událostí: *1. Případ užití začíná, když uživatel klikne na tlačítko "Začít samostatnou hru" v hlavním menu.* *2. Aplikace zobrazí okno s nastavením počtu umělých hráčů, které budou kontrolované počítačem.* *3. Uživatel zadá počet umělých hráčů, se kterými bude hrát, a klikne na tlačítko "Začít".* 4. Aplikace připojí uživatele ke hře a zobrazí herní stránku.
Důsledek: Uživatel je připojen ke hře.

Obrázek 4.4: Detailní popis případu užití UC3.

Případ užití: Zahájení hry více hráčů
ID: UC4
Rodičovský případ užití: UC2
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být identifikovaný.
Tok událostí: *1. Případ užití začína, když uživatel klikne na tlačítko "Začít hru více hráčů" v hlavním menu.* *2. Aplikace zobrazí okno s nastavením maximálního počtu hráčů v této partii.* *3. Uživatel zadá maximální počet hráčů a klikne na tlačítko "Začít".* **4. Aplikace zobrazí okno s klíčem hry, pomocí kterého ostatní hráči se mohou připojit.** **5. Uživatel sdělí klíč svým kamarádům a klikne na tlačítko "OK".** 6. Aplikace připojí uživatele ke hře a zobrazí herní stránku.
Důsledek: Uživatel je připojen ke hře.

Obrázek 4.5: Detailní popis případu užití UC4.

Případ užití: Připojení k existující hře
ID: UC5
Rodičovský případ užití: UC2
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být identifikovaný.
Tok událostí: *1. Případ užití začíná, když uživatel klikne na tlačítko "Začít hru více hráčů" v hlavním menu.* *2. Aplikace zobrazí okno s nastavením klíče existující hry.* *3. Uživatel zadá klíč a klikne na tlačítko "Připojit se ke hře".* 4. Aplikace připojí uživatele ke hře a zobrazí herní stránku.
Důsledek: Uživatel je připojen ke hře.

Obrázek 4.6: Detailní popis případu užití UC5.

4.2.1.3 Tabulka splnění požadavků

Tabulka splnění požadavků, která je zobrazená na obrázku 4.11, pomáhá sledovat, jaké požadavky plní jaký případ užití a jaké požadavky ještě mají být splněny. V daném případě představené případy užití pokrývají všechny specifikované požadavky.

Případ užití: Hra
ID: UC6
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být připojen ke hře.
Tok událostí: 1. Případ užití začíná, když se všichni hráči připojili ke hře. 2. Zatímco hra není ukončená 2.1 Aplikace zobrazuje aktuální pozice hráčů na mapě a stav hry. 2.2 Aplikace vyžaduje, aby další hráč udělal tah. 2.3 Jestliže je hod uživatele 2.3.1 inlude(Tah).
Důsledek:
Alternaitvní tok událostí: 1. Kdykoli uživatel může opustit hru.
Důsledek: Uživatel odpojí se od serveru.

Obrázek 4.7: Detailní popis případu užití UC6.

Případ užití: Tah
ID: UC7
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být připojen ke hře.
Hlavní scénář: 1. Případ užití začíná, když je uživatel na tahu. 2. Aplikace dovolí kliknout na tlačítka "Standardní hod" a "Riskantní hod". 3. Uživatel klikne na jedno z dvou tlačítek. 4. Aplikace zvýrazní pozice, na které se uživatel může přesunout. 5. Uživatel klikne na vybranou pozici. 6. Aplikace přesune figurku uživatele na tuto pozici.
Sekundární scénáře: Přesun na blokující pozici Stání na stejné pozici
Důsledek: Uživatel změní pozici své figurky v aktuální partii hry.

Obrázek 4.8: Detailní popis případu užití UC7.

Případ užití: Tah **Sekundární scénář: Přesun na blokující pozici**
ID: UC7
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být připojen ke hře.
Sekundární scénář: 1. Sekundární scénář začíná, od 5. bodu hlavního scénáře případu užití Tah. 2. Aplikace přesune figurku uživatele na blokující pozici. 3. Aplikace nedovolí uživateli udělat následující tah a nedovolí ostatním hráčům předjet tuto pozici.
Důsledek: Uživatel změní pozici své figurky v aktuální partii hry.

Obrázek 4.9: Detailní popis alternativního scénáře případu užití UC7.

Případ užití: Tah **Sekundární scénář: Stání na stejné pozici**
ID: UC7
Aktéry: Uživatel
Předběžné podmínky: Uživatel musí být připojen ke hře.
Sekundární scénář: 1. Sekundární scénář začíná, od 3. bodu hlavního scénáře případu užití Tah. 2. Aplikace nedostane žádné možností přesunu figurky a završí tah uživatele.
Důsledek: Uživatel zůstane na svém místě.

Obrázek 4.10: Detailní popis alternativního scénáře případu užití UC7.

	UC1	UC2	UC3	UC4	UC5	UC6	UC7
F1	+						
F2						+	+
F3							+
F4							+
F5							+
F6							+
F7							+
F8		+		+	+		
F9		+	+				

Obrázek 4.11: Tabulka splnění požadavků aplikace „Pouštní Rallye“.

Kapitola 5

Analýza

Během analýzy formuje se základní představa o tom, jak bude vypadat budoucí systém a jak bude plnit svou funkčnost. Cílem analýzy je zkoumání problémového doménu a modelování kostry budoucí aplikace pomocí nalezení možných tříd a relací mezi nimi. Na základě vytvořené kostry se odhadují možné způsoby interakce mezi jednotlivými třídami pro dosažení funkčnosti definované v požadavcích. Výsledkem této fáze je pak koncept fungujícího systému, který plní nejdůležitější požadavky a ve kterém nejsou specifikované žádné implementační detaily.

5.1 Doménový model tříd

Doménový model tříd je diagram klíčových tříd a realací mezi nimi. V takovém modelu se popisují základní možné funkce a atributy jednotlivých tříd, které jsou nutné ke splnění důležitých případů užití a tudíž i funkčních požadavků.[1, s. 98]

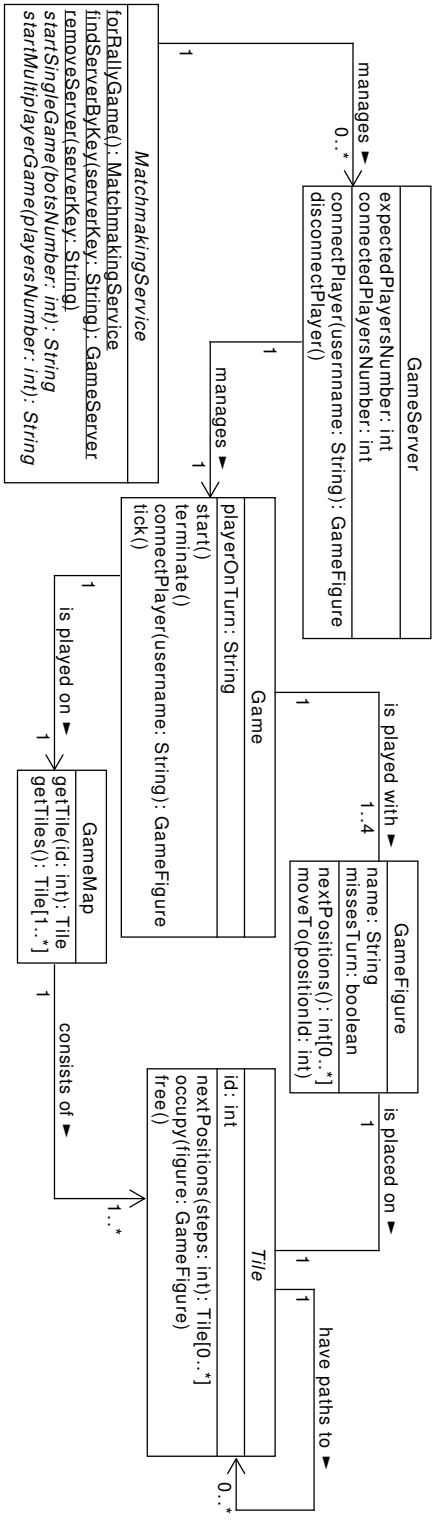
5.1.1 Doménový model tříd hry „Pouštní Rallye“

Na obrázku 5.1 je zobrazen doménový model hry „Pouštní Rallye“. Všechny třídy formují základní popis objektů, které se zúčastní procesu a vztahů mezi nimi. Doménový model byl vytvořen pomocí nástroje UMLet.[22]

MatchmakingService Tato třída je abstraktní a je zodpovědná za statické ukládání objektů třídy „GameServer“. Dědice této třídy umějí vytvářet nové herní servery pro jednoho a více hráčů.

GameServer Tato třída propojuje uživatele se hrou a sleduje aktuální počet připojených hráčů.

Game Tato třída představuje samotnou hru. Propojuje figurky uživatelů s herní mapou a sleduje aktuálního hráče na tahu.



Obrázek 5.1: Doménový model hry „Pouštní Rallye“.

GameFigure Tato třída reprezentuje figurku uživatele ve skutečné partii hry a poskytuje veškerou potřebnou funkčnost pro provedení tahu. Pomocí „GameFigure“ lze vypočítat další pozice a přemístit figurku na jednu z těchto pozic.

GameMap Tato třída reprezentuje herní mapu. Skládá se z dědiců třídy „Tile“ a umí je poskytovat podle ID.

Tile Toto je abstraktní třída představující dlaždici, po kterých se přemísťují hráče. Dlaždice zná své další sousedy a umí vypočítat další pozice v závislosti od počtu kroků.

5.2 Analytická realizace případů užití

Popsat realizaci případů užití je možné pomocí popisu interakcí mezi jednotlivými třídami a objetky. Jedním z několika způsobů takového popisu jsou sekvenční diagramy.

Sekvenční diagramy představují interakce mezi objekty v závislosti na času. V takových diagramech se ukazuje, jaké akce provádí aktér nad aplikací a jaké metody při tom volá u příslušných instancí tříd pro získání nutného výsledku.[1, s. 220]

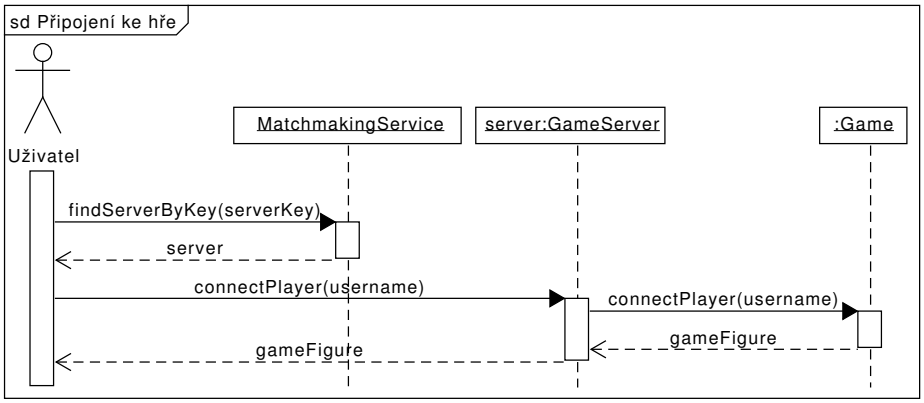
Analytické sekvenční diagramy se modelují na základě doménového modelu tříd, a proto popisují koncept na stejné úrovni abstrakce, tudíž jsou bez specifických implementačních detailů.

5.2.1 Analytické sekvenční diagramy hry „Pouštní Rallye“

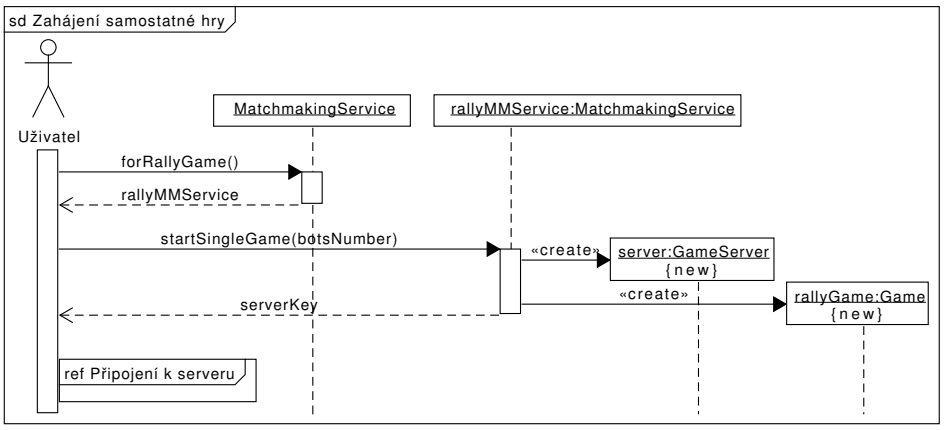
V této subsekci jsou představeny a popsány diagramy interakcí objektů hry „Pouštní Rallye“, které realizují nejdůležitější případy užití. Všechny diagramy byly vytvořeny pomocí nástroje UMLet.[22]

Připojení ke hře Na obrázku 5.2 je zobrazen princip komunikace objektů pro připojení ke hře. Když uživatel zná klíč, může se připojít k serveru pomocí třídy „MatchmakingService“, která nalezne příslušný server podle klíče. Server je představen objektem třídy „GameServer“ a pomocí něj se uživatel připojí ke skutečné hře. Tento sekvenční diagram popisuje realizaci případu užití „UC5“.

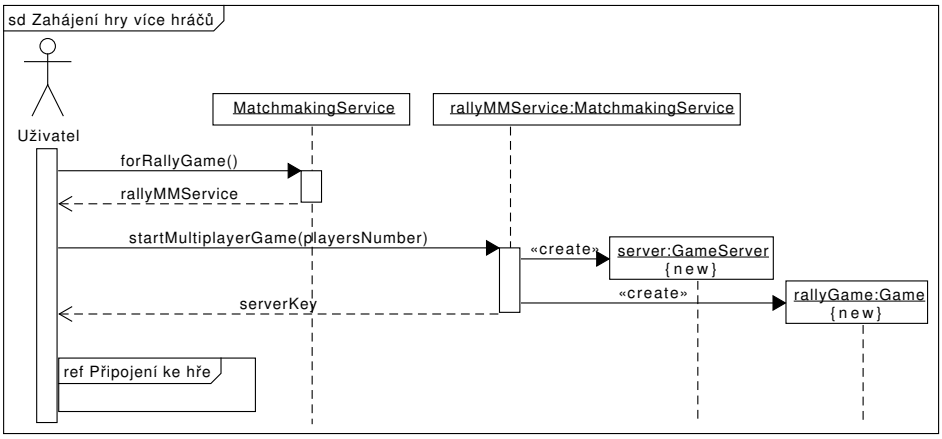
Zahájení samostatné hry Na obrázku 5.3 je zobrazen princip zahájení samostatné hry. Pomocí třídy „MatchmakingService“, uživatel dostane jejího dědice, který je určen k vytváření serverů hry „Pouštní Rallye“, vytvoří nový server a následně se k němu připojí. Tento sekvenční diagram popisuje realizaci případu užití „UC3“.



Obrázek 5.2: Analytický sekvenční diagram: Připojení ke hře.



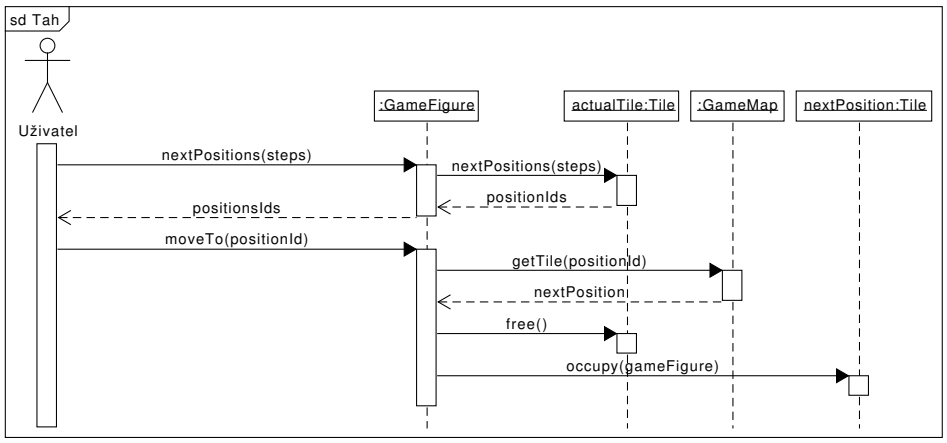
Obrázek 5.3: Analytický sekvenční diagram: Zahájení samostatné hry.



Obrázek 5.4: Analytický sekvenční diagram: Zahájení hry více hráčů.

Zahájení hry více hráčů Na obrázku 5.4 je zobrazen princip zahájení hry více hráčů. Pomocí třídy „MatchmakingService“, uživatel dostane jejího dědice, který je určen k vytváření serverů hry „Pouštní Rallye“, vytvoří nový server a následně se k němu připojí. Tento sekvenční diagram popisuje realizaci případu užití „UC4“.

Tah Na obrázku 5.5 je zobrazen princip provedení jednoho Tahu ve hře. Po připojení ke hře uživatel dostane objekt třídy „GameFigure“, pomocí nějž nalezne další možné pozice a přemístí se dále. Tento sekvenční diagram popisuje realizaci případu užití „UC7“.



Obrázek 5.5: Analytický sekvenční diagram: Tah.

Kapitola 6

Návrh

Hlavním cílem návrhové fáze je rozšíření analytického konceptu z předchozí fáze o implementační detaily. A proto je potřeba zvolit implementační technologie, které přivedou ke správnému řešení. Takové technologie zahrnují programovací jazyk, knihovny, frameworky a způsob ukládání dat. Na základě nových detailů se pak modeluje nový model tříd a sekvenční diagramy, které jsou maximálně podobné budoucímu zdrojovému kódu. Výsledkem této fáze je již hotový koncept, ze kterého se programuje systém.

6.1 Vybrané implementační technologie

6.1.1 Programovací jazyk

Již podle názvu práce je jasné, že implmentačním jazykem projektu musí být jazyk Java. Java je velmi populární programovací jazyk, který se nejlépe hodí k napsání velkých serverových aplikací díky svým knihovnám, specifikacím a profesionální komunitě, ale může být použit i v jakékoli jiné sféře, jelikož je docela výkonný a kompatibilní se skoro všemi platformami.

6.1.2 Framework

Zadání práce také specifikuje i framework, pomocí kterého musí být napsána aplikace. Takovým frameworkem je Vaadin, který od verze 10 se přesněji nazývá VaadinFlow. VaadinFlow umožňuje realizovat tenký klient webové aplikace podobným způsobem, jak se realizuje grafické uživatelské rozhraní u obvyklých počítačových aplikací, kde aplikační vrstva přímo kontaktuje s prezentační vrstvou. Ve frameworku Vaadin ale aplikační vrstva kontaktuje s modelem prezentační vrstvy. Když uživatel poprvé navštěvuje webovou stránku aplikace, stahuje si engine VaadinFlow, který již dále synchronizuje data s tímto modelem uloženým na serveru. Umožňuje to oboustrannou komunikaci mezi serverem a uživatelem, což se velmi hodí k vývoji her.

6.1.3 Perzistence dat

Daná aplikace nepotřebuje žádné ukládání dat, a proto se ten problém neřeší.

6.2 Architektonický vzor

Architektonickým vzorem aplikace byl zvolen Model-View-Controller čili MVC. MVC rozděluje logiku a strukturu zdrojového kódu na tři příslušné názvu komponenty. Tyto komponenty komunikují mezi sebou pomocí rozhraní a podle následujícího principu: View reprezentuje uživatelské rozhraní, zobrazuje aktuální stav Modelu a sleduje uživatelské akce; při provedení nějaké akce, View kontaktuje Controller, který volá příslušné metody rozhraní Modelu; Model následně mění svůj stav a notifikuje o tom View, aby zobrazilo nový stav Modelu.[23] Takový MVC se přesněji nazývá aktivní, protože jsou všechny komponenty propojené cyklicky. Framework VaadinFlow umožňuje komunikovat s webovými stránkami ze strany serveru, a proto lze uskutečnit takové propojení, což se stalo důvodem výběru vzoru MVC.

6.3 Návrhový model tříd

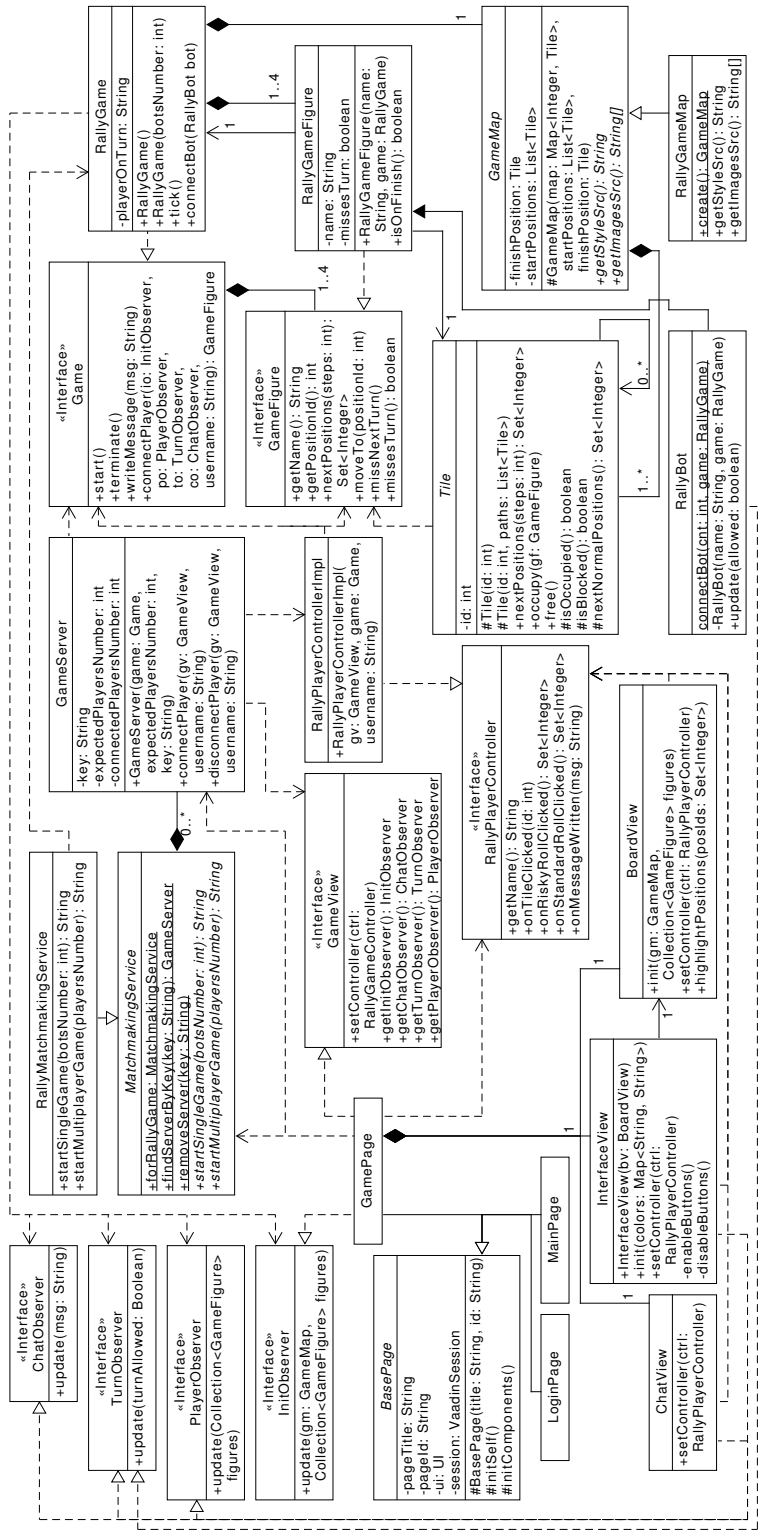
Návrhový model tříd je rozšířený o specifické detaily implementace doménový model, pomocí kterého se pak píše výsledný zdrojový kód. V takovém modelu skoro všechny původní relace mezi třídami musejí být změněny závislostmi, agregacemi nebo kompozicemi. Staré třídy mohou se rozdělit na menší nové kvůli rozdílné logice, mohou se přidat třídy definující rozhraní nebo týkající se uživatelského rozhraní, jelikož jsou v dané fázi vývoje již určené implementační technologie. Metody takových tříd teď mají reprezentovat konkretní jednoduché akce, které souvisejí s odpovědností své třídy.[1, s. 260-261, 276]

6.3.1 Návrhový model tříd hry „Pouštní Rallye“

Na obrázku 6.1 je zobrazen návrhový model tříd hry „Pouštní Rallye“. Byly přidány rozhraní, třídy reprezentující uživatelské rozhraní neboli webové stránky, protože se vyvíje webová aplikace, a třídy patřící k architektonickému vzoru MVC. Jelikož návrhový model je založen na doménovém, podrobně budou popsané pouze nové významné třídy a rozhraní. Návrhový model byl vytvořen pomocí nástroje UMLet.[22]

BasePage Tato třída je abstraktní, dědí od třídy „Div“ frameworku Vaadin a reprezentuje jednu webovou stránku aplikace. Má přístup k uživatelskému sezení, pomocí kterého rozlišuje uživatele.

LoginPage Tato třída reprezentuje stránku pro přihlášení do aplikace. Bez přihlášení se uživatel nedostane do hlavního menu aplikace a nebude identifikován v systému.



Obrázek 6.1: Návrhový model tříd hry „Pouštní Rallye“.

MainPage Tato třída reprezentuje hlavní menu aplikace, ve kterém lze vybrat režim hry a ji začít.

GamePage Tato třída reprezentuje stránku, na které se provádí samotná hra. Skládá se ze tří komponentů „BoardView“, „InterfaceView“ a „ChatView“. Implementuje rozhraní „GameView“, pomocí kterého je propojená se hrou, a rozhraní „InitObserver“, pomocí kterého iniciuje své komponenty.

BoardView Tato třída reprezentuje komponent stránky „GamePage“, který zobrazuje herní mapu a hráče. Implementuje rozhraní „PlayerObserver“, pomocí kterého dostává aktuální informace o pozicích hráčů.

InterfaceView Tato třída reprezentuje část stránky „GamePage“, která zobrazuje herní rozhraní. Toto rozhraní zahrnuje tlačítka s výběrem hodu kostkou. Implementuje rozhraní „TurnObserver“, pomocí kterého tato tlačítka blokuje a odblokuje.

ChatView Tato třída reprezentuje komponent stránky „GamePage“, který zobrazuje herní chat a umožňuje psání nových zpráv. Implementuje rozhraní „ChatObserver“, pomocí kterého dostává nové zprávy.

RallyPlayerController Toto rozhraní reprezentuje Controller vzoru MVC. Všechny uživatelské akce nad uživatelským rozhraním se přesměrují na třídu implementující toto rozhraní. S Controllerem komunikují třídy „BoardView“, „InterfaceView“ a „ChatView“.

RallyBot Tato třída reprezentuje herní umělou inteligenci. Dědí od třídy „RallyGameFigure“ a umí totéž, co umí hráč, ale dělá to náhodným způsobem. Navíc implementuje rozhraní „TurnObserver“, aby sledoval svoje pořadí tahu.

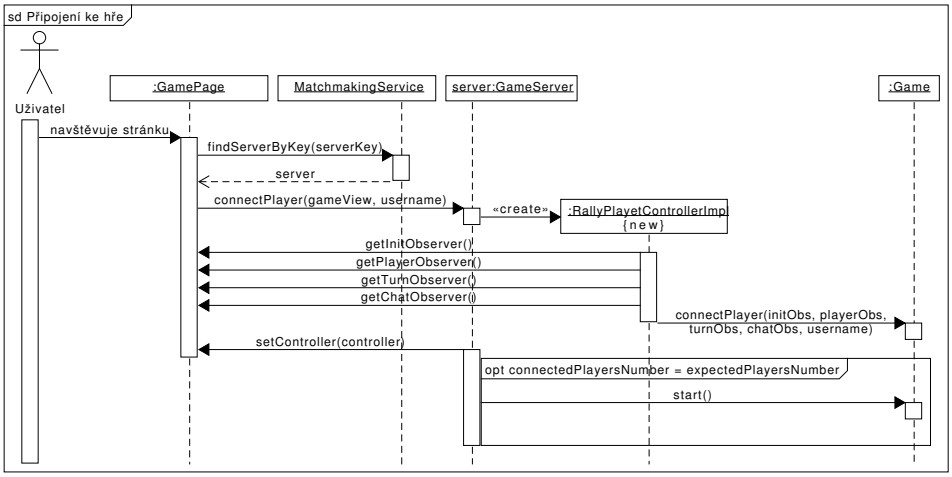
6.4 Návrhová realizace případů užití

Návrhová realizace případů užití rozšiřuje analytickou realizaci stejným způsobem, jako návrhový model rozšiřuje doménový. Sekvenční diagramy zachovávají svůj princip, jenže jsou doplněné o implementační detaily návrhového modelu.

6.4.1 Návrhové sekvenční diagramy hry „Pouštní Rallye“

Uživatel jako aktér systímu v návrhových diagramech komunikuje již ne s základním modelem aplikace, ale s plnohodnotným uživatelským rozhraním, které je představeno webovými stránkami.

Chybové situace nebyly ukázány pro lepší čitelnost. Všechny diagramy byly vytvořeny pomocí nástroje UMLet.[22]

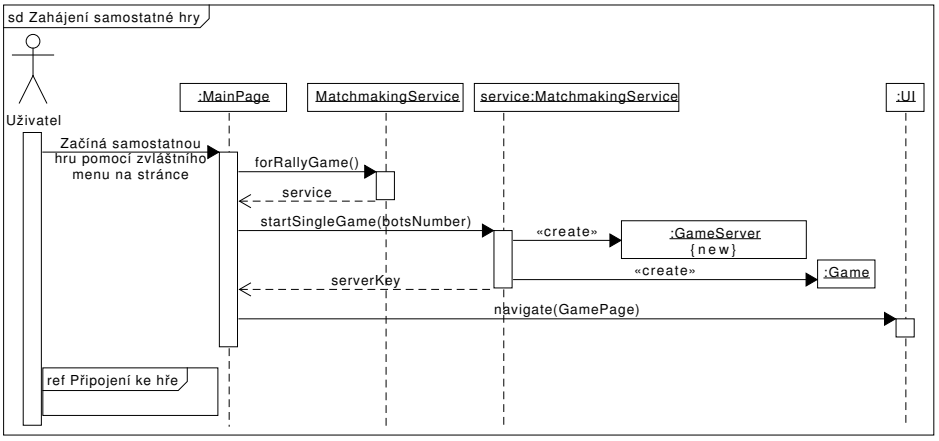


Obrázek 6.2: Návrhový sekvenční diagram: Připojení ke hře.

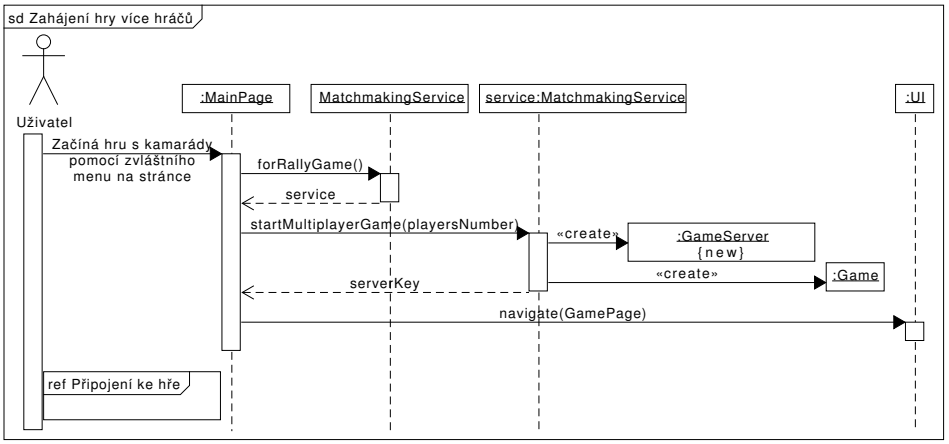
Připojení k hře Na obrázku 6.2 je zobrazena sekvence metod při připojování ke hře. Jakmile se uživatel dostane na herní stránku, aplikace se pokusí připojit k hernímu serveru na základě klíče, který je uložen jako atribut uživatelského sezení. Po připojení k serveru, uživatel automaticky bude připojen i ke hře: vytvoří se Controller, který propojí View a Model. Pokud je počet připojených uživatelů dostatečný, hra se nastartuje.

Zahájení samostatné hry Obrázek 6.3 ukazuje sekvenci metod při zahájení samostatné hry. Uživatel pomocí uživatelského rozhraní zadá parametry samostatné hry a podtvrdí její zahájení. Instanciuje se dědic třídy „MatchmakingService“, který následně vytvoří nové objekty „GameServer“ a „Game“ a je uloží. Klíč vytvořeného serveru dostane stránka „MainPage“, která ho uloží do atributů sezení a přesměruje na jinou stránku „GamePage“. Dále se uskuteční připojení ke hře podle předchozího diagramu. Stejně se tak dělá při zahájení hry více hráčů, diagram je na obrázku 6.4.

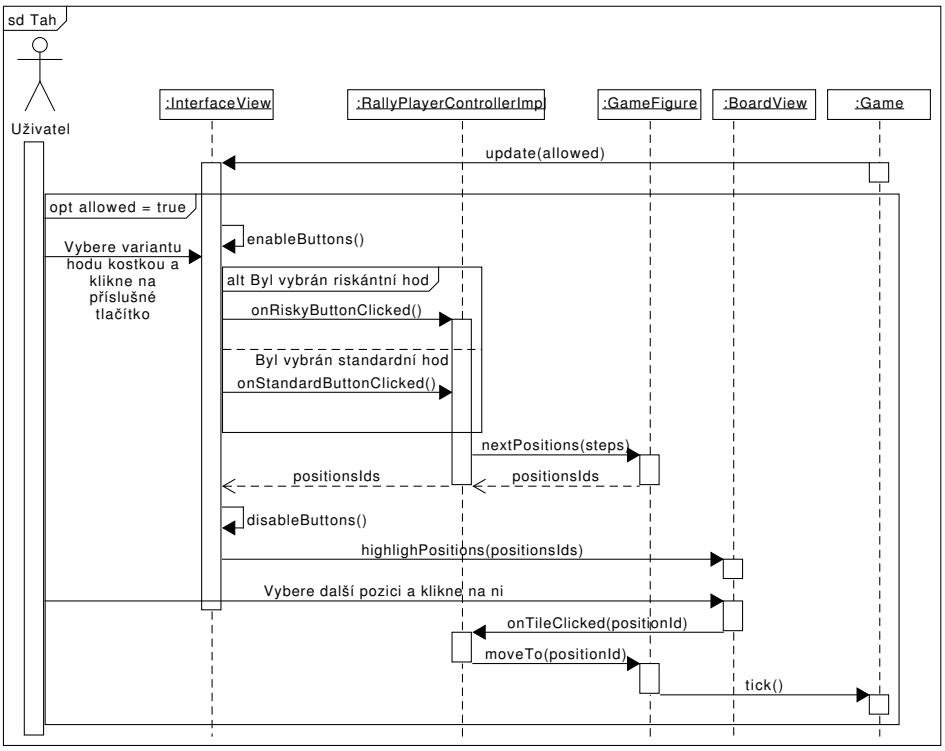
Tah Obrázek 6.5 ukazuje sekvenci metod při provedení jednoho tahu. Všechno se děje na stránce „GamePage“, jejímiž částmi jsou „InterfaceView“ a „BoardView“. Jakmile je tah uživatele, „InterfaceView“ odblokuje příslušná tlačítka. Uživatel klikne na jedno z tlačítek a „BoardView“ zvýrazní dostupné pozice. Uživatel pak klikne na vybranou pozici a aplikace přesune jeho figurku, aktualizuje View všech uživatelů partie a požádá dalšího v pořadí hráče udělat svůj tah. Pro lepší čitelnost nebyla zobrazena komunikace s objekty třídy „Tile“



Obrázek 6.3: Návrhový sekvenční diagram: Zahájení samostatné hry.



Obrázek 6.4: Návrhový sekvenční diagram: Zahájení hry více hráčů.



Obrázek 6.5: Návrhový sekvenční diagram: Tah.

6.5 Návrh uživatelského rozhraní

Poslední částí návrhové fáze je návrh uživatelského rozhraní. Předtím jak začít implementovat, programátor musí mít představu o tom, jak bude vypadat budoucí aplikace. V případě hry „Pouštní Rallye“ celá aplikace se skládá ze tří webových stránek:

1. přihlašovací stránka,
2. hlavní stránka a
3. herní stránka.

Všechny návrhy byly vytvořeny pomocí nástroje Figma.[24]

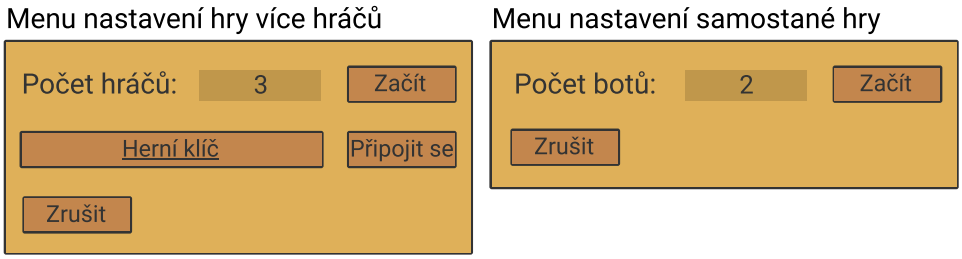
6.5.1 Přihlašovací stránka

„LoginPage“ je výchozí přihlašovací stránka. Skládá se z formy, kam uživatel zadá své jméno, a tlačítka, které následně klikne pro přihlášení.

Obrázek 6.6: Návrh stránky „LoginPage“.

6.5.2 Hlavní stránka

„MainPage“ je stránka aplikace zobrazující hlavní menu. Pomocí tohoto menu uživatel může začít hru, přečíst si pravidla hry a odhlásit se z aplikace. Před zahájením hry se uživateli zobrazí menu, ve kterém vybere příslušná nastavení.

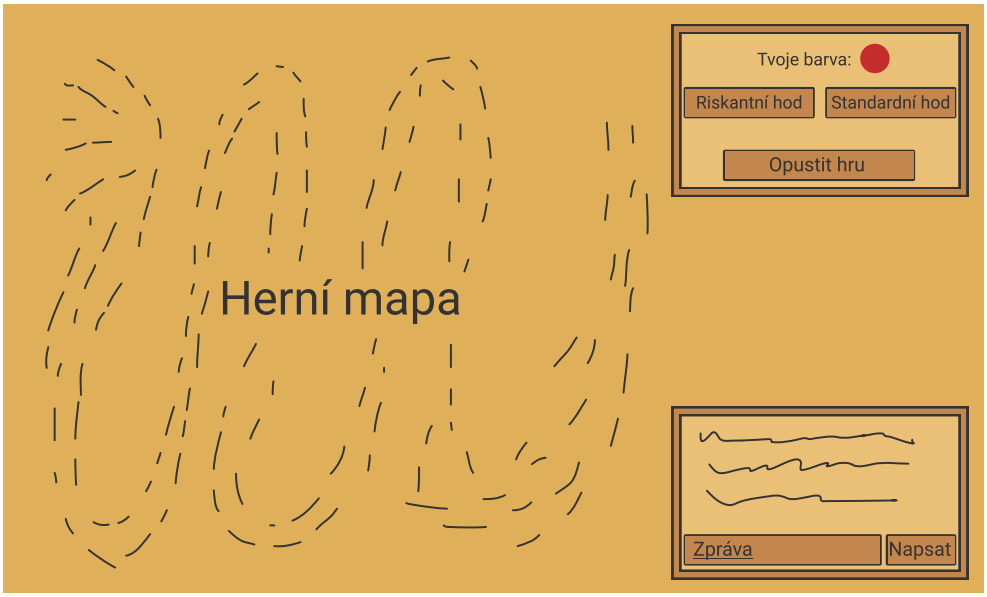


Obrázek 6.8: Návrh menu s nastaveními.

6.5.3 Herní stránka

„GamePage“ je stránka, na které prochází samotný herní proces. Zobrazuje herní mapu, rozhraní pro výběr hodu kostkou a herní chat.

Obrázek 6.7: Návrh stránky „MainPage“.



Obrázek 6.9: Návrh stránky GamePage.

Kapitola 7

Implementace

Implementace je jedinou fází vývoje softwaru, kterou nelze vynechat, a proto je nejdůležitější části celého procesu.[25] Většina času se v této fázi tráví napsaním kódu na základě již vypracovaného konceptu z návrhové fáze pomocí již určených instrumentů. V této kapitole avšak budou popsány použité při napsání kódu principy a vzory a také vymyšlená autorem řešení.

7.1 Principy napsání kódu

Při napsání kódu se autor pokoušel se držet základních principů „čistého kódu“:

- název třídy musí odrážet její podstatu, proč tato třída existuje a jaká je její funkčnost;
- podle názvu metod musí být zřejmé, jaké plní funkce;
- dlouhý a srozumitelný název je lepší než krátký a nesrozumitelný;
- třídy a metody musí být co nejkompaktnější pro lepší čitelnost a srozumitelnost;
- pokud funkce dostane na vstup chybná data, musí hodit výjimku;
- výjimky musí podrobně popisovat chybu;
- komentáře se nepoužívají, pokud lze namísto toho použít funkci nebo proměnnou.[26]

Také byl použit princip „Don’t repeat yourself“ neboli DRY, který se překládá jako „Neopakuj se“. DRY říká, že každá část systému musí mít jedinou reprezentaci, což ohledně kódu znamená, že žádné jeho řádky nesmí být použité několikrát.[27] Všechny opakující části se dávají do funkcí nebo konstant, které se už pak využívají.

7.2 Implementace hry „Pouštní Rallye“

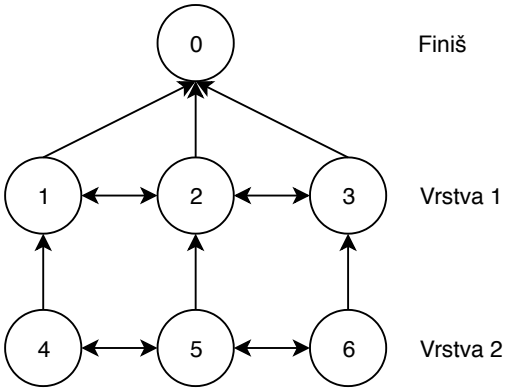
7.2.1 Reprezentace mapy ve hře

Je od začátku zřejmé, že se herní mapa musí skládat z dlaždic, jak to je u její deskové verze. Ale jak tyto dlaždice reprezentovat a ukládat v aplikaci? Jak se pak hledat cesty mezi těmito dlaždicemi? Řešením takového problému byl zvolen orientovaný graf.

Každá instance třídy „Tile“, která reprezentuje jednu dlaždici, představuje jeden vrchol grafu. Jelikož všechny vrcholy jsou unikátní, každá dlaždice má svůj unikátní ID. Orientované hrany jsou představene atributem třídy „Tile“, který je kolekcí odkazů na jiné sousední dlaždice. Všechny dlaždice jedné mapy se ukládají do slovníku, kde klíčem je ID dlaždice. Výsledná mapa představená dědicem třídy „GameMap“, která pak má tento slovník dlaždic a ví, které z nich jsou startovací a která je finiš. K vytvoření takové mapy slouží třída „GameMapBuilder“.

Pomocí takové reprezentace se dá lehce zjistit aktuální pozici figurky hráče, najít vrcholy, do kterých se možné dostat při požadovaném počtu kroků, a přemístit tuto figurku.

Přemíštění se uskutečňuje směrem orientací hran. Všechny hrany vedou k finiši, avšak se dá volně pohybovat mezi vrcholy jedné vrstvy. Vrstva je skupina vrcholů, které jsou ve stejné vzdálenosti od finiše. Příklad takového grafu je na obrázku 7.1.



Obrázek 7.1: Přiklad reprezentace mapy formou grafu.

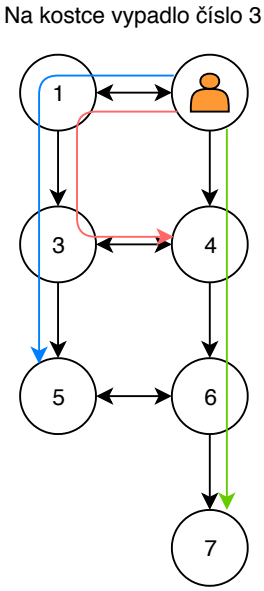
7.2.2 Hledání dalších pozic

Hledání dalších pozic prochází shodným způsobem jako algoritmem „Breadth-First Search“ neboli BFS, neboli „Hledání do šířky“. V tomto případě ale algoritmus se dělá rekurzivně a si nepamatuje, kterými vrcholy již prošel, avšak nevrací se do vrcholů, ze kterých byl spuštěn. Takové řešení je zvoleno

kvůli tomu, aby se spočítaly všechny možné varianty přemístění figurky, kromě těch, kde se pohybuje jedním vrcholem během jedné cesty více než jednou. Uskutečňuje se voláním metody instance třídy „Tile“:

```
Set<Integer> nextPositions(int steps),
```

kde „steps“ je počet kroků, který musí projít figurka od této pozice. Vrací množinu ID vrcholů, do kterých se dá se přemístit. Může vrátit i prázdnou množinu v případě, že po cestě před hráčem nějaký jiný hráč blokuje trať. Na obrázku 7.2 je zobrazen příklad hledání další pozice, kde různými barvami zobrazeny možné cesty.



Obrázek 7.2: Příklad hledání další pozice.

7.2.3 Zobrazení mapy v prohlížeči

Další otázkou po modelování mapy bylo „Jak tuto mapu zobrazit?“. Jelikož se vytváří webová aplikace, tenkým klientem je v takovém případě prohlížeč. Řešením bylo zvoleno použití standardních elementů HTML a CSS, což se lehce dá implementovat pomocí frameworku Vaadin.

Každá dlaždice je zobrazená elementem „div“, který má svůj ID stejný s ID vrcholu dlaždice, a má svoji unikátní pozici. Pozice se určují pomocí CSS pravidel pro každou dlaždici. Jelikož takových dlaždic je více než 300, není vhodné psát taková pravidla ručně. Pro tyto účely byl napsán jednoduchý skript na JavaScript, který lze spustit v prohlížeči v režimu offline. Pomocí něj lze myší nastavit pozici a vygenerovat CSS pravidla každého elementu. Tímto způsobem byly nastaveny všechny pozice. Výsledný CSS soubor byl

přemístěn do zdrojových souboru projektů. Tento soubor se vždycky stahuje uživatelem, když se navštěvuje herní stránka.

7.2.4 Umělá inteligence

Vzhledem k prostotě herního procesu, umělá inteligence je představena náhodně:

- při hodu kostkou bot s pravděpodobností 50 % na 50 % vybere jednu ze dvou variant;
- následně vybere jednu z dostupných dlaždic, kde pravděpodobnost výběru každé dlaždice je také stejná.

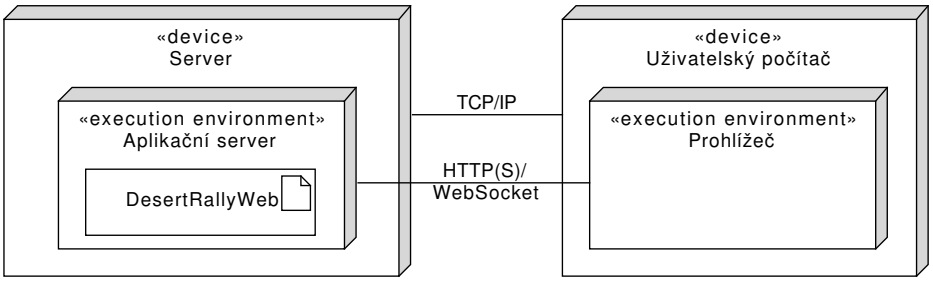
7.2.5 Verze použitých nástrojů

Aplikace byla vyvinuta v jazyce Java verze 8 a sestavena nástrojem Maven verze 3.6.0. Při sestavení aplikace byly použity následující závislosti:

- „vaadin-bom“ a „vaadin-core“ verze 13.0.6. Tyto knihovny představují framework VaadinFlow.
- „javax.servlet-api“ verze 3.1.0. Knihovna poskytovaná aplikačním serverem, na který je aplikace nasazená. Je potřebná pro zpracování odpovědí uživatelům a nezbytná pro framework VaadinFlow.
- „slf4j-simple“ verze 1.7.25. Knihovna potřebná pro vedení logu.
- „mockito-core“ verze 2.28.2. Knihovna potřebná pro vytváření mock-tříd.
- „junit-jupiter-engine“ 5.4.2. Knihovna poskytující testovací prostředí.

7.3 Nasazení

Nasazení je závěrečná fáze vývoje softwaru. Popsat nasazení je možné pomocí příslušného diagramu, který ukazuje, v jakém prostředí musí běžet aplikace a pomocí jakých protokolů probíhá komunikace mezi jednotlivými částmi.[1, s. 365] Diagram nasazení aplikace „Pouštní Rallye“ je zobrazen na obrázku 7.3. Diagram byl vytvořen pomocí nástroje UMLet.[22]



Obrázek 7.3: Diagram nasazení aplikace „Pouštní Rallye“.

Kapitola 8

Testování

Testování je nezbytnou součástí vývoje softwarového produktu. Cílem tohoto procesu je nalezení neočekávaných chyb systému a ověření, zda systém plní požadovanou funkčnost. Část testů se provádí samotnými programátory a testery, ale konečnou aplikaci již testují potenciální uživatele. Celý ten proces lze rozložit na jednotlivé body, které probíhají paralelně s vývojem: testování modulů, integrační testování, systémové testování a akceptační testování.[28]

8.1 Testovací prostředí

Testování probíhalo na vlastním počítači autora, aplikace běžela na lokálním serveru.

- Aplikační servery Jetty verze 9.4.11 a GlassFish verze 5.0.0.
- Rozlišení obrazovky 1920x1080 pixelů.
- Použité prohlížeče Mozilla Firefox verze 67 a Google Chrome verze 74.

8.2 Testování hry „Pouštní Rallye“

8.2.1 Testování modulů

Testování modulů neboli „unit testing“ je testování funkčností jednotlivých tříd, kde programátor testuje správnost chování každé metody. Cílem testů je nalezení takových kombinací vstupních dat, při kterých aplikace vygeneruje chybný výstup.[29]

Většina tříd přímo komunikují mezi sebou, a aby se najednou netestovalo několik modulů, používají se mock-třídy. Objekty takových tříd imitují chování originálních instancí, ale neposkytují potřebnou funkčnost. Navíc lze spočítat, kolikrát byla volána konkrétní metoda, a napřed zadat její výsledek.

Testy modulů byly provedeny na celé aplikace, kromě komponentu View, který je založen na frameworku Vaadin. Při testování byly použity knihovny JUnit5, která poskytuje testovací prostředí a funkce, a Mockito, která umožňuje vytváření mock-tříd. Největší pozornost byla věnována třídam „GameMapBuilder“, „RallyGameFigure“, „Tile“ a jejím dědicům, protože ony zakládají bázi hry.

8.2.2 Integrační a systémové testování

Integrační testování testuje souvislost jednotlivých modulu. Moduly nesmí mezi sebou kolidovat a musí správně spolu komunikovat pro dosažení složitějších výsledků. Cílem je nalezení a opravení takových špatných souvislostí, které přivádějí k chybám v systému.[30]

Systémové testování testuje, jestli správně integrované moduly opravdu korektně plní nějaký určitý požadavek. Testování nejčastěji probíhá podle scénářů definovaných v případech užití.[31]

Integrační a systémové testování se děla ne programátory, ale speciálními testery a pomocí doplňujících nástrojů nebo knihoven. Bohužel v případě aplikace „Pouštní Rallye“ takové testování není možné, jelikož se nedá otestovat uživatelské rozhraní vytvořené pomocí frameworku Vaadin. Testovací knihovna je dostupná pouze prémiovým uživatelům.

8.2.3 Akceptační testování

Akceptační testování je finální fáze testování, kde je skoro hotový produkt představen skupině potenciálních budoucích uživatelů. Od uživatelů se vyžaduje plnohodnotné využití všech funkcí aplikace. Cílem je nalezení možných zbylých chyb a získání zpětné vazby.[32]

Pro akceptační testování byli vybráni tři uživatelé. Každý z uživatelů musel vykonat následující seznam činností:

1. Přihlásit se do aplikace.
2. Vytvořit a zahrát partii samostatné hry.
3. Přejít zpět na hlavní menu.
4. Vytvořit a zahrát partii hry více hráčů s autorem.
5. Odhlásit se z aplikace.

Díky prostotě aplikace a jejímu intuitivnímu rozhraní všechy testy byly vykonané bez jakýchkoli problémů. Jedinou poznámkou bylo to, že pokud uživatel neuvidí notifikaci o tom, že je na tahu, musí to pochopit sám. Následně to bylo opraveno přidáním nových stylů příslušným tlačítkům, když jsou aktivní.

Závěr

Cílem dané práce bylo vytvořit webovou herní aplikaci, jejíž prototypem je desková hra „Pouštní Rallye“. Vývoj projektu byl proveden podle pravidel a principů unifikovaného procesu vývoje softwaru, ve fázích kterého byly splněny jednotlivé body zadání.

Nejprvé byly specifikovány požadavky a omezení aplikace, pomocí kterých byly následně modelovány případy užití. Případy užití pomohly zformovat představu o tom, z jakých částí se může skládat systém, což bylo zobrazeno v doménovém modelu tříd. Komunikace všech tříd tohoto modelu byla představena pomocí sekvenčních diagramů. Takové diagramy pomohly zachytit zodpovědnosti a funkce každého objektu systému. Následně byly zvoleny implementační technologie a architektonický vzor aplikace, kvůli kterým bylo potřeba rozšířit a upřesnit předchozí model a diagramy o implementační detaily. Podle nových diagramů a modelu již bylo možné psát zdrojový kód aplikace.

Výsledná aplikace pak byla otestována autorem a jinými uživateli a nasazena na aplikační server firmy „USPIN.cz“. Hru je možné zahrát jak samostaně, tak i s více hráči. Všechny body zadání byly splněny a aplikace je hotová k využití.

Díky použitému návrhu se tuto aplikaci dá lehce rozšířit o nové mapy a dokonce i nové herní režimy v podobném stylu. V budoucnu by bylo vhodné zlepšit editor map prostřednictvím integrace do herní aplikace, udělat responsivní design uživatelského rozhraní a přidat podporu osobních účtů.

Seznam použité literatury

1. ARLOW, Jim; NEUSTADT, Ila. *UML and the Unified Process: Practical object-oriented analysis and design*. 1. vyd. London: Pearson Education Limited, 2002. ISBN 0-201-77060-1.
2. A short history of the Web. *CERN* [online] [cit. 2019-04-23]. Dostupné z: `https://home.cern/science/computing/birth-web/short-history-web`.
3. BERNERS-LEE, T.; CONNOLLY, D. Hypertext Markup Language - 2.0. *IETF* [online]. 1995 [cit. 2019-06-19]. Dostupné z: `https://theblog.adobe.com/adobe-flash-update/`.
4. LIE, Håkon Wium; BOS, Bert. Cascading Style Sheets, level 1. *W3C* [online]. 1996 [cit. 2019-06-19]. Dostupné z: `https://theblog.adobe.com/adobe-flash-update/`.
5. MERCER, Jamie. A Short History of Java. *DZone* [online]. 2017 [cit. 2019-04-23]. Dostupné z: `https://dzone.com/articles/a-short-history-of-java`.
6. *JavaSE 7 Documentation* [online] [cit. 2019-06-19]. Dostupné z: `https://docs.oracle.com/javase/7/docs/api/java/awt/package-summary.html#package_description`.
7. *ChessApp* [online]. Peter Hunter, 2002 [cit. 2019-06-19]. Dostupné z: `https://web.archive.org/web/20090907072956/http://english.op.org/~peter/ChessApp/#applet`.
8. ANDREESSEN, Marc. Innovators of the Net: Brendan Eich and JavaScript. *Netscape TechVision* [online]. 1998 [cit. 2019-04-23]. Dostupné z: `https://web.archive.org/web/20080208124612/http://wp.netscape.com/comprod/columns/techvision/innovators_be.html`.
9. HOFFMANN, Jay. Flash And Its History On The Web. *The history of the Web* [online]. 2017 [cit. 2019-04-23]. Dostupné z: `https://thehistoryoftheweb.com/the-story-of-flash/`.

10. *Newgrounds Wiki: History* [online] [cit. 2019-04-23]. Dostupné z: https://www.newgrounds.com/wiki/about-newgrounds/history.

11. BOUHLEL, Yassine. A Nostalgic Rummage Through the History of Flash [online]. 2010 [cit. 2019-06-19]. Dostupné z: https://code.tutsplus.com/articles/a-nostalgic-rummage-through-the-history-of-flash--active-6733.

12. FESTA, Paul. Just a Flash in the Web video pan? *ZDNet* [online]. 2005 [cit. 2019-04-23]. Dostupné z: https://web.archive.org/web/20070516090612/http://news.zdnet.co.uk/internet/0%2C1000000097%2C39211831%2C00.htm.

13. Gold Miner. *CrazyGames* [online] [cit. 2019-06-25]. Dostupné z: https://www.crazygames.com/game/gold-miner.

14. *HTML5 Differences from HTML4* [online] [cit. 2019-04-23]. Dostupné z: https://www.w3.org/TR/html5-diff/.

15. JOBS, Steve. Thoughts on Flash. *Apple* [online]. 2010 [cit. 2019-06-19]. Dostupné z: https://www.apple.com/hotnews/thoughts-on-flash/.

16. *BrowserQuest* [online]. Mozilla Foundation, 2019 [cit. 2019-06-25]. Dostupné z: http://browserquest.mozilla.org/.

17. COMMUNICATIONS, Adobe Corporate. Flash And The Future of Interactive Content. *Adobe Blog* [online]. 2017 [cit. 2019-06-19]. Dostupné z: https://theblog.adobe.com/adobe-flash-update/.

18. *JEP 289: Deprecate the Applet API* [online]. Daniil Titov, 2016 [cit. 2019-06-19]. Dostupné z: http://openjdk.java.net/jeps/289.

19. PAPER, An Oracle White. Java Client Roadmap Update [online katalogový list]. 2018 [cit. 2019-06-19]. Dostupné z: https://www.oracle.com/technetwork/java/javase/javaclientroadmapupdate2018mar-4414431.pdf.

20. WebGL 2.0 Specification [online]. 2015 [cit. 2019-06-19]. Dostupné z: https://www.khronos.org/registry/webgl/specs/latest/2.0/.

21. MLEJNEK, Ing. Jiří. *Softwarové Inženýrství I: Analýza a sběr požadavků - případy užití* [online]. 2019 [cit. 2019-06-24]. Dostupné z: https://moodle.fit.cvut.cz/pluginfile.php/88280/mod_resource/content/3/03.prednaska.pdf [Soubor přístupný po přihlášení do sítě ČVUT – kopie souboru uložena na přiloženém USB disku].

22. *Free UML Tool for Fast UML Diagrams* [online]. UMLet, 2019 [cit. 2019-06-25]. Dostupné z: https://www.umlet.com/.

23. *Applications Programming in Smalltalk-80(TM): How to use Model-View-Controller (MVC)* [online]. Steve Burbeck, Ph.D., 1987 [cit. 2019-06-19]. Dostupné z: https://web.archive.org/web/20120729161926/http://st-www.cs.illinois.edu/users/smarch/st-docs/mvc.html.

24. *Figma: the collaborative interface design tool.* [online]. Figma, 2019 [cit. 2019-06-25]. Dostupné z: https://www.figma.com/.

25. MLEJNEK, Ing. Jiří. *Softwarové Inženýrství I: Implementace* [online]. 2019 [cit. 2019-06-24]. Dostupné z: https://moodle.fit.cvut.cz/pluginfile.php/88291/mod_resource/content/1/08.prednaska.pdf [Soubor přístupný po přihlášení do sítě ČVUT – kopie souboru uložena na přiloženém USB disku].

26. Summary of 'Clean code' by Robert C. Martin. *GitHub Gist* [online] [cit. 2019-06-24]. Dostupné z: https://gist.github.com/wojteklu/73c6914cc446146b8b533c0988cf8d29.

27. VENNERS, Bill. Orthogonality and the DRY Principle: A Conversation with Andy Hunt and Dave Thomas, Part II [online]. 2003 [cit. 2019-06-19]. Dostupné z: https://www.artima.com/intv/dry.html.

28. HLAVATÝ, Martin. *Softwarové Inženýrství II: Testování* [online]. 2018 [cit. 2019-06-24]. Dostupné z: https://moodle.fit.cvut.cz/pluginfile.php/172/course/section/11326/06_Testing.pdf [Soubor přístupný po přihlášení do sítě ČVUT – kopie souboru uložena na přiloženém USB disku].

29. Unit Testing. *Software Testing Fundamentals* [online] [cit. 2019-06-24]. Dostupné z: http://softwaretestingfundamentals.com/unit-testing/.

30. Integration Testing. *Software Testing Fundamentals* [online] [cit. 2019-06-24]. Dostupné z: http://softwaretestingfundamentals.com/integration-testing/.

31. System Testing. *Software Testing Fundamentals* [online] [cit. 2019-06-24]. Dostupné z: http://softwaretestingfundamentals.com/system-testing/.

32. Acceptance Testing. *Software Testing Fundamentals* [online] [cit. 2019-06-24]. Dostupné z: http://softwaretestingfundamentals.com/acceptance-testing/.

Příloha **A**

Seznam použitých zkratek

IT Informační technologie

WWW World Wide Web

HTML Hyper Text Markup Language

CSS Cascading Style Sheets

JVM Java Virtual Machine

AWT Abstract Window Toolkit

MVC Model-View-Controller

DRY Don't repeat yourself

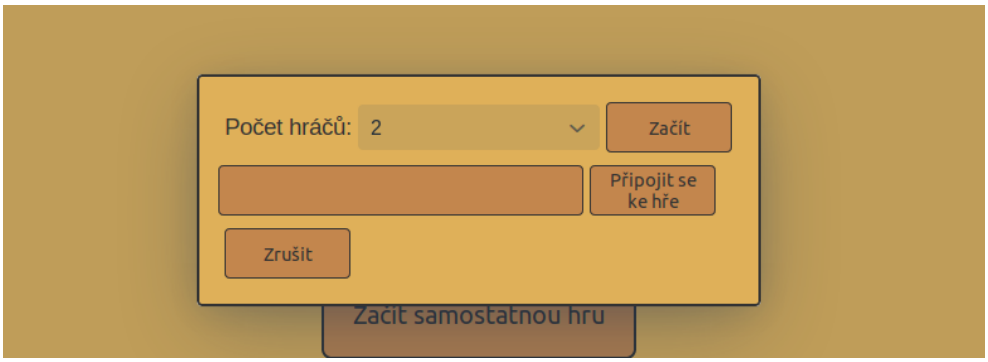
BFS Breadth-First Search

Příloha B

Finální podoba aplikace

Prázdné místo na přihlašovací a halvní stránce je odříznuté pro lepší čitelnost.

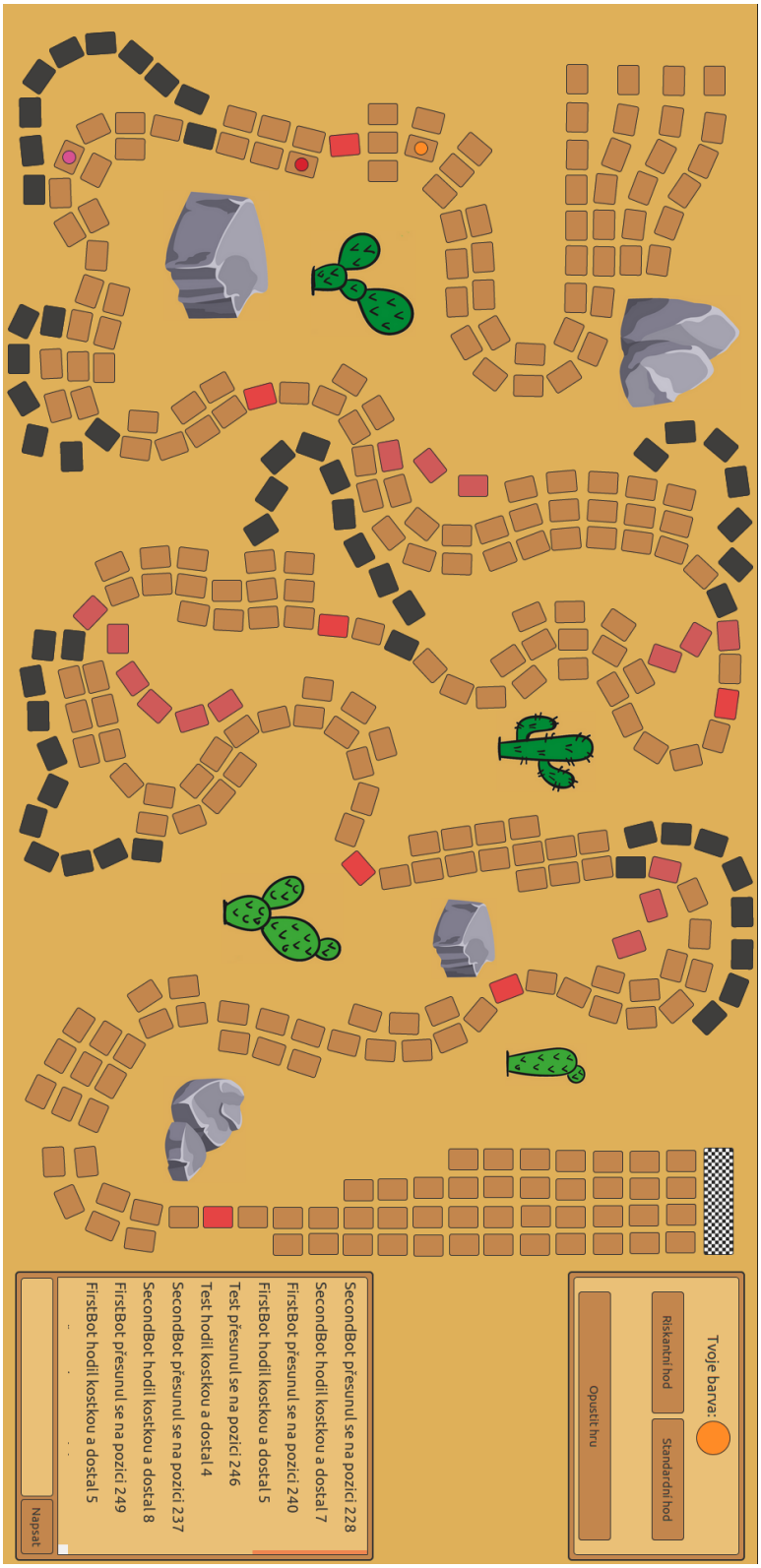
Obrázek B.1: Přihlašovací stránka.



Obrázek B.2: Menu nastavení hry více hráčů.

Obrázek B.3: Menu nastavení samostatné hry.

Obrázek B.4: Hlavní stránka.



Obrázek B.5: Herní stránka.

Příloha C

Obsah přiloženého media

- `readme.txt` ..... stručný popis obsahu media
- `desert-rally-web.war` ..... webový archiv aplikace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
  - `map-editor` ..... skript pro vytváření stylů herní mapy
  - `lectures` ..... citované přednášky
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF