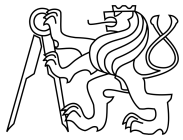
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Návrh a implementace backendu pro hru Backgammon
Student:	Václav Lipert DiS.
Vedoucí:	Mgr. Jan Dolejš
Studijní program:	Informatika
Studijní obor:	Softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Cílem práce je navrhnout a implementovat back-end pro hru Backgammon.

Postupujte dle následujících kroků:
1. Popište princip hry, specifikujte požadavky na vrstvu herní logiky ve formě případů užití.
2. Navrhněte SW architekturu celého řešení.
3. Navrhněte strukturu datového úložiště.
4. Na základě případů užití z bodu 1 navrhněte vrstvu herní logiky.
5. Implementujte datové úložiště na platformě MS SQL server.
6. Zvolte vhodný framework pro tvorbu serverů MMO her na platformě Java nebo C# a implementujte herní logiku.
7. Celé řešení zdokumentujte a vhodným způsobem otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 11. února 2016

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

Bakalářská práce

Návrh a implementace backendu pro hru Backgammon

Václav Lipert

Vedoucí práce: Mgr. Jan Dolejš

17. května 2016

Poděkování

Tímto bych rád poděkoval vedoucímu mé práce, Mgr. Janu Dolejšovi, a dalším zástupcům společnosti BGLab s.r.o. za konzultace a rady, které mi pomohly tuto práci dokončit.

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. Dále prohlašuji, že jsem s Českým vysokým učením technickým v Praze uzavřel dohodu, na základě níž se ČVUT vzdalo práva na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona. Tato skutečnost nemá vliv na ust. § 47b zákona č. 111/1998 Sb., o vysokých školách, ve znění pozdějších předpisů.

V Praze dne 17. května 2016 .....

Odkaz na tuto práci

Lipert, Václav. *Návrh a implementace backendu pro hru Backgammon.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato bakalářská práce se zabývá analýzou, návrhem a implementací backendového systému pro hru Backgammon pro firmu BGLab. Analýza je tvořena na základě požadavků definovaných zástupci firmy BGLab. Cílem je navrhnout a implementovat datový model, herní logiku a síťovou komunikaci s klientskou aplikací. Pro provoz serverů je využito cloudového řešení Azure, které zajišťuje i provoz databáze. Další technologie, kterých je v řešení využito, jsou Java, Hibernate, SmartFox server, MS SQL. Unikátnost řešení je v možnosti provozování klientské aplikace na všech nejrozšířenějších desktopových a mobilních platformách. Dále je kladen důraz na vysokou úroveň zabezpečení a spolehlivost komunikace mezi klientskou aplikací a serverovou částí. Součástí příloh této práce jsou kompletní zdrojové kódy a DDL skripty řešení.

Klíčová slova Backend, analýza, návrh, implementace, online hra, Backgammon, Java, MS SQL, cloud

Abstract

This thesis deals with analysis, design and implementation backend system of the Backgammon game for the BGLab company. The analysis is based on the requirements defined by the BGLab representatives. The goal is to design and implement a data model, game logic and network communication with client application. Azure is used as a platform for running servers and it is also used for running the database. Another technologies, which are used in the solutions, are Java, Hibernate, SmartFox server, MS SQL. The uniqueness of the solution is the possibility of running the client application on all major desktop and mobile platforms. Furthermore, the emphasis is on a high level of security and reliability of communication between the client application and server components. The appendixes of this work contains complete source code and DDL scripts of the solution.

Keywords Backend, analysis, design, implementation, online game, Backgammon, Java, MS SQL, cloud

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Herní průmysl ve světě zábavy nabývá stále většího významu. Zejména hry, kde spolu hraje více hráčů, jsou v posledních letech hlavně díky rozvoji internetu velmi populární. V tomto oboru je neustále prostor pro to přijít s něčím novým, nebo alespoň lepším, než je stávající nabídka her. Hra Backgammon, kterou se hodlám zabývat, je jednou z nejstarších her na světě a v některých zemích světa je neustále velmi populární. I přes relativně jednoduchá pravidla v sobě skrývá nepřeberné množství herních situací. Z těchto důvodů a z důvodů, že se o problematiku síťových her delší dobu zajímám, jsem se společně s firmou BGLab s.r.o. rozhodl zvolit jako téma své bakalářské práce implementaci herního serveru hry Backgammon.

Kapitola 1

Cíl práce

Mým úkolem je navrhnout, implementovat a otestovat serverovou část včetně datového modelu pro databázi, do které bude serverová část ukládat data. V této kapitole bych rád přiblížil strukturu projektu, současný stav jiných řešení a motivace projektu.

1.1 Struktura projektu

Celý projekt online hry Backgammon je rozvržen na tři hlavní části. Klientskou část, webovou část a herní server. Klientská část obsahuje grafické rozhraní. Hlavní obrazovky tvoří registrace, přihlášení, lobby a hra. Dále obsahuje komponentu pro komunikaci s herním serverem. Celá část je vyvíjena výhradně v prostředí Unity 3D, přičemž využívá některých dalších komponent pro Unity 3D. Webová část se dělí na část veřejnou a část neveřejnou. Veřejnou část tvoří webová prezentace a některé další formuláře umožňující správu hráčova účtu (např. formulář Zapomenuté heslo). Neveřejná část umožňuje přístup technických uživatelů k profilu hráčů, popř. herním datům. Uživatelé této části jsou primárně z technické podpory, popř. jsou to jiní pověření uživatelé. Poslední částí celkového řešení je herní server. Ten se stará o evidenci hráčů, evidenci her a obsahuje veškerou herní logiku pro hraní her. Já se ve své práci zabývám výhradně touto částí, ale na jiné části se v různých místech odkazuji tak, aby byl jasný kontext. Detailní požadavky na část herního serveru jsou uvedeny v kapitole 2.2.

Dále je projekt naplánován na několik dílčích etap. Ve své práci se zabývám výhradně první etapou s možným výhledem do dalších etap. První etapa z pohledu herního serveru zahrnuje analýzu, návrh a implementaci základní herní mechaniky. V dalších etapách je plánována implementace dalších typů her backgammon jako jsou např. Chouette, Nackgammon apod. (viz 2.1.3), návrh a implementace napojení webové části na herní data, rozkládání zátěže herních serverů a hledání nejvýhodnější lokace herního serveru vzhledem k časové prodlevě při komunikaci serveru s klientským zařízením atd. Další

budoucí neméně zajímavou oblastí je implementace automatického protihráče s pokročilou analýzou tahů.

1.2 Současný stav řešení problematiky

V současné době existuje několik uzavřených implementací online hry backgammon, ale žádná z nich nesplňuje zadání přenositelnosti na všechna nejpoužívanější zařízení. Jednou z implementací je open source řešení - GNU Backgammon. Tato implementace obsahuje základní herní logiku a hru s automatickým protihráčem (botem). Neobsahuje ale online hru a ani nepodporuje přenositelnost např. na mobilní platformy. Jednou z nejkvalitnějších implementací hry backgammon je eXtreme Gammon [1]. Tato implementace obsahuje i podrobnou analýzu již odehraných her s poukázáním na hráčovy chyby. Ovšem také neobsahuje online hru a je tak spíš určená ke zdokonalování hráčských dovedností. Dále existuje několik online serverů [2], které se zabývají hraním backgammonu, ale tyto zpravidla mají hru backgammon jako doplňkovou k ostatním hrám a navíc umožňují hrát pouze pomocí webového prohlížeče nebo klienta určeného pro jednu konkrétní platformu.

1.3 Motivace projektu

Z předchozí sekce vyplývá, že na trhu není příliš kvalitních backgammon online her. A to právě hodlá projekt firmy BGLab změnit. Cílem je implementace herního klienta pro nejpoužívanější desktopové i mobilní platformy. Hlavní zaměření je na země s vyšší popularitou hry backgammon. K seznamu těchto zemí lze dojít např. přes historickou tradici v těchto zemích [3] nebo přes vyhledání pořadatelů větších turnajů a složení hráčů na těchto turnajích. Dle hráčů, kteří se těchto turnajů zúčastní, jde např. o Velkou Británii, Japonsko, USA, Izrael, Čínu, Německo, Francii apod. Cílem je využít a rozvíjet popularitu ať už v těchto zemích, nebo i dalších, kde backgammon tak populární není. Měla by tomu pomoci široká základna hráčů, široká přenositelnost herního klienta, nabídka nejpoužívanějších typů her, později i podpora v médiích apod. Žádné stávající řešení nemá všechny tyto atributy.

Kapitola 2

Analýza

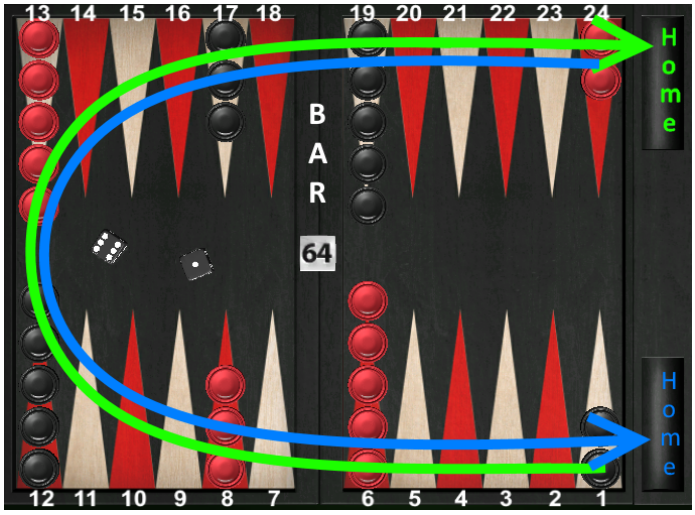
V této kapitole bych rád seznámil čtenáře se hrou backgammon a jejími pravidly, dále provedl analýzu formou sběru požadavků a navržení případů užití s využitím UML diagramů [4].

2.1 Hra backgammon

Historie hry backgammon (nebo-li česky také vrhcáby) sahá až do pátého tisíciletí před naším letopočtem a jedná se tedy o jednu z nejstarších deskových her vůbec. Z té doby totiž byly nalezeny na území někdejší Sumerské říše hrací desky, jež mají s hrací deskou Backgammon společné prvky [3, s. 7]. Kdy přesně se zformovala pravidla tak, jak je známe dnes, není a asi ani nebude nikdy úplně jasné [3, s. 13]. Původ hry je pevně spjat s egyptskou hrou Senet [3, s. 11], jež pochází z let okolo roku 3000 př. n. l. Další vývoj předchůdců této hry probíhal společně s šachy v Číně a Indii. Za podobnou hru lze považovat indickou parcheesi (14. st. n.l.), která měla některá pravidla odlišná: byla pro čtyři hráče a každý hráč měl na desce svůj vlastní kvadrant. Rovněž kameny měly výchozí pozice mimo desku. V Evropě se podobná forma hry poprvé objevila v Řecku (zmínil se o ní např. Platón ve 4. století př. n. l.) a hlavně v Římě (hra zvaná Ludus duodecim scriptorum [3, s. 17]). Odtud se pozvolna dostala i do západní Evropy. Zde byl zaznamenán největší rozmach hry okolo 16. - 17. st. n.l. a to především v Dánsku, Francii, Německu, Británii. Právě v Británii z roku 1645 pochází první použití názvu hry, tak jak ho známe dnes: „backgammon“ [3, s. 34]. S psanými pravidly poprvé přišel Edmund Hoyle v roce 1743 [3, s. 36].

2.1.1 Pravidla hry

Pravidla hry backgammon [5],[3] jsou relativně jednoduchá, i když pro některé začátečníky ne úplně hladce pochopitelná. V základní variantě je hra určená pro dva hráče, přičemž hráči sedí naproti sobě. Každý hráč má k dispozici



Obrázek 2.1: Hrací deska [6] se znázorněním směru pohybu obou hráčů a výchozími pozicemi kamenů

15 hracích kamenů, dvě hrací kostky a jednu násobící kostku. Cílem hráče je přesunout všechny svoje hrací kameny z výchozích pozic do svého domku (oblast mimo hrací pole). Každý hráč má svůj vlastní směr, ve kterém kameny přesunuje. Obrázek 2.1 ukazuje směry obou hráčů, přičemž modrý hráč má červené kameny a zelený hráč černé kameny.

Herní deska Herní desku tvoří čtyři hlavní kvadranty, 6 pozic pro hrací kameny v každém kvadrantu (celkem tedy 24 pozic), další dvě pozice jako domov pro každého hráče a jedna společná pozice pro vyhozené kameny (bar, dělící čára). Z obrázku 2.1 je patrná nejčastěji používaná výchozí pozice kamenů, pozice baru a jednotlivých domků hráčů (Home).

Hod kostkami V každém kole hází hráč oběma kostkami najednou. Hodnoty na kostkách pak může využít k posunu svých kamenů o příslušný počet polí, přičemž musí využít maximální počet kostek. Pokud na obou kostkách padnou dvě stejné hodnoty, hráč může tuto hodnotu použít celkem čtyřikrát. Je to tedy stejné, jako kdyby padly čtyři kostky se stejnou hodnotou. Kolo končí, pokud je vyčerpán maximální možný počet kostek.

Vyhazování kamenů a vracení do hry Pokud má hráč na dané pozici pouze jeden kámen, může být soupeřem vyhozen, pokud má validní tah některým ze svých kamenů na onu pozici. Kámen se pak odkládá na bar. Pokud má hráč umístěný některý svůj kámen na baru, musí nejprve tento kámen nasadit zpět. To se provádí přesunutím na pozici 25 − hodnota kostky. Pokud

na této pozici má soupeř minimálně dva kameny, nelze kámen nasadit a hráč musí čekat na další svůj tah.

Konec hry Hra končí, pokud kterýkoliv z hráčů přesune všechny svoje kameny do domku mimo hrací desku (tzv. vyvedení kamenů), odmítne nabídku násobení, vzdá se a nebo mu uplyne čas. Počet bodů, které jsou hráči za hru připsány, se určuje podle pozice soupeřových kamenů v době vyvedení posledního kamene. Pokud má soupeř vyvedený alespoň jeden kámen, jedná se o jednoduchou výhru (tedy $wp = 1 \times dv$, kde wp jsou vítězní body a dv hodnota násobící kostky). Pokud soupeř nemá vyvedený ani jeden kámen, ale všechny jeho hrací kameny jsou mimo výhercův domovský kvadrant (tzn. jsou někde v ostatních třech kvadrantech a žádný není na baru), jedná se o tzv. gammon ($wp = 2 \times dv$). Pokud neplatí ani jedno, jedná se o tzv. backgammon ($wp = 3 \times dv$).

Vyvádění kamenů Kámen je možné vyvést (tzn. přesunout do domku), pokud se všechny hráčovy kameny ve hře nachází v posledním kvadrantu před domkem. Je možné ovšem vyvést jen takový kámen, který má počet polí přesně do domku roven hodnotě na kostce, a nebo kámen, který je nejvzdálenější od domku.

Násobící kostka Násobící (nebo také doublovací) kostka přináší do hry zajímavý strategický prvek. Každý hráč, který má k dispozici násobící kostku, může nabídnout protihráči zdvojnásobení výsledných výherních bodů. Pokud tedy hráč zdvojnásobí hru a protihráč násobek přijme, už se nehraje o 1, 2 nebo 3 výherní body, ale o 2, 4 nebo 6 výherních bodů (podle typu výhry). Zároveň se násobící kostka přesunuje k němu a může s ní tedy disponovat jen on. Protihráč pak může v některých variantách hry výherní body znovu zdvojnásobit (4, 8, 12) a po přijmutí se kostka přesunuje opět zpět k původnímu hráči. Násobící kostka proto nabývá hodnot 2, 4, 8, 16, 32, 64.

Hra na čas V turnajových hrách se zápas omezuje na čas. Princip je obdobný jako v šachu. Hráč má na celý zápas přidělený maximální čas a ten se mu odečítá pokaždé, když je na řadě (hází kostkami, táhne, rozhoduje se v násobení), přičemž prvních několik sekund se hráči čas neodečítá - vždy tedy má určitý čas na tah, aniž by přicházel o celkový čas přidělený na zápas. Počet těchto „volných“ sekund se dohodne před začátkem hry, obvykle to bývá 12s.

2.1.2 Doplňková pravidla hry

Beaver Beaver (česky bobr) se používá hlavně při hraní her o peníze. Jde o rozšíření procesu zdvojnásobení počtu výherních bodů (doublování). Protihráč nemusí násobení jen akceptovat nebo odmítnout, ale může nabídnout

původnímu hráči další dvojnásobek ještě v rámci stávajícího tahu. Tázaný hráč pak může zdvojnásobení přijmout, odmítnout nebo dát tzv. Raccoon (viz. dále).

Raccoon Raccoon (česky mýval) navazuje na Beaver další možností zdvojnásobit výherní body ještě jako součást stávajícího tahu. Raccoon dává původně doublující hráč. Protihráč může reagovat přijetím nebo odmítnutím (pak hra končí).

Jacobyho pravidlo Používá se při hře o peníze. Pokud ve hře nebyla použita násobící kostka, počítá se gammon a i backgammon (viz. Výherní body 2.2.1) za jeden bod.

Crawfordovo pravidlo Používá se pouze při zápasu. Omezuje použít násobící kostku v první hře, ve které je kterýkoliv z hráčů jeden bod od vítězného počtu bodů.

2.1.3 Alternativní pravidla hry

V této sekci uvádím pouze nejznámější varianty hry, které by mohly být v budoucnu doimplementovány. Dalších variant je opravu mnoho [7].

Hypergammon Každý hráč má pouze tři kameny, které jsou na začátku hry rozmístěny na pozicích 24,23,22.

Nackgammon Liší se v počátečním rozmístění kamenů. Na rozdíl od základní hry jsou na počátku na pozici 6 umístěny pouze čtyři kameny, na pozici 13 také pouze čtyři kameny a na pozici 23 jsou umístěny dva kameny.

Chouette Hra tří a více hráčů, přičemž tým hraje proti jedinci. V týmu je vždy jeden hráč označen jako kapitán, který hraje (hází kostkou a posouvá kameny) a ostatní hráči ovládají každý svou násobící kostku. Samostatně pak mohou použít násobící kostku, popř. přijmout nebo odmítnout požadavek na zdvojnásobení od protihráče. Kapitán je na začátku hry hráč, který měl nejvyšší prvotní hod. Pokud padne při úvodním rozlosování mezi některými hráči stejná hodnota na kostce, tito určují svoji pozici v týmu už pouze rozlosováním kostkami s konkrétními hráči na stejné pozici. Spoluhráči v týmu mohou kapitánovi radit, ovšem předpokládá se, že kapitána hraje hlavně sám. Pokud hráč v týmu odmítne znásobení, hru končí a nesmí kapitánovi už radit. Pokud kapitán odmítne násobení protihráče, ztrácí pozici v týmu a na pozici kapitána postupuje další hráč dle pozic učených v počátečním rozlosování. Hra končí buď standardně, jako hra mezi dvěma hráči, a nebo když všichni hráči (popř. protihráč) odmítnou násobení.

2.2 Analýza požadavků

V této podkapitole uvádím slovník pojmů a analýzu požadavků pro herní server. Uvedené požadavky se týkají první etapy vývoje, která zahrnuje registraci a přihlášení uživatele, založení nové hry a možnost hrát hru (a dokončit) s dalším hráčem.

2.2.1 Slovník pojmů

Tah V rámci svého tahu hráč použije násobící kostku nebo rovnou hází hracími kostkami. Dále přesune své kameny dle hodnot hracích kostek a ukončí svůj tah sebráním hracích kostek z hrací desky.

Kolo Viz. Tah 2.2.1, případně může mít i význam ve smyslu kolo zápasu a kolo turnaje.

Hra Hra začíná úvodním hodem kostkami a končí, pokud nastanou podmínky k ukončení hry (viz. Konec hry 2.1.1).

Vítězné body Po každé hře vítězný hráč obdrží několik vítězných bodů, které se stanovují podle typu výsledku hry (viz. Konec hry 2.1.1).

Zápas Zápas je několik her v řadě. Končí, pokud nastanou předem dohodnuté podmínky, např. některý z hráčů dosáhne v součtu sedmi vítězných bodů nebo se hraje na pevně daný počet her apod.

Turnaj Několik kol zápasů. Dvojice hráčů jsou rozlosovány před každým kolem. Vyhrává hráč s nejvyšším počtem výher a vítězných bodů.

2.2.2 Funkční požadavky

F1 Evidence hráčů Systém bude evidovat hráče. Hráč se zaregistruje pomocí herního klienta, kde vyplní všechny povinné údaje. Systém nebude evidovat technické uživatele, ti budou evidováni v databázi webové části. Dále se může evidovaný uživatel přihlásit do systému pomocí uživatelského jména a hesla.

F2 Evidence účtů Ke každému hráči budou evidovány peněženky, přičemž každá peněženka bude pevně svázána s určitou měnou. V první etapě bude mít hráč pouze peněženku s herní měnou (tzv. playmoney), která bude vytvořena automaticky při registraci nového uživatele.

F3 Evidence her Systém bude evidovat všechny hry, ať už založené jednotlivými hráči, tak i automatizovaně založené nebo založené manuálně technickou podporou. Přihlášeným uživatelům server poskytuje seznam připravených nebo rozehraných her.

F4 Hraní hry Systém po výběru a odsouhlasení hry oběma hráči umožní hru těchto dvou hráčů, přičemž průběh hry bude ukládán do databáze. Zároveň během hry se průběžně kontrolují pravidla tak, aby nemohlo dojít k podvodům, jako jsou nesprávné a falešné tahy, nevalidní akce apod. Hráč musí být schopen prostřednictvím svého herního klienta hrát několik her najednou a architektura serverového řešení mu to musí umožnit. Systém v první etapě podporuje pouze základní hru (neobsahuje tedy pravidla alternativních her 2.1.3).

2.2.3 Nefunkční požadavky

N1 Komunikace s herním klientem v Unity3D Serverová část musí být schopna komunikovat s klientskou částí, která je implementována v prostředí Unity3D [8].

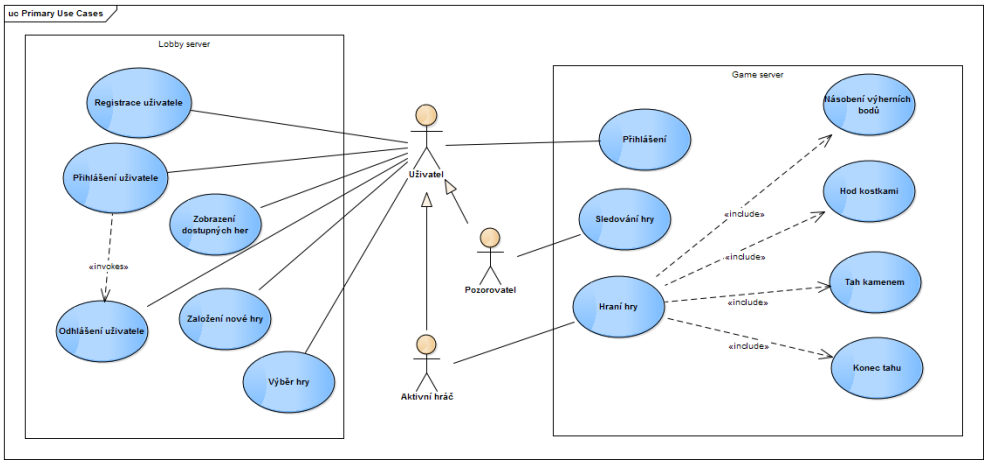
N2 Nízká latence Hráči musí být schopni hrát s minimálním zpožděním způsobeným síťovou komunikací i napříč různými kontinenty. Jednotlivé hry musí být spuštěny na serverech tak, aby tyto servery byly fyzicky vzdáleny od všech hráčů co nejméně. Architektura nesmí bránit přenesení a navázání hry z jednoho herního serveru na jiný (implementace bude předmětem dalších etap).

N3 Kontrola pravidel Velký důraz musí být kladen na kontrolu pravidel hry herním serverem tak, aby bylo znemožněno podvádění. Pravidla jsou implementována primárně na serveru, herní klient obsahuje jen některé fragmenty pravidel pro zajištění plynulosti hry.

N4 Bezpečnost Přenos citlivých dat musí probíhat šifrovanou komunikací s aktuálně dostupnou a bezpečnou šifrovací technologií.

N5 Využití komponent třetích stran V projektu je požadováno maximálního využití komponent třetích stran, které jsou licenčně kompatibilní. Dosáhne se tak kratšího času potřebného na implementaci řešení a sníží náklady na testování.

N6 Rozšiřitelnost Systém musí být rozšiřitelný o další typy her bez zásahů do architektury.



Obrázek 2.2: Diagram případů užití herního serveru

N7 Dostupnost Řešení musí být od začátku koncipováno jako řešení s vysokou dostupností 24x7. V případě výpadku kteréhokoliv ze serverů musí být systém schopný nahradit tento server jiným záložním serverem v řádu minut.

2.3 Případy užití

V následující sekci jsou popsány scénáře případů užití. Scénáře byly sestaveny na základě získaných požadavků (2.2) a dalších konzultací se zástupci společnosti BGLab. Diagram na obrázku 2.2 obsahuje vazby mezi aktéry, případy užití a částmi herního serveru. Uživatel je jediným aktérem v systému z pohledu herního serveru, může ovšem vystupovat v roli aktivního hráče (účastní se aktivně hry) nebo pozorovatele (sleduje hru jiných aktivních hráčů).

UC1 Registrace uživatele Neregistrovaný uživatel zašle povinné údaje registrace: uživatelské jméno, heslo, email, státní příslušnost, telefonní číslo. Dále zvolí bezpečnostní otázku a bezpečnostní odpověď. Tyto dva údaje budou použity jako doplňující údaje při procesu obnovy hesla (bude mít na starosti webová část). Server tyto údaje zkontroluje a provede registraci. O registraci je proveden záznam do auditního logu a vrácen výsledek uživateli.

UC2 Přihlášení uživatele Registrovaný uživatel je oprávněn se do systému přihlásit pomocí uživatelského jména a hesla. Systém povolí přihlášení pouze pokud uživatel uvede správné heslo. Pokud už je uživatel v systému veden jako přihlášený, provede se nejprve automatické odhlášení z původního zařízení.

UC3 Odhlášení uživatele Přihlášený uživatel má možnost se ze systému odhlásit. Systém zruší z paměti server záznam o tom, že je přihlášený.

UC4 Zobrazení dostupných her Přihlášenému uživateli je umožněno získat ze serveru seznam připravených a probíhajících her.

UC5 Založení nové hry Přihlášený uživatel založí novou hru. Dále čeká, dokud se jiný hráč do hry nepřipojí pomocí UC6 Výběr hry.

UC6 Výběr hry Přihlášený uživatel vybere jednu hru, kterou chce hrát. Pokud je tato hra rozehraná, stává se z něj v kontextu této hry pozorovatel. Pokud hra rozehraná není, stává se z něj v kontextu této hry aktivní hráč.

UC7 Sledování hry Hráč v roli pozorovatele si vybere rozehranou hru. Herní server mu umožní sledovat danou hru, tzn. bude pozorovateli odesílat veškeré události o hře, ovšem pozorovatel do ní nemůže nijak zasahovat.

UC8 Hraní hry Hraní se skládá z několika dalších dílčích případů užití, které na sebe vzájemně navazují. Hra je iniciována úvodním hodem kostkami, kdy server provede hod kostkou pro každého hráče a ten s vyšší hodnotou na kostce začíná. Dále hra pokračuje rovnou UC8.2 Přesunutím kamenu.

UC8.1 Hod kostkami Aktivní hráč iniciuje hod kostkami. Herní server zkontroluje, zda je hráč oprávněn hodit kostkami a dále vygeneruje dvojici náhodných čísel, kterou pošle klientovi. Ten pak pokračuje UC8.2 Přesunutím kamenu.

UC8.2 Přesunutí kamenu Aktivní hráč iniciuje přesunutí kamenu. Herní server zkontroluje, zda je přesun kamene legitimní dle daných pravidel hry a provede přesun kamene (kamenů).

Alternativní scénář Pokud se aktivní hráč ve svém tahu zmýlí nebo si ho rozmyslí, je oprávněn přesun kamenem vrátit zpět. V takovém případě herní server zkontroluje, zda hráč předtím opravdu provedl tento tah a provede vrácení kamenů tak, jak byly před zrušeným tahem.

UC8.3 Násobení Aktivní hráč může před hodem kostkami provést zdvojnásobení výherních bodů, pokud není násobící kostka u protihráče. Ten může násobení přijmout (pak hra pokračuje automaticky hodem kostek) nebo zamítnout, pak hra končí, přičemž platí poslední hodnota na násobící kostce před násobením. U některých typů her může hráč ještě provést okamžité „znovu-zdvojnásobení“ (beaver) a protihráč může provést ještě další „znovu-zdvojnásobení“ (raccoon). Další úrovně už možné nejsou.

UC8.4 Konec tahu Pokud aktivní hráč vyčerpal všechny možné přesuny hracích kamenů v souladu s hodnotami na hracích kostkách, může provést ukončení kola. Herní server zkontroluje, zda je hráč oprávněn ukončit tah (je ve správném stavu a vyčerpal všechny přesuny kamenů) a nastaví druhého hráče jako hráče na tahu.

Kapitola 3

Návrh

Následující kapitola se zabývá výběrem z možných řešení, popisem navrhované architektury řešení, popisem komponent herního serveru, návrhem doménového modelu a některými stavovými diagramy popisujícími herní stavy.

3.1 Možná řešení

Vzhledem k požadavku N5 (využití komponent třetích stran) jsme se společně s vedoucím práce snažili nalézt platformu, která by nám urychlila vývoj herního serveru. Je k dispozici několik řešení, která implementují základní stavební prvky herních serverů. Jejich hlavním cílem je vyřešit síťovou komunikaci klienta se serverem a implementovat některé základní funkčnosti, jako jsou např. přihlášení, vytváření herních místností, připojení do herních místností, chat mezi hráči apod. Tyto frameworky bývají rozděleny na klientskou a serverovou část, přičemž klientských implementací je více, pro každou podporovanou platformu jedno. Serverová část už je implementována pro jednu konkrétní platformu, což ale nijak neodporuje požadavkům.

Po vyloučení řešení, která nejsou vhodná pro naše zadání (např. nesplňují podporu klientské implementace pro Unity 3D (požadavek N1) apod.), zbyly ve výběru herního online serveru dva hlavní kandidáti: Photon server [9] a Smartfox server [10]. Obě řešení jsou komerční, z čehož vyplývá, že od určitého počtu současně připojených hráčů je potřeba zakoupit licenci. Photon server je implementován v jazyce C# a jeho síťové jádro v C++. Je zaměřen zejména na podporu síťové komunikace (TCP, UDP). Logiku herních místností a lobby je potřeba doprogramovat pomocí tzv. serverů, což jsou logické celky, které obsluhují nějaký logický celek komunikace (lobby, chat, hru, přihlášení apod.). Smartfox server je implementován v Javě. Zaměřuje se jak na podporu síťové komunikace (TCP, UDP, http, https), tak již v základním řešení obsahuje podporu pro přihlašování, správu zón a místností v lobby. Stávající implementace lze do jisté míry ovlivňovat pomocí tzv. extensions, což jsou moduly, který pomocí událostí komunikují se Smartfox serverem. Lze

v nich ovlivňovat mimo jiné proces přihlášení, správu účastníků v místnostech, samotnou herní logiku apod.

Další oblastí, kterou je v rámci návrhu řešení potřeba vybrat, je platforma, na které bude řešení provozováno. Jelikož ze zadání vyplývá, že řešení musí být provozovatelné v několika světadílech bez znatelných prodlev v síťové komunikaci (nefunkční požadavek N2), jako nejvýhodnější (alespoň prozatím) se jeví využít tzv. cloudových služeb. Ty nabízí mimo jiné provozování vlastních virtuálních serverů v různých datacentrech po celém světě a jsou tak pro naše řešení vhodné. Cloudové řešení také napomáhá splnění nefunkčního požadavku N7, jelikož je možné mít připravené záložní servery s minimálními náklady. V rámci dalších služeb je pak poskytován i databázový engine a databázový prostor, což je pro naše řešení také výhodné. Řešení by bylo možné provozovat i dalšími způsoby, jako např. pronájem serverů u lokálních poskytovatelů a provozování vlastní databáze, ovšem po zvážení těchto alternativ jsme přeci jen přistoupili k provozování v cloudové službě. Hlavní důvod byl kratší harmonogram projektu v případě volby tohoto řešení. Ovšem herní server je vyvíjen tak, aby nebyl závislý na jediné konkrétní cloudové službě a byl přenositelný buď na jinou, nebo ho bylo možné provozovat vlastními (popř. pronajatými) technickými prostředky.

V neposlední řadě je potřeba navrhnout datový model tak, aby splňoval požadavky na hru. Ten bude vytvořen pomocí analytického přístupu sběr požadavků, sestavení doménového modelu a návrh fyzického datového modelu s využitím modelovacího jazyka UML 2 [4]. Vzhledem k použitému cloudovému řešení bude fyzický datový model koncipován pro Microsoft SQL Server 2012. Nicméně díky použití ORM frameworku bude serverová část nezávislá na konkrétní databázové technologii a bude ji případně možné provozovat na jiných databázových technologiích beze změny implementace. Pak by bylo samozřejmě nutné tomu přizpůsobit i fyzický datový model. Další oblasti k řešení jsou např. implementace generátoru náhodných čísel, zabezpečení přihlášení, šifrování komunikace, automatický hráč pro testovací účely apod.

3.2 Vybraná řešení

Po dlouhém zvažování jsme spolu s vedoucím práce vybrali nakonec Smartfox server, který je pro naše řešení vhodnější. Implementuje totiž více činností, které využijeme a museli bychom je tak jako tak implementovat. Také podpora produktu nám přišla na vyšší úrovni. Vyplývá z toho ovšem nutnost využít pro implementaci herní logiky a herních služeb jazyk Java, jelikož v jiném programovacím jazyce nelze pro tento server rozšíření (extensions) programovat.

Po výběru základních serverových technologií jsme mohli přistoupit k výběru cloudové platformy. Na základě srovnání jak cenového, tak i nabízených možností, jsme vybrali řešení Azure od firmy Microsoft. To vyšlo finančně výhodnější hlavně z důvodu možnosti využívat vývojové a testovací

prostředí zdarma (v rámci programu BizSpark [11]), což je pro projekt značná úspora nákladů. Ovšem jak už je uvedeno výše, řešení je možné provozovat i na jiných cloudových platformách jako např. Google Cloud Platform nebo Amazon Webservices. Cenové porovnání těchto tří globálních cloudových platforem je umístěno formou dynamické excelovské tabulky na přiloženém CD *(cloud_kalkulace.xlsx)*.

ORM framework Hibernate [12] jsem vybral čistě z důvodu mých osobních zkušeností a preferencí. Používal jsem ho na několika projektech a už s ním mám jisté zkušenosti.

Jako generátor náhodných čísel jsme vybrali implementaci Fortuna [13] hlavně pro své široké možnosti výběru zdrojů entropie.

3.3 Architektura

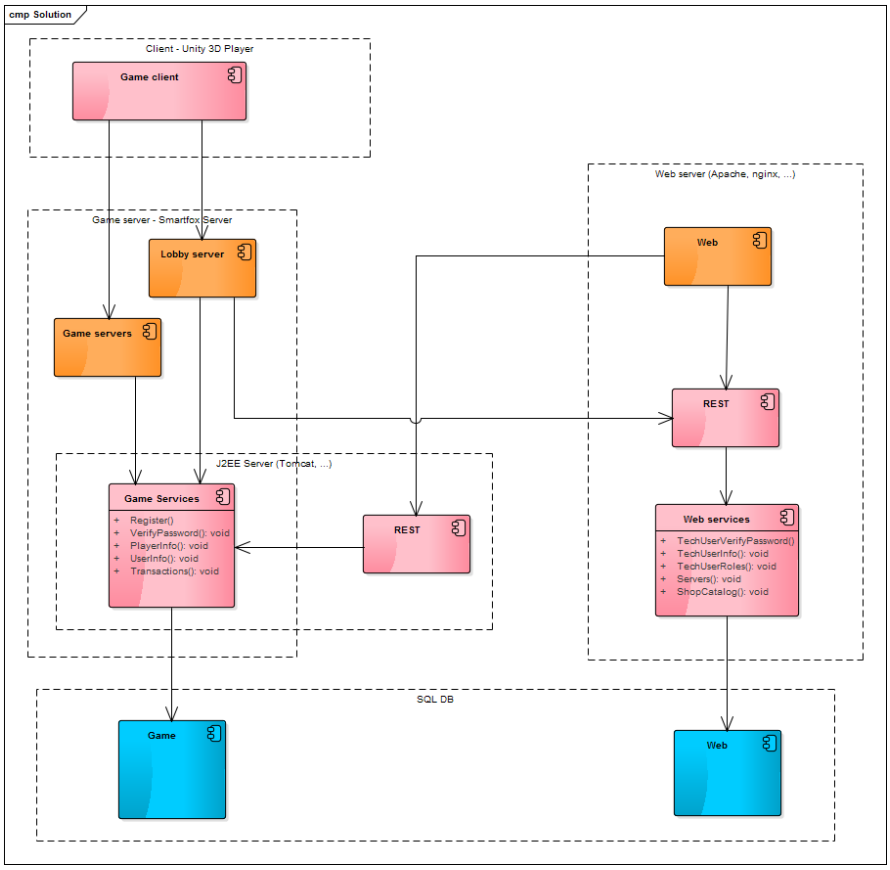
Diagram architektury na obrázku 3.1 zobrazuje hlavní části celého budoucího systému a vazby mezi nimi. Všechny tyto části budou provozovány ve třech oddělených prostředích: vývojovém, testovacím a produkčním. Následuje popis jednotlivých částí diagramu.

Lobby server je reprezentován komponentou bgmon-lobby-server 3.4 a je provozován v běhovém prostředí Smartfox serveru. Udržuje seznam všech připravených a probíhajících her prostřednictvím herních místností. Je v provozu vždy právě pouze jeden pro jedno prostředí. V případě výpadku je k dispozici záložní lobby-server, který obnoví seznam herních místností pomocí perzistentních záznamů her v databázi. Dále udržuje seznam aktivních herních serverů aby mezi nimi mohl rozkládat zátěž (součást dalších etap projektu).

Game server (herní server) je reprezentován komponentou bgmon-game-server 3.4 a stejně jako Lobby server je provozován v běhovém prostředí Smartfoxu. Ovšem herních serverů může běžet paralelně víc, cílově se počítá s jedním serverem na jeden region, ale počet serverů a v jakých regionech budou, bude možné škálovat podle aktuálních provozních potřeb.

Game DB (herní databáze) obsahuje informace o hráčích, jejich peněženkách, o turnajích, hrách, auditní log hráče a další herní informace.

Game services (herní server) je modul reprezentovaný komponentou bgmon-server-services a je jediným prostředníkem mezi herní databází a okolím. Herní server s tímto modulem pracuje napřímo bez dalších komunikačních rozhraní, aby bylo dosaženo co nejmenší latence (požadavek N2). Nad tímto rozhraním bude postaveno REST rozhraní pro potřeby webové části, kde není taková potřeba držet nízkou latenci, ale naopak je vítáno oddělení od technologií Java a Smartfox.



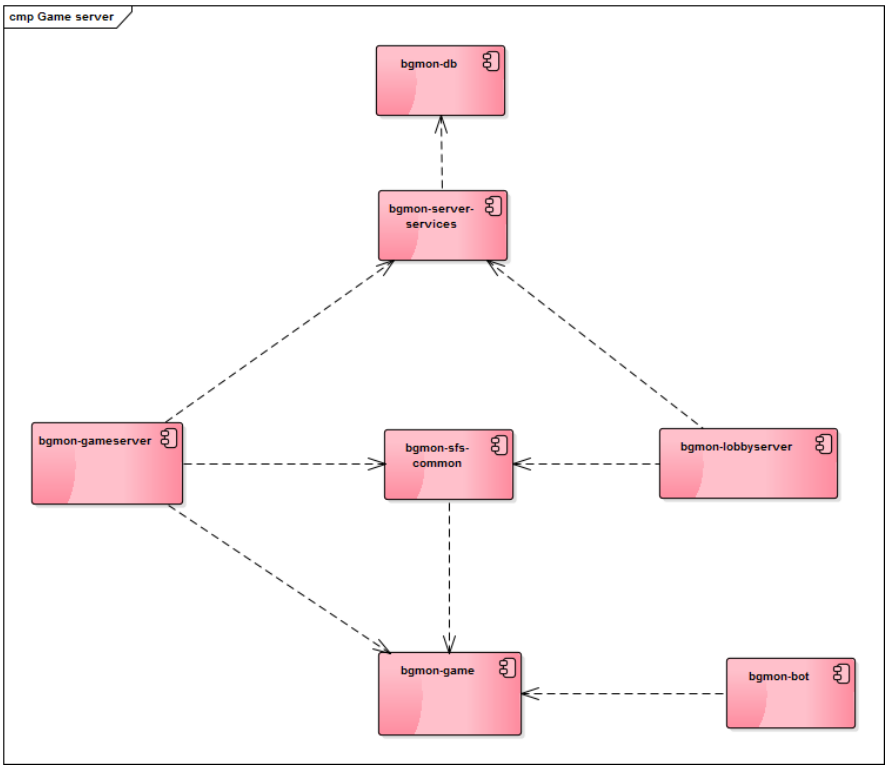
Obrázek 3.1: Základní architektura game serveru a vazby na ostatní části

Herní klient je zařízení, na kterém běží klientská část. Ta obsahuje komunikační knihovnu od tvůrců Smartfox serveru, která komunikuje s lobby serverem a herními servery.

Webová část a webová databáze se stará o webovou prezentaci a o některé dílčí činnosti správy hráčova účtu. Také slouží jako hlavní nástroj technické podpoře, jelikož pověřeným technickým uživatelům umožňuje nahlížet na hráčovy údaje. V době psaní této bakalářské práce probíhá analýza této části.

3.4 Model komponent

Obrázek 3.2 obsahuje komponentní diagram herního serveru. Vysvětleme si v následujících odstavcích jejich význam.



Obrázek 3.2: Komponenty game serveru a závislosti mezi nimi

bgmon-game Tato komponenta obsahuje základní herní logiku a třídy reprezentující jednotlivé části hry (hráč, herní deska, násobící kostka apod.). Komponenta je koncipovaná tak, že je přenositelná, tzn. není nijak vázaná např. na Smartfox server nebo jiné běhové prostředí.

bgmon-lobby-server Komponenta reprezentující lobby server. Je implementována formou rozšíření pro Smartfox server a je tedy závislá na jeho běhovém prostředí. Zprostředkovává komunikaci s klientem pro přihlášení, registraci, poskytování seznamu herních místností a výběr hry. V budoucnu bude poskytovat např. i chat hráčů.

bgmon-game-server Komponenta reprezentující herní server. Jakmile jsou oba hráči připraveni odstartovat hru (odehrává se na lobby serveru), tak lobby server předá klientům údaje o serveru, na kterém se hra bude odehrávat. Klienti se na tento server připojí a požádají o start hry. Klient dále zůstává připojen na lobby server, ale samotná hra (nebo více her) už probíhá výhradně na herním serveru. Ten prostřednictvím rozšíření pro Smartfox server zajišťuje komunikaci s klientským zařízením. Po zpracování jednotlivých událostí volá knihovnu bgmon-game 3.4, která provádí samotnou herní logiku a kontroly.

bgmon-sfs-common Jedná se o komponentu, která sdružuje společné funkčnosti týkající se používání Smartfox serveru. Jedná se hlavně o serializaci a deserializaci datových objektů, pomocích kterých komunikuje klient se serverem.

bgmon-db Komponenta obsahuje databázové entity vygenerované pomocí Hibernate knihovny přímo z MS SQL databáze. Dále obsahuje pomocné funkčnosti pro zjednodušení práce s JPA/Hibernate.

bgmon-server-services Komponenta obsahuje základní služby pro práci s herní databází a je to také jediná komponenta, která přistupuje do databáze přímo (respektive s pomocí komponenty Hibernate). Jednotlivé služby jsou více rozepsány v kapitole Serverové služby 4.6. V budoucnu se na tuto komponentu napojí prostřednictvím REST rozhraní i webová část a komponenta tedy zůstane jediným místem pro přístup k herní databázi.

bgmon-bot Implementace automatického hráče pro testovací účely. Tato funkčnost není v požadavcích, avšak ulehčila testování herní mechaniky, jelikož nebylo nutné otevírat dvě herní okna nebo mít k dispozici druhého testovacího hráče. Komponenta bude dále rozvíjena až v dalších etapách projektu.

3.5 Doménový model

Následující podkapitola popisuje jednotlivé entity doménového modelu, který byl vytvořen na základě požadavků, slovníku a pravidel hry. Přehledový diagram je umístěn na obrázku 3.3.

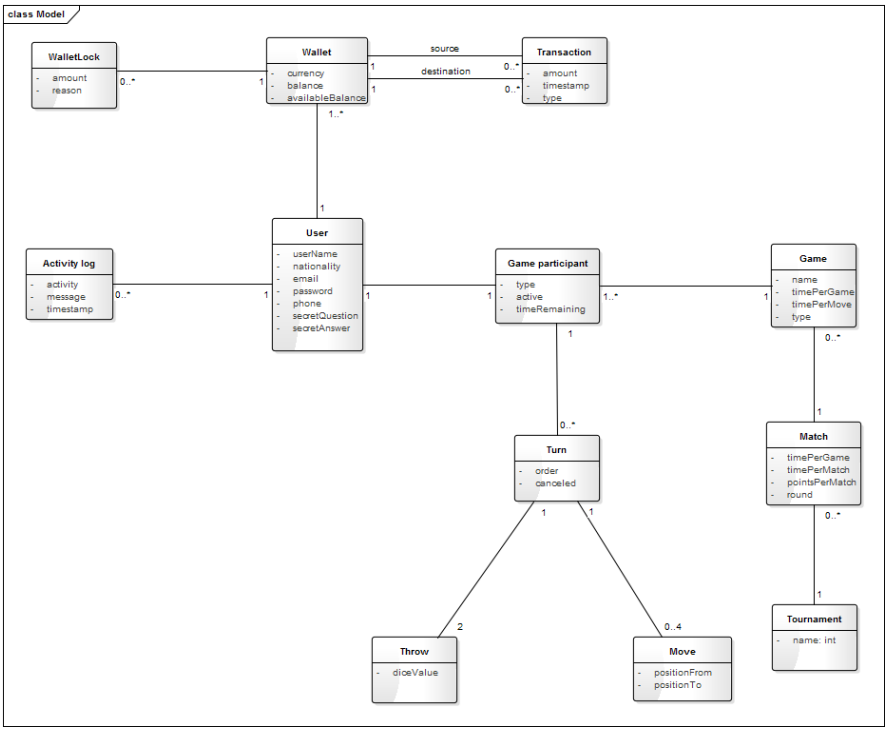
3.5.1 Entita User

Entita představuje zaregistrovaného uživatele v systému.

Atribut	Popis
userName	Uživatelské jméno
email	E-mailová adresa uživatele
password	Hash hesla + sůl (nikoliv tedy heslo samotné)
nationality	Státní příslušnost
phone	Telefonní číslo
secretQuestion	Bezpečnostní otázka při zapomenutí hesla (číselník)
secretAnswer	Odpověď na bezpečnostní otázky

3.5.2 Entita Wallet

Entita reprezentuje peněženku uživatele.



Obrázek 3.3: Doménový model

Atribut	Popis
currency	Měna peněženky
balance	Aktuální zůstatek peněženky
availableBalance	Disponibilní zůstatek peněženky (Aktuální zůstatek peněženky beze všech zamknutých částek)

3.5.3 Entita WalletLock

Entita reprezentuje blokovanou částku na peněžence.

Atribut	Popis
amount	Výše částky blokace
reason	Důvod blokace

3.5.4 Entita Transaction

Entita reprezentuje transakci mezi jednotlivými peněženkami.

Atribut	Popis
amount	Výše částky transakce
type	Typ transakce (peněženka x peněženka; systém x peněženka)
timestamp	Datum a čas, kdy transakce proběhla

3.5.5 Entita Activity log

Entita slouží k záznamu aktivity uživatele.

Atribut	Popis
activity	Typ aktivity (registrace, přihlášení, vstup do hry, dohrání hry, založení herní místnosti, atd.)
message	Volný text upřesňující aktivitu uživatele
timestamp	Datum a čas, kdy k události došlo

3.5.6 Entita Game

Entita představuje jednu konkrétní hru.

Atribut	Popis
name	Název hry
timePerGame	Celkový čas pro jednoho hráče na zápas (bez jednotlivých časů na kolo)
timePerMove	Čas v rámci jednoho tahu, po který nebude odpočítáván celkový čas na zápas
type	Toto pole je určeno k odlišení jednotlivých typů her.

3.5.7 Entita GameParticipant

Entita představuje účastníka hry: může jít o aktivního hráče nebo o pozorovatele.

Atribut	Popis
type	Typ účastníka hry Hráč; Pozorovatel
active	Příznak, zda je účastník hry aktivní
timeRemaining	Zbývající čas hráče

3.5.8 Entita Turn

Entita reprezentující jeden tah hry 3.5.

Atribut	Popis
order	Pořadí tahu v rámci hry. Jednotlivá pořadí se můžou opakovat, pokud bylo použito Undo tahu.
canceled	Zda tah byl zrušený pomocí Undo. Pokud dojde k ukončení tahu, musí existovat právě jeden záznam o tahu, který má nastaveno canceled na False

3.5.9 Entita Throw

Entita představuje hod hrací kostkou.

Atribut	Popis
diceValue	Hodnota hozené hrací kostky

3.5.10 Entita Move

Entita reprezentuje přesunutí hracího kamenu hráčem.

Atribut	Popis
positionFrom	Pozice, ze které byl hrací kámen přesunut
positionTo	Pozice, na kterou byl hrací kámen přesunut

3.5.11 Entita Match

Entita představuje jeden zápas (definice viz. 2.2.1)

Atribut	Popis
timePerGame	Celkový čas pro jednoho hráče na zápas (bez jednotlivých časů na kolo)
timePerMove	Čas v rámci jednoho tahu, po který nebude odpočítáván celkový čas na zápas
pointsPerMatch	Počet bodů potřebných k výhře zápasu
round	Kolo v rámci turnaje (pokud se jedná o zápas, který spadá po turnaj)

3.5.12 Entita Tournament

Entita představuje jeden zápas (definice viz. 2.2.1)

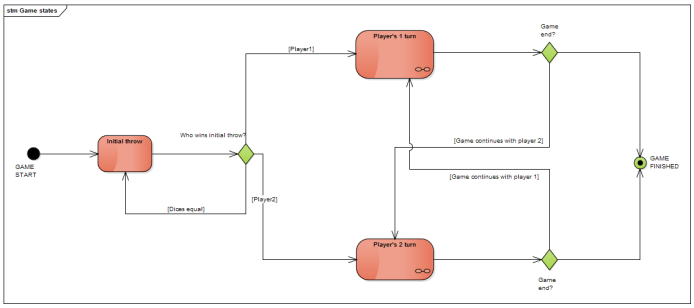
Atribut	Popis
name	Název turnaje

3.6 Herní logika

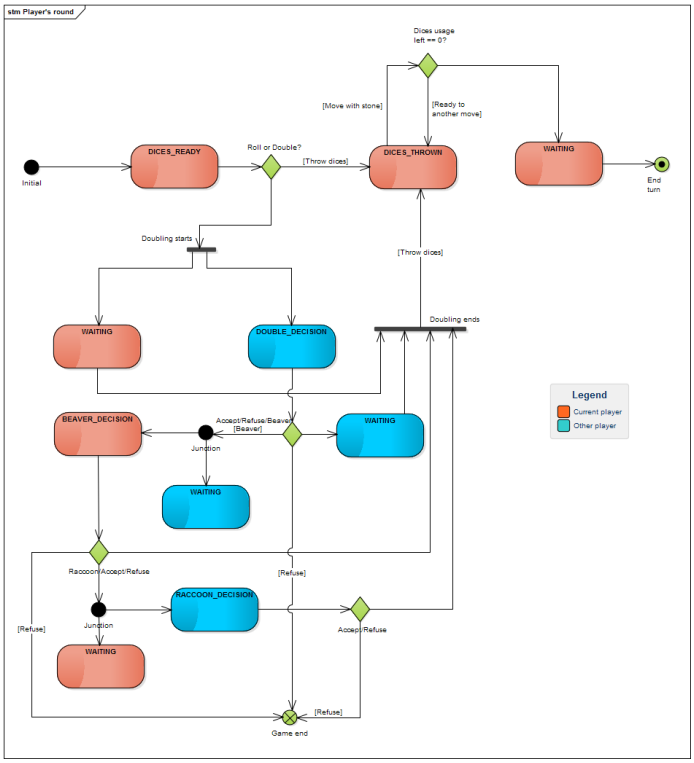
Část kontrol herních pravidel probíhá pomocí stavového stroje, kdy jsou striktně kontrolovány možné přechody mezi jednotlivými stavy. Následující sekce se snaží průběh přechodů těchto stavů popsat.

3.6.1 Stavový diagram průběhu hry

Stavový diagram na obrázku 3.4 zobrazuje průběh celé hry. Hra začíná úvodním hodem, podle kterého se rozhoduje, který hráč začíná. Pokud padnou stejná čísla, hází se znovu. Celý jeden tah hráče je vyobrazen na obrázku 3.5. Hráč je na začátku tahu ve stavu DICES_READY (kostky připraveny). Zde může hráč buď použít násobící kostku a nebo hodit kostkami. V případě použití násobící kostky hráč čeká na rozhodnutí druhého hráče. Ten může buď přijmout (pak původní hráč pokračuje hodem kostkami), odmítnout (pak hra končí a druhý hráč prohrává), a nebo dát tzv. Beaver (viz 2.1.3). Jde o okamžité další znásobení původnímu hráči, který může buď přijmout (pak hra pokračuje hodem kostkami), odmítnout (pak hra končí a původní hráč prohrává), a nebo může dát další okamžité znásobení druhému hráči - Raccoon



Obrázek 3.4: Stavový diagram hry

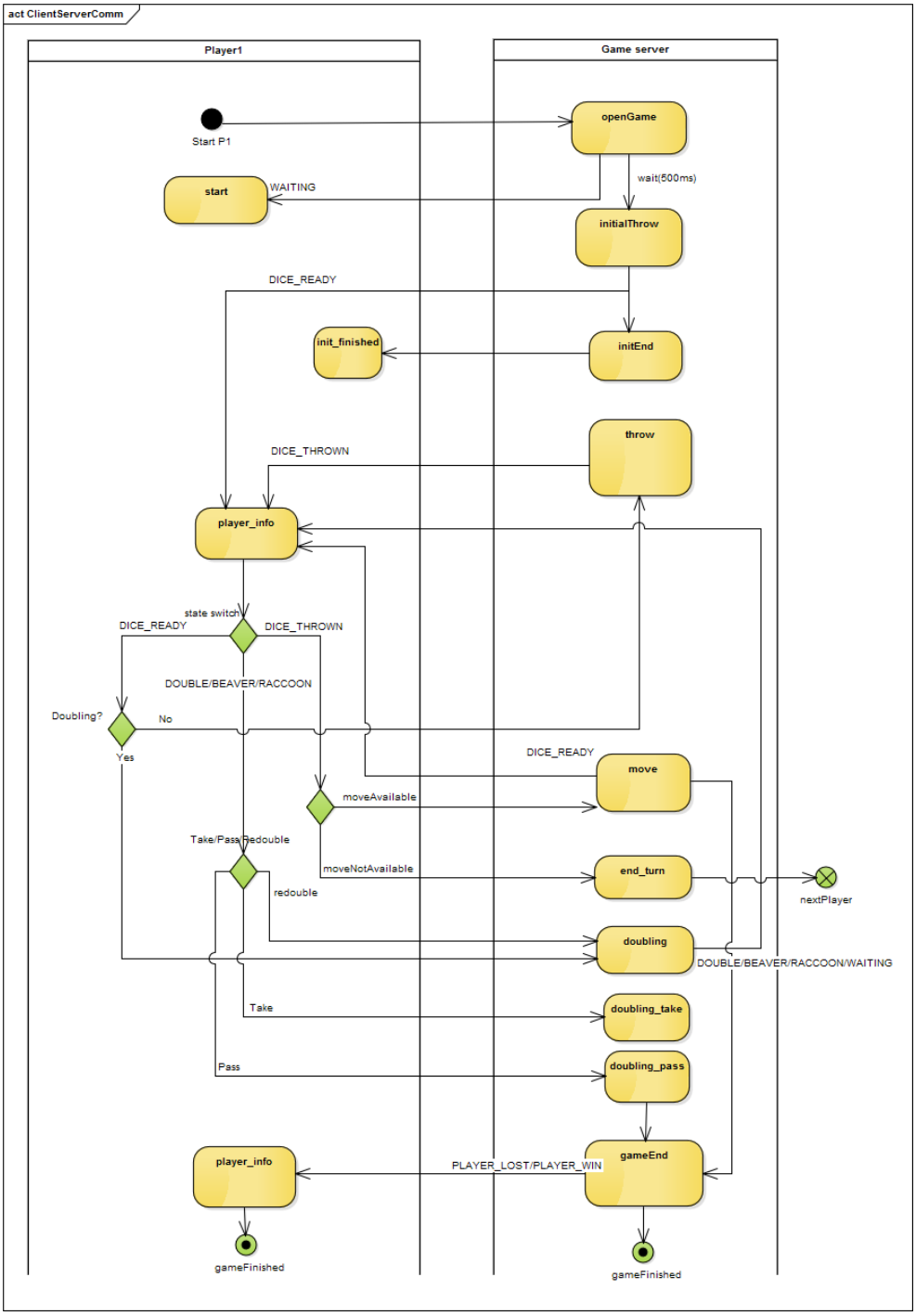


Obrázek 3.5: Stavový diagram hráče na tahu

(viz 2.1.3). V tom případě se opět rozhoduje druhý hráč a ten může buď přijmout, (pak hra pokračuje hodem kostkami), a nebo odmítnout, což znamená, že druhý hráč prohrává (platí poslední přijaté násobení a základní výhra, tzn. 1 bod). Po hodu kostek (stav DICES_THROWN) hráč pokračuje přesunutím svých hracích kamenů, až dokud nevyčerpá maximální možný počet použití hracích kostek. Pokud v některém z přesunů kamenů přesune poslední kámen do svého domku, vyhrává a hra okamžitě končí. Jinak tah končí ukončením tahu hráčem (v reálné hře sebere kostky z hrací desky a přepne hodiny na druhého hráče).

3.7 Komunikace klient/server

Diagram aktivit na obrázku 3.6 zobrazuje komunikaci klient/server. Pro udržení alespoň určité úrovně přehlednosti je uveden pouze případ pro hráče číslo 1, který zároveň vyhrál úvodní hod. Z aktivity *end_turn* by pak došlo k přepnutí na hráče 2, jehož komunikace je pak shodná jako tato. Velkými písmeny jsou uvedeny stavy hráče, které pak korespondují se stavy hráče na tahu 3.5. Aktivity pak představují volané funkce na klientovi, respektive na serveru.



Obrázek 3.6: Aktivity diagram komunikace klient/server

Kapitola 4

Realizace

V této kapitole se budu zabývat stručným popisem použitých technologií a některými zajímavými aspekty implementace herní logiky.

4.1 Použité technologie

Výběr použitých technologií proběhl na základě zvážení možných variant, které jsou popsány v sekci 3.1. Tato kapitola obsahuje jejich podrobnější popis.

4.1.1 Smartfox server

Smartfox server je běhové prostředí, ve kterém náš herní server běží. Toto prostředí přichází už s celou řadou funkčností, mezi které patří možnost připojení klienta přes síťové protokoly UDP/TCP/HTTP/SSL, implementovaný proces přihlášení, logické dělení serveru na tzv. zóny (uživatel se vždy přihlašuje do konkrétní zóny), správu herních místností, chat hráčů v rámci lobby nebo v rámci místnosti atd. Kompletní výčet vlastností je uveden zde [10].

Vlastní funkčnost (tedy např. herní logika) se pak doplňuje pomocí rozšíření ve formě .jar souborů obsahující třídy, které implementují různá rozhraní Smartfoxu. Server tyto třídy dle konfigurace serveru zaregistruje pro další použití při startu serveru. K dispozici je možnost rozšíření zóny a herní místnosti. Tato rozšíření si po inicializaci zaregistrují posluchače událostí Smartfox server, podle kterých pak vykonávají další činnost.

Smartfox má dvě základní edice: komunitní edici (kterou aktuálně využíváme pro vývojové účely) a komerční. Komunitní edice je omezena na maximální počet 100 současně přihlášených uživatelů. Ačkoliv je to pro vývojové účely plně dostačující, později počítáme s přechodem na komerční edici. Komerční edice má též výhodu v tom, že podporuje rozklad zátěže mezi více serverů, ovšem nepovedlo se mi o této funkčnosti dohledat více informací než jen kusé útržky v oficiálním fóru.

4.1.2 Java

Java je objektově orientovaný programovací jazyk, se kterým přichází i sada základních funkčností v podobě přenositelného runtime frameworku. Verze Javy, kterou využívá Smartfox server, je 1.7, takže tuto verzi je nutné využívat i pro naše účely. Pro účely správy závislostí balíčků je počítáno s využitím prostředí Maven, ovšem zatím vzhledem k nízkému počtu závislostí nebylo toto prostředí potřeba.

4.1.3 Hibernate ORM

Hibernate je komponenta pro objektově relační mapování. Umožňuje vývojářům jednoduše programovat napojení na databázový systém [12]. My využíváme její funkci vygenerování atributovaných entit z databáze a práci s entity managerem v JPA režimu. Dále jsou dodefinovány pomocí jazyka HSQL některé vyhledávací dotazy prostřednictvím atributů u jednotlivých entit.

4.1.4 Azure cloud

Azure je cloudové řešení od společnosti Microsoft. Používáme ho k provozování virtuálních serverů, na nichž je nainstalován OS Linux Debian 8.1, a k provozování databázového stroje (viz další odstavec). Azure nabízí celou řadu dalších služeb, ale jelikož nechceme být silně závislí na jedné konkrétní cloudové platformě, dlouhodobě uvažujeme pouze o využití těchto dvou služeb.

4.1.5 Microsoft SQL Server

MS SQL Server ve verzi 2012 používáme jako relační databázový systém. Jeho provoz je využíván v rámci Azure.

4.2 Implementace pravidel hry

V této kapitole se zaměřuji na část řešení kontroly herních pravidel z jedné herní situace. K přiblížení této herní situace si zopakujme pravidla přesunu kamenů. Kamenem je na cílovou pozici možné táhnout, pokud:

1. Hráč nemá žádný vlastní kámen vyhozený (na baru)

2. Cílová pozice obsahuje buď hráčovy kameny, žádné kameny nebo jeden soupeřův kámen (v takovém případě dojde k vyhození soupeřova kamene na bar)

3. Neexistuje jiný tah ve hře, který by umožňoval využít vyšší počet hozených hodnot

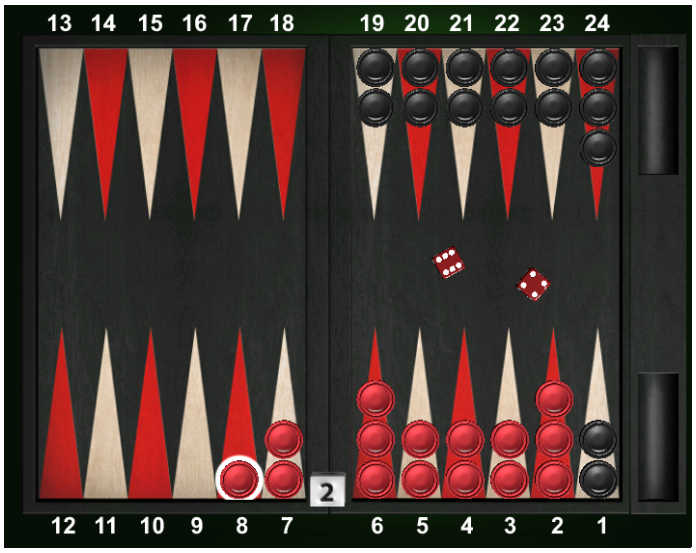
Vzhledem k výše uvedenému pravidlu 3 je v komponentě bgmon-game 3.4 implementován algoritmus k nalezení všech posloupností tahů, která vedou k maximálnímu počtu využití hodnot kostek. To se provádí sestavením stromu tahů, kde jednotlivé uzly stromu představují stav desky po provedeném tahu. Stav desky je tedy rozmístění herních kamenů, ale v paměti stačí udržovat pouze počet kamenů na jednotlivých pozicích. V každém stavu je pak potřeba vyhodnotit pomocí pravidel 1 a 2, zda je možný tah provést vzhledem k aktuální pozici kamenů. Mějme například herní situaci z obrázku 4.1. Zde je zřejmé, že buď můžeme táhnout pomocí kostky hodnoty 6 kamenem z pozice 8 na pozici 2. Následně můžeme táhnout pomocí kostky s hodnotou 4 z pozice 7 na pozici 3 a z pozice 6 na pozici 2. Také bychom mohli začít tah pomocí kostky s hodnotou 4 z pozic 8, 7, 6 na pozice 4, 3, 2, ale pokud bychom táhli jako první kamenem z pozice 8, kostku s hodnotou 6 už by nebylo možné využít. Proto tuto kombinaci tahů nemůže hráč použít. Existují i herní situace, kdy je možné táhnout buď jednou kostkou nebo žádnou kostkou, a kdy je nutné použít pouze určitou kombinaci tahů tak, aby byl vyčerpán maximální počet kostek. Strom pro hledání nejvyššího počtu použití kostek z této herní situace je pak na obrázku 4.1. Je z něj patrné, že v tomto případě lze využít obě kostky v libovolném pořadí. Obdobně se řeší situace, kdy padnou na kostkách dvě stejná čísla - kostky k použití jsou pak virtuálně 4. Situace je jednoduší v tom, že zde nezáleží na pořadí použití kostek, ale naopak výpočetně složitější v tom, že strom může dosáhnout až hloubky 4, což může v nejhorším případě vygenerovat až 15^4 uzlů. Algoritmus sestavování stromu ovšem zastaví zpracování, pokud už dojde k nalezení maximálního počtu kostek (v tomto případě 4). Reálně je tedy počet vygenerovaných uzlů řádově menší.

4.3 Generátor pseudo-náhodných čísel

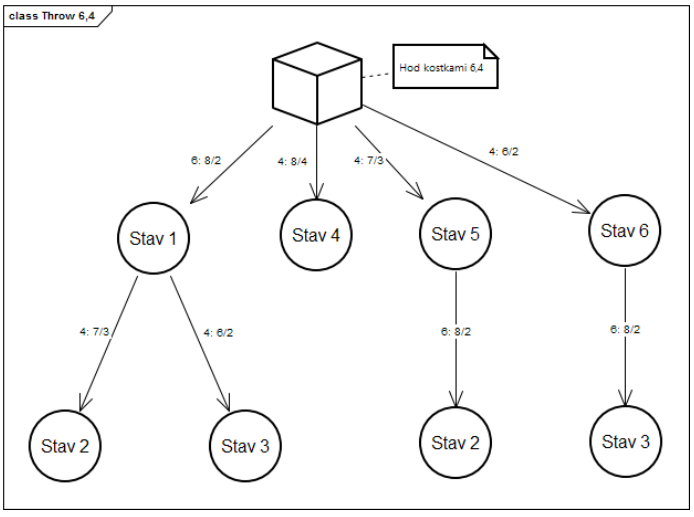
Náhodná složka je důležitou součástí hry, je proto nutné použít generátor pseudo-náhodných čísel s co nejrovnoměrnějším rozdělením náhodných čísel (ideálně rovnoměrné rozdělení). Aby toho bylo možné dosáhnout, je potřeba, aby generátor používal pro generování číselné řady co největší míru entropie. Standardní generátor dalšího čísla typu Integer má v Java runtime knihovnách následující implementaci [14]:

```
protected synchronized int next(int bits)
{
  seed = (seed * 0x5DEECE66DL + 0xBL) & ((1L << 48) - 1);
  return (int) (seed >>> (48 - bits));
}
```

Jedná se tedy svým zařazením o lineární kongruentní generátor [15, s. 10–26]. Jako vstupní seed používá tato standardní implementace pouze aktuální systémový čas v nanosekundách navýšený o číslo, které se při každém volání konstruktoru třídy Random inkrementuje o jedničku.



Obrázek 4.1: Herní situace I



Obrázek 4.2: Strom stavů herní situace I

Pokus č.	Počet iterací	JUR	JSS	FOR
1	1000000000	5.047	3.772	1.777
2	10000000	4.299	5.944	5.892
3	1000000000	2.495	4.772	2.038

Tabulka 4.1: Výsledky chí-kvadrát testu generátorů náhodných čísel, hodnoty představují hodnotu V z [15]

Java verze 1.7 ve svých runtime knihovnách obsahuje ještě další „bezpečnou“ implementaci generátoru pseudo-náhodných čísel (java.security.SecureRandom) vhodnou pro použití v šifrovacích algoritmech. Tato implementace splňuje minimální požadavky na použití v šifrovacích algoritmech dle statistických a dalších testů definovaných organizací NIST pro generátory určené k šifrování [16].

Provedl jsem chí-kvadrát test dle [15] a zde jsou výsledky třech pokusů pro generátor java.util.Random (JUR), java.security.SecureRandom (JSS) a Fortuna (FOR). Test byl proveden na skupinách čísel 1 - 6, tzn. hod kostkou. Výsledky ukazují, že odchylky od rovnoměrného rozdělení všech tří generátorů jsou obdobné.

Ovšem standardní implementace jsou více náchylné na zjištění seedu, což umožňuje přesné určení celé číselné řady [17]. S knihovnou fortuna je nalezení všech vstupních entropických hodnot téměř nemožné.

4.4 Bezpečnost

Tato kapitola pojednává o hlavních bezpečnostních aspektech řešení jako je autentifikace uživatele a šifrování citlivých dat.

4.4.1 Autentifikace

Smartfox server má zabudovaný mechanismus přihlášení, který lze z části modifikovat. Bez jakékoliv modifikace funguje tak, že server nejprve zašle náhodný řetězec bytů (sůl), klientská komunikační knihovna pak za heslo přidá sůl a provede hashování pomocí algoritmu MD5. Výsledek pak odešle spolu s uživatelským jménem na server, kde se vykoná stejný proces, jako na klientovi, a pokud hash souhlasí, server uživatele vede jako přihlášeného. Uvedený postup má ale několik nevýhod. Tou první je, že v dnešní době už není doporučováno algoritmus MD5 k bezpečnostním účelům používat [18]. Na dnešní poměry velmi rychlý, což usnadňuje útok hrubou silou. Další nevýhodou je, že hash musí být i v databázi uložena jako MD5 a nemůže být ani osolena, jelikož na server už přijde osolený tvar a není možné se vrátit zpět k původní čisté MD5 hash. Existuje několik řešení, jak tyto nevýhody odstranit. Já použil stejné řešení (ale vlastní), jako používá pomocná komponenta Smartfox

serveru - Login Assistant [19]. Ta spoléhá na integrované šifrování Smartfox serveru. Heslo pak odešle sice v textové podobě, ale zašifrované pomocí AES. Na serveru lze pak heslo zahashovat společně se solí uloženou v databázi.

4.4.2 Šifrování komunikace

Smartfox má v posledních verzích (od verze 2.10 [20]) zabudovanou podporu šifrování pomoci symetrické šifry AES. Funguje to tak, že na obou stranách komunikace musí být šifrování explicitně zapnuto. Pokud ho jedna ze stran zapnuté nemá, komunikace je odmítnuta. Před samotným procesem autentifikace dojde k předání šifrovacího klíče a inicializačního vektoru zabezpečeným kanálem přes SSL. Smartfox server tedy nutně musí mít zapnutou podporu SSL na jiném komunikačním portu a vystavený certifikát pro webový server. Zde se nachází jeden z bezpečnostních nedostatků Smartfox serveru. Klientské knihovny Smartfox neumožňují kontrolu otisku certifikátu použitého pro SSL komunikaci, a proto jeho šifrování není odolné proti man in the middle útoku [21]. Zde se budeme snažit vyřešit problém s výrobcem ještě před spuštěním ostrého provozu, abychom měli odolné řešení i proti tomuto typu útoku. Po té, co proběhne získání klíče a inicializačního vektoru, je už veškerá komunikace šifrována - tedy i přihlašovací údaje, čehož využíváme právě při vlastní autentifikaci.

4.5 Databázový model

Databázový model vychází z doménového modelu 3.5. Je už však přizpůsoben konkrétní relační databázi (v našem případě MS SQL 2012). U sloupců jsou definovány adekvátní datové typy, každá tabulka obsahuje primární klíč, tabulky obsahují integritní omezení, některé primární klíče mají definovaný AUTO INCREMENT apod. U entit Throw, Move a Round byla výjimečně provedena denormalizace do jediné tabulky ThrowMove z důvodu minimalizace relačních spojení. V této tabulce je očekáván velký počet řádků, a další spojení by tak mohla ubírat výkonnost častých dotazů. Ostatní entity se mapují 1:1 na tabulky. Vygenerovaná dokumentace z modelu pomocí nástroje Enterprise Architect a vygenerované DDL skripty jsou umístěny na přiloženém CD.

4.6 Serverové služby

Následující kapitola popisuje herní služby, které jsou implementované v komponentě bgmon-server-services.

4.6.1 info.bglab.server.services.register.RegisterService

Služba, která provádí registraci uživatele. Vstupní parametr je entita User. Po provedení kontrol, zda jsou údaje o uživateli správné a uživatel ještě není evidován, provede v jedné transakci vložení záznamu do tabulky User a vytvoří účet s herní měnou.

4.6.2 info.bglab.server.services.login.LoginService

Ověří, zda uživatel je zavedený v databázi a zda zahashované a osolené heslo odpovídá hash v databázi. Pokud je vše v pořádku, vrátí ID uživatele.

4.6.3 info.bglab.server.services.users.UserService

Vyhledá uživatele podle jeho hlavních identifikátorů.

4.6.4 info.bglab.server.services.users.ActivityLogService

Zaznamená aktivitu k danému uživateli. Podporované aktivity jsou prozatím registrace, přihlášení, přidání peněženky.

4.6.5 info.bglab.server.services.games.GameParticipantService

Vyhledá účastníka hry.

4.6.6 info.bglab.server.services.games.GameService

Vyhledá nebo založí hru.

4.6.7 info.bglab.server.services.games.ThrowMoveService

Umožňuje zaznamenat nový hod kostkou nebo tah kamenem.

4.6.8 info.bglab.server.services.random.RandomGenerator

Vrací referenci na objekt reprezentující generátor náhodných čísel.

4.6.9 info.bglab.server.services.wallet.WalletService

Služba vrací informace o peněžence uživatele.

4.6.10 info.bglab.server.services.wallet.TransactionService

Služba provádí transakci částky mezi peněženkami. Kontroluje, zda zdrojová i cílová peněženka mají stejnou měnu a zda na zdrojové peněžence je dostatečný zůstatek. Pokud je vše v pořádku, odečte částku ve zdrojové peněžence,

přičte částku k cílové peněžence a vytvoří záznam o transakci do tabulky WalletTransaction.

Kapitola 5

Vyhodnocení

Vzhledem k povaze řešení (serverová část bez klientské části) jsou možnosti vyhodnocení funkčnosti lehce omezené. Hlavní testování probíhalo pomocí klientské části hraním jednotlivých her a pozorováním chování, což je ovšem špatně zdokumentovatelné. Pro zdokumentování proběhlých testů jsem využil jednotkových testů. Jednotlivé implementované jednotkové testy popisuji v následující podkapitole. V budoucnu jsou plánovány i další testy, jako např. výkonnostní, ovšem ty by v současné etapě projektu nebyly příliš vypovídající, jelikož v dalších etapách před samotným spuštěním dojde k implementaci dalších částí, které také budou mít vliv na výkonnost. Dále budou po dokončení klientské části probíhat testy samotné klientské aplikace dle testovacích scénářů, přičemž i ty budou z principu zahrnovat testování komunikace se serverem. V rámci testování byl naprogramován automatický hráč (bot), který obsahuje základní funkčnost hraní hry se spoluhráčem. Vzhledem k tomu, že byl implementován jen a pouze pro účely testování a není v požadavcích na tuto první etapu vývoje, není zde jeho detailnější rozbor. Zdrojový kód aktuálního testovacího bota je součástí příloh.

5.1 Testy

Následuje výčet implementovaných jednotkových testů pro prostředí JUnit. Jednotkové testy netestují veškeré třídy v projektu, ale jen ty nejdůležitější části. Report z provedení testů je přiložen na CD.

info.bglab.backgammon.tests.DicesOrderTest Jednotkový test specifické herní situaci, kdy server musí vyhodnotit, že první hozenou kostkou nelze táhnout, ovšem druhou lze. Je tedy nutné pořadí kostek prohodit tak, aby klientovi rovnou nabídl druhou kostku jako první.

info.bglab.backgammon.tests.GamePlayTest Jednotkový test hraní hry. Obsahuje komplexnější test od začátku hry až po předání hry druhému hráči,

přičemž test záměrně zkouší nepovolené akce v rámci svého tahu i tahu jiného hráče.

info.bglab.backgammon.tests.HigherDiceFirstTest Jednotkový test určité herní situace, kdy by hráč mohl buď použít jednu a nebo druhou kostku (nevyužije ovšem obě dvě), ale podle pravidel musí použít tu s vyšší hodnotou.

info.bglab.backgammon.tests.ThrowDicesTest Jednotkový test na správný výpočet počtu použití kostek v určité herní situaci.

info.bglab.server.services.tests.RNGDistributionTest Jednotkový test generátorů náhodných čísel vypočítávající hodnotu V chí-kvadrát testu, viz kapitola 4.3.

info.bglab.server.services.tests.RegistrationLoginTest Jednotkový test registrace a následného ověření hesla uživatele.

info.bglab.server.services.tests.GameTest Jednotkový test založení nové hry a ukládání herních dat.

info.bglab.server.services.tests.TransactionTest Jednotkový test provedení transakce částky mezi peněženkami.

Závěr

Cílem této práce bylo analyzovat, navrhnout a implementovat backend pro online hru backgammon a současně naplnit požadavky deklarované zástupci společnosti BGLab. Řešení pak bylo nutné ještě adekvátně otestovat.

Cíle považuji za splněné, což dokazuje i fakt, že v době psaní této práce je klientská aplikace funkční a lze bez problémů dohrát celou hru s druhým protihráčem. Jak už jsem několikrát zmínil, projekt touto etapou nekončí a jeho vývoj pokračuje dále tak, aby celé řešení mohlo být v co nejbližší době spuštěno v produkčním prostředí.

V dalších etapách rozvoje je plánováno rozšíření implementace o další typy her, popř. implementovat analytické nástroje pro vyhodnocování herních dat. V budoucnu je dále možné se věnovat zdokonalení automatického protihráče tak, aby byl ve hře konkurenceschopný i pokročilejším hráčům. Co se týče správy samotných zdrojových souborů, mohlo by se ukázat jako vhodné převést projekt pod Maven, jelikož počet závislostí se bude jistě během dalšího vývoje zvyšovat.

Ačkoliv jsem se s většinou zde použitých technologií už setkal na jiných předchozích projektech, přesto mi tato práce přinesla nové zkušenosti s některými dalšími technologiemi. Za velmi cennou zkušenost považuji hlavně seznámení se s cloudovými službami Azure, kterou určitě využiji i v budoucnu. Také zkušenost se Smartfox serverem je pro mě určitě přínosem.

Literatura

[1] GameSite 2000 Ltd.: *Independent sudies [online].* [cit. 2016-05-09]. Dostupné z: `http://www.extremegammon.com/studies.aspx`

[2] Keith, T.: Backgammon Play Sites [online]. [cit. 2016-05-09]. Dostupné z: `http://www.bkgm.com/servers.html`

[3] Oswald Jacoby, J. R. C.: *The Backgammon Book.* New York: The Viking Press, 1970, ISBN 0-670-14409-6.

[4] Arlow J., N. I.: *UML 2 a unifikovaný proces vývoje aplikací.* Brno: Computer Press, první vydání, 2007, ISBN 978-80-251-1503-9.

[5] U.S. Backgammon Federation: *Backgammon Rules – And How To Play.* [cit. 2016-05-06]. Dostupné z: `http://usbgf.org/learn-backgammon/rules-of-backgammon/`

[6] Obrázek hrací desky poskytnut s laskavým svolením společnosti BGLab s.r.o.

[7] Keith, T.: Backgammon Variants [online]. [cit. 2016-05-09]. Dostupné z: `http://www.bkgm.com/variants/`

[8] Unity Technologies: *Unity 3D [online].* [cit. 2016-05-2]. Dostupné z: `https://unity3d.com/`

[9] Exit Games: *Photon Server [online].* [cit. 2016-05-2]. Dostupné z: `https://www.photonengine.com/`

[10] gotoAndPlay(): *SmartFoxServer 2X - features [online].* [cit. 2016-05-11]. Dostupné z: `http://www.smartfoxserver.com/products/sfs2x#p=features`

[11] Microsoft Corporation: *BizSpark [online].* [cit. 2016-05-09]. Dostupné z: `https://www.microsoft.com/bizspark`

[12] Red Hat, Inc.: *Hibernate ORM - About [online].* [cit. 2016-05-09]. Dostupné z: http://hibernate.org/orm/

[13] Wästerlund, J.: Fortuna [online]. [cit. 2016-05-16]. Dostupné z: https://github.com/grunka/Fortuna

[14] Oracle: *GNU Classpath[online].* [cit. 2016-05-07]. Dostupné z: https://docs.oracle.com/javase/7/docs/api/java/util/Random.html

[15] Knuth, D. E.: *The Art of Computer Programming.* Volume 2: Seminumerical Algorithms, Addison-Wesley, třetí vydání, 1997, ISBN 0-201-89684-2.

[16] National Institute of Standards and Technology: *Recommendation for Random Number Generation Using Deterministic Random Bit Generators [online].* [cit. 2016-05-06]. Dostupné z: http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-90Ar1.pdf

[17] Ta, F.: Predicting the next Math.random() in Java [online]. 2014, [cit. 2016-05-02]. Dostupné z: http://franklinta.com/2014/08/31/predicting-the-next-math-random-in-java/

[18] Whittaker, Z.: MD5 password scrambler 'no longer safe' [online]. 2012, [cit. 2016-05-11]. Dostupné z: http://www.zdnet.com/article/md5-password-scrambler-no-longer-safe/

[19] gotoAndPlay(): *The Login Assistant component [online].* [cit. 2016-05-11]. Dostupné z: http://docs2x.smartfoxserver.com/DevelopmentBasics/login-assistant

[20] gotoAndPlay(): *Protocol Cryptography [online].* [cit. 2016-05-11]. Dostupné z: http://docs2x.smartfoxserver.com/GettingStarted/cryptography

[21] Sanders, C.: Understanding Man-In-The-Middle Attacks - Part 4: SSL Hijacking [online]. 2010, [cit. 2016-05-11]. Dostupné z: http://www.windowsecurity.com/articles-tutorials/authentication_and_encryption/Understanding-Man-in-the-Middle-Attacks-ARP-Part4.html

Příloha A

Seznam použitých zkratek

DDL Data definition language

REST Representational State Transfer

API Application Programming Interface

JPA Java Persistence API

OS Operační systém

MS SQL Microsoft SQL Server

AES Advanced Encryption Standard

UML Unified Modeling Language

SSL Secure Sockets Layer

TCP Transmission Control Protocol

UDP User Datagram Protocol

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

Příloha **B**

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `doc`
  - `cloud_kalkulace.xlsx` ..... cenové porovnání cloudových řešení
  - `BG_DB.rtf` ..... popis fyzického datového modelu
  - `test_report` ..... reporty spuštění jednotkových testů
  - `prog` ..... programátorská dokumentace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
  - `db` ..... DDL skripty pro vytvoření databáze
- `text` ..... text práce
  - `zadani.pdf` ..... zadání této práce
  - `lipervac.pdf` ..... text této práce ve formátu PDF
  - `lipervac.ps` ..... text této práce ve formátu PS