Západočeská univerzita v Plzni

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

Diplomová práce

Architektura a implementace webové hry pro více hráčů

Plzeň, 2012 Bc. Martin Štěpánek

Prohlášení

Prohlašuji, že jsem diplomovou práci vypracoval samostatně a výhradně s použitím citovaných pramenů.

V Plzni dne 3. července 2012

Bc. Martin Štěpánek

Architecture and implementation of a web multiplayer online game

by

Martin Štěpánek

Department of Computer Science and Engineering, Faculty of Applied Sciences, University of West Bohemia in Plzen, 2012

Thesis supervisor: Doc. Ing. Přemysl Brada, MSc. PhD.

Abstract

This thesis deals with design and implementation of web multiplayer online game developed as student project on Department of Computer Science and Engineering, University of West Bohemia in Pilsen. The project called Space Traffic aims to improve publicity of the said department among the high school students and also to present work of it's own students.

This work reviews existing game design of developed game and proposes changes and improvements. It also presents base architecture of developed game as modular web application build on Microsoft .NET technology. Key elements of this architecture were implemented with team of students during the academic year 2011/2012. This work also mentions some selected topics about videogames and game design. At the end it provides suggestions and recommendations for further continuation of the project.

Obsah

Poděkování

Tímto bych chtěl poděkovat všem studentům, kteří se na projektu Space Traffic podíleli po celou dobu jeho existence a především těm, se kterými jsem měl tu čest spolupracovat osobně.

Dále bych chtěl poděkovat všem pracovníkům Katedry informatiky a výpočetní techniky za podporu a spolupráci při realizaci projektu Space Traffic.

Speciální dík patří vedoucímu práce Doc. Ing. Přemyslu Bradovi, MSc. PhD., za cenné rady a připomínky a všeobecnou podporu celého projektu.

Nakonec bych chtěl poděkovat rodině a přátelům za podporu a pomoc během celého studia.

1 Úvod

Tato diplomová práce se zabývá problematikou game designu a návrhu architektury webové hry a způsobem její implementace jako studentského projektu na Katedře informatiky a výpočetní techniky Fakulty aplikovaných věd Západočeské univerzity v Plzni[1]. Tato práce navazuje na diplomovou práci Zbyňka Neuberta s názvem *Analýza, návrh a vedení týmu v projektu webové hry* [1], která byla obhájena na téže katedře roku 2010.

Počítačové hry jsou interaktivní médium, které postupně hledá své místo v kultuře moderní společnosti. Za čtyřicet let svého vývoje se hry postupně staly samozřejmou součástí životů mnoha lidí a přestože zatím nejsou všeobecně uznávány jako forma umění, mnozí je za něj považují.

Z pohledu softwarového inženýrství představuje jejich tvorba složitý problém, pro jehož řešení je třeba uplatňovat poznatky nejen z informatiky, ale i z mnoha dalších oborů. Tento široký záběr činí z tvorby her výzvu pro kreativitu a stává se obohacující a naplňující zkušeností pro jejich tvůrce. Dalo by se říct, že tvorba her je hrou sama o sobě.

1.1 Cíle práce

Cílem této práce je vytvoření game designu a architektury webové hry *Space Traffic* v návaznosti na předchozí práci v rámci studentského projektu na KIV. Součástí je také implementace klíčových komponent, ověření jejich funkčnosti a návrh doporučení pro další pokračování projektu.

Tento text seznamuje čtenáře s projektem Space Traffic, vzniklého na KIV jako možného prostředku propagace studia na katedře mezi středoškolskými studenty. Přináší detailní popis game designu hry a prezentuje návrh modulární architektury pro její realizaci. Také shrnuje praktické zkušenosti z práce na projektu v akademickém roce 2011/2012, týkající se především spolupráce se studenty bakalářských a navazujících magisterských studijních programů KIV, organizace projektu, vedení dokumentace a předkládá doporučení pro další vývoj.

[1] Ve zbytku práce užívány zkratky KIV, FAV a ZČU.

1.2 Struktura práce

Tato práce je členěna do 7 kapitol (včetně tohoto úvodu), jejichž účel je následující:

Kapitola 2 seznamuje čtenáře s problematikou počítačových her se zaměřením na oblast her webových.

Kapitola 3 představuje studentský projekt Space Traffic, seznamuje s jeho cíli, historií a postupem prací v akademickém roce 2011/2012.

Kapitola 4 se zabývá game designem hry *Space Traffic*. Nejprve poskytuje úvod do problematiky a následně rozebírá původní koncept a předkládá jeho změny a rozšíření. Také detailně popisuje jednotlivé herní prvky a mechanismy.

Kapitola 5 popisuje návrh architektury pro realizace hry a některé klíčové detaily její implementace.

Kapitola 6 shrnuje praktické zkušenosti z práce na projektu v akademickém roce 2011/2012, především pak zkušenosti s úrovní znalostí studentů a organizací. Dále nabízí doporučení pro další pokračování projektu.

Kapitola 7 je závěrem, který shrnuje získané znalosti a hodnotí dosažená řešení.

2 Webové hry

Tato kapitola se věnuje počítačovým a především pak webovým hrám. Poskytuje stručný pohled do historie, shrnuje hlavní technologie a nahlíží do budoucnosti webu jako herní platformy.

Myšlenka užití počítačů pro hraní her provází výpočetní techniku už od samotného počátku. Jak uvádí Wikipedie [2], prvním počítačem sestrojeným speciálně za účelem hraní hry byl počítač NIMROD z roku 1951 (hra *NIM*). V roce 1952 vytvořil Alexander S. Douglas hru *OXO* na počítači EDSAC Cambridgeské univerzity, první hru využívající grafický display. Vývoj pokračoval hrou *Tennis for Two* z roku 1958, jejímž autorem byl William Higinbotham, která využívala osciloskop jako zobrazovací zařízení. V následujícím období vznikla řada zajímavých her, které však byly většinou vnímány jako zajímavé technologické rarity.

Toto vnímání změnila hra *PONG* firmy Atari z roku 1972, která byla první komerčně úspěšnou arkádovou videohrou. V této době už také hry začaly pronikat do domácností v podobě zábavních herních zařízení, připojitelných k televizi. Tato zařízení jsou dnes označována jako herní konzole.

S příchodem mikropočítačů a osobních počítačů v 80. letech minulého století se hry staly neodmyslitelnou součástí této technologie. Vyvinuly se z jednoduchých hříček v komplikované simulace rozličných herních světů, které jsou omezeny jen fantazií jejich tvůrců (a výkonem cílových zařízení).

2.1 Počítačové hry současnosti

Přestože v minulosti vznikaly počítačové hry na řadu různých zařízení[1], současné prostředí je mnohem více homogenizováno. Domácí herní konzole jsou zastoupeny především produkty firem Microsoft (XBox), Nintendo (Wii) a Sony (PlayStation). Trhu s přenosnými herními konzolemi pak dominují produkty firem Nintento a Sony. Dále je zde trh her pro mobilní telefony a tablety (ať už produktech společnosti Apple nebo zařízeních založených na operačních systémech Android nebo Windows Phone). A pak je tu samozřejmě zastoupení her pro PC (především s operačním systémem Microsoft Windows).

Co se obsahové stránky současných her týče, je k dispozici celá řada žánrů od hlavolamů a logických her, přes různé simulátory, budovatelské a strategické hry až po výpravné hry jako RPG (Role-playing game) nebo adventury. Navíc se autoři často snaží o prolínání žánrů ve snaze vytvořit unikátní herní zážitek.

[1] Jen v seznamu herních konzolí na Wikipedii [3] je uvedeno přes 100 zařízení od historických až po současná.

Ovládání současných her se již neomezuje pouze na klasická vstupní zařízení jako klávesnice, myš, gamepad nebo joystick. Všechny moderní konzole nabízejí i některou z forem ovládání pohybem,[2] které je také k dispozici u mobilních zařízení díky vestavěným senzorům. Tento způsob ovládání je využíván především ve sportovních hrách. Některé hry podporují také ovládání hlasem, jehož využití je však stále vzácné (spíše jako zajímavý doplněk).

Velký vliv na vývoj her má internet a to hned ve dvou oblastech. První je zjednodušení hry více hráčů (multiplayer[3]), která je díky internetu snáze dostupná a umožňuje kdykoli najít spoluhráče/protihráče pro sdílení herního zážitku. Druhou je digitální distribuce her popularizovaná službou Steam[4] společnosti Valve. Výhodou této formy distribuce je především okamžitá dostupnost hry zákazníkovi. Nepřímo pak internet ovlivnil i vznik nové platformy pro hraní her – internetového prohlížeče.

2.2 Co jsou webové hry a jejich stručná historie

Za webové hry lze označit jakoukoli hru, která ke své prezentaci využívá internetový prohlížeč. Díky tomu nevyžadují žádnou instalaci a jsou okamžitě dostupné hráči.

Webové hry se vyvíjely společně s vývojem webových technologií. Technologie, které umožnily využití webových stránek jako platformy pro hraní her, jsou především HTML[5] 2, DHTML[6] a JavaScript v letech 1995-1996. Dalšího pokroku, především v audiovizuálním zpracování, došlo s uvedením Macromedia Flash Player 2 v roce 1997, který rozšířil Flash o možnost využití programovacího jazyka ActionScript.

První webové hry se omezovaly na žánr tahových MMO (Massive Multiplayer Online) strategií (někdy označovány jako MMOBG – Massively Multiplayer Online Browser Games). V tomto typu hry jednotliví hráči provedou své tahy najednou ve vymezeném časovém okně. Po jeho uplynutí jsou vyhodnoceny serverem a dojde k posunu simulace herního světa. Celý proces se opakuje novým časovým oknem pro tahy hráčů. Hratelnost v této podobě nevyžaduje graficky náročné uživatelské rozhraní a vystačí si v té době dostupnými webovými formuláři.

Jednou z prvních[7] takových her byla *Earth: 2025* společnosti Solaria Games (později Solaria Interactive, Echelon Entertainment a nakonec Swirve.com), která byla spuštěna v roce 1996 a ukončena v roce 2009. Jednalo se o strategickou hru o budování státu v postapokalyptickém světě. Jejím duchovním nástupcem je hra *Earth Empires*, vytvořená a dále rozvíjená fanoušky původní hry.

[2] Detekce pohybu těla a gest pomocí systému kamer nebo ovladač s pohybovými senzory.
[3] V českých médiích je hra více hráčů běžně označována anglickým slovem multiplayer.
[4] Web služby Steam - `http://www.steampowered.com`
[5] HTML - HyperText Markup Language
[6] DHTML - Dynamic HyperText Markup Language
[7] Zdroj: článek *Browser Games: Basics and History* na webu CTrust Network [4] a Earth Empires Wiki [5].

Další vývoj webových her odrážel především pokroky ve webových technologiích. S příchodem Flashe začaly vznikat i graficky náročnější hry, nejprve pro jednoho hráče a později i multiplayerové. Zároveň se rozvíjel žánr webových strategií vznikem her jako *Travian* firmy Travian Games, *Divoké kmeny* firmy InnoGames a řady dalších. Zajímavým projektem je také MMORPG[8] hra *RuneScape* společnosti Jagex Game Studios, která již v roce 2001 nabídla 3D grafiku s využitím Javy.

2.3 Technologie

Moderní webové technologie používané pro tvorbu her lze rozdělit na dvě základní skupiny:

- technologie podporované nativně prohlížeči, k nimž patří HTML, Cascade Style Sheets (CSS) a JavaScript;
- technologie využívající vlastní rozšíření prohlížečů pro svůj chod, jako Java, Abode Flash, Silverlight a Unity.

HTML a CSS představují standardní formát pro tvorbu webového obsahu. Poskytují dostatečné prostředky pro tvorbu základních her. Doplněním o JavaScript je pak možné dosáhnout přímé interaktivity webové stránky bez nutnosti komunikace se serverem. Pro tvorbu jednoduchých her je tato kombinace dostačující, nevyhovuje však pro vytváření graficky náročných titulů. Tento nedostatek řeší **HTML5**, nově vyvíjený standard HTML, který přináší řadu zásadních vylepšení:

- prvek **Canvas**, umožňující kreslení 2D grafiky pomocí JavaScriptu,
- přímá podpora vektorového formátu **Scalable Vector Graphics (SVG)**,
- optimalizace pro zvýšení výkonu grafických webových aplikací (především JavaScript funkce **requestAnimationFrame**),
- podpora plnohodnotné hardwarově akcelerované 3D grafiky v podobě technologie **WebGL**.

Přestože úroveň podpory těchto nových technologií jednotlivými prohlížeči se různí, většina z nich je již nyní dostupná (výjimkou je WebGL). Kompletní přehled novinek v HTML5 a rozsah jejich podpory je možné nalézt například na webu `http://caniuse.com/`.

Technologie **Flash** firmy Adobe je již součástí webu již přes 15 let. Původní nástroj určený pro tvorbu animované vektorové grafiky se postupem času vyvinul v komplexní

[8] MMORPG - Massively multiplayer online role-playing game

platformu, která je v současnosti široce využívána pro tvorbu her. Důležitým milníkem je pak udevení Abode Flash 11, který přináší podporu hardwarově akcelerované 3D grafiky.[9]

Flash aplikace jsou spouštěny pomocí tzv. Flash Playeru, který je pomocí pluginu nainstalován do prohlížeče. Podporovány jsou všechny hlavní prohlížeče včetně mobilních.

Microsoft Silverlight je technologie uvedená v roce 2007, která nabízí podobnou funkcionalitu jako Abobe Flash, kterému měla konkurovat. Funguje na stejném principu instalace rozšíření do prohlížeče, její adaptace je však na mnohem nižší úrovni, jak po stránce podpory prohlížečů, tak po stránce existujících aplikací.

Poskytovanými možnostmi je s Flashem porovnatelná, především ve své poslední, 5. verzi. Přesto se spíše prosazuje jako prostředek pro vytváření webových a mobilních aplikací jako součást .NET platformy, než pro tvorbu her.

Poslední technologií, která zde bude zmíněna je Unity[10] 3 firmy Unity Technologies. Jedná se o herní engine a sadu vývojových nástrojů, jejichž cílem je poskytnout společné vývojové prostředí pro různé cílové platformy. Snaha je především o sjednocení vývoje a tím snížení nákladů na vývoj multiplatformních titulů.

Hru vytvořenou pomocí Unity je možné publikovat na všechny současné herní platformy, tj. konzole XBox 360, PS3, Nintendo Wii, Windows, Mac, Android, iOS a web. Pro běh hry v prohlížeči je opět použit plugin. Vyvíjena je také alternativa používající Flash.

2.3.1 Specifika webových her oproti ostatním počítačovým hrám

Webové hry mají určitá technická omezení, se kterými je třeba počítat. Historicky bylo hlavním omezením webových her jejich grafické zpracování. Bez použití Adobe Flash nebylo možné dosáhnout grafické kvality běžných her. Současné technologie tento problém odbourávají, i když je stále problém s pokročilou 3D grafikou.

Druhým důležitým omezením je objem dat, který hra ke svému chodu vyžaduje. Ten je limitován faktem, že hra musí být hráči stažena při každém spuštění. Současné AAA[11] tituly mohou vyžadovat instalaci přesahující 10 GB, což u webové hry není možné.

Protože webové hry vyžadují už z principu online přítomnost, je snaha o její další využití. Tomu odpovídají nejen obchodní modely, ale také zaměření her jako takových (multiplayer, integrace se sociálními sítěmi, apod.).

[9] Zdroj: web firmy Adobe - `http://www.adobe.com/products/flashplayer/features.html`

[10] Web produktu Unity 3 - `http://unity3d.com/`

[11] Jako AAA neboli „tříáčkové“ (v angličtině Tripply-A) tituly jsou označovány hry vyvíjené pod záštitou velkých vydavatelů. Tyto hry se vyznačují vysokou úrovní technického i uměleckého zpracování. Jejich vývoj obvykle trvá několik let. Cílovými platformami jsou konzole a PC.

2.3.2 Současnost a budoucnost webu jako herní platformy

V současné době se oblast webových her mění z okrajové na plnohodnotnou herní platformu. Díky novým technologiím jako HTML5, WebGL, Abode Flash 11, Unity a dalším, je možné produkovat hry, které se kvalitou stále více přibližují hrám na klasické platformy.

Dalším důležitým vlivem na rozvoj webových her jsou sociální sítě, především pak sít Facebook. Tyto sítě vytvářejí jednotný bod, kterým jsou webové hry distribuovány hráčům a stávají se tak jejich integrální součástí. Pro toto prostředí vzniká nový druh tzv. sociálních her, které jsou cíleny na příležitostné hráče, snaží se o jednoduchost a většinou se jedná o hry s otevřeným koncem, které se snaží využít specifické prostředí sociálních sítí.

Další oblastí, ve které se webová platforma prosazuje pro hraní her, jsou mobilní zařízení. Díky rostoucí dostupnosti mobilního internetu a podpoře moderních webových technologií mobilními prohlížeči je webová platforma atraktivní alternativou pro tvorbu her cílených na tato zařízení. Jednou z velkých předností je především opravdová multiplatformita těchto produktů a tím i úspora nákladů na vývoj.

Do budoucna lze předpokládat, že se prohlížeč a web stanou jednou z hlavních herních platforem. Současná technologická omezení již nejsou překážkou pro vytváření kvalitních titulů všech žánrů a jediným nedostatkem je v současné době nižší výkon platformy pro 3D hry úrovně AAA bez použití technologií jako Adobe Flash 11 nebo Unity. Přesto lze předpokládat, že se situace bude dále zlepšovat.

,

3 Projekt Space Traffic

V této kapitole je představen studentský projekt Space Traffic. Je zde popsán účel projektu, jeho stručná historie a jeho stav a cíle v akademickém roce 2011/2012. Podrobně se těmto tématům věnují diplomové práce Petra Vogla [6] a Richarda Kocmana [7].

3.1 O projektu

Studentský projekt Space Traffic podporovaný KIV má za cíl vytvoření webové hry (MMO) na téma vesmírného obchodování s programováním jako prvkem hratelnosti. To znamená, že hráči mohou ve hře vytvářet programy pro automatizaci některých herní činností. (Například ovládání kosmických lodí.)

Projekt přináší studentům KIV možnost využít studiem získávané znalosti v řešení netriviálních úloh a zároveň uplatnit i vlastní kreativitu.

3.1.1 Účel projektu

Záměrem KIV je prezentovat veřejnosti činnost svých studentů a zvýšit zájem o studium nabízených oborů u žáků středních škol. Prostředkem se má stát vytvářená hra, jejíž popis je předmětem této práce. Forma vývoje jako studentský projekt poskytuje studentům KIV příležitost prezentovat své dovednosti a získat cenné zkušenosti v oblasti týmové práce, vedení projektů a softwarového vývoje.

3.1.2 Stručná historie projektu

Historií projektu se podrobně zabývají již zmíněné diplomové práce [6] a [7]. Zde bude uvedeno jen její stručné shrnutí.

Projekt Space Traffic vznikl díky iniciativě Zbyňka Neuberta ve spolupráci s KIV v roce 2009. Neubert vytvořil základní koncept hry a poté se jej pokusil realizovat s týmem několika studentů. Jak popisuje ve své práci [1], byly počátky projektu obtížné. Hlavním problémem byl především nedostatek pracovních sil. Také rozsah cílů stanovených pro ak. rok 2009/2010 neodpovídal obtížnosti úlohy. Přesto se týmu pod jeho vedením podařilo vytvořit funkční prototyp hry, založený na technologiích PHP,[1] HTML a JavaScript.

V následujícím ak. roce (2010/2011) se však ukázalo, že vytvořené řešení nemá parametry, které by umožňovaly novému týmu pod vedením Richarda Kocmana v pracích

[1] PHP je skriptovací jazyk pro vytváření webových aplikací - `http://www.php.net/`.

pokračovat. Předchozí tým se příliš soustředil na praktické výsledky a podcenil[2] problematiku předání projektu do dalších let. Neuchopitelná architektura implementaci spolu s nedostatečnou dokumentací vedly nakonec k rozhodnutí opustit původní řešení a vytvořit nové, založené na PHP frameworku Nette.[3] To zdrželo celý vývoj projektu, který byl následně přerušen ztrátou projektového manažera. V druhé polovině března 2011 musel Richard Kocman práce na projektu zanechat z vážných rodinných důvodu.

V ak. roce 2011/2012 byl projekt převzat Petrem Voglem a autorem této práce. Protože předchozí rok bylo provedeno jen velmi málo implementace a záměr nepokračovat v implementaci z 2009/2010 trval, bylo rozhodnuto o změně technologie na .NET.

3.2 Stav projektu na začátku ak. roku 2011/2012

Stav projektu na začátku ak. roku 2011/2012 lze považovat za „nový začátek“. Práce provedené v ak. roce 2010/2011 posloužily jako prototypy pro některé prvky nové implementace. Jednalo se především o uživatelské rozhraní. Původní verze hry z roku 2010 byla využita jen jako zdroj inspirace.

Samotná koncepce hry zůstala v zásadě zachována, došlo však k jejímu upřesnění a několika klíčovým změnám v game designu. Ty jsou popsány v následující kapitole.

V rámci přípravných prací v létě 2011 vznikla kostra aplikace, koncept rozšíření game designu a především strategie pro řízení projektu, zaměřující se na získávání pracovních sil formou semestrálních prací. Podrobnosti lze nalézt v práci Petra Vogla [6].

3.3 Cíle pro 2011/2012

Cíle projektu pro akademický rok 2011/2012 jsou uvedeny v příloze A tak, jak byly navrženy v květnu 2011 a předneseny na schůzce se zástupci KIV. Následuje jejich stručné shrnutí.

Cíle se týkaly především vytvoření reprezentace herního světa a základní funkcionality hry. Důraz byl kladen především na kvalitu řešení. Také zmiňují prvky podpory projektu jako takového, které by umožnily efektivnější spolupráci členů a snazší předání projektu. Oproti původnímu dokumentu došlo k dohodě o změně technologie z PHP na .NET.

Míru jejich naplnění lze posoudit v přiloženém dokumentu z května 2012, viz. příloha B.

[2] Je třeba uvést, že příčinou tohoto podcenění byl nedostatek zkušeností, nikoli nedbalost.
[3] Nette je český PHP framework - `http://nette.org/cs/`.

4 Game design Space Traffic

Předchozí kapitola představila projekt webové hry *Space Traffic*. V této kapitole bude podrobně rozebrán game design této hry, jeho obecné zásady, změny oproti původnímu konceptu, jednotlivé herní prvky a mechaniky, a také požadavky na audiovizuální zpracování hry. Také se zmiňuje o některých dalších rozšířeních, která lze do hry začlenit buď už při jejím vývoji, nebo jako dodatečná rozšíření po jejím spuštění. Nejprve je však nutné věnovat se samotné problematice game designu.

4.1 Game design obecně

Game design[1] je proces návrhu hry (počítačové), tj. jejích pravidel, cílů, herního prostředí, příběhu a formy zpracování atd. Zároveň je tímto termínem označován i popis hry jako takový, tj. celkový popis všech výše uvedených částí. Tento význam přejímá i tato práce. Proces vzniku game designu bude užíván český výraz návrh hry. Obě dvě užití termínu ale úzce souvisí s tím, co je to vlastně hra. Proto je třeba se této otázce stručně věnovat.

Jesse Schell ve snaze definovat pojem hra věnuje ve své knize *The Art of Game Design: A Book of Lenses* celou podkapitolu [8, kap. 3,]. V ní je shrnuta řada možných definic, které spíše pojem hry rozšiřují, než aby jej definovaly. Nakonec autor snahu o definici hry vzdává a končí podkapitolu slovy:

„So let's not dawdle here. We've spent enough time thinking about what a game is. Now let's go see what a game is made of.“

Pro potřeby této práce budeme hru chápat jako softwarovou aplikaci, jejímž účelem je poskytnutí aktivní zábavy uživateli – hráči. Zde samozřejmě vyvstává otázka, co je to zábava. A právě hledání odpovědi na tuto otázku je možné vnímat jako cíl celého procesu game designu.

Výstupem game designu je většinou dokument, označovaný jako **Game Design Document** (běžně zkracovaný na GDD), popisující všechny tyto detaily v míře potřebné pro jejich realizaci. Tento dokument se vyvíjí společně s projektem hry a odráží změny, které byly v návrhu provedeny dodatečně. Druhá část této kapitoly slouží jako výchozí verze GDD projektu Space Traffic. Z praktických důvodů je pak obsah tohoto dokumentu duplikován v projektové wiki v části *Game design*.[2] Tam bude dále aktualizován spolu s realizací jednotlivých herních prvků a jejich testováním.

[1] Anglický termín, běžně používaný i v češtině; česky herní design nebo herní návrh.

[2] Space Traffic Wiki, *Game design* - `http://spacetraffic.kiv.zcu.cz/code/tiki-index.php?page=Game+design`

4.1.1 Postup vzniku game designu

Počítačová hra je aktivní médium. Základem je tedy navrhnout, co bude hráč ve hře dělat, aby to pro něj byla zábava. Interakce hráče se hrou se nazývá hratelnost (anglicky gameplay) a je nejdůležitějším prostředkem komunikace mezi designérem a hráčem.

Neméně důležité je zasazení hry. To je reprezentováno příběhem, podobou a pravidly herního světa, cíli hry a neherními prvky pro dokreslení zážitku.

Obě tyto základní složky hry je v průběhu jejího návrhu třeba definovat. Všechno však vždy začíná u nápadu. Tento nápad většinou spadá do dvou kategorií:

- nápad zajímavého zasazení hry (prostředí, příběh, apod.);
- nápad zajímavé herní činnosti (mechanika, zajímavá zbraň, apod.);

Tyto dvě kategorie zároveň určují, z jakého úhlu pohledu je možné nápad dále rozvinout v plnohodnotný návrh hry. Je to buď pohled **shora dolů** nebo **zdola nahoru**, které podrobně popisuje článek *Game Design Cognition: The Bottom-Up And Top-Down Approaches* autorů Gillarda Lopes a Rafaela Kuhnena [9].

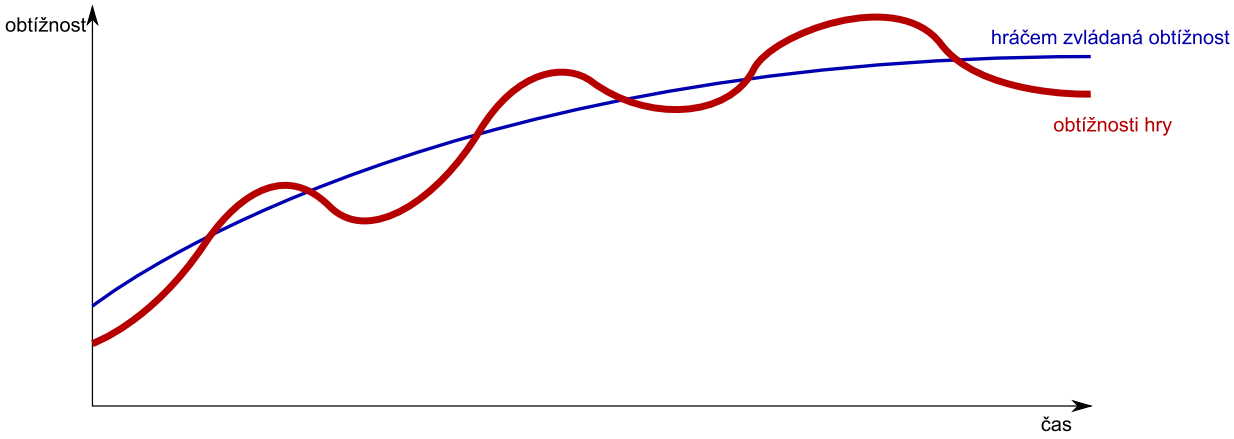
Při pohledu shora dolů designér nejprve definuje celkový koncept hry, její zasazení, nosné prvky příběhu a celkového vyznění a poté jednotlivé oblasti detailně rozebírá. Řeší tak otázku, co ve hře bude hráč dělat. Příkladem hry navržené tímto způsobem je RPG hra *Zaklínač* polské firmy CD Projekt, vzniklá na motivy stejnojmenné série knih polského spisovatele Andrzeje Sapkowskiho.

Naopak při pohledu zdola nahoru jsou na začátku k dispozici prvky hratelnosti, které designér vnímá jako zábavné. Na jejich základě se pak snaží vytvořit celkový koncept, který by „omluvil“ existenci těchto prvků ve hře. Dobrý příklad hry vzniklé tímto způsobem je *Portal* firmy Valve, který vznikl na základě myšlenky „portálové pušky“, umožňující vytvářet průchozí portály v 3D prostoru.

V obou případech je pro rozvinutí nápadu využíváno mnoho různých přístupů od ryze inženýrského stanovení požadavků až po formování umělecké vize. Jsou uplatňovány znalosti z matematiky, fyziky, ekonomie, informatiky, psychologie, dalších společenských a kulturních věd, filozofie a v závislosti na konkrétní hře jakékoli další znalosti a zkušenosti designéra, které lze do hry promítnout.

4.1.2 Prostředky návrhu her

Protože tvorba počítačových her probíhá už přes 40 let, byly za tuto dobu nashromážděny mnohé poznatky a osvědčené postupy, které je možné využívat. Jejich popis dalece přesahuje rámec této práce, proto zde budou uvedeny jen některá vybraná témata, která se přímo vztahují k předkládanému game designu Space Traffic. Popisované prostředky vychází z [8], kde je také možné nalézt mnoho dalších informací.



Obrázek 4.1: Graf obtížnosti hry v závislosti na schopnosti hráče - kontinuální výzva.

Motivace hráče

Hráče je třeba ke hraní hry motivovat. Zábava je přirozeným zdrojem motivace, někdy je však třeba přistoupit k dodatečným technikám, především pokud hra obsahuje obtížnější momenty.

Samotné vyvážení obtížnosti je klíčovou úlohou při návrhu hry. Jednou z aplikovatelných metod je **kontinuální výzva**. Jejím principem je kolísání úrovně obtížnosti kolem úrovně obtížnost, kterou hráč zvládá (viz. Obrázek 4.1). Hráč je tak periodicky vystaven obdobím zvýšenému tlaku, který je vystřídán obdobím relaxace. Tím se u něj střídá pocit výzvy s pocitem vlastního zlepšení. Zároveň je dlouhodobě obtížnost ustálená a nevede k situaci, kdy by její růst proměnil zábavu ve frustraci.

Dlouhodobé a krátkodobé cíle a odměna jsou další důležitou součástí každé hry. Hráč potřebuje mít ve hře jasný dlouhodobý cíl a možnost sledovat, jak se k jeho splnění blíží. Po jeho dosažení je pak odměněn. Pokud by ale cíl byl příliš vzdálen, hráč by mohl ztratit zájem. Proto je možné mu nabídnout krátkodobé cíle jako pomyslný most k tomuto dlouhému cíli. Nejlepší hry pak umožňují hráčům hledat si tyto krátkodobé a někdy i dlouhodobé cíle sami a tím si vytvářet vlastní herní obsah.

Vyvážení hry

Jedním z konceptů, běžně používaný při vyvažování herních prvků a mechanik, je princip „**kámen, nůžky, papír,**“. Jedná se o techniku s cílem předejít vzniku dominantní strategie.[3] Jak již název napovídá, vychází z všeobecně známé „losovací“ hry. V praxi tento princip znamená definování herních prvků (a mechanik) tak, že existuje-li prvek A, který má/poskytuje výhodu nad prvkem B, existuje další prvek C, poskytující/mající výhodu nad prvkem A.

[3] Pojem z teorie her znamenající, že pro hráče je tato strategie nejvýhodnější bez ohledu na to, jakou strategii zvolí ostatní hráči. Prakticky vede k ustálení stylu hraní všech hráčů a ignorování ostatních aspektů hry.

4.1.3 Specifika MMO her

MMO hry jsou určené pro velké množství současně hrajících hráčů. Tyto hry mají mnohá specifika, která je odlišují od ostatních typů her.

Jedním ze základních odlišností je *persistence světa* – herní svět existuje i bez přítomnosti hráče. V praxi to znamená, že hra ubíhá dál podle daných pravidel a hráči do ní vstupují a zase ji opouštějí průběžně.

Mnoho MMO her je navrženo jako *„nekonečné“,* není možné je dohrát v pravém slova smyslu. Herní doba se u jednotlivých hráčů počítá na měsíce, v některých případech i na roky. Návrh těchto her má buď charakter *„pískoviště“*, podporující kreativitu hráčů a umožňující jim vytvářet vlastní herní obsah, nebo má charakter *„zábavního parku“*, snažící se hráče motivovat a provést postupně celým herním obsahem. Je také běžnou praxí průběžné přidávání nového obsahu po vydání MMO hry, čímž se prodlužuje její „životnost“.

Přítomnost velkého množství hráčů v herním světě je dalším důležitým prvkem odlišující MMO hry od ostatních multiplayerových her. Sociální interakce mezi hráči je mnohdy komplikovaná, většinou zahrnuje jak vztahy spoluhráčů, tak i protihráčů a hra obsahuje prvky a mechaniky, které jí podporují. (Například: chat, podpora gest avatarů,[4] vzájemný obchod, odměňování pomoci jinému hráči, nutnost týmů pro některé typy herního obsahu apod.)

V souvislosti s MMO hrami je třeba se ještě zmínit o vlivu jejich obchodního modelu na samotný game design. Většina MMO her je dlouhodobými projekty, které mají poskytnout svým autorům zisk na několik následujících let. Mnoho her proto využívá tzv. pay-to-play (zkracováno na P2P) obchodní model, který je vlastně periodickou platbou za poskytování služby – hry. K tomuto klasickému modelu se v posledních letech připojil ještě tzv. free-to-play (F2P) model, ve kterém je základní hra zdarma a projekt vydělává pomocí mikro transakcí.[5]

Oba tyto modely mají však znatelný vliv na volbu prvků ve hře s cílem podpořit jeho efektivitu. Ať už se jedná o „zdržování“ postupu hráče hrou pro jeho udržení nebo prodávání různých výhod nad ostatními hráči, v obou případech se jedná o praktiky, které jsou minimálně znepokojující. Problémy využívání game designu k jinému účelu než k vytvoření zábavné hry a související etické otázky, jsou diskutovány v závěru této podkapitoly.

4.1.4 Výukové prvky ve hrách

I když hra není přímo výuková (navržená za účelem výuky), může obsahovat výukové prvky. Tím je možné obohatit hráčovu zkušenost se hrou nejen o zábavu samotnou, ale i o získání cenných znalostí a zkušeností.

Nejjednodušším výukovým prvkem je podněcování zvědavosti. Hra může hráči nabí-

[4] Avatar je označení pro personifikaci hráče ve hře.

[5] Platby v řádu jednotek dolarů za malé porce dodatečného herního obsahu.

zet doplňující informace, vztahující se k hernímu obsahu. Obchodní hra může hráči nenápadně nabízet poznatky z ekonomie. Historická hra může obsahovat informace o klíčových historických postavách. Dokonce jen používáním skutečných názvů pro imaginární herní elementy je možné hráče inspirovat k samostatnému dohledání informací.

Důmyslnější přístup je využití hratelnosti k nenápadnému seznámení hráče se zamýšlenými koncepty. Hráč tak může nevědomky získat praktické zkušenosti v oblastech, se kterými by se v reálném životě nemohl setkat. Tento způsob začlenění výukových prvků je však velmi náročný a vyžaduje notnou míru kreativity, aby byl hráč dostatečně odstíněn od faktu, že je hrou „vyučován“. To by mohlo odvést jeho pozornost od hry samotné.

Příkladem her, které úspěšně využívají oba zmíněné přístupy je série *Civilization*, jejímž autorem je Sid Meier. Jedná se o 5 dílů tahové strategie, ve které hráč ovládá vybraný národ a snaží se ho provést od prehistorie přes jednotlivé epochy lidských dějin až do nepříliš vzdálené budoucnosti. Kromě samotného tématu a vlastní hratelnosti je ve hře k dispozici podrobná encyklopedie všech herních prvků a konceptů, která je uvádí do historického kontextu.

4.1.5 Morální horizont game designu

Díky uplatňování znalostí z psychologie, sociologie a praktickým zkušenostem herních vývojářů v navrhování her vzniklo mnoho běžně užívaných technik, které ve své podstatě přímo manipulují s chováním hráče. Navíc dochází k uplatňování znalostí z oblasti navrhování her do skutečného světa (označováno jako gamifikace).

Přestože využívání některých prvků a technik v návrhu hry může být lákavé, je třeba se zamyslet i nad tím, zda je jejich využívání etické. Této problematice se dlouhodobě věnuje americký nezávislý herní vývojář Jonathan Blow. Ve své přednášce *Video Games and the Human Condition* [10] předkládá myšlenku, že využitím některých běžně využívaných game designových postupů lze přeměnit nudnou a nesmyslnou aktivitu do podoby vyvolávající chuť a nutkání tuto aktivitu vykonávat. Je možné vytvořit nudnou a nesmyslnou hru, kterou bude chtít hrát velké množství lidí. Taková hra „ničí čas“, kterého všichni máme nedostatek.

Blow dále uvádí, že nudu je nutné chápat jako přirozený obranný mechanismus, který člověka chrání před prováděním neproduktivních činností. Vidí potenciální nebezpečí ve schopnosti tento obranný mechanismus obejít a přesměrovat přirozenou motivaci dělat věci, které člověku přináší užitek k prázdným a zbytečným činnostem.

Při návrhu her je proto nutné uvažovat i o tomto problému a klást si otázku zda je činnost, kterou hra nutí hráče dělat, pro něho přínosná. Samotná zábava možná nestačí.

Přednášku Blow zakončuje citátem amerického spisovatele Alana Moora:

„Writers and people who had command of words were respected and feared as people who manipulated magic. In latter times I think that artists and writers have allowed themselves to be sold down the river. They have accepted the prevailing belief that art and

writing are merely forms of entertainment. They're not seen as transformative forces that can change a human being; that can change a society. They are seen as simple entertainment; things with which we can fill 20 minutes, half an hour, while we're waiting to die. It's not the job of the artist to give the audience what the audience wants. If the audience knew what they needed, then they wouldn't be the audience. They would be the artists. It is the job of artists to give the audience what they need.“

-Alan Moore

4.2 Charakteristika hry Space Traffic

Space Traffic je strategická MMO hra o vesmírném obchodování. Hra je určena k nepravidelnému hraní s herní dobou kolem tří měsíců.

4.2.1 Role hráče

Hráč je v roli obchodníka, který se zabývá mezihvězdným obchodem s komoditami. Pomocí svých kosmických lodí přepravuje zboží z jedné planety na druhou po celé galaxii za účelem zisku.

Ve hře není hráč reprezentován přímo, místo toho se stává ředitelem vlastní obchodní společnosti a udílí pokyny svým virtuálním pracovníkům (lodím). Pro potřeby komunikace s ostatními hráči si hráč volí herní jméno, pod kterým vystupuje.

4.2.2 Interakce mezi hráči

Vzájemná interakce hráčů je ekonomická. Hráči soutěží na herním trhu o obchodní příležitosti. Zároveň mohou budovat partnerské obchodní vztahy a usilovat o společné posílení svých pozic na trhu.

4.2.3 Výukové prvky ve hře

Hra obsahuje výukové prvky se zaměřením na informatiku a programování. Stěžejním prvkem je v tomto případě možnost vytvářet programy pro ovládání vlastních lodí – vlastní umělou inteligenci.

4.3 Cíle návrhu hry Space Traffic

Cílem game designu *Space Traffic* je vytvoření zábavné nenásilné hry, která by hráče seznámila se základy programování. Hra bude mít vlastnosti *„pískoviště“* i *„zábavního*

parku“, bude povzbuzovat spolupráci hráčů a celkově nabízet hodnotnou zábavu.

Při rozvíjení návrhu bude snaha o aplikaci výukových prvků. Hra tím poskytne hráčům další přínos, kromě zábavy samotné. Při hledání vhodných prvků budou upřednostňovány témata informatiky, matematiky, fyziky a dalších technických oborů.

V neposlední řadě bude při návrhu dbáno na zásady obecné slušnosti a respektování principů svobody, demokracie a lidských práv.

4.3.1 Charakteristika cílové skupiny hráčů

Hra je určena především studentům středních škol ve věku od 15 do 19 let, kteří by se mohli stát potenciálními uchazeči o studium na FAV, především ve studijních programech nabízených KIV. Díky tomuto vymezení lze předpokládat následující vlastnosti hráčů:

- Nejedná se o začínající hráče.
- Mají zájem o vědu a techniku.
- Mají alespoň základní zájem o výpočetní techniku.
- Lze u nich předpokládat minimální znalost programování.

Dále se předpokládá, že by hra mohla zaujmout i studenty bakalářských studijních programů KIV, především prvních ročníků. To by umožnilo vytvoření společné komunity hráčů a mohlo dále napomoci v propagaci studia na KIV.

4.3.2 Zásady herního designu Space Traffic

Protože je hra vyvíjena jako nekomerční produkt, nevzniká žádný tlak na zahrnutí prvků uměle prodlužujících hrací dobu. Nebudou použity prostředky prodlužující přítomnost hráče ve hře nebo takové, které by hráče nutily hru kontrolovat v pravidelných intervalech. Také bude bezpečné hru na delší dobu opustit, aniž by se tím hráč dostal do nevýhodné situace.

Hra nebude hráče nutit k opakování stejné činnosti[6] k dalšímu postupu. Pokud bude třeba něco dělat opakovaně, půjde to vyřešit díky vytvoření programu. Dobrý pocit ze hry musí vycházet z kvalitně stráveného času, nikoli z překonání dlouhého období nudy.

Obtížnost hry bude nastavena tak, aby průměrný hráč cítil výzvu. Obtížnost hry bude založena na dostatečně složitém a dynamickém ekonomickém modelu. Odměňování hráče nebude založeno na náhodě plynoucí ze hry, ale především na jeho inteligenci.

Všechny tyto zásady lze shrnout do následujících pěti hesel:

[6] Označováno anglickým slovem grind.

1. **No grief** – Hra neobsahuje prvky, které by umožnily hráčům vzájemně si škodit.
2. **No grind** – Hra nevyžaduje po hráči bezmyšlenkovité opakování stejných akcí pro postup.
3. **No addiction** – Hru lze kdykoli přerušit, aniž by hráč po delší absenci musel začínat znovu.
4. **Smart play** – Hra zvýhodňuje hráče, kteří hrají „chytře“. Pochopení herních mechanismů a jejich využití ve svůj prospěch by mělo být výhodnější, než bezmyšlenkovité provádění náhodných akcí.
5. **Fun learning** – Hra nabízí herní prvky, které přesahují z oblasti zábavy do výuky. Motivace k učení však musí vždy vycházet ze zábavy.

Každý herní prvek zahrnutý ve hře by měl být posuzován na základě těchto zásad. Herní prvky by neměly žádnou z nich porušovat.

4.4 Inspirace

V přednášce Vladimíra Geršla *Co je game design a jak vlastně vzniká* [11] byla uvedena jako jedna z důležitých zdrojů inspirace game designéra znalost ostatních her. Pochopení fungování herních prvků a mechanik v již existujících hrách a jejich kritické zhodnocení je důležitým aspektem návrhu her. Proto je třeba se seznámit s hrami, jejichž prvky jsou blízké konceptu vytvářené hry. Zde je možné čerpat nejenom inspiraci, ale také nalézt chyby, kterých je třeba se vyvarovat.

Inspirací pro tvorbu návrhu *Space Traffic* byly hry s podobnou tématikou a to především ze dvou směrů: strategické obchodní hry a vesmírné simulátory. Následuje seznam her a jejich stručný popis.

4.4.1 Transporter Tycoon Deluxe

Transporter Tycoon Deluxe je strategická hra z roku 1994, jejímž autorem je Chris Sawyer. Jedná se o obchodní strategii, ve které hráč vlastní přepravní společnost zabývající se železniční, silniční, námořní a leteckou přepravou zboží a pasažérů. Hráč ve hře zprostředkovává zásobování mezi několika typy průmyslových závodů a tím vytváří řetěz začínající vstupními surovinami (uhlí, železo, dřevo, ropa, obilí a dobytek) a končící „zbožím“ dodávaným ke spotřebě do měst. Dále může hráč přepravovat pasažéry a poštu mezi městy. Zásobování měst způsobuje jejich růst.

Jednou z nejzajímavějších složek hry je železniční přeprava, díky možnosti vytvářet komplexní železniční sítě. Po nich je možné provozovat vlaky naváděné pomocí jednoduchého systému příkazů. Protože může dojít ke srážce vlaků, je třeba celou síť správně

Obrázek 4.2: Obrázek tržiště ze hry Port Royale 2 firmy Ascaron Entertainment.

zajistit semafory a při delším hraní je hlavním cílem optimalizace celé sítě tak, aby měla co největší propustnost na všech hlavních tratích.

Tato hra je dále vyvíjena v rámci projektu *OpenTTD*,[7] který je open source přepracováním původní hry s mnoha podstatnými vylepšeními. Jedním z vylepšení je i rozšíření příkazového systému vlaků o podmínky, což umožňuje pružněji reagovat na měnící se ekonomickou situaci ve hře (kolísání rychlosti produkce zdrojů surovin) a dosažení mnohem větší efektivity celé dopravní sítě.

4.4.2 Port Royale 2

Port Royale 2 je hra firmy Ascaron Entertainment, vydaná v roce 2004. Tato obchodní strategie se zabývá námořním obchodem v Karibiku v 17. století. Hráč si může zvolit kariéru dle svého přání počínaje vybudováním obchodního impéria po pirátství. Obchodní složka hry pak nabízí komplexní simulaci s možností vlastnit obchodní loďstvo, dílny, plantáže a vlastní sklady po celé karibské oblasti. Hra má otevřený konec.

[7] Web projektu OpenTTD - `http://www.openttd.org/`

Zpracování volného obchodu ve hře je realizováno následujícím způsobem: Každé město vykupuje a prodává všechny druhy zboží. Ceny jsou závislé na velikosti zásob a také na potřebách města. Změny cen se tedy mění dle nabídky a poptávky. Celé obchodní rozhraní je vidět na obrázku 4.2.

4.4.3 Freelancer

Hra *Freelancer* firmy Digital Anvil je vesmírným simulátorem vydaným roku 2003. Hráč se v roli vesmírného pilota volně pohybuje se svou kosmickou lodí po vesmíru složeném ze 48 hvězdných systému propojených skokovými branami.

Hra nabízí obchod i boj, propracovaný systém frakcí a reputace, možnost vylepšovat si vlastní loď nebo si koupit jinou a mnoho dalších zajímavých prvků. Součástí hry je také příběh, který hráče postupně provede herním světem, seznámí ho s jeho fungováním a se vzájemnými vztahy jednotlivých frakcí.

4.4.4 Série X

Série her *X* firmy Egosoft[8] – *X: Beyond the Frontier* (1999), *X-Tension* (2000), *X²: The Threat* (2003), *X³: Reunion* (2005), *X³: Terran Conflict* (2008) a *X³: Albion Prelude* (2011) – je sérií komplexních vesmírných simulátorů, které nabízejí hráči otevřené prostředí s možností boje i obchodu.

Hra je obsahově podobná dříve uvedené hře *Freelancer*. Rozdílem je, že hráč může vlastnit libovolné množství lodí a vesmírných stanic (továrny, doky). Aby bylo možné ovládat takto velké množství objektů, hra nabízí hráči možnost upravovat umělou inteligenci jednotlivých lodí pomocí skriptovacího jazyka. Přestože rozhraní není příliš intuitivní, jsou možnosti velmi rozsáhlé (lze například naprogramovat loď pro systematické vypouštění komunikačních satelitů, zásobování továren nebo automatické obchodování).

4.4.5 Další hry

K dalším hrám, jejichž znalost měla vliv na game design Space Traffic patří: *EVE Online*[9] společnosti CCP Games (MMORPG vesmírný simulátor s otevřenou ekonomikou), *World of Warcraft* firmy Activision Blizzard a *Travian*[10] firmy Travian Games.

[8] Web společnosti Egosoft - `http://www.egosoft.com/`

[9] Web *EVE Online* - `http://www.eveonline.com/`

[10] *Travian* je strategická webová hra; web - `http://www.travian.cz/`

4.5 Změny v původním game designu Space Traffic

Následující část popisuje změny v game designu oproti původnímu návrhu Zbyňka Neuberta [1]. Přestože celková podoba hry je zachována, došlo k řadě změn v detailech jejího fungování, k vypuštění některých herních prvků a doplnění jiných.

K hlavním změnám patří především:

- vypuštění pirátů a bojových lodí,
- vypuštění aliancí,
- změna zaměření programování,
- změny ve způsobu ovládání lodí.

Jednotlivé změny jsou diskutovány dále.

4.5.1 Vypuštění pirátů a bojových lodí

Prvek náhodných útoků pirátů ovlivněných vztahem pirátů k jednotlivým aliancím byl ze hry vypuštěn, protože se jednalo o prvek porušující zásadu č. 1 – No grief. Navíc by tento prvek nutil hráče k častějším kontrolám stavu hry, což porušuje i zásadu č. 3 – No addiction. Na závěr je třeba říct, že tento prvek je v podstatě „placení výpalného“ a je tedy otázkou, zda má po stránce obsahové v „univerzitní“ hře být.

Bojové lodě byly také vypuštěny, protože bez pirátů postrádají smysl.

4.5.2 Vypuštění aliancí

Úlohou aliancí, jak jsou popsány v [1], bylo snížení nákladů na přistávání ve společných docích, společná obrana proti pirátům a rozšíření komunikace mezi hráči. Vzhledem k vypuštění pirátů tento prvek ztrácí většinu svého významu.

Z tohoto důvodu jsou aliance vypuštěny a nahrazeny prvkem společných korporací, které umožňují mnohem užší spolupráci mezi hráči. Tento prvek je považován za dodatečné rozšíření game designu a je uveden na konci kapitoly.

4.5.3 Změna zaměření programování lodí

Původním záměrem bylo umožnit hráči programovat vlastní navigaci lodí. Loď se měla pohybovat k planetám jejich „stíháním“ – průběžnou korekcí kurzu k současné pozici planety. Hráč měl mít možnost tuto navigaci vylepšit pomocí vlastního programu.

Takový herní prvek má však jen velmi omezený prostor pro zlepšování a dá se říci, že pro hru není zásadní. Jedná se jen o vyřešení problému pomocí znalostí z matematiky, fyziky a geometrie a výsledkem je numerická metoda, která vypočte pro konkrétní dvě tělesa bod jejich setkání. Díky zamýšlenému obchodu s těmito programy by pak ve hře vzniklo postupem času kvalitní řešení, které by stačilo nakoupit a dál by bylo možné celou programovací část hry ignorovat.

Je zde navíc problém se samotnou realizací. Pokud by bylo nutné každou loď v reálném čase řídit, bylo by vykonávání programů na serveru s rostoucím počtem hráčů velmi náročné a pokud by řízení bylo spuštěno na straně klienta, byl by problém s hráči, kteří nejsou momentálně ke hře připojeni (není vhodné, aby hráč byl nucen sledovat 10 minut letící loď jen proto, aby mohla letět po optimální dráze).

Dalším praktickým problémem bylo schvalování programů před jejich použitím ve hře. To jednak znesnadňuje přístup k tomuto hernímu prvku a také by tento proces vyžadoval podpůrný tým, který by schvalování vykonával. Vzhledem k faktu, že nelze automaticky kontrolovat, zda program neobsahuje nekonečné cykly,[11] přichází v úvahu jen ruční analýza kódu, kterou by s největší pravděpodobností nebylo možné personálně zajistit.

Z těchto důvodů bylo změněno programování lodí z pouhé navigace a nakládání lodí na možnost kompletního programového řízení ekonomické činnosti hráče. Hra by umožnila vytvoření vlastní umělé inteligence, která by prováděla herní akce v zastoupení hráče.

Vzhledem k tomu, že u takto pojatého systému neexistuje optimální řešení, je zde velký prostor pro kreativitu a experimentování. Podrobnosti této herní mechaniky jsou popsány v podkapitole 4.6.12.

4.5.4 Změny způsobu ovládání lodí

Jak již bylo zmíněno v předchozím bodě, hráčova loď měla „stíhat“ cíl svého letu. Tento způsob pohybu je však poměrně složitý a navíc je velmi vzdálen od reality vesmírného cestování. V neposlední řadě pak ztratil smysl v souvislosti se změnou zaměření programování, popsanou v předchozím bodě.

Místo toho bylo rozhodnuto, že se lodě po herním vesmíru pohybují automaticky. Výpočty drah jsou prováděny přímo a nevyžadují hráčův zásah. Dráha je počítána pro bod setkání lodi s planetou při zachování konstantní rychlosti pohybu. Pro navození větší realističnosti je vizualizace pohybu lodi transformována tak, aby se dráha lodi nepřiblížila k hvězdě blíže, než na stanovenou minimální bezpečnou vzdálenost. Tento efekt je pouze vizuální.

Hráč určuje dráhu lodi vybráním sekvence červích děr a cílové planety (základny). Loď pak po této trase letí automaticky. Změna trasy je možná pouze po výstupu z červí díry. Dynamická změna trasy je možná, ale pokud by byla do hry přidána, pak jako vylepšení

[11] Jedná se o problém zastavení algoritmu, jehož algoritmickou nerozhodnutelnost dokázal Alan Turing [12].

lodi, nikoli výchozí chování – nemožnost změnit trasu kdykoli zvyšuje vážnost rozhodování, především v případě hráčovy vlastní AI.

Dalším prvkem ovládání lodí, který byl z původního návrhu vypuštěn, bylo „ruční“ nakládání zásilek. Náklad je do lodi naložen automaticky a obchodní rozhraní zajistí kontrolu dostupných kapacit. Činnost nakládání by ve hře byla opakována příliš často a stala by se otravnou. (viz. zásada č. 2 – No grind)

4.5.5 Rozšíření původního konceptu

Dále byla hra rozšířena o následující prvky:

- vlastní továrny,
- možnost vylepšování lodí,
- nové formy obchodu (aukce, volný obchod),
- úrovně hráče, achievementy.

Popis těchto prvků je uveden v následující podkapitole.

4.6 Popis hlavních herních prvků a mechanik

V této podkapitole jsou popsány jednotlivé herní prvky a mechaniky. Tento popis je výchozím bodem pro implementaci těchto prvků. Popis na některých místech úmyslně vynechává konkrétní hodnoty různých parametrů (především ceny), které se budou v průběhu vývoje měnit na základě testování a celkového balancování hry.

4.6.1 Herní svět a mapa galaxie

Herní svět představuje část galaxie obydlená lidmi v budoucnosti (24. století). Galaxie se skládá z obydlených hvězdných systémů, které obsahují herní objekty a jsou propojeny pomocí „červích děr“, které v kontextu hry poskytují okamžitou přepravu z jednoho hvězdného systému do jiného.

Mapa galaxie je tedy graf, ve kterém hvězdné systémy reprezentují vrcholy a červí díry tvoří hrany. Ve hře je zobrazena jako rovinný graf se zářícími body reprezentujícími hvězdné systémy a linkami, představujícími propojení červími děrami. Tato mapa zároveň hráče informuje o přítomnosti jeho vlastních lodí v jednotlivých systémech (vhodná indikace je ikona lodi doplněná číslem).

Rozsah mapy galaxie by se měl pohybovat mezi 30 a 50 hvězdnými systémy. Větší množství by bylo matoucí. Rozložení mapy by mělo být nepravidelné.

4.6.2 Hvězdný systém

Hvězdný systém je tvořen hvězdou (systémy s více hvězdami nejsou v základní verzi zahrnuty), planetami a koncovými body červích děr. Celý systém je viděn z pohledu „shora“ ve 2D. Hvězda je stacionární, planety a koncové body červích děr obíhají po kruhových či eliptických oběžných drahách.

Reprezentace hvězdných systémů nemá za cíl zachování realističnosti. Cílem je poskytnout hráči vizuální orientace. Důraz je tedy spíše kladen na unikátnost jednotlivých systémů. Je třeba také dodržet praktické vzdálenosti mezi jednotlivými oběžnými drahami pro jejich snadné rozlišení.

Množství planet ve hvězdném systému by se mělo pohybovat od 4 do 10. Systémy s více planetami by měly obsahovat jednu nebo dvě větší planety (plynné obry), po vzoru Sluneční soustavy.

Oběžné doby planet by se v každém systému měly výrazně lišit. Planety blíže vnějšímu okraji mají oběžnou dobu větší, než planety blíže hvězdě. Oběžné doby by se měly pohybovat v řádu jednotek hodin reálného času.

Každý koncový bod červí díry propojuje hvězdný systém s jedním dalším hvězdným systémem. Tyto koncové body obíhají po vzdálených oběžných drahách s výrazně delší oběžnou dobou oproti planetám (kolem jednoho dne). Běžný systém obsahuje 2 červí díry, minimum je 1, maximum 6.

Obydlené planety jsou ve hře reprezentovány základnami (popis funkce základen je uveden v 4.6.4). Pouze tyto planety jsou aktivními herními objekty. Neobydlené planety mají pouze dekorativní účel. Herní vesmír by měl obsahovat několik hustě zalidněných systémů a s rostoucí vzdáleností od nich by měla hustota osídlení klesat.

4.6.3 NPC

Úlohou NPC (Non-player character, postava neovládaná hráčem) ve hře je oživit herní svět a také zajistit základní služby pro potřeby hráčů. Lze je také využít jako nástroj pro balacování hry, kterým je možné nepřímo ovlivňovat chování hráčů. Ve hře se vyskytují dva základní typy NPC: NPC přepravci a NPC korporace/frakce.

NPC přepravci jsou lodě řízené počítačem, vykonávající ve hře stejnou přepravní činnost jako hráč. Zaměřují se na volný obchod (viz. dále). Jejich primárním účelem je oživit svět. NPC lodě se pohybují po předem stanovených obchodních cestách, nesnaží se chovat inteligentně na základě stavu trhu.

NPC korporace jsou virtuální společnosti, pod jejichž názvy se skrývají hrou poskytované základní služby. Jejich smyslem je oživit herní prostředí a zajistit kontext pro některé herní prvky, například společnosti, které vyrábí kosmické lodi nebo výrobci zboží, které hráč přepravuje. Dále vystupují v roli zadavatelů kontraktů (viz. 4.6.7)

NPC frakce mají stejný účel jako NPC korporace, představují například populaci planety, politická uskupení apod.

4.6.4 Základny

Každá obydlená planeta je reprezentována základnou. Tu si lze představit jako kolonii nebo hlavní město. Hráčova loď zde může přistát a provádět interakci se službami poskytovanými základnou. Funkčně pak základna představuje herní objekt sloužící k interakci s herní mapou.

Každá základna je vybavena následujícími službami:

- kosmodrom s možností nákupu a opravy lodí,
- nákup pozemků,
- služby spojené s obchodem: NPC obchodník, depozit
- NPC továrny
- seznam dostupných kontraktů,
- depozit zboží (pro obchodování v aukci).

Přistání lodí v kosmodromu je spojeno s fixním poplatkem. Oprava lodí je také standardní funkce s pevně danou cenou. Dále kosmodrom zásobuje hráčovu loď palivem (automatický poplatek při odletu). Nákup lodi je dostupný globálně, hráč si na jakékoli základně může koupit libovolnou loď.

Obchod je rozebírán samostatně v bodě 4.6.7. V tom samém bodě je rozebrán i systém kontraktů.

NPC továrny zajišťují výrobu zboží pro potřeby volného obchodu. Jsou spojeny s fungování NPC Obchodníka, který je také popsán v bodě 4.6.7.

Pozemky

Velmi důležitou funkcí základny je možnost vlastnit pozemek. Ten je reprezentován jako plocha složená ze čtvercových polí. Na této ploše může hráč stavět vlastní budovy (popsány dále v 4.6.6).

Hráč nakupuje jednotlivá pole po skupinách tak, aby vznikla vždy obdélníková plocha. Maximální množství polí je omezeno na každou základnu, maximum je 8x8 polí. Pozemky je také možné nakupovat a prodávat v aukci (prodány jsou i s postavenými budovami).

Hráč platí při každém nákupu pozemku daň, která se zvyšuje úměrně s celkovým množstvím jím vlastněných pozemků v celém herním světě (růst je exponenciální). Tato daň je uplatněna i při nákupu pozemku v aukci (započítáno v ceně pro individuální hráče).

Budovy, které je možné na pozemcích stavět, mají ve většině případů rozlohu jednoho pole. Některé dražší budovy pak mohou obsadit i více, maximálně však plochu o rozloze 3x3. Omezení by měla především sloužit k větší specializaci základen hráčů a tím vytvořit zajímavé prostředí.

4.6.5 Zboží

Jak již bylo uvedeno, hlavní náplní hry je přeprava zboží. Pod tímto obecným pojmem si lze představit libovolnou kombinaci surovin a výrobků, kterými lze v galaxii obchodovat.

Druhy zboží jsou rozděleny do dvou základních kategorií: **běžné** a **speciální** zboží.

Běžné zboží může převážet každá hráčova loď. Jedná se například o potraviny, náhradní díly, prefabrikáty, elektroniku, textil, lodní komponenty, pitnou vodu a fúzní články (zdroj energie). Dále se jedná o některé typy nerostných surovin, které nevyžadují zvláštní zacházení.

Tato kategorie druhů zboží je základ, který hráče provází od počátku hry. Trh s těmito druhy zboží je větší, nepřináší proto takové zisky, zato je však více stabilní a nevyžaduje speciální vybavení.

Speciální zboží vyžaduje, aby byla hráčova loď speciálně vybavena pro jeho přepravu. Do této kategorie patří:

- nebezpečný náklad, jako jsou radioaktivní, hořlavé a výbušné látky,
- živé organismy (pasažéři, dobytek), vyžadující podporu života v přepravních prostorách,
- kryogenní náklad, vyžadující udržování stálého chlazení po dobu přepravy.

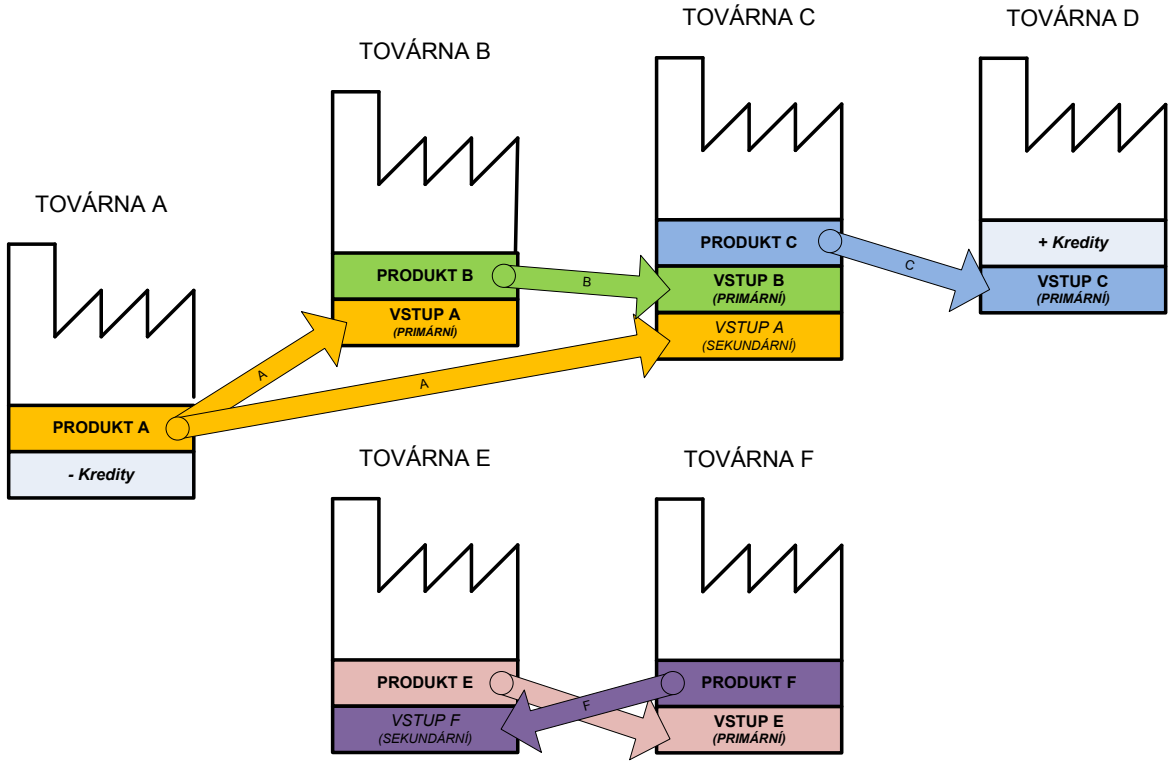
Přeprava speciálního zboží je pro hráče obecně výhodnější, je však spojena s vyššími vstupními náklady (speciální vybavení lodí je drahé) a může při jeho přepravě docházet k nehodám.

Jednotka nákladu (U)

Protože hlavním herním prvkem je přeprava nákladu, hra využívá pro vyjádření množství speciální jednotku **jednotku nákladu** (anglicky cargo unit) se zkratkou **U**. Tato jednotka vyjadřuje množství zboží přepočítané na počet přepravních kontejnerů s omezenou maximální hmotností a objemem. Na různé druhy nákladu se aplikuje jedno z těchto dvou omezení tak, aby žádný z limitů nebyl překročen.

Jednotky nákladu jsou vyjadřovány desetinným číslem s přesností na desetiny.[12] 1 U odpovídá[13] hmotnosti 100 t nebo objemu 100 m^3.

Zavedení těchto jednotek poskytuje hráči snazší porovnání cen jednotlivých druhů zboží vzhledem k volným přepravním kapacitám, které má k dispozici. Může přímo porovnávat ceny různých druhů zboží pro konkrétní počty U.



Obrázek 4.3: Diagram vzájemných zásobovacích vztahů továren.

4.6.6 Budovy a továrny

Jak již bylo uvedeno, na jednotlivých základnách lze stavět budovy. Jsou tři základní typy budov: **dok** (angl. dock), **sklad** (angl. warehouse) a **továrna** (angl. factory).

[12] Desetinné místo je využito pouze proto, aby působily tyto jednotky „technicky“. Jedná se tedy jen o efekt.

[13] Myšlenka přepravních kontejnerů a jejich parametry jsou inspirovány standardizovanými kontejnery používanými v intermodální dopravě [13]. Pro použití v mezihvězdné přepravě bylo uvažováno o větších objemech a hmotnostech. Jako základ pro zvolené hodnoty posloužila představa jedné lokomotivy nebo naloženého železničního nákladního vozu.

Dok

Postavením doku získá hráč vlastní prostředek pro opravy a doplňování paliva lodí. Jeho velikost je 3x3 a hráč jej na každé základně může postavit jen jednou.

Sklad

Sklad slouží k uchovávání zásob. Umožňuje hráči interagovat s trhem planety i bez přítomnosti vlastní lodi (popsáno v 4.6.7). K dispozici jsou tři velikosti skladů: 1x1, 2x2 a 3x3. Postavením dalšího skladu na základně hráč zvýší kapacitu, funkčně se tyto sklady chovají jako jeden.

Továrna

Továrny jsou základním prvkem herní ekonomiky. Jejich účelem je produkovat zboží. Každá továrna může produkovat jeden druh zboží (je tedy více druhů továren), svůj produkt. Pro tuto produkci jsou třeba výrobní vstupy – zboží spotřebované při výrobě produktu.

Vstupy továrny jsou dvojího druhu: *primární* a *sekundární*. **Primární vstup** je hlavním vstupem výroby. Spotřebou jednotek primárního vstupu jsou vytvářeny jednotky produktu. **Sekundární vstup** je nutný k provozu továrny, jeho spotřeba je ale výrazně nižší. Poměry jsou vždy vztaženy k 1 U produktu. Například:

1 U palivový článek = 2 U lithium + 0,1 U plast + 0,2 U kyslík;

kde plast a kyslík jsou sekundární vstupy. Maximální poměr primárního vstupu k produktu je 1:5 a sekundárního 1:0,5.

Továrna vyrábí produkt konstantní rychlostí s periodou společnou pro všechny továrny. Pro každý druh zboží sloužící jako vstup má továrna samostatný zásobník. Zároveň má také zásobník na produkt. Výroba probíhá, dokud je v zásobnících vstupů dostatek jednotek pro výrobu a zároveň dokud je místo v zásobníku na produkt. [14]. Továrny si umí automaticky doplňovat zásobníky ze skladu hráče, pokud je k dispozici na stejné základně. Vyrobený produkt je také možné automaticky ukládat do skladu, továrna sama jej však nemůže prodávat (to je funkcí obchodníka ve skladu, viz. 4.6.7). Pokud dojde k zastavení výroby, dojde k jejímu obnovení automaticky po vyřešení příčiny.

Obrázek 4.3 ukazuje různé typy továren a jejich vzájemného propojení do zásobovacích řetězců. Jednotlivé druhy zboží jsou tedy transformovány postupně od základních surovin přes meziprodukty až po konečný produkt, který je spotřebován. Tento řetěz je znázorněn továrnami **A**, **B**, **C** a **D**.

[14] Výroba nemá žádné další náklady. Spojení výroby s hrazením finanční částky by se mohlo zdát jako vhodné, tento poplatek by však mohl „zruinovat“ hráče v jeho nepřítomnosti. Cena výroby je proto přenesena na spotřebu zboží, které je třeba aktivně nakupovat/produkovat.

Továrna **A** představuje zdroj zboží, jehož vstupem je herní měna. Může se jednat například o důl, který produkuje některou ze základních surovin na začátku. Tento typ továrny jako jediný nemůže být vlastněn hráči. Jeho účelem je regulování herní ekonomiky nastavováním cen výchozích surovin prostřednictvím NPC.

Továrna **B** je jednoduchá továrna s jedním primárním vstupem **A** a jejím produktem je **B**. Továrna **C** je pak ukázkou Továrny s primárním i sekundárním vstupem. Pro jednoduchost systému by žádná továrna neměla mít v kombinaci více než 5 vstupů.

Továrna **D** je spotřebitelem zboží. Zboží je zde přeměňováno na zisk. Tento typ továrny je opět určen pro NPC jako způsob pro odebírání zboží z oběhu.

Továrny **E** a **F** demonstrují cyklickou závislost mezi továrnami. Továrna **E** má produkt **F** jako sekundární zdroj, což přeneseně znamená, že na výrobu produktu **E** je jeho část spotřebována. Protože se však jedná jen o sekundární spotřebu, je zajištěno, že bude bilance produktu **E** v cyklu vždy kladná.

Tento systém umožňuje na základě jednoduchých pravidel vytvořit komplexní obchodní vztahy a vzniklá ekonomika může být velmi dynamická. Zároveň obsahuje jasné balanční nástroje v podobě zdrojů a spotřebitelů.

4.6.7 Obchod

Hra poskytuje hráči tři základní typy obchodních příležitostí:

- volný obchod,
- kontrakty,
- vesmírná aukce.

Smyslem je nabídnout hráči rozmanitost v hlavní náplni hry a zároveň zajistit dostatečnou pružnost herního systému pro případné rozšiřování.

Volný obchod

V rámci volného obchodu může hráč nakupovat a prodávat zboží. Každá obydlená planeta (planeta se základnou) umožňuje hráči nakoupit a prodat zboží za místní cenu. Tato cena se mění na základě nabídky a poptávky. Díky proměnlivosti cen nelze předem zajistit ziskovost každé cesty a hráč proto musí dobře zvažovat jednotlivé nákupy a prodeje v kontextu aktuální situace ve hře.

Hlavní výhoda této formy obchodu je její okamžitá přístupnost. Hráč může nakoupit zboží, jehož cena je momentálně výhodná a teprve potom hledat místo pro odbyt. Pokud se mu daná obchodní cesta nevydaří, může se v momentě rozhodnout prodat zboží jinde.

Volný obchod je zprostředkováván pomocí **obchodníků**. Základním je **NPC obchodník**, každý hráč však může vytvořit vlastního obchodníka, pokud postaví příslušnou budovu (sklad). Obchodník pracuje s omezenou zásobou zboží, kterému je nastavena výkupní a prodejní cena pro jednotlivé druhy zboží a počty zásob, které musí udržovat (v případě výkupu).

Trh je rozhraní volného obchodu (inspirováno hrou Port Royale 2, viz. obrázek 4.2). Obsahuje výpis druhů zboží. Ke každému druhu je uvedena aktuální prodejní cena, aktuální výkupní cena, množství jednotek na skladě a odhadovaná cena za požadované množství. Tyto ceny se stanovují na základě nabídek a poptávek obchodníků na základně.

- Hráč **nakupuje** zboží tak, že označí příslušný druh zboží, zadá požadované množství a maximální přijatelnou cenu na jednotku. Poté může provést nákup. Zboží bude nakoupeno od obchodníka s nejlepší cenou (nejlepší nabídka). Pokud jeho zásoby nebudou stačit, bude zbylé množství nakoupeno od dalšího obchodníka v seznamu seřazeného podle ceny. Není-li v okamžiku transakce k dispozici dostatečné množství zboží s cenou nižší nebo rovnou maximální přijatelné ceně, je zakoupeno pouze dostupné množství.
- **Prodej** zboží je realizován obdobně. Hráč zadává svou minimální prodejní cenu zboží. Transakce proběhne jen pro množství, které lze prodat za cenu vyšší nebo rovnou, počínaje nejlepší nabídkou. Pokud hráč vlastní sklad na dané základně, může navíc vytvořit nabídku stanovením pevné prodejní ceny druhů zboží, které má na skladě a případně minimální zásobu, která musí ve skladu zůstat. Publikování této nabídky je provedeno se zpožděním (aplikováno také, pokud je změněna cena; množství lze měnit libovolně).
- **Výkup** zboží je možné provést pomocí vlastního skladu na příslušné základně. U každého typu zboží může hráč nastavit výkupní cenu a cílovou úroveň skladu. Nákup je pak prováděn automaticky z dostupných nabídek.

Aby nebylo možné manipulovat s trhem pomocí tohoto systému a způsobit kolaps, NPC obchodník poskytuje nejdražší prodej a nejnevýhodnější výkup každého typu zboží.

Kontrakt

Kontrakty lze chápat jako obdobu „misí“ nebo „questů“ v jiných hrách. Jedná se o časově omezené smlouvy, které zavazují ke splnění určité sady úkolů. Jejich smyslem je poskytnout hráči větší rozmanitost ve stylu hraní hry a jasné cíle.

Zadavatelem kontraktů jsou NPC korporace. Jejich generování je automatické a v dostatečném množství, odrážejícím počet aktivních hráčů.

Každá planeta nabízí přehled kontraktů, které se k ní vztahují (je zdrojem nebo cílem kontraktu). Hráč se může rozhodnout přijmout kontrakt, čímž se mu přidá do seznamu úkolů a hra dále sleduje jeho plnění. Pokud hráč nesplní úkoly, ale odvedl část práce, bude odměněn s penalizací, odvíjející se od množství splněné práce.

Kromě zpestření běžné hry je možné využít kontrakty v tutoriálu (postupné plnění úkolů, které seznámí hráče s jednotlivými prvky hry) a případně potřeby i jako nástroj vyprávění příběhu ve hře. Lze také uvažovat o možnosti zadávat kontrakty přímo hráči. Tak by vznikla další cesta, jakou hráči mohou generovat herní obsah a rozšiřovat tak herní zážitek sobě i ostatním.

Aukce

Poslední formou obchodu je aukce. Ta umožňuje hráčům obchodovat vzájemně se zbožím, které vlastní na globálním trhu. Aukce je rozdělena na dvě sekce: prodej zboží a jeho nákup.

Prodej zboží je aukce podle modelu výkupní aukce, ve které prodávající hráči uveřejňují nabídku zboží s vyvolávací cenou a kupující mohou k této ceně přihazovat. Vyhrává nejvyšší nabídka po skončení aukce. Navíc může mít nabídka stanovenu výkupní cenu, kterou lze přijmout v jakémkoli okamžiku aukce a tím získat zboží. Pokud není uvedena, aukce probíhá podle anglického modelu.

Nákup zboží probíhá jako obrácená aukce. Nakupující uveřejňují nabídky nákupu s poptávaným množstvím a **maximální výkupní cenou**, ukládanou do **depositu** do skončení aukce. Jednotliví prodávající nabízejí toto množství s cenou nižší než je aktuální nejnižší nabízená cena. Vyhrává nejnižší nabídka po skončení aukce. I v tomto případě může být nastavena výkupní cena, která představuje minimální cenu aukce a pokud některá nabídka bude odpovídat této ceně, automaticky vyhrává.

Cena podání nabídky v aukci je stanovena fixním poplatkem. Dále hra odečte 5% z vítězné částky (v případě prodeje z konečné prodejní ceny, v případě nákupu z částky zaplacené nabízejícímu). Přihazování v aukci není zpoplatněno.

Finanční částky odpovídající jednotlivým nabídkám jsou uloženy do depozitu (hráč musí mít k dispozici sumu při podávání nabídky). Pokud je nabídka „přebita“, je tato částka vrácena na konto hráče po uplynutí „transakční lhůty“ 1 den.

Zboží je uloženo v depozitu na planetě, kde se nacházelo při podání nabídky. Hráč může opakovat nabídku v aukci bez vyzvednutí tohoto zboží, podmínkou je, že množství nabízeného zboží je stejné. Pokud se rozhodne nabídku znovu nepodat, musí si zboží vyzvednout lodí nebo přesunout do svého skladu na téže planetě. To samé platí o zboží, získané v aukci.

Aby bylo zajištěno, že si hráči zboží vyzvednou a nebudou deposit aukce používat jako alternativu skladů, je za každé zboží uložené v depositu déle než 24 hodin od skončení aukce připočteno penále k poplatku za novou aukci. Tento poplatek roste exponenciálně s množstvím zapomenutého zboží v depositu a hráč je při vypisování nové aukce upozorněn na rostoucí cenu.

Dále jsou obě formy aukce časově omezeny s minimální dobou trvání 1 den. Délka aukce je nastavována ve dnech, maximum je 7 dní. Pro zabránění vykalkulované manipulace s aukcí na poslední chvíli je zbývající doba v posledním dni aukce uváděna v šestihodino-

vých intervalech: „méně než 6 hodin“, „méně než 12 hodin“, atd. Pokud je zbývající doba větší, než 24 hodin, stačí uvádět počet zbývajících dní.

Herní měna a cenové relace ve hře

Hra využívá jednotnou měnu s názvem *kredity* (anglicky *credits*), jejichž zkratka je **CD**. Jedná se o silnou měnu, která by měla mít větší hodnotu, než současná hodnota eura (rok 2012). Jako základ pro oceňování ve hře je proto možné vzít reálné ceny.

Protože se jedná o sci-fi prostředí, je samozřejmě třeba ocenit i věci, které v našem světě ještě neexistují a jejich vytvoření by vyžadovalo vynaložení enormních prostředků. Zde je však možné vycházet z cen věcí, které plní danou úlohu v našem světě. Například při oceňování kosmických lodí lze vycházet z cen námořních lodí nebo velkých letadel, které v současném světě plní podobnou úlohu, jakou by plnily nákladní kosmické lodě v budoucnosti.

4.6.8 Kosmická loď

Kosmická loď je základním prostředkem pro plnění herních cílů. Hráč získává na začátku hry jednu loď, model „Shipunto“ a další si může v průběhu hry koupit. Každá loď se skládá z **komponent**, které určují její vlastnosti. **Model** lodi definuje, které komponenty lze na loď nainstalovat a také množství **slotů** pro volitelné komponenty.

Komponenty lodi

Komponenty lodi představují jednotlivé subsystémy lodi, které hráč může volit z poskytnuté palety. Jsou instalovány do slotů, které definuje model lodi, a jsou rozděleny na tři funkčně specifické typy:

Moduly[15] jsou část lodního trupu jako takového. Každá loď je z nich složená a změna konfigurace modulů může ovlivnit fyzický vzhled lodi. Moduly jsou základním stavebním kamenem pro návrh lodí.

Interní subsystémy[16] (dále používána zkratka INS) představují vnitřní technické vybavení, které je společné pro celou loď a týká se jejího vnitřního chodu. Nemají viditelný efekt, ovlivňují však statistiky lodi.

Externí subsystémy[17] (dále používána zkratka EXS) jsou zařízení, sloužící k interakci s okolním vesmírem. Jejich instalace přidává lodi schopnosti, které nejsou součástí běžného modelu.

Základní loď se vždy skládá z následujících komponent:[18]

- modul motoru (povinný),
- modul palivové nádrže (povinný),
- modul základního nákladového prostoru (musí existovat nákladový prostor, lze zaměnit za jiný typ),
- INS navigační počítač (povinný),
- konečný počet prázdných slotů pro INS, EXS a moduly (maximum 5 od jednoho typu, 10 celkem).

Do volných slotů nelze instalovat další modul motoru, navigační počítač ani palivovou nádrž. Nákladní prostory a palivovou nádrž lze rozšiřovat (v praxi náhrada za větší model). Instalace některých modulů může poskytnout jeden další volný slot konkrétního typu.

Statistiky lodí

Každá loď je reprezentována sadou statistik, které popisují její výkon v různých oblastech. Jejich porovnáním může hráč rozhodnout o tom, která loď se mu víc hodí. Pro zpestření hry je vhodné nastavovat konkrétní hodnoty tak, aby výsledný efekt byl „kámen, nůžky, papír“, tzn. ve výsledku není žádný nejlepší model lodi, ale každý se hodí na některý typ úlohy.

Dále je loď popsána těmito statistikami:

- Základní statistikou lodi je její **maximální kapacita nákladu** vyjádřená v U. Ta je dána výkonem motoru a určuje, jak „velká“ loď může být. Instalováním modulů je tato kapacita spotřebovávána. Nákladový prostor je speciální modul, který umožňuje kapacitu jím obsazenou využít jako přepravní kapacitu pro zboží.
- Dalším parametrem lodi je její **míra opotřebení**. Toto je provozní parametr udávaný v procentech. Loď je provozem opotřebovávána a vyžaduje pravidelnou opravu.
- **Dolet lodi** je dán kapacitou palivové nádrže a určuje maximální vzdálenost cíle lodi v počtu hvězdných systémů, které je třeba k cíli procestovat bez mezipřistání. Protože loď nemůže znát polohu planet a koncových bodů červích děr v jiných systémech, nese si zásobu paliva potřebnou pro nejhorší možnou spotřebu. Bezpečnostní systém neumožní lodi naplánovat cestu, kterou by nemohla dokončit.
- **Rychlost lodi** je udávána v jednotkách ISP (specifický impulz). Jedná se o poměr tahu ke spotřebě paliva za jednotku času. Použití této jednotky je pouze pro efekt, hlavní úlohou hodnot je možnost porovnat, která ze dvou lodí je rychlejší.

[18] Seznam nezahrnuje systémy pro podporu života, předpokládá se, že základní loď je bezpilotní.

- **Velikost paměti navigačního počítače** lodi souvisí s vytváření programů pro jejich řízení. Omezuje množství příkazů programu a množství proměnných. Podrobně je celý systém popsán v 4.6.12.
- Posledním důležitým parametrem lodi je **velikost jednotlivých nákladových prostorů a jejich obsazení**, obojí udávané v U.

Provoz lodi

Loď je hráčem ovládána způsobem již popsaným v 4.5.4. Loď při letu spotřebovává palivo, které je třeba doplňovat.

Kromě toho dochází k opotřebení lodi, jak již bylo zmíněno. Týká se to pouze lodí, které jsou na cestě nebo aktivně vykonávají program na základně. Role opotřebení lodi souvisí především s programováním lodí a je blíže vysvětlena v 4.6.12.

Pokud je loď opotřebena na více než 60%, začne na ní docházet k nehodám a tento fakt je signalizován hráči. Pokud dojde k 100% opotřebení, loď nemůže dál fungovat – pokud k tomu dojde za letu, na lodi dojde ke katastrofickému selhání a bude ztracena (zničena).

Opotřebení lodi je relativně pomalé, loď musí být schopna absolvovat více cest, než vznikne potřeba její opravy. Dále je vztah cena opravy lodi k míře jejího opotřebení nelineární, více opotřebená loď je dražší na opravu, takže je hráč motivován k provádění oprav včas. Z druhé strany však minimální cena opravy znevýhodňuje bezmyšlenkové opravování lodi po každé cestě.

4.6.9 Úroveň hráče a zkušenostní body

Úroveň hráče neboli jeho **level** je číslo, vyjadřující hráčův postup ve hře. Dosažená úroveň zároveň vymezuje možnosti, které jsou hráči k dispozici. To umožňuje otevírat hráči hru postupně tak, aby měl dostatek času se s každým představeným herním prvkem seznámit a nebyl hned zahlcen množstvím voleb, které může/musí učinit. Zároveň mohou být hráči znemožněny volby, které v dané fázi hry nemají smysl (například nákup pozemků na začátku hry, kdy hráč ještě nemá dostatečné bohatství na jejich efektivní využití).

Každý hráč začíná na 1. úrovni, maximální úroveň je 30. Pro dosažení vyšší úrovně musí hráč získat určené množství tzv. **zkušenostních bodů** (anglicky experience points), kterými je odměňován v průběhu hraní za plnění jednotlivých herních aktivit (plnění kontraktů, obchodní činnost, průzkum, případně další).

Množství potřebných bodů mezi jednotlivými úrovněmi roste s každým dalším levelem. Nemělo by se jednat přímo o exponenciální růst, ale dosažení každého dalšího levelu by mělo být náročnější. Vzhledem k tomu, že schopnost hráče plnit herní aktivity v čase přirozeně roste (hráč bohatne), je třeba rychlost získávání zkušenostních bodů a množství bodů potřebných pro dosažení jednotlivých úrovní řádně otestovat a vybalancovat s ohle-

dem na zamýšlenou herní dobu. Určení přesných hodnot je proto záležitostí pro pozdější fáze vývoje.

4.6.10 Achievementy

Achievement (anglicky; lze přeložit jako úspěch; v souvislosti s hrami se většinou používá počeštěný anglický výraz) představuje dílčí cíl, který může hráč dobrovolně plnit v průběhu hry. Účelem systému achievementů je nabídnout hráči milníky v postupu hrou, motivovat ho a zároveň dokládat jeho postup hrou nad rámec běžného systému levelů. Zároveň slouží jako forma odměny pro splnění konkrétních částí hry a v neposlední řadě pro vyjádření sociálního postavení hráčů ve hře ve smyslu dokladu o jejich „veteránství“ (v případě viditelnosti ostatním hráčům).

Využití systému achievementů je v současných hrách běžné a v zásadě jsou si v mnohém podobné. Jako referenční implementaci pro inspiraci lze vzít systém, který ve svých hrách využívá společnost Blizzard (*World of Warcraft*, *StarCraft II*, *Diablo III*). Dalším běžně známým systémem je implementace achievementů, která je součástí digitální distribuční platformy Steam.

Achievementy lze dělit na tři základní skupiny (převzato z Episody *Achievements* pořadu Extra Credits [14]):

- *Nevyhnutelné* – získané normálním hraním hry.
- *Dobrovolné* – představují úkoly a cíle, jejichž splnění si může hráč sám zvolit nad rámec hlavního zaměření hry.
- *Inspirující* – předkládají hráči výzvy, které mění herní zážitek a umožňují hráči vyzkoušet alternativní způsoby hraní.

Při návrhu konkrétních achievementů pro hru je třeba mít se na pozoru před velkým množstvím nevyhnutelným achievementům, protože nevnášejí do hry žádnou přidanou hodnotu. Naopak dobrovolné a zejména inspirující achievementy obohacují herní zážitek a měly by proto tvořit většinu z celkového počtu.

Pro potřeby hry *Space Traffic* rozeznáváme několik typů achievementů, lišící se svou formou a způsobem, jakým jsou plněny. Jsou to:

Jednorázový achievement je získán při dosažení určitého kritéria. Například nákup první lodě, postavení první továrny a podobně. Většinou se jedná o *nevyhnutelný achievement*, proto by se jich ve hře nemělo vyskytovat moc.

Víceúrovňový achievement získává hráč na základě kritérií s odstupňovanou obtížností. Tato forma achievementu je vhodná především pro úkoly, které jsou časově náročné nebo mají vysokou obtížnost. Hráč je tak motivován/odměňován postupně.

Typ seznam je plněn postupným "sbíráním" jednotlivých položek seznamu. Například navštívení všech hvězdných systémů na herní mapě.

Typ nekompletní seznam je obdoba seznamu, kde pro splnění achievementu je nutné získat určitý počet položek seznamu. Tuto formu lze využít například jako součást víceúrovňového achievementu se seznamem.

Typ počítadlo získává se sbíráním bodů, měl by obsahovat několik úrovní splnění (např.: bronzová, stříbrná, zlatá). Většinou jde o dlouhodobé achievementy, například množství peněz získaných obchodem (1 000, 100 000, 1 000 000).

Meta-achievement funguje jako seznam, ale položkami jsou achievementy. Lze ho využít jako shrnující achievement pro splnění určitého tematicky souvisejícího seznamu achievementů. Z důvodu přehlednosti by nemělo být využíváno více jak jedné úrovně zanoření. Dále má tento typ achievementu význam pouze u komplikovaného systému, který by nebyl v případě lineárního seznamu achievementů dostatečně přehledný.

Každý achievement má své unikátní jméno, které vtipně vyjadřuje jeho kritéria nebo na ně naráží. Dobrým zdrojem inspirace jsou známé citáty. Název achievementu je doplněn výstižnou ikonou. Přesná kritéria získání achievementu jsou pak součástí jeho popisu. Je třeba dbát na to, aby tato kritéria nebyla vylučující – zisk jednoho achievementu nesmí znemožnit získání jiného.

Dosažení achievementu je hráči signalizováno ve hře dostatečně výrazným způsobem, nejlépe zprávou uprostřed obrazovky, která po několika sekundách sama zmizí. Zároveň by měla být doplněna zvukovým efektem.

Seznam dostupných achievementů včetně vyznačení těch splněných je každému hráči přístupný ve hře. Lze uvažovat i o jejich dostupnosti pro ostatní hráče jako součást hráčského veřejného profilu. Součástí rozhraní je i zobrazení počtu splněných achievementů a jejich celkového počtu, které vyjadřuje míru dokončení hry.

Příklady achievementů pro Space Traffic:

- „Království za osla." – Hráč si zakoupí svou druhou loď.
- „Nabídka, která se neodmítá" – Hráč splnil první kontrakt.
- „Všechny tyto světy jsou vaše!" – Hráč prozkoumal celou mapu.

4.6.11 Hodnocení hráče, skóre a žebříčky

Kromě získávání zkušeností a úrovní nabízí hra ještě počítání skóre. Toto skóre je počítáno pro omezená časová období (jeden den, týden a měsíc) a jeho hodnota odpovídá čistému zisku hráče v tomto období. Z těchto hodnot jsou pak sestaveny žebříčky, které umožňují hráčům porovnat svoji úspěšnost ve hře.

Dále hra sleduje denní rekordy, jako například nejvyšší jednorázová obchodní transakce, nejvyšší produkce a podobně. Denní rekordy jsou publikovány vždy následující den.

Tyto mechaniky mají za cíl povzbudit soutěživost hráčů a dodat jim motivaci ke zdokonalování svého hraní.

4.6.12 Programování lodí a hráčem vytvořená umělá inteligence

Možnost vytvářet programy, které budou automaticky řídit hráčovo obchodní impérium je jednou z charakteristických vlastností hry *Space Traffic*. Tento prvek je koncipován tak, aby hráči postupně otevíral možnosti a seznámil ho se základními aspekty programování jako takového.

Hra obsahuje dva programovací jazyky, rozdělující programování ve hře na dvě úrovně. První z nich je jednoduchý jazyk **Starship Basic**, jehož předlohou je jazyk **Sinclair BASIC** (viz. *Sinclair ZX Spectrum: Basic Programming* [15]). Druhým je pak skriptovací jazyk **Lua**.[19]

Starship Basic a řídící program lodi

Jak již bylo řečeno, jazyk Starship Basic vychází z jazyka Sinclair BASIC, používaných v počítačích Sinclair ZX Spectrum. Tyto osmibitové počítače používaly jednoduchý dialekt BASICu, vykonávaný vestavěným interpretem. Volba tohoto jazyka jako základu vychází z myšlenky, že moderní programovací jazyky jsou příliš komplikované pro použití ve hře a mnoho jejich vlastností je dalece nad rámec zaměření tohoto herního prvku. Navíc je díky ploché struktuře programu a číslování řádků možné seznámit hráče se základními principy fungování počítače. [20]

Přehled jednotlivých příkazů a jejich význam je k dispozici v příloze C. Program je vykonávány v „navigačním počítači“ kosmické lodi.

Každá loď má k dispozici omezenou „paměť“ o kapacitě N (výchozí 20), která je rozdělena na „program“ a „proměnné“. Program je zapisován po jednotlivých příkazech s očíslováním jednotlivých řádků programu od 0 do N-1 (více příkazů na řádek oddělených dvojtečkou není na rozdíl od předlohy podporováno). Na rozdíl od původního systému budou řádky programu číslovány automaticky, a pokud bude příkaz vložen mezi, budou automaticky přečíslovány (včetně příkazů GO TO a GO SUB; pokud by v uvedených příkazech byla použita proměnná, budou tyto řádky hráči indikovány jako potenciální problém).

[19] Web jazyka Lua - `http://www.lua.org/`

[20] Moderní programovací jazyky jako Java nebo C# dosahují takové úrovně abstrakce, že je velmi obtížné pod nimi vidět principy, na kterých počítače fungují. Doplníme-li je o výkonné procesory, operační paměť v řádu gigabytů a pokročilé frameworky, je obtížné si uvědomit důsledky lineární povahy správy paměti, výkonnostních omezení daných složitostí algoritmů, omezenosti dostupné paměti apod. Volba jednoduchého jazyka jako je BASIC a jeho omezení na virtuální prostředí s tvrdými limity může umožnit hráčům získat cenné zkušenosti.

Rozdělení paměti na program a proměnné je nutné určit, nelze jej měnit za běhu programu. Každá proměnná zabírá místo o velikosti 1, výjimkou jsou pole, která zabírají místo rovné počtu prvků.

Kromě této paměti má každá loď k dispozici „lodní deník“, který dokáže uchovat 8 řádek po 30 znacích textu.[21] Řádky jsou vkládány příkazem PRINT s běžným mechanismem rolování. Tento deník slouží pro ladící výpisy a provozní hlášení, která by hráče mohla zajímat.

Jedním z možných rozšíření je vytvoření grafického editoru progamů v tomto jazyce, ve kterém by byly jednotlivé příkazy nahrazeny čtvercovými ikonami, které by se skládali do dlouhého řádku/sloupce. Skoky by byly naznačeny šipkami. Tato vizuální pomůcka by usnadnila vstup hráčů do problematiky programování a zároveň pomohla s orientací.

Vykonávání programu lodi

Vykonávání programů lodí běží lineárně, hráči sdílejí strojový čas. Jedním ze základních problémů tohoto zpracování je nebezpečí vzniku nekonečných smyček, které by nakonec způsobily nefunkčnost celého systému. Tomuto problému nelze zabránit, lze však omezit jeho dopad.

Příkaz FLY TO představuje přirozenou pauzu v provádění programu lodi. Lze předpokládat, že snahou hráče bude udržet lodi v pohybu. Po přistání lodi je automaticky provedeno omezené množství „prioritních příkazů“, které jsou zpracovány v sekvenci. Pokud v rámci tohoto krátkého úseku loď znovu odstartuje, není nijak ovlivněna.

Pokud lodi při pobytu na základně nestačí vymezené množství příkazů, je provádění programu převedeno na společný plánovač, který rozdělí vyhrazený strojový čas mezi všechny čekající lodě. Každé lodi je počítán počet příkazů od dokončení posledního FLY TO – čítač příkazů od přistání. Čím vyšší je hodnota tohoto čítače, tím pomalejší je vykonávání programu. Plánování přiděleného času je hierarchické[22] (hráč -> loď).

Provádění příkazů způsobuje opotřebení lodi. Míra opotřebení na příkaz mírně roste s hodnotou čítače příkazů od přistání. Loď, která dosáhne 100% opotřebení je vyřazena z plánovače a její program tak zastaven. Účelem mechaniky opotřebení lodi je zajistit vyřazení lodí, jejichž program je příliš náročný nebo potenciálně chybný.[23]

[21] Velikost je zvolena s přihlédnutím na uložení v databázi jako varchar o 255 znacích. Na základě konkrétní implementace lze zvolit jiné hodnoty, avšak počet řádků by neměl překročit 20, aby byl hráč nucen pečlivě vážit zápis výstupu a to v důsledku vedlo k menšímu zatížení na straně serveru.

[22] Cílem je zajištění spravedlivého dělení času mezi hráče tak, aby hráči s velkým počtem lodí „neokrádali“ začínající hráče.

[23] Příkaz REPAIR funguje jako přepínač a oprava lodi je provedena až při odletu. Tak nemůže dojít k narušení účelu mechanismu opotřebení.

Jazyk Lua a Master Control Computer

Pro pokročilé hráče je k dispozici možnost pronajmout si **Master Control Computer** (MCC, jeden na hráče), speciální „počítač“, který jim umožní kontinuálně provádět program v jazyku Lua. To jim umožní manipulovat s vlastními loděmi, továrnami a sklady na centrální úrovni a provádět globální řízení svého obchodního impéria.

Na této úrovni může hráč řídit běh programů v jednotlivých lodích (nahrát a spustit program, přerušit jeho běh, znovu spustit, restartovat) a reagovat na události (chyba v programu lodi, loď přistála atd.). Provádění některých příkazů uměle zpomaleno, aby byli hráči nuceni psát efektivní programy.

Jazyk Lua je univerzální rozšiřitelný skriptovací jazyk, který se těší široké oblibě ve světě her. Je použit v celé řadě titulů, především pro vytváření GUI. [24] Volba jazyka Lua byla provedena na základě zkušeností a nelze jí brát jako konečnou. Hra může být postavena na jiném skriptovacím jazyku, nebo jich dokonce může podporovat více.

Vliv použití programování na ekonomickou situaci hráče

Obecně by programování ve hře nemělo mít zásadní ekonomický dopad na hráče, aby motivace k využití tohoto herního prvku byla založena na zvědavosti a snaze o sebezdokonalení. Všechna rizika spojená s použitím automatizace obchodní činnosti je třeba pečlivě hodnotit a poskytnout hráči nástroje pro jejich kontrolu.

Základním nástrojem je možnost nastavit limity na automatické transakce a minimální úroveň konta, která zajistí, že hráči zůstane dostatek finančních prostředků pro pokračování ve hře.

Speciálním případem je pak užívání MCC spojené s periodickou platbou nájemného. Důvodem zavedení poplatku je potenciální problém s dostupností potřebných kapacit pro všechny hráče. Pokud tento problém v praxi nenastane, je vhodné tento poplatek nahradit jednorázovou vysokou pořizovací cenou.

Knihovna programů a jejich sdílení

Každý hráč má k dispozici vlastní knihovnu programů, která může být omezena kapacitou úložiště. Tato knihovna má podobu seznamu programů, jejichž jména splňují běžné konvence pro pojmenovávání souborů.

Hráči mohou ve hře sdílet programy jejich publikací. Publikovaný program může být buď „open source“ zdarma, nebo můžou prodávat ostatním hráčům licence. Každá taková licence umožňuje jejímu držiteli spustit daný program na jedné své lodi. Zdrojový kód není v tomto případě vidět.

[24] Za zmínku určitě stojí hra *World of Warcraft* firmy Activision Blizzard, kde je jazyk Lua použit pro přizpůsobování GUI hráči. V tomto jazyce tak vznikají velmi komplexní modifikace a nástroje pro tuto hru.

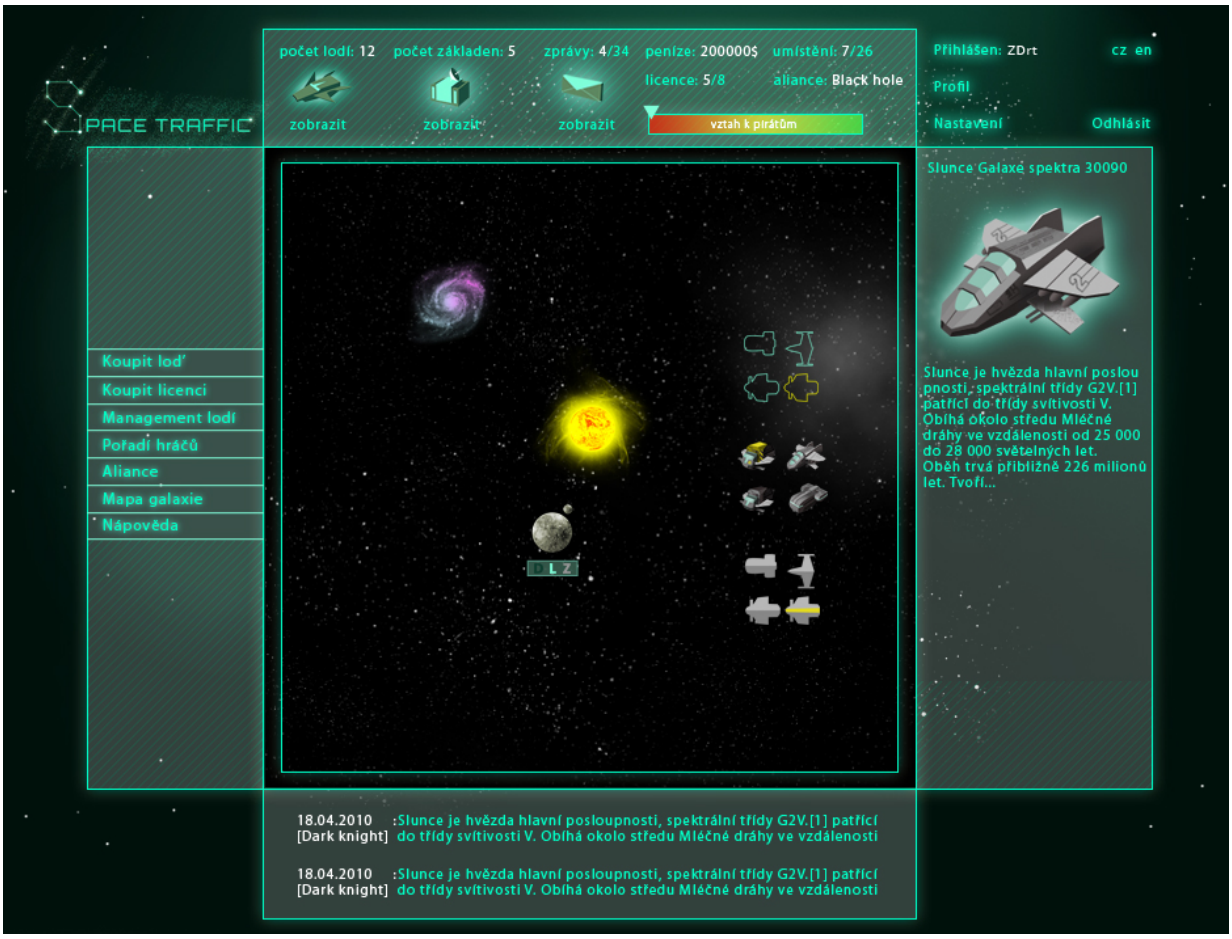
Úroveň jednotlivých programů mohou hráči hodnotit systémem dobrý/špatný. Hra je k tomu bude aktivně vyzývat.

Programy půjde také importovat a exportovat v textovém formátu.

4.7 Audiovizuální ztvárnění hry

Díky zasazení hry není její audiovizuální ztvárnění náročné. Přesto je třeba dbát na jeho kvalitu, protože je vizitkou celého projektu a vytváří klíčový první dojem.

4.7.1 Grafická podoba PHP verze a důvody proč nevyhovovala

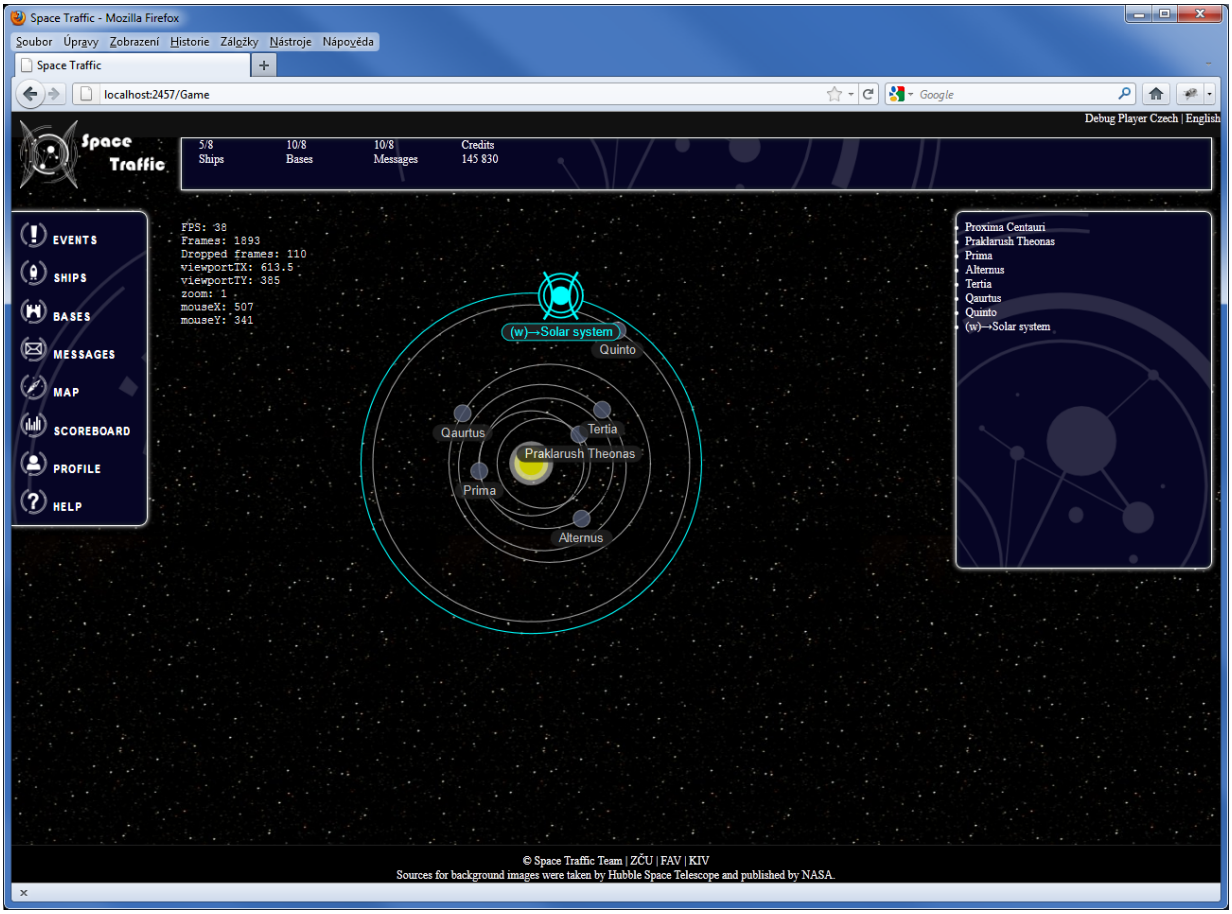


Obrázek 4.4: Ukázka GUI PHP verze z roku 2010.

Grafická podoba PHP verze byla poměrně zdařilá, jak je možné posoudit na obrázku 4.4. Komplikace v jejím nasazení byla především v nedostupnosti podkladů, bez nichž nebylo možné provádět potřebné úpravy.

Problém byl především v tom, že dostupné grafické zdroje nejsou uloženy ve formátu s oddělenými vrstvami pozadí a popředí. Pozadí je v tomto případě hvězdná plocha, která prosvítá jednotlivými grafickými prvky a pokud by došlo k manipulaci s rozložením jednotlivých elementů, výsledek by obsahoval rušivé artefakty. To se bohužel týkalo i takových prvků, jako je hlavní menu.

Z tohoto důvodu bylo nakonec rozhodnuto o jejím opuštění, i když byla vzata jako inspirace pro vytvoření nového prototypu.



Obrázek 4.5: Ukázka GUI nové verze z roku 2012.

Na obráku 4.5 je k nahlédnutí ukázka uživatelského rozhraní nové verze implementace. Současné GUI je tvořeno prototypy, které slouží především k ověření struktury uživatelského rozhraní. Jeho konečná podoba bude vyžadovat profesionální grafickou práci, kterou tým ve složení v akademickém roce 2011/2012 nemohl zajistit.

4.7.2 Stylizace hry

Pro grafické ztvárnění hry byla zvolena stylizace „vektorové grafiky“. Pro zobrazení herních objektů jsou použity jednoduché geometrické tvary, které jsou znázorněny především svým obrysem a barvou.

Obrázek 4.6: Ukázka ze hry *DEFCON* firmy Introversion Software.

Styl by měl připomínat obrazovku radaru nebo „počítačové“ znázornění bojiště, běžně k vidění v mnoha filmech. Za hlavní předlohu zde lze vzít film *Válečné hry* (*WarGames*, 1983, [16] a především hru *DEFCON* společnosti Introversion Software (screenshot ze hry *DEFCON* na obrázku 4.6).

Výhody této stylizace jsou především v její jednoduchosti, přehlednosti a srozumitelnosti. Navíc toto ztvárnění dobře zapadá do zvoleného způsobu technické realizace pomocí vektorového formátu SVG.

4.7.3 Zásady návrhu GUI

Grafické rozhraní hry by mělo splňovat následující požadavky[25]:

Intuitivnost Hráč očekává určité chování objektů a voleb ve hře. Je třeba toto očekávání naplnit a dělat věci tak, jak je u her tohoto typu obvyklé. Stejně tak je třeba vážit formu poskytovaných informací tak, aby vyhovovaly účelu, pro který je hráč chce získat.

[25] Vychází z obecně odvoditelných pravidel pro tvorbu GUI, cílem je aby výsledek co nejvíce odpovídal očekávání hráčů na základě předchozích zkušeností z ovládáním počítačů a her.

Jednoznačnost Všechny volby jsou od sebe jasně odlišitelné a jejich účel je zřejmý. Související volby jsou sdružovány do logických skupin.

Konzistence Veškeré chování UI musí být konzistentní v rámci celku. Pokud se určitý prvek UI chová na jednom místě jako odkaz, bude se tak chovat vždy a bude odkazovat na stejnou věc (např.: jméno planety nebo pozice tlačítka Ok na pravé straně dialogu vlevo od Storno).

Tři kroky Každá informace nebo volba je dostupná po maximálně třech krocích.

Více cest Pokud v daném místě existuje relevantní akce, měla by být dostupná přímo z místa. (např.: Jméno planety jako odkaz na systém kde se nachází.)

Okamžitá nápověda Nápověda k jednotlivým prvkům a objektům musí být dostupná přímo u těchto objektů (ne hledáním v manuálu).

4.7.4 Zvuková stránka hry

Jak již je uvedeno v [1], zvuková stránka hry se omezuje na zvukové efekty potvrzující hráčovy akce nebo informující o důležité aktualizaci stavu hry (např. změna výše konta hráče). Hudební doprovod není potřeba, i když by mohl kvalitní hudební doprovod pomoci v dotvoření herního prostředí. Nezbytnou funkcí je však možnost vypnout jak případný hudební doprovod, tak i zvukové efekty.

4.8 Náměty na další rozšíření game designu Space Traffic

V této podkapitole jsou popsány některé další náměty nad rámec game designu hry *Space Traffic*. Jejich implementace není pro samotné fungování hry nutná, lze jimi však zpestřit herní zážitek. Také je možné je implementovat až po dokončení a uveřejnění hry, jako dodatečné rozšíření.

4.8.1 Minihry

Do projektu můžou být včleněny minihry. Jednalo by se o hry odlišných žánrů (hlavolamy, akční hry, znalostní kvízy apod.), které by byly integrovány do hlavní hry a nabízely by odlišný herní zážitek.

Nejjednodušší způsob jejich integrace je přes mechanizmus kontraktů, je však možné přijít i na další způsoby.

Realizací by se pak mohlo jednat o samostatné aplikace, které by pomocí služeb komunikovaly s herním serverem. Lze uvažovat i o aplikacích pro mobilní platformy.

4.8.2 Hráčské korporace

Hráčská korporace je užší forma spolupráce skupiny hráčů. Umožňuje hráčům založit společnost podobnou skutečným podnikům, která jim umožňuje sdílet zdroje s cílem je zhodnotit. Vzniká tak společný majetek, který hráči kolektivně spravují.

Založením korporace do ní každý hráč vkládá svůj podíl v herní měně (všichni ve stejné výši). Tak vznikne základní jmění společnosti. Jednotliví hráči zároveň mohou společnosti věnovat část svých lodí, pozemků a budov. Ty se stávají majetkem korporace. Věnování majetku zvyšuje podíl hráče v korporaci (výchozí je rovnoměrné rozdělení podílu).

Každý hráč může za korporaci rozhodovat, nakupovat a prodávat zboží za peníze korporace, přijímat a plnit mise, kupovat lodi a továrny, apod. Všechen takto získaný majetek je vždy majetkem korporace.

Hráči se mohou shodnout na vyplácení dividend – částky, která je převedena jednotlivým hráčům z konta korporace. Výše částky je závislá na podílech jednotlivých hráčů na korporaci. O vyplacení hráči rozhodují většinovým hlasováním podle svých podílů.

Při zrušení korporace je její movitý majetek prodán a finanční částka z prodeje, doplněná o zůstatek na kontě korporace je rozdělena dle podílů mezi jednotlivé hráče.

Tento koncept lze rozšířit o možnost obchodování s podíly nebo možnost vytvoření hrou podporované organizační struktury.

4.8.3 Vzájemná komunikace lodí zasíláním zpráv

Zajímavou rozšířením jazyka Starship Basic by byla možnost vzájemné komunikace lodí zasíláním zpráv. K tomu odeslání zprávy by sloužil příkaz SEND, který by umožňoval odeslat číslo, řetězec nebo pole konkrétní lodi. Tato zpráva by byla doručena do fronty zpráv cílové lodi. Pokud by tato fronta byla plná, došlo by k chybě. Vyzvednutí zprávy by pak bylo možné příkazem RECEIVE.

Možnost komunikace by zásadním způsobem rozšířila možnosti umělé inteligence lodí, které by se takto mohly chovat jako multi-agentní systém.

5 Architektura Space Traffic a její implementace

Tématem této kapitoly je softwarová architektura implementace hry *Space Traffic*. Seznamuje čtenáře s požadavky, použitými technologiemi, celkovou architekturou a jejími jednotlivými částmi. Také poskytuje náhled do důležitých prvků implementace.

Softwarová architektura popisuje softwarový systém jako strukturu složenou z částí, jejich role a vzájemné vztahy[1]. Umožňuje tak zvládnout vytváření složitých systémů díky jejich dekompozici na menší celky.

Původní implementace hry z akademického roku 2009/2010 nebyla založena na žádné uchopitelné architektuře. K určité dekompozici zde došlo, ale výsledné komponenty architektury byly příliš těsně provázány. Navíc byla nedostatečně zdokumentována. Z tohoto důvodu bylo na začátku ak. roku 2010/2011 rozhodnuto o vytvoření nové implementace, která bude založena na existujícím frameworku pro vytváření webových aplikací (viz. diplomová práce Richarda Kocmana [7]). Bohužel k žádné relevantní implementaci nedošlo (důvody jsou opět popsány v [7]).

Proto bylo rozhodnutí o nové implementaci založené na vhodné architektuře přejato i do ak. roku 2011/2012 a bylo také rozhodnuto o změně platformy z PHP na .NET (změna je diskutována v 5.3).

Poznámka: Slovo *projekt* je v této práci mimo tuto kapitolu chápáno ve významu softwarový projekt. V této kapitole slovo *projekt* využíváno výhradně ve významu projektu Microsoft Visual Studia – struktury zdrojových souborů a dalších objektů, potřebných pro vytvoření aplikace nebo její části.

5.1 Požadavky na novou architekturu a na její implementace

Při zkoumání implementace z ak. roku 2010/2011 byly identifikovány následující nedostatky (popsáno také v [7]):

- Nedodržení vzoru Model-view-controller (MVC)[2] - jednotlivé vrstvy vzájemně „prorostly“).

[1] Zdroj – Paul Clements, *Documenting Software Architecture* [17]

[2] MVC – Model-view-controller; architektonický vzor oddělující datový model, uživatelské rozhraní a řídící logiku.

- Špatná soudržnost tříd – nekonzistentní zpracování požadavků, které byly vyhodnocovány na několika úrovních, bez zjevných pravidel, která by umožňovala vysledovat konkrétní funkčnost.
- CSS vložené přímo do kódu pomocí atributů *style*.
- Zcela špatná implementace View vrstvy – používání příkazu echo místo html kódu s vloženými hodnotami na konkrétních místech.
- Nekonzistence v názvech (funkcí, proměnných, tříd).
- JavaScript implementace nevyužívala objektově orientované programování. Jednalo se o funkcionální program s množstvím různě zřetězených zpětných volání. Veškerý kód byl navíc v jediném zdrojovém souboru s minimálním množstvím komentářů.

Nová architektura je opět založena na MVC. Dále bylo na základě schůzky se zástupci KIV, Petra Vogla a autora práce v květnu 2011 stanoveny tyto obecné požadavky:

- použití ověřených a standardizovaných technologií;
- skutečné oddělení jednotlivých komponent architektury;
- vhodná kombinace technologií, jejichž vzájemná spolupráce je vyzkoušená;
- věnovat zvýšenou pozornost dokumentaci;
- zahrnout potenciál pro další rozšíření;
- automatické testování;

Tyto obecné požadavky byly dále doplněny o požadavek na oddělení definice hry a jejích parametrů od programového kódu (realizují konfigurační soubory a podpora skriptů).

Všechny uvedené požadavky byly zohledněny při návrhu dále popisované architektury.

5.2 Užité technologie

Projekt je založen na následujících technologiích:

- Microsoft .NET Framework 4,
- Microsoft ASP.NET MVC 3,
- Microsoft ADO.NET Entity Framework,

- Microsoft Windows Communication Foundation (WCF),
- Microsoft SQL Server 2008
- NLog,
- LESS, dotless,
- HTML5, CSS3, JavaScript,
- jQuery, jQuery UI, jQuery SVG,
- AJAX, JSON,
- Mootools.Class,
- XML, XSD.

Microsoft .NET Framework 4[3] je aplikační vývojová platforma firmy Microsoft, poskytující nástroje pro vytváření, nasazení a běh aplikací pro desktopy, web a mobilní zařízení. Skládá se ze dvou základních částí: Common Language Runtime (CLR) , zajišťující běh .NET aplikací a knihovny tříd, poskytující běžně používané programové prostředky.

Microsoft ASP.NET MVC 3[4] je volitelná část webové platformy ASP.NET, která poskytuje elementy a nástroje pro vývoj aplikací dle architektury MVC. Jejími přednostmi jsou především podpora "convention over configuration"přístupu, podpora Test-driven development (TDD) a bezproblémová integrace s ostatními technologiemi firmy Microsoft.

Microsoft ADO.NET Entity Framework[5] (použitá verze 4.3) představuje rozšíření vrstvy pro přístup do datových zdrojů ADO.NET, založené na principu objektově-relačního mapování (ORM – Object Relational Mapping). Tato technologie poskytuje mezivrstvu mezi persistencí objektů a jejich fyzickým uložením. Umožňuje toto mapování měnit bez nutnosti změny aplikace. Také umožňuje využívat více mapování pro různá datová úložiště. V neposlední řadě podporuje Language-integrated Query (LINQ).

Microsoft Windows Communication Foundation (WCF)[6] je komplexní API a běhové prostředí pro vytváření servisně orientovaných aplikací na bázi SOA (service-oriented architecture). Služby založené na WCF jsou nezávislé na komunikačním protokolu, který je určen konfigurací publikované služby. Formát zpráv pak využívá ve většině případů standartu SOAP, podporovány jsou ale i další formáty (XML, RSS, JSON). [7]

Microsoft SQL Server 2008 je relační databázový systém firmy Microsoft, nabízející obvyklou funkcionalitu databázových systémů. Důvod k jeho použití je především ve snadné integraci s ostatními technologiemi firmy Microsoft.

[3] Web .NET – `http://www.microsoft.com/net`

[4] Web ASP.NET MVC 3 – `http://www.asp.net/mvc/overview`

[5] Dokumentace k Entity Framework - `http://msdn.microsoft.com/en-us/library/bb399567.aspx`

[6] Dokumentace k WCF - `http://msdn.microsoft.com/en-us/library/dd560536.aspx`

[7] Další informace viz. Scott Klein, *Professional WCF programming: .NET development with the Windows Communication Foundation* [18].

Více informací o všech technologiích firmy Microsoft lze nalézt na webu *Microsoft Developer Network* (MSDN – `http://msdn.microsoft.com`).

NLog[8] je logovací knihovna pro .NET, podobná známému log4j.

LESS[9] je dynamický jazyk pro vytváření šablon stylů, rozšiřující jazyk CSS o proměnné, mixiny (třída společných vlastností, určená pro využití v ostatních třídách, bez samostatného použití), operace a funkce. Dále nabízí alternativní strukturování CSS deklarací pomocí hierarchické struktury. Knihovna **dotless**[10] je implementací překladače LESS v C#, sloužící pro dynamický překlad na straně serveru za chodu webové aplikace. Způsob použití LESS v projektu je popsán v 5.5.2.

HTML5 spolu s **CSS3** je budoucí standard webových technologií. Tento rodící se standard zároveň rozšiřuje **JavaScript** o mnoho prvků pro lepší podporu webových aplikací. Využití těchto technologií v projektu je popsáno dále v 5.5.1.

jQuery[11] je JavaScript knihovna poskytující funkce pro snadnou manipulaci s HTML pomocí JavaScriptu. Projekt zároveň využívá některá její rozšíření, především **jQuery UI** (sada aktivních ovládacích prvků) a jQuery SVG[12] (doplnění jQuery pro manipulaci se SVG).

AJAX (Asynchronous JavaScript and XML) je obecná technologie pro výměnu dat mezi webovým serverem a klientem bez načítání nové stránky v prohlížeči. **JSON** (JavaScript Object Notation) je datový formát, inspirovaný syntaxí deklarace objektů v JavaScriptu.

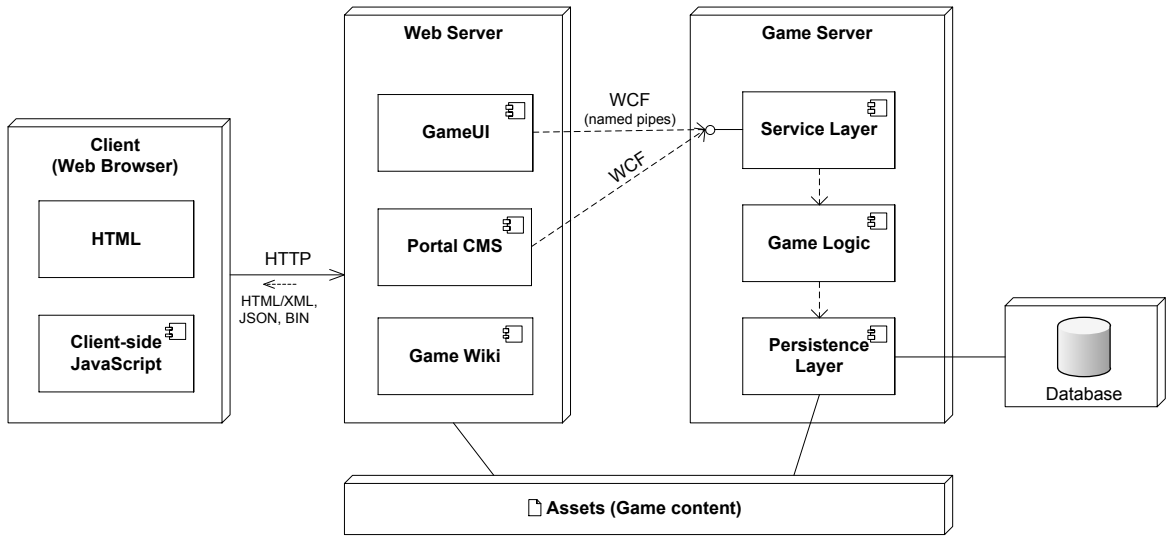
MooTools[13] (My Object-Oriented Tools) je javascriptová knihovna nabízející podobnou funkcionalitu jako jQuery. V projektu je použita jen její část MooTools.Class, která poskytuje funkce pro vytváření „tříd“.

XML (Extensible Markup Language)[14] je standardizovaný obecný značkovací jazyk, vyvinutý konsorciem W3C, využívaný jako univerzální formát pro publikování a výměnu dat. Jazyk je přímo podporován .NET Frameworkem. **XSD** (XML Schema Definition) je jazyk pro popis struktury konkrétního uživatelského XML jazyka. Je využíván pro validaci dokumentů.

5.3 Celkový pohled na architekturu

Architektura hry vychází z architektonického vzoru MVC. Celkový pohled na systém je znázorněn na obrázku 5.1. Systém je rozdělen do pěti základních modulů:

[8] Web projektu NLog – `http://nlog-project.org/`
[9] Web LESS – `http://lesscss.org/`
[10] Web dotless – `http://www.dotlesscss.org/`
[11] Web jQuery – `http://jquery.com/`
[12] Autorem jQuery SVG je Keith Wood – `http://keith-wood.name/svg.html`
[13] Web Mootools – `http://mootools.net/`
[14] XML na webu W3C - `http://www.w3.org/XML/`



Obrázek 5.1: Diagram architektury aplikace Space Traffic.

- klient (na obrázku Client),
- webový server (Web Server),
- herní server (Game Server),
- databáze,
- herní obsah (Assets).

Klient je klientská část webové aplikace běžící v internetovém prohlížeči, využívající HTML5 a JavaScript. Komunikace mezi klientem a webovým serverem probíhá pomocí protokolu HTTP (z anglického Hypertext Transfer Protocol) s využitím datových formátů HTML, XML, JSON a dalších formátů pro předávání obsahu (obrázky, mediální soubory, apod.; na obrázku označeno jako BIN). Pro aktualizaci webových stránek je využívána technologie AJAX.

Na **webovém serveru** se počítá s nasazení tří nezávislých webových aplikací, které představují front-end hry. Jsou to:

- **Uživatelské rozhraní hry** (GameUI - dále používáno toto označení) je ASP.NET webová aplikace postavená na framworku ASP.NET MVC 3, slouží pro sestavování uživatelského rozhraní hry a zprostředkování komunikace mezi herním serverem a klientem.
- **Herní portál** (Portal CMS) je aplikace typu Content Management System (CMS; Systém pro zprávu obsahu, též označován jako redakční nebo publikační systém). Jejím účelem je zajištění webové prezentace hry a projektu jako takového.

- **Herní wiki** (Game Wiki), sloužící jako nápověda ve hře.

Všechny tyto tři aplikace jsou integrovány převážně na úrovni uživatelského rozhraní.

Herní server představuje jádro celého systému hry. Tento modul je složen ze tří komponent uspořádaných do tří vrstev. První je **vrstva služeb** (Service Layer) obsahuje služby publikované herním serverem, sloužící ke komunikaci mezi uživatelským rozhraním a hrou. Komunikace probíhá pomocí technologie WCF, umožňující publikovat služby pomocí různých protokolů prostřednictvím různých komunikačních kanálů.[15] Tato vrstva zprostředkovává interakci s **herní logikou** (Game Logic), která realizuje simulaci herního světa. Poslední vrstvou je **persistenční vrstva** (Persistence Layer), která jak již název napovídá, zprostředkovává persistenci herních objektů. Přístup do herní databáze je realizován pomocí ADO.NET Entity Frameworku na bázi návrhového vzoru Data Access Objects (DAO).

Databáze a **herní obsah** (Assets) jsou posledními moduly architektury. Databáze uchovává data nutná k obnovení stavu hry při výpadku a také data, která nejsou nutná pro fungování modelu herního světa a lze se na ně odkazovat až v případě potřeby. Herní obsah je úložiště statických datových souborů obsahujících konfigurace herního modelu, doplňkové texty, audiovizuální zdroje a další digitální zdroje, které mohou být předány klientovi přímo a lze u nich využít mechanismus cachování obsahu.

5.3.1 Odůvodnění změny technologie z PHP na .NET a C#

Původní projekt počítal s implementací v PHP na základě analýzy [[bp-minar]]} a [1]. Od tohoto záměru bylo v akademickém roce 2011/2012 upuštěno ve prospěch technologie .NET a jazyka C#. Důvody pro opuštění PHP byly především tyto:

- Znalost jazyka PHP není mezi studenty běžná.
- Práce v jazyce PHP vyžaduje odlišný přístup než práce v jazyce Java, který je na FAV základem výuky programování.
- PHP nelze použít pro vytváření doplňkových nástrojů potřebných pro vývoj hry.
- Pro vytvoření nezávisle běžící simulace herního světa je třeba periodicky spouštět skript, toto řešení není ideální.
- Implementace převzatá z akademického roku 2010/2011 nebyla rozsáhlá a přechodem na novou technologii neznamenal zásadní zdržení.

[15] Nasazení na jednom fyzickém serveru předpokládá komunikaci prostřednictvím pojmenovaných rour (named pipes), které dosahují vyššího výkonu než lokální TCP/IP komunikace.

Diskuze volby .NET a C# oproti jazyku Java (Servlety+JSP, případně Spring MVC)

Na základě již zmíněných předchozích analýz technologií bylo nakonec rozhodnuto o použití prostředí .NET a jazyka C#. Vedly k tomu následující úvahy:

- Jazyky C# a Java jsou si velmi podobné.
- .NET je komplexní platforma, která nabízí prostředky pro vývoj většiny typů aplikací. Většina potřebných technologií je dodávána přímo Microsoftem s konzistentní úrovní kvality.
- Prostředí Javy není homogenní, výběr vhodného frameworku je komplikované rozhodnutí, zejména bez dostatečných zkušeností s jednotlivými technologiemi.
- Microsoft poskytuje rozsáhlou dokumentaci ke všem poskytovaným technologiím v podobě již zmíněného MSDN.

5.3.2 Důvody pro oddělení simulace hry od webové aplikace

Přestože by bylo možné spustit vlákno starající se o běh herní simulace přímo z webové aplikace, byla tato varianta řešení zavrhnuta z následujících důvodů:

- Aplikační doména webová aplikace je spravována IIS serverem. Je spuštěna při požadavku a může být ukončena dle potřeby IIS;
- Oddělení kontinuálně běžící úlohy od webové aplikace je běžně doporučovaná praktika;
- Oddělení umožňuje nezávislé testování obou modulů;

Hlavní výhodou tohoto řešení je škálovatelnost, dosažená možností spuštění obou aplikací na různých serverech.[16]

5.3.3 Struktura projektu a nasazení aplikace

Projekt je na nejvyšší úrovni uložen jako solution[17] SpaceTraffic Visual Studia 2010 (dále jen VS) – soubor SpaceTraffic.sln. Ten obsahuje následující projekty:

Core – knihovna společných tříd, využívaných ostatními projekty.

Core.Tests – unit testy projektu Core

[16] Prakticky se nepředpokládá tak velký zájem o hru, že by to bylo nutné.

[17] Anglický název pro skupinu projektů Visual Studia. Sdružuje jednotlivé části aplikace do jednoho celku pro jejich lepší organizaci. Termín nebude v práci pro zachování jednoznačnosti překládán.

GameServer – aplikace herního serveru.

GameServer.Tests – unit testy projektu GameServer

GameServerScripts – projekt obsahující skripty, které jsou součástí herního serveru, překládají se dynamicky při startu serveru.

GameUi – klientská webová aplikace

GameUi.Tests – unit testy projektu GameUi

StarSystemEditor – editor hvězdných systémů.

Kromě těchto projektů je součástí solution ještě adresář `Assets` s herním obsahem, adresář `scripts`, obsahující skripty používané při překladu a manipulaci s projektem a adresář packages, obsahující distribuce sdílených knihoven.

Celý solution je uložen v systému pro správu verzí Subversion (SVN).[18]

5.3.4 Podrobnosti k překladu projektů

Projekty jsou překládány automaticky, je zde však několik specifických úprav procesu překladu.

- Projekty GameServer a GameServer.Tests využívají databázi, proto při vývoji vyžadují connection string pro připojení k lokální databázi vývojáře. Aby mohl mít každý vývojář vlastní konfiguraci, byl vytvořen mechanismus, který zajistí využití lokální konfigurace v případě její existence.

  Tato lokální konfigurace je uložena v podadresáři `.config` příslušného projektu. Pomocí Post-Build Event je volán skript `\scripts\copyconfig.bat`. Ten zajistí zkopírování příslušného konfiguračního souboru z adresáře `.config` do cílového adresáře překladu. Pokud lokální konfigurace neexistuje, je využita výchozí s příponou `.default` v adresáři projektu.

- Projekt GameServerScripts je překládán, jeho výstup však není součástí výstupního adrešáře. Místo toho je prostřednictvím Post-Build Event zkopírován zdrojový kód do podadresáře `scripts` cílového adresáře.

  Skripty jsou překládány dynamicky při startu serveru. Zde je třeba poznamenat, že překlad skriptů sice může odhalit programové chyby, ale dynamický překlad nemusí nutně importovat všechna assembly, která jsou importována v projektu. Skutečné importy jsou dány konfigurací serveru.

[18] Umístěno na svn+ssh://students.kiv.zcu.cz/home/subversion/diplomka/spacetraffic - přístup není veřejný.

5.3.5 Nasazení aplikace

Nasazení aplikace vychází z diagramu na obrázku 5.1. Základní varianta počítá s následujícím provedení:

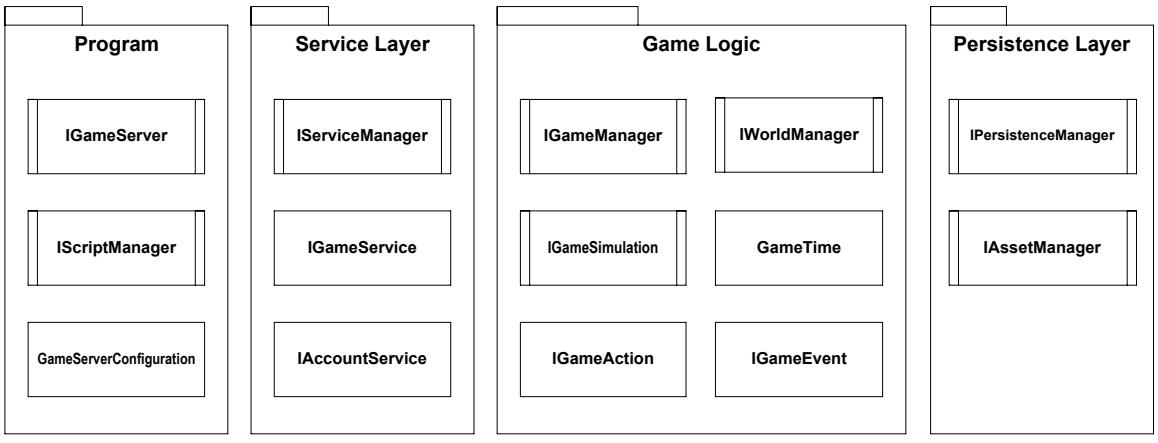
- Web Server a Game Server jsou na jediném fyzickém stroji (OS MS Windows 2008 Server, IIS 7);
- Assets - herní obsah - je uložen na témž serveru a zpřístupněn pomocí protokolu HTTP;
- Databáze běží na nezávislém serveru s instalací MS SQL Server 2008.

Dalšími variantami je možnost oddělit jednotlivé webové aplikace na fyzicky samostatné webové servery. Dále je možné oddělit i Game UI a Game Server na dva fyzické stroje.

Nasazení webových aplikací včetně Game UI je prováděno běžnou cestou. V případě Game UI je nutné správně nastavit konfigurační soubor.

Game Server je samostatná konzolová aplikace, běžící nepřetržitě. Po nastavení konfiguračního souboru je třeba zajistit, že tato aplikace bude spuštěna po startu serveru. Dále je nutné zajistit povolení komunikace na příslušných portech v použitém firewallu, pokud je použita komunikace po síti.

5.4 Herní server (Game Server)



Obrázek 5.2: Game Server - diagram objektového modelu.

Herní server (Game Server) je aplikace realizující simulaci herního světa. Dále zprostředkovává persistenci herního světa a komunikaci hráčů. Jak již bylo zmíněno v 5.3, je tato

aplikace složena ze tří vrstev: vrstvy služeb, herní logiky a persistenční vrsty. Kromě těchto tří vrstev obsahuje implementace ještě programovou vrstvu, která plní funkce spojené s fyzickým během aplikace. Na obrázku 5.2 jsou znázorněny nejdůležitější třídy a rozhraní objektového modelu herního serveru.

Základem návrhu herní server reprezentovaný rozhraním `IGameServer`, které slouží jako vstupní bod ke všem objektům relevantním pro běh hry. Dále je zde sada správců:

- **správce skriptů** (`IScriptManager`),
- **správce služeb** (`IServiceManager`),
- **správce herního světa** (`IWorldManager`),
- **správce hry** (`IGameManager`),
- **správce herního obsahu** (`IAssetManager`),
- **správce persistence** (`IPersistenceManager`).

Role a funkce jednotlivých správců bude podrobně popsána dále v této podkapitole.

Kromě těchto základních rozhraní jsou pro architekturu herního serveru klíčové ještě následující třídy a rozhraní:

- třída `GameTime`, sloužící jako reprezentace herního času;
- třída `GameServerConfiguration`, poskytující přístup ke konfiguraci herního serveru;
- rozhraní `IGameAction`, reprezentující akci vykonatelnou hráčem;
- rozhraní `IGameEvent`, reprezentující herní událost;
- rozhraní `IGameSimulation`, reprezentující simulaci vykonávanou jako součást hry.

Poslední tři uvedená rozhraní jsou podrobněji rozebrány dále v této podkapitole.

Výčet prvků uzavírají dvě služby: **služba pro správu hráčských účtů** (`IAccountService`) a samotná **služba hry** (`IGameService`).

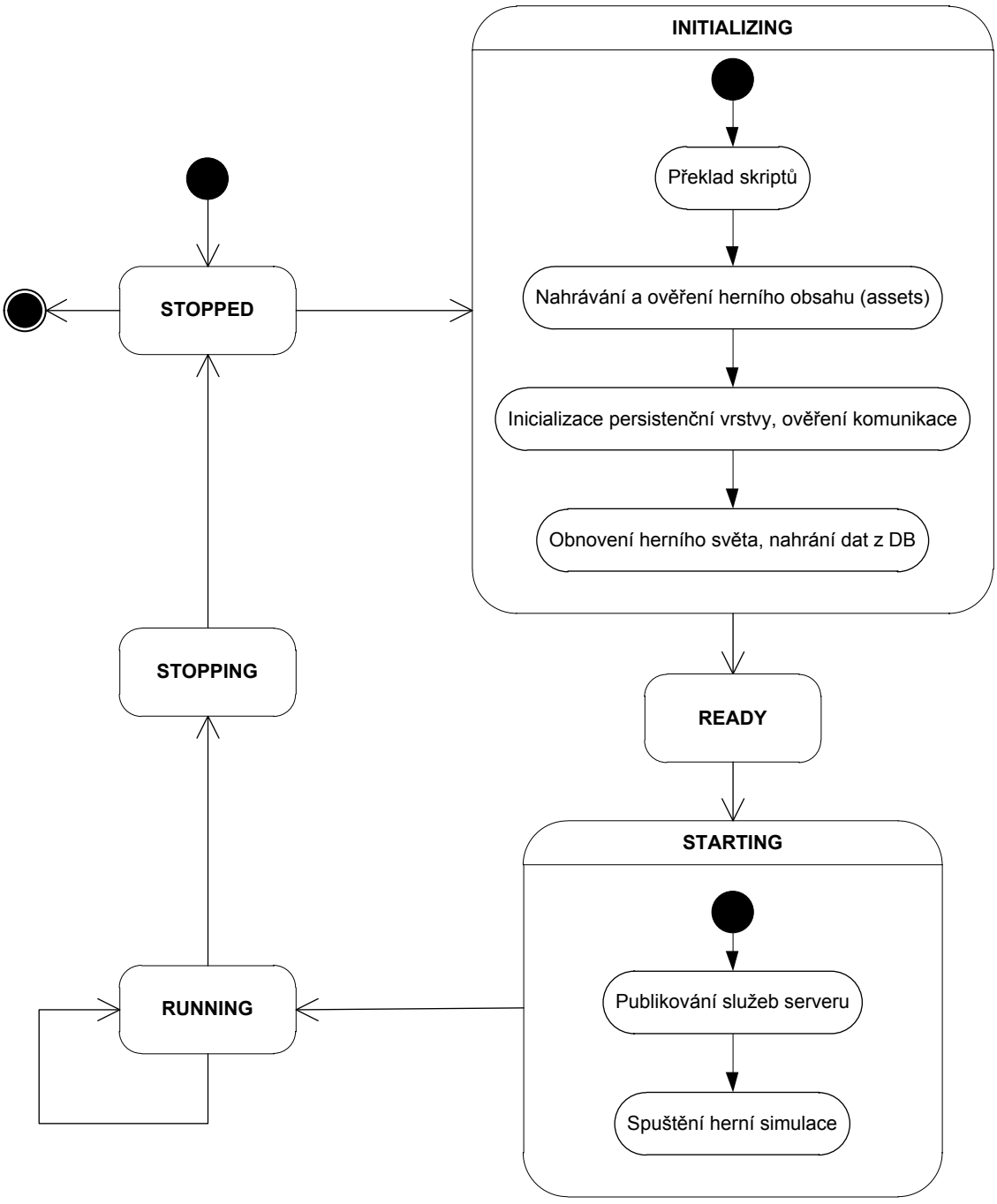
Následuje popis jednotlivých vrstev herního serveru a důležité prvky jejich implementace.

5.4.1 Programová vrstva

Programová vrstva je tvořena implementací rozhraní `IGameServer`. Instance této implementace řídí životní cyklus herního serveru, který je znázorněný na obrázku 5.3.

Server se může nacházet v jednom z 6 stavů:

- zastavený (STOPPED),



Obrázek 5.3: Game Server - diagram životního cyklu.

- probíhá inicializace (INITIALIZING),
- připraven (READY),
- startuje (STARTING),

- běžící (RUNNING),
- zastavuje (STOPPING.

Význam jednotlivých stavů je zřejmý, za zmínku stojí pouze stavy INITIALIZING a STARTING.

Inicializace herního serveru

Při inicializaci serveru jsou vykonávány čtyři důležité operace. První je překlad skriptů zajišťovaná **správcem skriptů** (`IScriptManager`), který přeloží všechny zdrojové soubory z podadresáře `scripts` pomocí C# překladače. Výstupem překladu je knihovna `SpaceTraffic.Scripts.dll`, která je po překladu nahrána do aplikační domény serveru. Tím je zajištěna dostupnost skriptů v průběhu zbytku inicializace.

Druhou operací je nahrání a kontrola herního obsahu, prováděná **správcem herního obsahu** (`IAssetManager`). Přestože tato kontrola prodlužuje inicializaci a některá ověřovaná data nejsou serverem používána, je její provedení nezbytné pro vyloučení chyb vzniklých sestavováním aplikace do distribuční podoby.

Inicializace persistenční vrstvy je třetí operací prováděnou při inicializaci. Zajišťuje jí **správce persistence** (IPersistenceManager). V jejím průběhu je v závislosti na konfiguraci serveru vytvořena databáze a ověřena komunikace s ní.

Poslední operací inicializace je obnovení světa, tj. vytvoření struktury objektů pro jeho reprezentaci a načtení jejich stavů z databáze. O tuto činnost se starají **správce herního světa** (IWorldManager) a **správce hry** (IGameManager).

Start herního serveru

Startem herního serveru se rozumí spuštění jeho hlavního vlákna. To provede nejprve publikování služeb herního serveru pomocí správce služeb a poté je spuštěn cyklus aktualizaci herního světa. Tím je spuštěna hra. Ukázka běhu aplikace herního serveru je k dispozici na obrázku D.1 v příloze D.

5.4.2 Servisní vrstva (Service Layer)

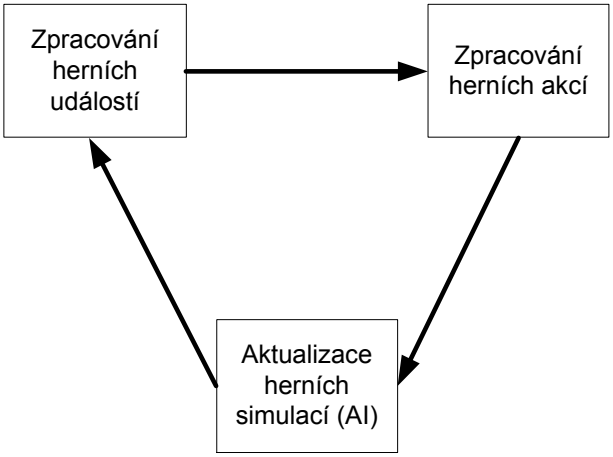
Servisní vrstva je řízena implementací rozhraní IServiceManager, zajišťující řízení životního cyklu hostitelů[19] jednotlivých publikovaných WCF služeb. Samotné služby jsou implementovány na základě kontraktů WCF.[20] Model počítá se dvěma základními službami, AccountService pro správu hráčských účtů a GameService pro ovládání hry. Jejich kontrakty jsou sdíleny prostřednictvím knihovny Core s webovou aplikací GameUi.

[19] `System.ServiceModel.ServiceHost`, dokumentace - `http://msdn.microsoft.com/en-us/library/system.servicemodel.servicehost.aspx`

[20] Kontrakt služby - rozhraní definující metody služby, viz. *Professional WCF programming* [18].

5.4.3 Herní logika (Game Logic)

Jak již bylo zmíněno výše, v hlavním vlákně herního serveru je vykonáván cyklus, starající se o aktualizaci herního světa. Tuto aktualizaci provádí IGameManager. Proces je znázorněn na obrázku 5.4.



Obrázek 5.4: Game Logic - cylus aktualizace herního světa.

Pro pochopení tohoto cyklu je nutno nejprve vysvětlit pojmy **herní akce**, **herní událost** a **herní simulace**.

Herní akce

Herní akce (`IGameAction`) reprezentuje hráčovu akci. Lze ji chápat jako workflow, které je odstartováno na příkaz hráče (nebo NPC) a jeho výsledkem je persistentní změna herního světa. Tyto akce jsou řazeny do fronty a spouštěny sekvenčně. *Výsledek* akce je obecný objekt, na který se lze programově dotázat. Slouží k aktualizaci uživatelského rozhraní.

Akce se může nacházet ve čtyřech stavech:

- připravena (PREPARED) – akce je připravena k prvnímu spuštění;
- naplánována (PLANNED) – akce byla již alespoň jednou spuštěna, je naplánována k dalšímu spuštění pomocí herní události;
- dokončená (FINISHED) – akce skončila úspěšně, *výsledek* je k dispozici pro vyzvednutí;
- selhání (FAILED) – akce selhala, nedošlo k žádné změně herního světa.

Jak je vidět, akce může být spouštěna opakovaně. Musí pro ní však platit následující pravidla:

1. Úspěšné dokončení akce vždy mění herní svět;
2. Selhání akce nemění herní svět;
3. Akce je konečná (plánování jejího spouštění neprobíhá donekonečna);
4. Průběh akce je persistentní (lze jej obnovit při restartu serveru).

Příkladem akce je let lodi z jedné základny na druhou (FLY TO, viz. 4.6.12):

- První spuštění akce provede naplánování letu dle zadané cesty (sekvence hvězdných systémů, kterými je třeba proletět na cestě k cíli) a loď je vyjmuta ze seznamu lodí na příslušné základně a přidána do seznamu letících lodí odpovídajícího systému. Dále jsou uloženy údaje o cestě a aktuální poloze lodi do databáze. Tato akce je pak naplánována pro další spuštění v čase, kdy loď dorazí k prvnímu koncovému bodu červí díry, kterým má proletět do dalšího systému.
- Po uplynutí daného množství času je tato akce opět vyvolána pomocí naplánované události. Její další spuštění provede přesunutí lodi do následujícího hvězdného systému, zapíše aktualizaci stavu do databáze a provede naplánování akce v čase dalšího průletu červí dírou.
- Toto opakované plánování probíhá až do okamžiku příletu do cíle, kdy je akce úspěšně ukončena.

Sekvenční zpracování akcí má za cíl zajistit konzistenci ve změnách herního světa bez nutnosti složitých synchronizačních strategií a využívání transakcí.

Herní události

Herní událost je objekt nesoucí informaci o herní akci, která má být spuštěna v některém určitém budoucím čase. To je potřeba pro veškeré změny, které mají být vykonány se zpožděním (především let lodi, ale také reakce na herní ekonomiku, vypršení aukcí apod.).

Herní simulace

Herní simulace je objekt, který provádí určitou podmnožinu operací nad herním světem při každé aktualizaci herního světa. Jednotlivé simulace řeší úlohy jako rozhodování umělé inteligence nebo spouštění hráčských programů. Provádějí také automatické operace herního systému, které mají reagovat na aktuální stav (např. sklad, který má nakoupit zboží při poklesu ceny) a periodické změny jako "výroba"nových *jednotek* zboží.

Reprezentace herního světa

O reprezentaci herního světa se stará správce herního světa `IWorldManager`. Jeho implementace obsahuje všechny datové struktury popisující mapu hvězdných systémů a herní objekty v ní, aktivní hráče a lodě, továrny, základny atd. Dále pak obsahuje všechny operace, které je nad těmito objekty možné provádět.

Důvodem pro implementaci obslužných metod na úrovni manažera místo v konkrétních objektech je snaha zajistit jednotnou úroveň pro řešení synchronizace. Přestože zpracování herních událostí a akcí je sériové, čtení dat probíhá paralelně díky povaze zpracování služeb WCF. Navíc je nutné počítat s budoucí optimalizací zpracování herní logiky pro běh ve více vláknech.

Persistenční vrstrva (Persistence Layer)

Persistenční vrstva je tvořena správcem persistence `IPersistenceManager`, sloužící jako továrna DAO objektů pro přístup k databázi a správce herního obsahu IAssetManager, který poskytuje funkce spojené s načítáním datových souborů. Formáty dat herního obsahu jsou popsány dále v podkapitole 5.7.

Databázový model je vytvořen dle objektového modelu metodikou Code First.[21] Konfiguraci modelu zajišťuje třída `SpaceTraffic.Persistence.SpaceTrafficContext`.

Implementace persistenční vrstvy byla provedena studenty v rámci předmětu KIV/ASWI.[22] Dokumentaci lze najít v projektové wiki[23].

Databázi je možné vytvořit automaticky nastavením hodnoty CreateDatabaseIfNotExists atributu type elementu initializer v sekci gameServerConfig konfiguračního souboru herního serveru.

Další možné volby jsou:

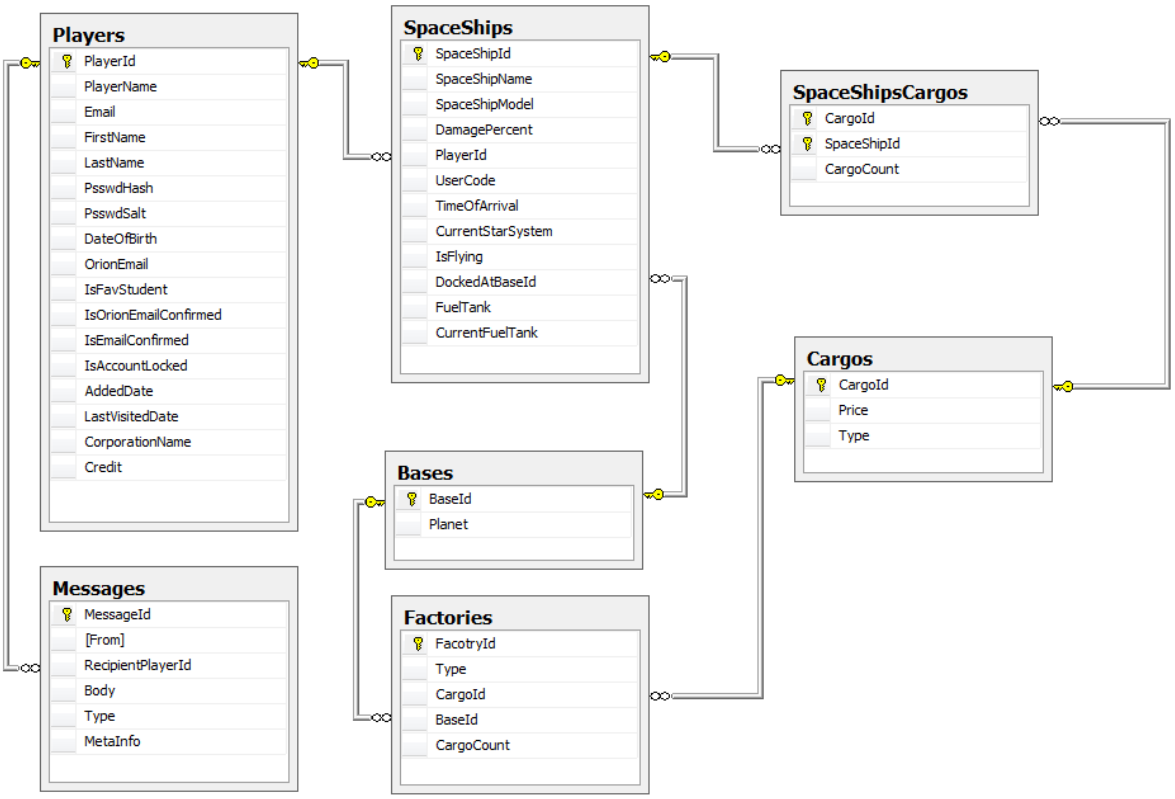
- DropCreateDatabaseIfModelChanges – při změně modelu dojde ke smazání databáze (i s daty!) a opětovnému vytvoření prázdné;
- DropCreateDatabaseAlways – databázové schéma bude vždy vymazáno a znovu vytvořeno.

V době psaní této práce byl k dispozici pouze provizorní databázový model, určený především pro testování mechanismu persistence dat, jeho schéma je k dispozici na obrázku 5.5. Jeho upřesnění a doplnění bude provedeno při další implementaci.

[21] Jedna z vývojových metodik pro Entity Framework, ve které je na základě doménových tříd automaticky vygenerováno databázový model za běhu aplikace. Více o této metodice lze nalézt v Jason Oliviera, *Tutorial: Code First with EF 4.1* [19]

[22] KIV/ASWI – Pokročilé softwarové inženýrství.

[23] Projektová wiki, *Persistence Layer* – `http://spacetraffic.kiv.zcu.cz/code/tiki-index.php?page=Persistence+Layer`



Obrázek 5.5: ERA testovací databáze.

5.4.4 Konfigurace herního serveru

Konfigurace herního serveru je v podobě standardního XML konfiguračního souboru .NET Frameworku, doplněný o vlastní konfigurační sekci. Soubor s konfigurací je nahráván automaticky. Nastavovány jsou následující parametry:

- adresář s herním obsahem (element assets),
- jméno souboru s mapou galaxie (element map),
- způsob inicializace persistenční vrstvy (element initializer), který byl popsán v předchozím bodě.

Podrobnosti o nastaveních a možných hodnotách jsou součástí dokumentace třídy `GameServerConfigurationSection`. Konfigurace také obsahuje nastavení koncových bodů služeb a připojení k databázi.

5.5 Uživatelské rozhraní hry (GameUI)

Uživatelské rozhraní hry má podobu webových stránek s využitím dynamických změn jejich obsahu. Je realizováno jako webová aplikace, založená na frameworku ASP.NET MVC 3. Tento open source framework firmy Microsoft umožňuje vývoj aplikací na bázi MVC a poskytuje robustní základ pro další vývoj.

Webová aplikace je vytvořena na základě výchozí struktury projektu. Na nejvyšší úrovni je veřejná část MVC aplikace. Na nejvyšší úrovni jsou veřejné kontrolery jako přihlašování, registrace apod. Samotná hra je umístěna v oddělené oblasti `Game` (umístění `Areas\Game`), ke které je možné přistoupit pouze po přihlášení.

Současná implementace má spíše charakter prototypu a neobsahuje žádnou zajímavou funkčnost. Skládá se převážně z jednotlivých formulářů herního uživatelského rozhraní.

5.5.1 Využití HTML5

HTML5 není zatím kompletní technologie a její podpora prohlížeči není kompletní. Přesto bylo rozhodnuto pro její použití z následujících důvodů:

- Hra bude ve vývoji několik let, za tu dobu se může podpora využitých funkcí zlepšit.
- HTML5 nabízí zásadní výhody, které mohou zjednodušit implementaci.
- Koncový uživatel – hráč – má většinou k dispozici aktualizovanou verzi prohlížeče.
- Využití HTML5 dává studentům pracujícím na projektu možnost seznámit se s novou technologií, která se v čase stane požadavkem pro mnoho pracovních příležitostí.

Jako referenční prohlížeče jsou zvoleny Firefox a Chrome. Jejich minimální verze nejsou stanoveny, očekává se vývoj na nejaktuálnějších verzích do dokončení alfa verze hry[24]. Pak budou stanovenoveny konkrétní podporované verze pro konečnou verzi hry a nutné úpravy budou provedeny jako součást beta fáze vývoje.

5.5.2 CSS a využití jazyka LESS

Použití jazyka CSS pro definování vzhledu aplikace přináší především možnost vytvářet herní rozhraní po funkční stránce a detaily jeho vzhledu měnit nezávisle na implementaci. Proto je využívání CSS v moderních webových aplikacích běžné.

Jeho jedinou nevýhodou je přehlednost v rozsáhlých aplikacích. V tom pomáhá právě jazyk LESS, který rozšiřuje CSS o řadu prvků napomáhající právě zpřehlednit a zjednodušit

[24] Alfa verze je funkčně kompletní verze hry, další vývoj se soustředí na odstraňování chyb.

zdrojový kód šablon. Zároveň je s CSS kompatibilní, lze proto využít stávající CSS kód a postupně jej přepracovat s využitím rozšíření, která LESS nabízí.

Pro použití LESS šablon je třeba zajistit jejich překlad do CSS. To je zajistit třemi způsoby:

- staticky, vhodným nástrojem;[25]
- dynamicky na straně serveru;
- dynamicky na straně klienta pomocí JavaScriptu.

V aplikaci je LESS zakomponován pomocí knihovny dotless. Ta je registrována jako HTTP handler pro soubory s příponou `.less` v konfiguračním souboru GameUi (soubor `Web.config`). Upřesňující nastavení dotless (stejnojmená sekce ve `Web.config`) umožňují caching šablon a jejich minifikaci.[26] Obě tato nastavení nejsou povolena ve vývojové verzi.

Samotné šablony jsou organizovány do sady souborů umístěných v adresáři `Content` projektu GameUi, nastavující konkrétní prvky uživatelského rozhraní. Ty jsou importovány souborem `Game.less` v adresáři `Content`, který je využíván jako finální šablona pro klienta.

Ukázka LESS kódu:

```
.wormholeEndpoint {
  .center {
    fill:white;
    stroke-width:0px;
  }
  .outer-rings {
    // Neviditelná výplň; pro události myší
    // v prostoru mezi kružnicemi
    fill: rgba(0,0,0,1);
    stroke-width:2px;
    stroke:white;
  }
  .curves {
    fill:none;
    stroke:white;
    stroke-width:3px;
  }
}
```

[25] Například SimpLESS – `http://wearekiss.com/simpless`

[26] Z anglického minification - odebrání všech nepotřebných znaků z kódu za účelem jeho zmenšení.

5.5.3 Přístup ke službám herního serveru

Pro přístup k jednotlivým službám herního serveru (viz. 5.4.2) je vytvořen mechanismus založený na sadě klientských tříd, které poskytují stejné rozhraní, jako volané služby.[27] Jmenný prostor implementace je SpaceTraffic.GameUi.GameServerClient.

Základem celého mechanismu jsou třídy:

- ServiceChannelFactory,
- ServiceClientBase,
- GameServerClientFactory.

Statická třída ServiceChannelFactory vytváří kanál pro přístup ke konkrétní službě prostřednictvím generické metody GetClientChannel. Ta pro konkrétní rozhraní kontraktu vytváří instance IClientCannel pro konkrétní službu, pro zadaný koncový bod (endpoint).

Generická třída ServiceClientBase slouží jako základ implementace pro klienta konkrétní služby (v tomto případě AccountService a GameService). Automatizuje identifikaci koncového bodu služby na základě názvu kontraktu.

Výsledkem této kombinace tříd je výrazné zjednodušení implementace konkétních klientů jednotlivých služeb, jak demonstruje následující ukázka:

```
public class GameServiceClient : ServiceClientBase<IGameService>,
  IGameService
{
  public IList<WormholeEndpointDestination> GetStarSystemConnections(
      string starSystem)
   {
      using (var channel = this.GetClientChannel())
      {
         return
            (channel as IGameService).GetStarSystemConnections(starSystem);
      }
   }
}
```

Implementace konkrétní služby zároveň implementuje její kontrakt. Jednotlivé metody jsou pak redukovány na zavolání metody služby nad vytvořeným kanálem.

Celý mechanismus zastřešuje třída GameServerClientFactory. Ta zpřístupňuje instanci objektu (rozhraní IGameServerClient) obsahujícího reference na všechny klienty služeb herního serveru.

[27] Rozhraní služeb – kontrakty – jsou sdíleny mezi `GameUi` a `GameServer` pomocí knihovny `Core`. (jmenný prostor SpaceTraffic.Services.Contracts).

Praktické využití mechanismu vypadá následujícím způsobem:

```
// Vytvoření klienta ke službám serveru

private readonly IGameServerClient GSClient =
      GameServerClientFactory.GetClientInstance();

Návratový_typ data = GSClient.Služba.Metoda(parametry_metody_služby...);
```

5.5.4 Předávání hráčských akcí serveru

Jak již bylo popsáno v 5.4.3, základem změn herního světa jsou herní akce. Pokud obsluha požadavku klienta webovou aplikací vyžaduje vykonání herní akce, je provedeno asynchronně a je odpovědností klienta, dotázat se na výsledek.

Celý proces komunikace je vidět na diagramu 5.6. V průběhu zpracování dotazu kontrolerem je zavolána metodu PerformAction služby GameService. Ta zajistí přidání akce do fronty ke zpracování. Vrací objekt, identifikující požadovanou akci pro získání výsledku a další relevantní data.[28] Na jeho základě může klient získat výsledek akce voláním RequestActionResult.[29]

5.6 Klientský JavaScript

Pro dosažení lepší úrovně interaktivity webového uživatelského rozhraní hry bylo rozhodnuto o použití JavaScriptu s novými prvky HTML5. To sebou nese následující výhody:

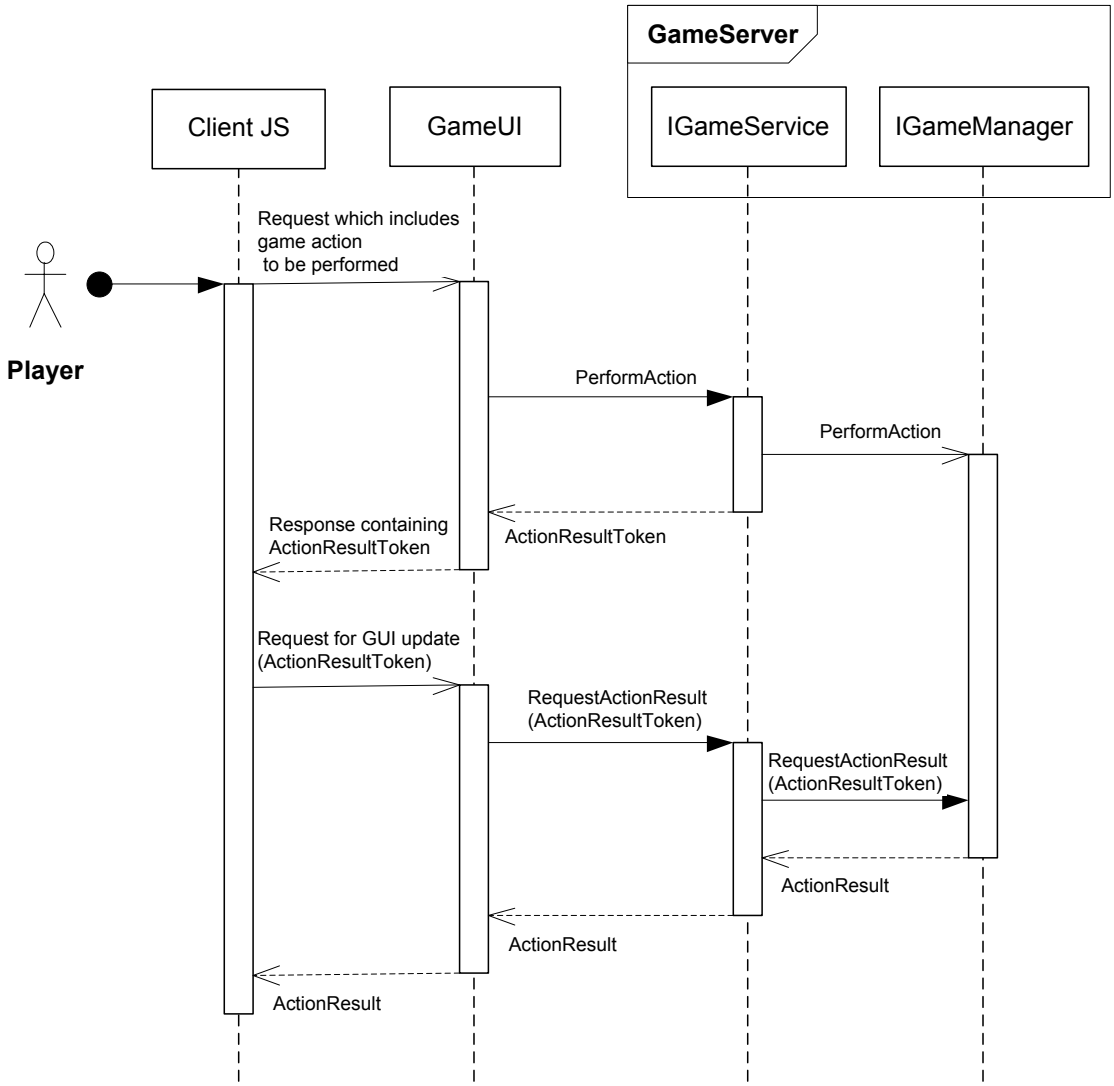
- Lepší odezvu GUI díky obnovování části stránky pomocí technologie AJAX.
- Další zlepšení odezvy asynchronním nahráváním dat na pozadí a ukládáním do vyrovnávací paměti.
- Vytvoření aktivního grafického GUI, které více připomíná klasickou hru.
- Využití různých efektů pro indikaci změn ve hře.
- Přenesení části zátěže ze serveru na klienta, které samo o sobě přináší řadu výhod[30].

Na druhou stranu přináší JavaScript i řadu nevýhod:

[28] Například odhadovaný čas dokončení akce – zatím není podporováno.

[29] Výsledek má většinou vliv na aktualizaci GUI, v mnoha případech nebude třeba. Jeho specifické využití je u modálních dialogů – například nákup zboží.

[30] Zmenšení objemu přenášených dat a počty požadavků, obojí vede k zlepšení odezvy komunikace. Také část zpracování odpadá, takže jsou požadavky vyhodnocovány rychleji.



Obrázek 5.6: Diagram předávání hráčských akcí serveru.

- Jazyk není mezi studenty příliš znám.
- JavaScript se v mnoha principech liší od jazyků vyučovaných KIV (Java, C#, C++). Pro studenty je těžší do něj proniknout.
- KIV nenabízí předmět, který by se mu dostatečně věnoval.
- JavaScript obsahuje řadu konstrukcí, jejichž špatné použití může vést k neočekávanému chování.[31]

[31] Pro ilutraci: příkaz `for(i = 0; i < ...; i++)` vypadá na první pohled v pořádku. Je v něm však

Alternativami k tomuto řešení jsou technologie Adobe Flash, Microsoft Silverlight nebo Java. Všechny tyto technologie nabízí podobné možnosti, sdílí však stejný základní problém: studenti je neznají na potřebné úrovni. Proto byl výběr technologie spíše otázkou preferencí.

5.6.1 Struktura JavaScript implementace

Zdrojový projekt GameUi obsahuje dva adresáře s JavaScriptem:

- `Scripts`, obsahující JavaScript knihovny, potřebné pro běh aplikace;
- `JS`, obsahující klientský JavaScript.

Protože psaní veškerého kódu do jednoho zdrojového souboru je špatná praktika a na druhou stranu každý JavaScript soubor musí být vložen samostatným tagem script, byl vytvořen nástroj pro vkládání JavaScriptových souborů pomocí jednoho konfiguračního souboru – `jsimport`.

Ten má podobu seznamu souborů s relativními cestami, vztahující se k jeho umístění a zajistí přidání odpovídajících script tagů v aplikaci. Tento mechanismus je možné časem nahradit minifikací zdrojových souborů do jediného.

5.6.2 Vykreslování mapy hvězdného systému

Mapa hvězdného systému je vykreslována dynamicky s využití nových technologií HTML5. Jejím základem je generování SVG přímo do HTML pomocí JavaScriptu. Aktualizace obrazu je pak zajištěna změnami konkrétních atributů nebo manipulacemi s CSS.

Mapa má 4 vrstvy:

- background – pro objekty na pozadí (např. znázornění trajektorií těles);
- object – pro grafické reprezentace objektů;
- overlay – pro popisky a grafické indikátory;
- top – pro vykreslení objektu na popředí.

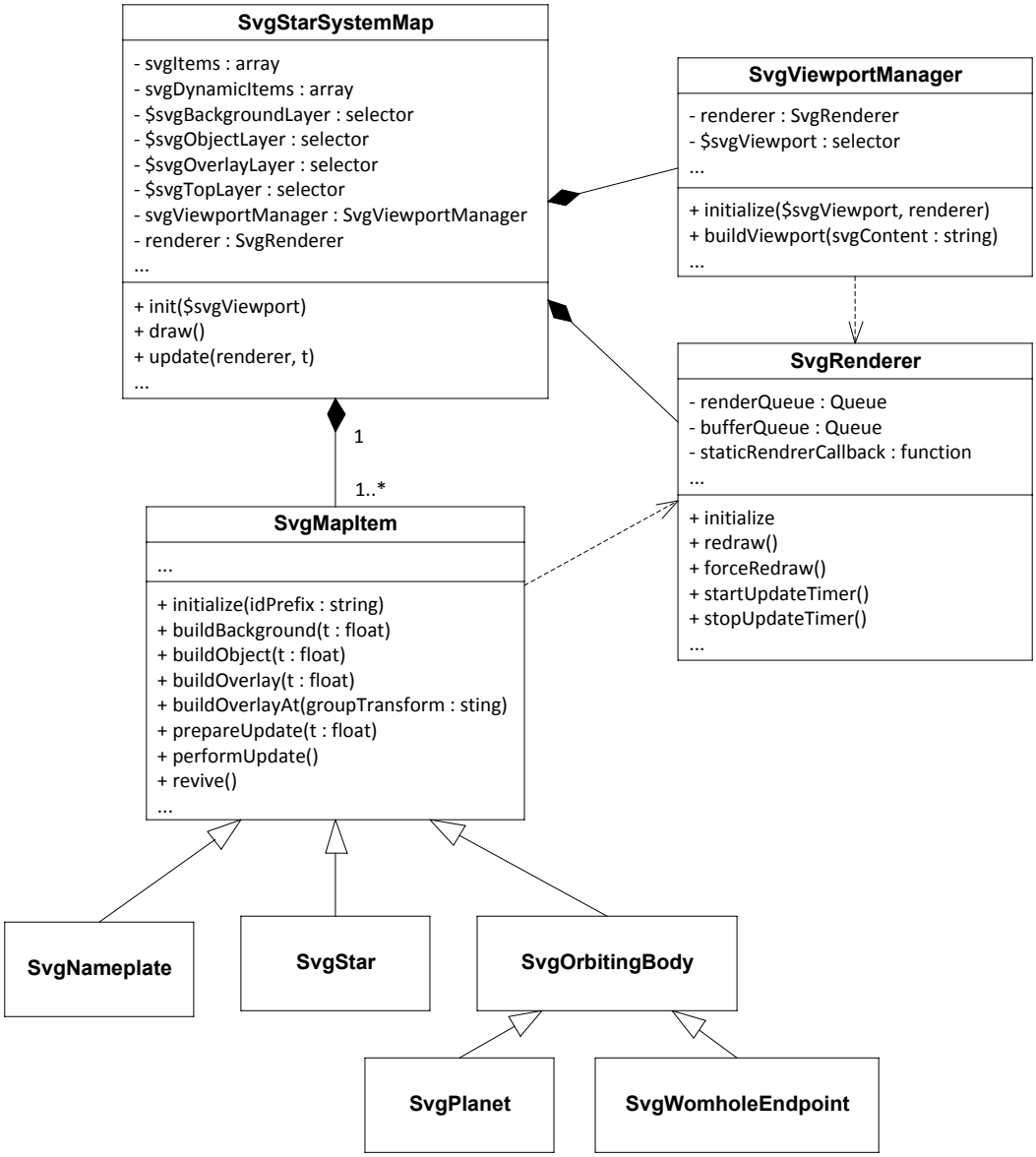
Tyto vrstvy slouží ke správnému uspořádání překrývání objektů [32].

Implementace mapy je tvořena sadou objektů [33] odpovídajícím třídám, znázorněným na obrázku 5.7.

skrytá chyba - proměnná `i` je v tomto případě globální. To může vést k neočekávanému chování programu a těžko odhalitelným chybám. Správně je `for(var i = 0; i <...; i++)`.

[32] SVG vykresluje elementy v pořadí jejich uvedení v kódu, popředí je vždy poslední zapsaný element.

[33] Objektově orientované programování v jazyce JavaScript je založeno na prototypech.



Obrázek 5.7: Diagram tříd vykreslování mapy galaxie.

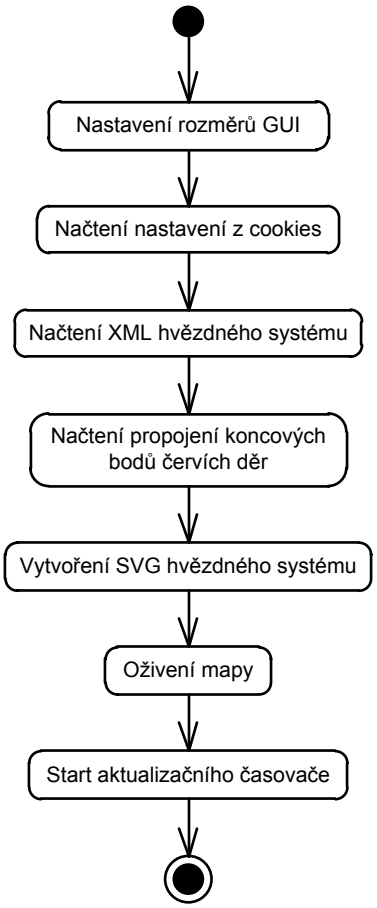
Jednotlivé vykreslované objekty jsou vytvářeny na základě prototypu odpovídajícímu SvgMapItem. Ten obsahuje metody pro vytváření jednotlivých vrstev pro objekt, aktualizaci a oživení objektu.

Objekt SvgStarSystemMap reprezentuje samotnou mapu. Mapa je vykreslena pomocí metody buildViewport objektu SvgViewportManager. Ten se zároveň stará o ovládání zobrazení: posun mapy tažením, zoom apod.

K aktualizaci mapy je využíván objekt SvgRenderer. Ten zajišťuje aktualizaci všech vykreslených objektů. Pro zajištění synchronizace překreslování SVG a okna prohlížeče je

použita funkce requestAnimationFrame (novinka HTML5). [34]

Proces vytvoření mapy hvězdného systému je znázorněn na diagramu aktivit na obrázku 5.8. Nejprve dojde k nastavení rozměrů GUI, které ovlivňuje vycentrování mapy. Poté je načtena konfigurace z cookies, která umožňuje změnit detaily vykreslování. Následuje načtení hvězdného systému z XML a jeho propojení (pro zobrazení názvů cílových hvězdných systémů).



Obrázek 5.8: Diagram tříd vykreslování mapy galaxie.

Po načtení dat je mapa vykreslena. Důležité je její tzv. oživení – zaregistrování obslužných funkcí pro události objektů na mapě. Speciálně je zde zajištěna změna velikostí prvků, jejichž velikost je závislá na velikosti textu[35].

Aktualizace animace mapy probíhá periodicky pomocí časovače. Je rozdělena na dvě fáze:

[34] Dokumentace na – `https://developer.mozilla.org/en/DOM/window.requestAnimationFrame`
[35] Velikost textu je možné zjistit až po jeho vykreslení.

- příprava změn (výpočty, sestavení řetězců apod.; funkce prepareUpdate jednotlivých objektů),
- provedení změn v DOM[36] (funkce performUpdate).

Toto rozdělení je nutné pro zajištění co nejplynulejšího překreslování. Při změnách v dokumentu provádí prohlížeč překreslení automaticky. Při větším množství změn je nebezpečí, že k tomuto překreslení dojde před jejich dokončení. To má za následek vícenásobné překreslování před dokončením sady změn. Tím, že jsou změny provedeny v rychlém sledu a nejsou přerušovány jinými operacemi se toto nebezpečí snižuje. Problém zcela eliminuje použití již zmíněné funkce requestAnimationFrame, která však zatím není zatím dostatečně podporována.

5.7 Formáty dat herního obsahu

Dalším důležitým aspektem architektury je využívání datových souborů definujících různé herní objekty ve formátu XML, které mohou být sdíleny mezi oběma hlavními částmi a mohou být předány v nezměněné podobě přímo do prohlížeče uživatele. Díky tomu je možné využít již existující mechanismy pro caching a také oddělit samotný herní obsah od funkcionality.

Formát XML byl zvolen pro jeho širokou rozšířenost a podporu. Navíc je možné s ním manipulovat přímo pomocí JQuery a také je ho možné přímo kombinovat s XHTML, pokud by tato potřeba někdy vznikla. Výhoda je také možnost využití XML schémat pro validaci.

Současná verze hry pracuje s dvěma typy datových souborů: definice mapy galaxie a definice hvězdného systému.

5.7.1 Soubor definice hvězdného systému

Soubor pro definici hvězdného systému obsahuje všechny údaje o vesmírných tělesech patřících do konkrétního hvězdného systému. Soubor je využit pro sestavení reprezentace mapy na serveru, vykreslení hvězdného systému u klienta a jako zdroj doplňujících informací, sloužících ke „vtažení“ hráče do herního světa (v angličtině označováno „immersion“).

Soubor definuje následující tělesa:

- hvězdu,
- seznam planet s jejich popisem,
- seznam koncových bodů červích děr.

[36] DOM – Document Object Model, objektová reprezentace HTML dokumentu v prohlížeči.

Koncové body červích děr nejsou v této reprezentaci napojeny na okolní hvězdné systémy. Toto napojení je definováno v mapě galaxie a díky tomu je možné jej změnit nezávisle.

Diagram struktury XML Schéma tohoto formátu je k dispozici v projektové wiki[37].

Ukázka:

```
<starsystem name="Proxima Centauri">
    <star name="Proxima Centauri">
        <trajectory>
            <stacionary x="0" y="0" />
        </trajectory>
        <details>
            <gravity>0.0</gravity>
            <mass>0.0</mass>
            <description> Star of Proxima Centauri system. </description>
        </details>
    </star>
    <planets>
        <planet altName="Praklarush Theonas" name="Proxima Centauri 1">
            <trajectory>
              <ellipticOrbit period="283" a="55"
                  b="53" angle="0" initialAngle="0.0" />
            </trajectory>
            <details>
                <gravity>0</gravity>
                <mass>3.00163783302852e+023</mass>
                <description>
                    Description of Praklarush Theonas placeholder.
                </description>
            </details>
        </planet>
    </planets>
    <wormholeEndpoints>
            <wormholeEndpoint id="0">
                <trajectory>
                    <circularOrbit period="62000"
                        initialAngle="10.0" radius="201"/>
                </trajectory>
            </wormholeEndpoint>
    </wormholeEndpoints>
</starsystem>
```

[37] Projektová wiki, *Star System XML* – http://spacetraffic.kiv.zcu.cz/code/tiki-index.php?page=Star+System+XML

5.7.2 Soubor definice mapy galaxie

Soubor pro definici hvězdného systému je určen pro sestavení grafu mapy galaxie. Jeho obsah se skládá ze dvou částí: seznamu hvězdných systémů a seznamu jejich propojení. Navíc jsou pro každý hvězdný systém v seznamu doplněny souřadnice bodu na ploše, které představují polohu hvězdy na dvourozměrném zobrazení této mapy.

Formát je definován XML schématem, jehož struktura je znázorněna v diagramu na obrázku D.2 uvedeného v příloze D.

Ukázka:

```
<galaxy name="Milky Way">
  <starsystems>
    <starsystem name="Solar system" x="300" y="275" />
    <starsystem name="Proxima Centauri" x="300" y="175" />
  </starsystems>
  <wormholes>
    <wormhole id="0">
      <endpoint system="Solar system" id="0"/>
      <endpoint system="Proxima Centauri" id="0"/>
    </wormhole>
  </wormholes>
</galaxy>
```

5.7.3 Další definiční soubory

Pro budoucí rozšíření hry vznikly ještě dva další formáty definičních souborů: definice modelu lodi a definice komponent. Schémata pro tyto formáty a jejich ukázky jsou součástí zdrojových kódů projektu, jejich konečná podoba však nebyla k dispozici v době vzniku této práce.

5.8 Doplňkové části (herní wiki, portál)

Herní wiki a portál doplňují architekturu celé aplikace. Přestože se můžou zdát nepodstatné, je jejich existence klíčová pro úspěch projektu. Počítá se s využitím existujících open source řešení se snahou o co nejjednodušší integraci do zbytku systému.

Jak již bylo zmíněno, **herní wiki** má sloužit jako nápověda ve hře. Výhoda použití wiki je především umožnění hráčům dále rozšiřovat její obsah a tím ulehčení práce vývojovému týmu. Vhodnou aplikací by v tomto případě byla MediaWiki[38], jejíž používání je všeobecně známé díky *Wikipedii*. Je třeba však provést další analýzu.

[38] Web MediaWiki – `http://www.mediawiki.org/wiki/MediaWiki`

Portál hry je pak systém typu CMS, který zajistí základní prezentaci hry a projektu na webu. Zajišťuje následující funkce:

- základní informace o projektu;
- základní informace o hře;
- informace o možnostech spolupráce na projektu;
- galerie obrázků a dalších prezentačních materiálů;
- blog projektu
- fórum projektu (veřejné).

Jedná se o důležitý prostředek komunikace s veřejností a potenciálními hráči, který může začít budovat povědomí o projektu ještě před vydáním hratelné verze hry. Tento portál může také napomoci v naplňování samotného účelu projektu – propagace studia na KIV.

Rozhodnutí o použití konkrétní aplikace nebylo v době psaní této práce k dispozici. Byly diskutovány open source aplikace WordPress[39] a Orchard[40]. Je zde však jasný zájem využít již existující řešení před vytvářením vlastního.

[39] Web WordPress – `http://wordpress.org/`
[40] Orchard – `http://www.orchardproject.net/`

6 Zkušenosti z vývoje v akademickém roce 2011/2012

Tato kapitola má za cíl předání získaných praktických zkušeností z práce na projektu Space Traffic v akademickém roce 2011/2012. Je určena především studentům odpovědným za vedení projektu v budoucích letech, kteří mohou tyto zkušenosti přímo využít, může mít však přínos i pro čtenáře, kteří se zabývají projekty s podobným charakterem.

Protože je zde hodnocena práce studentů, je třeba v úvodu uvést, že uvedené závěry je třeba brát jako subjektivní, protože skupina studentů, která se na projektu podílela, není dostatečně velkým reprezentativním vzorkem pro obecné závěry. Přesto mohou popisované zkušenosti a předkládaná doporučení posloužit jako vodítko v budoucí práci na tomto projektu a na projektech podobných.

V akademickém roce 2011/2012 se do projektu zapojil jeden student z prvního, 6 studentů z druhého ročníku bakalářského studia a 4 studenti z prvního ročníku studia navazujícího. Studenti se zapojili do projektu prostřednictvím semestrálních prací v předmětech KIV/PT[1], KIV/ZIM[2], KIV/ZSWI[3], KIV/ASWI[4] + KIV/DB2[5] a KIV/NET[6]. Dva studenti se ještě zapojili prostřednictvím předmětů Projekt.

6.1 Zadané úlohy

Hlavním úkolem studentů bakalářského studia bylo vytvořit plánovač letu lodi, který by umožnil sestavit časový plán letu lodi mezi jednotlivými pohybujícími se tělesy. Bod setkání byl určován na základě výpočtů pohybů těles po oběžných drahách a průsečík metodou půlení intervalů. Tento úkol byl řešen v rámci předmětů KIV/PT v zimním semestru a dále dopracováván v letním semestru v rámci KIV/ZSWI. Dále byl vytvořen v rámci předmětu KIV/NET editor hvězdných soustav.

Studenti navazujícího studia řešily implementaci persistenční vrstvy pomocí ADO.NET Entity Frameworku. Tuto úlohu zpracovávali v letním semestru jako tým v rámci předmětů KIV/ASWI a KIV/DB.

Zbylá práce studentů měla podpůrný charakter, zahrnovala analýzu dostupných nástrojů pro podporu vývoje a analýzu nástrojů pro tvorbu videodokumentace k projektu.

[1] KIV/PT – Programovací techniky
[2] KIV/ZIM – Znalostní a informační management
[3] KIV/ZSWI – Základy softwarového inženýrství
[4] KIV/ASWI – Pokročilé softwarové inženýrství
[5] KIV/DB2 – Databázové systémy 2
[6] KIV/NET – Programování v prostředí .NET

6.2 Hodnocení úrovně znalostí studentů

V prvé řadě je třeba uvést, že obě skupiny studentů (bakalářské a navazující studium) odvedly velmi kvalitní práci, která v rámci možností vyřešila zadané úlohy. Většinu nedostatků se podařilo odstranit průběžně a výsledky práce byly do projektu již integrovány, nebo se s jejich integrací počítá.

Přesto je zde několik obecných nedostatků. Největším problémem při řešení úlohy plánovače byl nedostatek zkušeností s laděním a testováním aplikací. Důsledkem bylo to, že odevzdávaná řešení fungovala jen za určitých podmínek a nebyla řádně odzkoušena pro krajní hodnoty vstupu. Chybělo zde zamyšlení nad tím, co je nutné otestovat a jaká jsou kritéria toho, že daný algoritmus funguje.

To vedlo například k tomu, že plánovač nebyl řádně ošetřen proti malým nebo naopak extrémně velkým hodnotám rychlosti lodi a chyba způsobila nekonečný cyklus. Druhou chybou bylo to, že studenti testovali plánovač vždy s jiným počátečním časem (aktuální čas), takže pokud došlo k chybě, nebylo jí možné znovu navodit a odhalit příčinu (použitý čas se ani nevypisoval).

Přestože se tyto problémy podařilo nakonec překonat, lze doporučit, aby v budoucnu byli studenti na tuto problematiku upozorněni a tato činnost s nimi byla konzultována.

Dalším problémem bylo nedostatečné pochopení významu chybových stavů a výjimek v aplikaci, především domněnka, že je třeba ošetřovat všechny chyby bez výjimky. Protože se práce studentů týkala většinou jádra programu, nebylo vhodné ošetřovat chyby, o jejichž významu nebylo na dané programové úrovni možné rozhodnout. I na tuto oblast je vhodné předem studenty upozornit.

Posledním významnějším problémem, byla nedostačující úroveň znalosti anglického jazyka. To mimo upuštění od anglických komentářů v kódu znamenalo nutnost revidovat názvy některých tříd, metod a proměnných, protože v použité formulaci nedávaly smysl.

6.3 Zkušenosti s organizací

Organizace prací na projektu probíhala bez větších problémů. Se studenty byly domluveny pravidelné schůzky, na které se studenti řádně dostavovali. Tak bylo možné probírat řešení úloh průběžně a vzniklé problémy řešit včas. Přesto se vyskytlo několik problémů, které zde budou popsány, aby se zamezilo jejich opakování.

Prvním problémem bylo rozdělení schůzek na různé dny, kdy bylo třeba opakovat stejnou informaci dvěma různým skupinám studentů, řešících stejný problém. To se ukázalo jako velký problém a docházelo k nekonzistencím v předávaných informacích.

V letním semestru byla proto snaha mít jednu pravidelnou schůzku se všemi účastníky projektu tak, aby měli všichni členové přehled o tom, co se v projektu děje. Tato společná schůzka měla formu inspirovanou standup schůzkou z metodiky SCRUM (popsáno v Jason

Yip, *It's Not Just Standing Up* [20]). Pokud bylo třeba řešit s některými studenty konkrétní problém, byla dohodnuta individuální schůzka s účastí všech na tomto konkrétním problému zainteresovaných. Bližší podrobnosti o celé organizaci lze nalézt v diplomové práci Petra Vogla [6, kap. 5].

Druhým problémem byl forma, jakou byla práce studentům zadána. Po zkušenosti z minulých let bylo upuštěno od hledání dobrovolníků. Místo toho byla zvolena cesta zadávání semestrálních prací. Byla nalezena témata, která bylo možné v rané fázi vývoje oddělit od zbytku implementace a ta byla zadána studentům ke zpracování na vytypované předměty.

V praxi se však ukázalo, že tento způsob je vysoce neefektivní. Komplikací byl už samotný fakt, že bylo třeba najít téma práce pro následující dva měsíce, na kterém by studenti mohli pracovat nezávisle, aniž by byl dostatek času na přípravu architektury celé aplikace. Také určení vhodného rozsahu úlohy představovalo značný problém, protože požadavky nebylo možné dostatečně připravit. Navíc studenti pracovali nezávisle na hlavním vývoji, a proto byla integrace jejich práce možná až po jejím dokončení.

Problém integrace byl v letním semestru vyřešen realizací práce v hlavní vývojové větvi projektu. To přineslo jiné komplikace, ale ve směs se to ukázalo jako dobrý tah.

Přesto byla hlavním nedostatkem nutnost zadávat v rámci projektu „miniprojekty" v podobě semestrálních prací, které nebyly přímo navázány na aktuální potřeby projektu hlavního. Po úvaze nad tímto problémem vznikla myšlenka jiné formy zapojení studentů do projektu a to formou jakési praxe. Student by v rámci konkrétního předmětu nedělal klasickou semestrální práci, ale byla by mu práce zadávána průběžně podle potřeb projektu. Jednotlivé úkoly by tak mnohem blíže kopírovaly aktuální stav a potřeby a práce by byla mnohem cílenější. Přidělované úkoly by musely být voleny tak, aby byly naplněny pedagogické požadavky předmětů, na které je touto formou semestrální práce odváděna. Výstupem by byl seznam odvedené činnosti se slovním hodnocením, na základě kterého by bylo možné posoudit aktivitu a kvalitu práce studenta.

Tato forma práce by vyžadovala užší zapojení katedry a vyučujících jednotlivých předmětů především pro konzultaci vhodnosti přidělované práce. Přesto by ve výsledku nejen zlepšit efektivitu práce na projektu, ale také být přínosem pro studenty, kteří by na projektu nasbírali zkušenosti, které by při plnění standardních zadání nemohli získat. Zároveň by jejich práce byla kontrolována staršími studenty a získali by tak i lepší zpětnou vazbu na svou práci nad rámec toho, co mohou vyučující zajistit při běžné výuce.

Posledním důležitý problém v organizaci projektu bylo obtížné předávání informací potřebných pro pochopení podstaty řešených úkolů mezi zadavateli (vedením projektu) a studenty řešícími konkrétní úkol. Studenti nemají dostatečné zkušenosti s řešením specifických úloh, které nejsou podrobně zadány a nepokládají dostatek doplňujících otázek. Většinou se domnívají, že probírané problematice rozumí, aniž by se nad ní dostatečně zamysleli a ujistili se o tom s příslušnou osobou. Případné detaily se pak snaží domyslet. To mnohdy vede k nedorozuměním, která se odhalí až při kontrole implementace. Je zde třeba poukázat na fakt, že studenti nemají přehled o celém projektu a mnohdy jim důležité souvislosti připadají jako nepodstatné detaily.

Jediným doporučením, které zde lze uvést, je upozorňovat studenty na věci, které je třeba si poznamenat a také si aktivně ověřovat, zda studenti probíranou tématiku správně pochopili. Přesné a kompletní předávání informací je v zájmu všech zúčastněných stran.

Další problémy s organizací jsou spíše běžné nebo důsledkem nedostatku zkušeností vedení projektu i jeho členů.

6.4 Metodika vedení dokumentace projektu

Kvalitní dokumentace je jednou z klíčových potřeb projektu. Vzhledem k jeho postupném předávání mezi studenty je dokumentace jediným nositelem návaznosti v projektu. Přesto se v praxi ukázalo vedení dokumentace jako závažný problém, vycházející z povahy fungování akademického prostředí.

6.4.1 Problém pořizování dokumentace ve školním projektu

V prvé řadě je třeba říct, že největším problémem není samotné pořizování dokumentací, které studenti vytvářejí jako součásti semestrálních prací. Problém je v jejich studiu. Studenti nejsou ochotni věnovat velmi mnoho času čtení rozsáhlých dokumentů nad rámec svých povinností.

Kvalita vytvářených dokumentací je také kolísavá. Je zde patrná snaha o splnění pomyslných povinných rozsahů stran, bez zamyšlení se nad tím, co je vlastně třeba říci. Vzhledem k výše uvedenému problému se čtením dokumentací by mnohem lépe vyhovovaly dokumenty shrnující základní myšlenky v podobě bodů, doplněné o vysvětlující diagramy a obrázky. Souvislý text je spíše na obtíž.

Klíčový význam má komentování zdrojových kódů. Jak již bylo uvedeno v předchozí podkapitole, úroveň znalosti anglického jazyka mezi studenty ještě stále neodpovídá obecným potřebám. Při požadavku na pojmenovávání proměnných, metod a tříd anglicky, není díky chybám v termínech a slovosledu vznikající kód vždy přehledný.

Posledním problémem je aktualizace již vzniklé dokumentace. Udržování rozsáhlých sad strukturovaných dokumentů vyžaduje příliš mnoho úsilí, které nakonec nepřináší dostatečný přínos. Udržování většího množství takových dokumentů je proto nežádoucí.

6.4.2 Klíčové oblasti dokumentace

Pro projekt jsou klíčové následující oblasti dokumentace:

- návody pro rychlý vstup do projektu, především pro orientaci v nástrojích a jejich instalaci;

- hlavní principy fungování vyvíjených aplikací, především vhodné diagramy a seznamy důležitých prvků[7];
- instrukce pro orientaci v kódu určenou začátečníkům (seznamy, co je důležité a kde co najít);
- checklisty pro opakovaně vykonávané práce;
- komentování kódu, včetně myšlenek a nápadů;

Pokrytí těchto oblastí by mělo pomoci ve zvládání dokumentování projektu. Bohužel je možné tyto zkušenosti zhodnotit až zpětně a v akademickém roce 2011/2012 se jimi nebylo možné řídit.

6.4.3 Využití videotutoriálů a videoblogu

Při úvahách nad metodikou vedení dokumentace vznikla myšlenka vytvoření sady videotutoriálů, jejichž cílem by bylo uvedení do jednotlivých oblastí projektu. Výhodou tohoto řešení je především efektivita předávání znalostí.

Protože vytvoření videa je mnohem náročnější, než úprava textového dokumentu, měla by tato činnost smysl jen pro pokrytí nejvyšší úrovně znalostí o projektu. Tak by se předešlo nutnosti častých aktualizací.

Druhou možností využití videa je vytváření videoblogu[8] projektu. Jeho forma by byla v krátkých videích, shrnujících aktuální myšlenky, problémy, úspěchy i neúspěchy. Videa by bylo možné natáčet poměrně rychle a nevyžadovala by žádnou komplikovanou přípravu. Vhodný interval pro jejich natáčení by byl jeden až dva týdny.

Hlavním přínosem takového videoblogu by bylo průběžné dokumentování historie projektu, které by umožňovalo zpětně vysledovat důležitá rozhodnutí a zároveň by posloužilo týmu jako jednotný zdroj pro diskuzi.

6.4.4 Wiki projektu

Pro vedení dokumentace, která nemůže být součástí zdrojového projektu, byla zřízena wiki.[9] Jejím účelem je poskytnout jednotný portál k projektovým znalostem, ke kterému mají přístup všichni členové projektu. Cílem je, aby editace prováděli všichni a udržovali tak informace aktuální.

[7] Někdy je potřeba dozvědět se, co je potřeba vědět stejně významná, jako potřeba konkrétní informace.

[8] Pro technickou realizaci lze využít službu YouTube, která nabízí přímou podporu této formy publikování videí.

[9] Dostupná na `http://spacetraffic.kiv.zcu.cz/code/`

Wiki je klíčovým nástrojem projektu a mělo by k ní být takto přistupováno. Její zřízení bylo provázeno celou řadou problémů[10] a na jejím nasazení je ještě třeba dále pracovat.

6.5 Doporučení pro další vývoj

Následují návrhy a doporučení pokračování vývoje projektu.

6.5.1 Návrh cílů pro akademický rok 2012/2013

Pro akademický rok 2012/2013 navrhuje autor práce stanovení následujících cílů (jedná se o navázání na návrh uvedený v příloze **??**):

1. Dokončení integrace výsledků práce z letního semestru 2011/2012.
2. Vytvoření komunity projektu (portál, blog, sociální sítě).
3. Dokončení GUI pro ovládání lodí.
4. Implementace programování lodí.
5. Implementace obchodu (pouze trh).
6. Minimální potřebná implementace základen.
7. Základní funkce NPC.
8. Implementace podpory achievementů, levelování.
9. Tvorba herního obsahu.
10. Testování.
11. Základní integrace herní wiki.
12. Vylepšení konfigurace projektové wiki.

6.5.2 Návrhy zadání semestrálních prací

Následuje seznam možných témat semestrálních prací na vytypované předměty:

- Systém achievementů jako samostatná práce na KIV/NET nebo KIV/PT.

[10] Především nalezení vyhovující implementace, které se podařilo až v polovině letního semestru.

- Prototyp[11] fungování obchodu na KIV/PT.
- Prototyp fungování základen na KIV/PT.
- Práce na implementaci jazyka Starship Basic na KIV/FJP. Lze zadat podmnožinu problému.
- Úpravy databáze pro implementaci nových prvků na KIV/DB1.
- Implementace jazyka Starship Basic jako oborový projekt.
- Implementace pohybu lodí jako oborový projekt – vzájemně se doplňují.
- Vylepšení konfigurace projektové wiki na KIV/ZIM.
- Práce na GUI na předměty KIV/PIA a KIV/ASWI.
- Vytvoření nezávislých nástrojů na KIV/NET.

Nelze očekávat, že se podaří všechna tato zadání obsadit. Také je dobré oslovit stejné studenty v rámci všech jejich předmětů, usnadní to komunikaci a může to vést k dlouhodobější spolupráci.

[11] Prototypem chápeme řešení založené na doménovém modelu hry, ale vyvíjené zvlášť. Pokud by bylo vytvořené řešení kvalitní, bylo by s potřebnými změnami integrováno do projektu. Cílem práce je ověřit koncept.

7 Závěr

Hlavní náplní této diplomové práce bylo pokračování realizace projektu Space Traffic, Konkrétně se jednalo o zhodnocení game designu, jeho úpravy a především doplnění nezbytných detailů. Dále byla navržena architektura herní aplikace, umožňující implementovat navržené herní prvky. Implementace základních částí této architektury byla provedena ve spolupráci se studenty KIV v rámci semestrálních prací na některé předměty. Obě tato témata shrnuje text diplomové práce.

Projekt Space Traffic vznikl v roce 2009 pod vedením Zbyňka Neuberta, který na toto téma obhájil diplomovou práci *Analýza, návrh a vedení týmu v projektu webové hry* [1] v roce 2010. Jím předložený game design hry byl touto prací revidován. Provedené změny měly za cíl především rozšíření konceptu programování jako herní mechaniky, který se stal nosným prvkem celé hry.

Předchozí implementace hry byla použita jen jako referenční, důvody, proč nebyla převzata, jsou uvedeny v této práci. Nově vytvořená architektura a její impelementace založená na technologii Microsoft .NET vytváří podmínky pro implementaci jednotlivých herních prvků a poskytuje prostor pro další rozšiřování.

Práce na implementaci byla vedena autorem práce, který v realizačním týmu zastával funkce hlavního programátora, analytika a do jisté míry také „kouče“. Výsledkem je vytvoření kostry webové aplikace, vyřešení persistence dat, komunikace mezi jednotlivými částmi aplikace, konfigurace, podpora skriptů a další specifické funkcionality, popsané v této práci. Přestože nebyla zatím vytvořena hratelná verze hry, hlavním cílem bylo vytvoření základu, na kterém bude možné dále stavět.

Literatura

[1] NEUDERT, Zbyněk. *Analýza, návrh a vedení týmu v projektu webové hry.* Plzeň, 2010. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky.

[2] First video game. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-06-26]. Dostupné z: `http://en.wikipedia.org/wiki/First_video_game`

[3] List of video game consoles. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001- [cit. 2012-06-27]. Dostupné z: `http://en.wikipedia.org/wiki/List_of_video_game_consoles`

[4] Browser Games: Basics and History, In: *CTrust Network* [online], 27.02.2012 [cit. 2012-6-10]. Dostupné z: `http://www.ctrustnetwork.com/a36/gaming/browser-games/articles/growser-games-history.html`

[5] Earth: 2025. In: *Earth Empires Wiki* [online]. 11.6.2011 [cit. 2012-6-10]. Dostupné z: `http://wiki.earthempires.com/index.php/Earth:_2025`

[6] VOGL, Petr. *Podpora vývoje webové hry pro více hráčů.* Plzeň, 2012. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky.

[7] KOCMAN, Richard. *Řízení projektu webové hry.* Plzeň, 2012. Diplomová práce. Západočeská univerzita v Plzni, Fakulta aplikovaných věd, Katedra informatiky a výpočetní techniky.

[8] SCHELL, Jesse. *The Art of Game Design: A Book of Lenses.* USA: Elsevier Inc., 2008. ISBN: 978-0-12-369496-6.

[9] Gilliard Lopes. Game Design Cognition: The Bottom-Up And Top-Down Approaches. In: Gamasutra. *Gamasutra: The Art & Business of Making Games.* [online]. Rafael Kuhnen. 14.11.2007, [cit. 2012-6-10]. Dostupné z: `http://www.gamasutra.com/view/feature/2129/game_design_cognition_the_.php`

[10] Jonathan Blow. Video Games and the Human Condition. In: *Youtube.* [online]. 27.9.2012 [cit. 2012-6-10]. Dostupné z: `http://www.youtube.com/watch?v=SqFu5O-oPmU`

[11] GERŠL, Vladimír. Co je game-design a jak vlastně vzniká In: *games.zcu.cz* [online]. ZČU v Plzni, FAV, KIV, 27.04.2012 [cit. 2012-06-28].

[12] TURING, Alan. *On computable numbers, with an application to the Entscheidungsproblem*, Proceedings of the London Mathematical Society, Series 2, 42 (1936), pp 230–265. [online] Dostupné z: `http://www.turingarchive.org/browse.php/B/12`

[13] Intermodal container. In: *Wikipedia: the free encyclopedia* [online]. St. Petersburg (Florida): Wikipedia Foundation, 2.2.2003, last modified on 24.6.2012 [cit. 2012-6-10]. Dostupné z: `http://en.wikipedia.org/wiki/Intermodal_container`

[14] *Extra Credits*.Season 2, Ep. 4, Achievements.Online stream. Dostupné z: `http://www.penny-arcade.com/patv/episode/achievements`

[15] Steven Vickers. Sinclair ZX Spectrum: Basic Programming. In: Pete Robinson. *World of Spectrum*. [online]. 1995 [cit. 2012-6-10]. Dostupné z: `http://www.worldofspectrum.org/ZXBasicManual/`

[16] John Badham. *WarGames* [DVD]. Lawrence Lasker,Walter F.Parkes. USA: 1983

[17] CLEMENTS, Paul. *Documenting software architectures: views and beyond*. Boston. ISBN 02-017-0372-6.

[18] KLEIN, Scott. *Professional WCF programming: .NET development with the Windows Communication Foundation*. Indianapolis, IN: Wrox/Wiley Pub., c2007, xix, 430 p. ISBN 04-700-8984-9.

[19] OLIVEIRA, Jason. *Tutorial: Code First with EF 4.1* [online]. 2011-10-17. [cit. 2012-06-26]. Dostupné z: `http://codefirst.codeplex.com/`

[20] YIP, Jason. It's Not Just Standing Up: Patterns for Daily Standup Meetings. In: *Martinfowler.com* [online]. 2011 [cit. 2012-06-19]. Dostupné z: `http://www.martinfowler.com/articles/itsNotJustStandingUp.html`

Seznam použitých symbolů a zkratek

2D	dvourozměrná (grafika)
3D	trojrozměrná (grafika)
AJAX	Asynchronous JavaScript and XML
CD	Herní měna - kretidy; viz. 4.6.7
CMS	Content Management System
CSS	Cascade Style Sheets
DAO	Data Access Objects
DHTML	Dynamic HTML
DOM	Document Object Model
EXS	Externí subsystém lodi
F2P	free-to-play
FAV	Fakulta aplikovaných věd
GB	gigabyte
GDD	Game Design Document
GUI	graphic user interface; grafické uživatelské rozhraní
HTML	HyperText Markup Language
INS	Interní subsystém lodi
JSON	JavaScript Object Notation
KIV	Katedra informatiky a výpočetní techniky
LINQ	Language-integrated Query
MMO	Massively multiplayer online

MMOBG	Massively multiplayer online browser games
MMORPG	Massively multiplayer online role-playing game
MS	Microsoft
MVC	Model-view-controller
NPC	Non-player character
ORM	Object Relational Mapping
OS	operační systém
P2P	pay-to-play
PC	Personal Computer
PS3	(Sony) PlayStation 3
RPG	Role-playing game
SOA	service-oriented architecture
SOAP	Simple Object Access Protocol
SVG	Scalable Vector Graphics
SVN	Subversion
TDD	Test-driven development
U	jednotka nákladu; viz. 4.6.5
URL	Uniform resource locator
VS	(Microsoft) Visual Studio
XHTML	eXtensible HyperText Markup Language
XML	Extensible Markup Language
ZČU	Západočeská univerzita v Plzni

Seznam obrázků

A Plán pro akademický rok 2011/2012

Space Traffic 2011/2012

Popis projektu

Vytvoření on-line webové hry jako nástroje pro propagaci Katedry informatiky a výpočetní techniky.

Cílová skupina

Hráči hry budou převážně studenti středních škol. Hra má za cíl poskytnou hráčům krátkodobou zábavu při prezentaci na dni otevřených dveří a několik následujících týdnů po jeho skončení.

Popis hry

- Hráč začíná s jednou nákladní kosmickou lodí.
- Hráč se pohybuje z planety na planetu, mezi hvězdnými systémy pak prostřednictvím umělých červích děr.
- Loďě je třeba zásobovat palivem.
- Na každé planetě je k dispozici NPC obchodník, který nabízí a poptává zboží různého druhu. (ceny se pohybují)
- Náplní hry je meziplanetární obchod - hráč je v roli dopravce.
- Hráč je odměňován herní měnou.
- Hra obsahuje achievementy.

Cíle pro rok 2011/2012

1. Dokončit registraci a přihlašování

2. Navrhnout a zdokumentovat architekturu (server + client), vychází se z Nette frameworku.
3. Navrhnout a implementovat engine pro vykreslování objektů na straně klienta na základě prototypů (JavaScript, SVG, HTML5).
4. Implementovat mapu hvězdného systému a galaxie.
5. Vytvořit nástroje pro vytváření map hvězdných systémů a galaxie (nemusí být implementovány jako webová aplikace).
6. Navrhnout a implementovat pohyb lodi po mapě, ovládání, provedení a zobrazení.
7. Navrhnout a implementovat podporu achievementů.
8. Navrhnout a implementovat podporu pro herní poštu, případně další formy komunikace (chat).
9. Implementace herního GUI.
10. Vytvoření tutorialu.
11. Vytvoření textových popisů herních objektů.
12. Vytvoření grafiky gui (nakreslení).
13. Vytvoření grafiky pro herní objekty (nakreslení).
14. Vytvoření testů.
15. Navrhnout a implementovat zakončení hry (např. po splnění všech achievementů).
16. Vytvoření portálu pro hru a blog, komentující vývoj.

Tyto body představují kompletní první verzi hry.

Další rozšíření

- Návrh programovacího jazyku, umožňujícího vytvořit automatické ovládání hráčovy obchodní flotily (automatické vyhledávání a realizace zakázek).
- Možnost budovat základny s vlastními výrobními prostředky a konkurovat NPC a ostatním hráčům na jednotlivých planetách.
- Zakládání korporací, sdružujících hráče a jejich výrobní prostředky (je třeba dále navrhnout důsledky takového sdružování). Podpora pro vytváření miniher (služby pro komunikaci s minihrou, jejich organizace a model odměňování).
- Události v herním světě, například živelné katastrofy.

- Nápověda ve stylu "Civilopedie"– herní encyklopedie s popisem herních prvků a objektů a odkazy na reálný svět, implementováno jako wiki.
- Herní oznamování o přednáškách pořádaných KIV a dalších informacích zajímavých pro hráče.
- Vylepšování lodí pomocí komponent.

Použité technologie

JavaScript, frameworky Mono a *JQuery*.

PHP5, frameworky *Nette 2*, *Dibi* a

SimpleTest

HTML5 a *CSS3*, *SVG*.

.NET a *Java* pro vytváření vývojových a editačních nástrojů.

B Zpráva o stavu prací, léto 2011/2012

Space Traffic 2012/2013

Co funguje

- Game design na několik let dopředu.
- Architektura na nejvyšší úrovni - GameServer, Webové Ui, JS engine.
- Definice XML formátů pro mapu a hvězdné systémy (včetně načítání), modely lodí, komponenty.
- Podpora skriptů v C#, dynamický překlad.
- Databáze, přístup přes DAO, snadné rozšíření.
- JS engine pro vykreslování grafiky, lze rozšiřovat, je tam pár bugů, které by se mělo povést odstranit.
- Přihlašování, Registrace na velmi základní úrovni.
- Nákup a pohyb lodí v základní verzi (rozpracováno, zbývá pospojovat - léto).
- Testy na některé části jádra.
- Testovací data (assety).
- Editor hvězdných systémů.
- Nakonfigurována wiki projektu.
- Návody a dokumentace, DP 2x.

Použité technologie

JavaScript, frameworky Mono a *JQuery*.

.NET, *ASP.NET MVC 3*, *Entity Framework*, *NLog*

HTML5 a *CSS3*, *SVG*, *Less* (dotless).

Cíle pro rok 2012/2013

1. Vytvoření komunity kolem projektu (portál, blog, sociální sítě)
2. Implementace obchodu
3. Implementace misí
4. NPC
5. Implementace programování lodí
6. Implementace podpory achievementů
7. Tvorba herního obsahu
8. Balancování, testování
9. Integrace herní wiki
10. Dokončení gui pro ovládání lodí.

C Příkazy jazyka Starship Basic

C.1 Elementární příkazy

Jazyk Starship Basic využívá následující podmnožinu klíčových slov Sinclair BASIC:

ABS mat. funkce absolutní hodnota

ACS mat. funkce arccos

AND log. funkce konjunkce

ASN mat. funkce arcsin

ATN mat. funkce arctan

CLS vymaže lodní deník

CONTINUE pokračování ve vykonávání programu (opak pause)

COS mat. funkce cosinus.

DIM deklarace pole

EXP mat. funkce exponenciální funkce (hodnota Eulerova čísla EXP 1)

FN volání definované uživatelské funkce

FOR cyklus for; syntaxe FOR *řídící proměnná* = *počáteční hodnota* TO *limit* [STEP *krok*]

GO SUB příkaz skoku s návratem pomocí příkazu RETURN

GO TO příkaz skoku

IF podmíněný příkaz; syntaxe IF podmínka THEN příkaz; větev ELSE není podporována.

INT převedení čísla float na integer oříznutím desetinné části.

LEN funkce vracející délku řetězce

LN mat funkce přirozený logaritmus

NEXT zvýší řídící proměnnou cyklu FOR na další hodnotu a skočí na jeho začátek

NOT log. funkce negace

ON ERR nastavení příkazu, který se provede při chybě; syntaxe ON ERR příkaz; základní použití ON ERR GO TO *číslo řádku* nebo ON ERR CONTINUE

OR log. funkce disjunkce

PAUSE zastavení provádění programu, lze jej opět ručně spustit

PI mat. konstanta \pi

PRINT vytiskne zprávu do lodního deníku

REM komentář v kódu

RETURN návrat z podprogramu (volání GO SUB)

RND vrací náhodné číslo v intervalu <0,1). [1]

SGN mat. funkce signum

SIN mat. funkce sinus

SQR mat. funkce (druhá) odmocnina

STEP velikost kroku inkrementace řídící proměnné cyklu FOR.

STR$ převede číslo na řetězec

TAB tisk tabulátoru

TAN mat. funkce tangens

THEN výchozí větev podmíněného příkazu, viz příkaz IF

TO horní mez pro řídící proměnnou cyklu FOR. Použito také pro definování rozsahů.

VAL převede řetězec na číslo

Kromě těchto základních klíčových slov jazyka lze ještě uvažovat o těchto rozšířeních:

AT pozice pro tisk pomocí příkazu PRINT

CHR$ vrací řetězec obsahující znak s ASCII kódem číselného parametru

CODE vrací ASCII kód prvního znaku v řetězci

DEF FN definování uživatelské funkce

LET přiřazovací příkaz, je možné použít jednoduchou formu bez klíčového slova: a = 5

LOAD nahraje a spustí program z hráčova úložiště; syntaxe LOAD "jméno programu"

[1] Generátor náhodných čísel je nutný pro možnost náhodného rozhodování umělé inteligence lodí. Lze uvažovat i o zavedení příkazu RANDOMIZE pro inicializace tohoto generátoru s konkrétní hodnotou.

VAL$ obdoba příkazu VAL, ale výsledné číslo je převedeno na řetězec

Dále jsou k dispozici následující operátory: ^ (mocnina), <, =, <=, >=, <>, +, -, * a /.

C.2 Elementární akce lodí

Kromě výše uvedených jazykových konstrukcí jsou k dispozici příkazy, které odrážejí specifické akce spojené s hrou. Základní akce lodí jsou:

FLY TO leť do; syntaxe FLY TO "jméno hvězdného systému", ...,"cílová základna"

LDCARGO naložení nákladu; syntaxe LDCARGO "*druh zboží*" AMNT *množství* FROM "*jméno vlastní lodi, skladu nebo továrny*"

ULDCARGO vyložení nákladu; syntaxe ULDCARGO "*druh zboží*" AMNT *množství* TO "*jméno vlastní lodi, skladu nebo továrny*"

BUY koupení zboží na trhu a naloží jej; syntaxe BUY "*druh zboží*" AMNT *množství* MAXP *maximální cena za jednotku* [FROM "*jméno hráče*"]

SELL prodej zboží na trhu; syntaxe SELL "*druh zboží*" AMNT *množství* MINP *minimální cena za jednotku* [TO "*jméno hráče*"]

REPAIR loď bude opravena před odletem.

Tyto akce mohou být provedeny i jako rozkazy, tj. jsou provedeny jednorázově na „ruční příkaz“ hráče. Provádění programu lodi je při takovém příkazu pozastaveno a je třeba ho znova spustit. Řízení chodu programu je možné pomocí rozkazů RUN, CONTINUE, PAUSE, STOP.

C.3 Sbírání informací o herním světě

Aby bylo možné programově rozhodovat o chování lodí, je třeba poskytnout možnosti získávat informace o herním světě. Čím rozsáhlejší budou, tím lepší a komplikovanější programy lze vytvářet. Základní příkazy pro zkoumání herního světa jsou:

GET PRICE vrátí cenu zboží na trhu; syntaxe GET PRICE "*druh zboží*" [FROM "*jméno hráče*"]; pokud není uvedeno jméno hráče, zjistí nejnižší cenu; pouze na základně.

GET BASES vrátí pole s názvy základen; syntaxe BASES ["*jméno hvězdného systému*"]; pokud není uvedeno jméno hvězdného systému, vrátí seznam pro aktuální.

GET PLANETS obdoba GET BASES, vrací pole názvů planet

GET STARSYSTEMS vrací pole s názvy hvězdných systémů

GET FUEL vrací zbývající množství paliva

GET SPACE vrací volné místo v nákladových prostorech pro zadaný druh zboží

GET FLYTIME vrací odhad doby letu do zadaného hvězdného systému

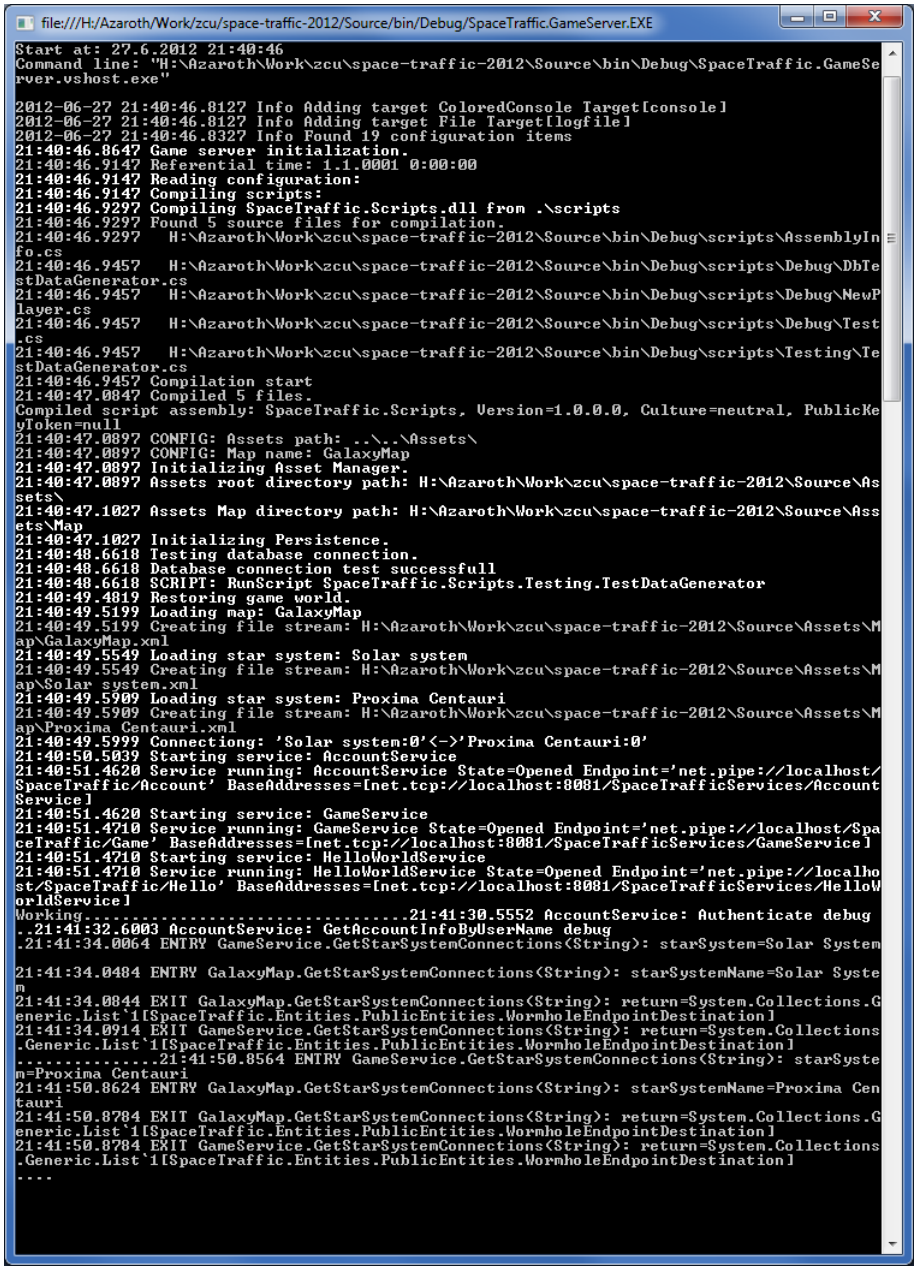
GET SUPPLY vrací hodnotu zásoby uvedeného druhu zboží; syntaxe GET SUPPLY "*druh zboží*" [FROM "*jméno hráče*"]|[IN "*jméno vlastní lodi, skladu nebo továrny*"]; Pouze na základně.

GET EXITS vrací pole názvů hvězdných systémů, do kterých lze cestovat z aktuálního.

Množství dostupných příkazů lze dále rozšiřovat. Jejich funkce by se však měla omezovat na základní operace. Pro získání odvozených informací je tady programovací jazyk.

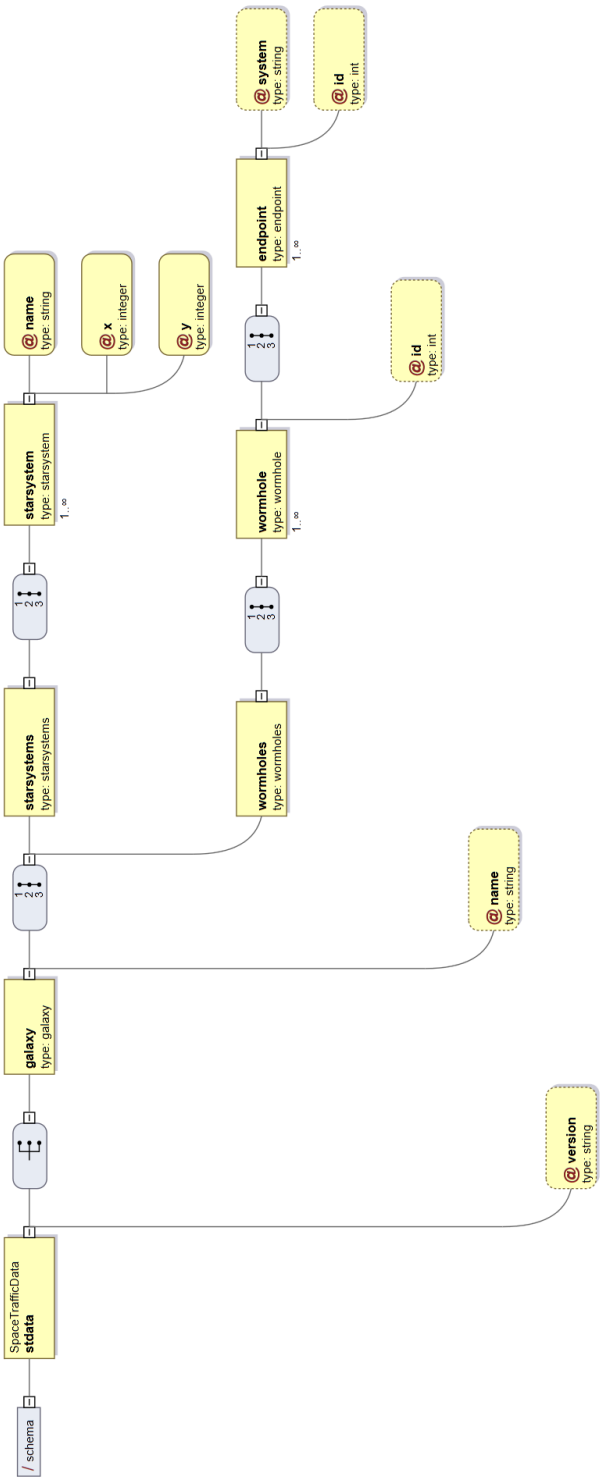
D Doplňující obrázky

D.1 Ukázky běhu herního serveru



Obrázek D.1: Game Server - ukázka běhu serveru.

D.2 Struktura XML formátu mapy galaxie



Obrázek D.2: Struktura XML formátu mapy galaxie. (Originál dostupný na `http://spacetraffic.kiv.zcu.cz/code/tiki-index.php?page=Galaxy+Map+XML`

E Obsah DVD

Přiložené DVD obsahuje:

`readme.txt` - soubor s popisem obsahu DVD.

`mstepanek-dp-2012.pdf` - text práce ve formátu PDF

`mstepanek-dp-zadani.pdf` - zadání práce ve formátu PDF

`doc` - adresář s doplňkovými materiály

`src` - adresář zdrojových kódů projektu ke dni odevzdání práce

`tex` - adresář s textem práce v LaTex