Západočeská univerzita v Plzni
Fakulta aplikovaných věd
Katedra informatiky a výpočetní techniky

Diplomová práce

Podpora vývoje v projektu Space Traffic nástroji Microsoft TFS

Plzeň 2013 Bc. Jan Dyrczyk

Prohlášení

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

V Plzni dne 14. srpna 2013

Bc. Jan Dyrczyk

Abstract

Using Microsoft TFS to support development in the Space Traffic project

Master's thesis deals with the using of the Microsoft Visual Studio Team Foundation Server (TFS) to support development in the Space Traffic project. The project is being developed as student project on the Department of Computer Science and Engineering at the University of West Bohemia. This thesis practically verifies development methodology designed by Vogl [3], suggests its possible modifications and practically verifies the designed improvements. It further explores the possibilities of the TFS for scheduling and tracking the progress of works and verifies the use of the TFS in real university project. In conclusion, the work presents the author's experiences and recommendations for further development of the project.

Poděkování

Tímto bych chtěl poděkovat vedoucímu diplomové práce Doc. Ing. Přemyslu Bradovi, MSc., Ph.D. za cenné rady, připomínky a vedení práce.

Dále všem studentům podílejícím se na vývoji projektu Space Traffic a pracovníkům Katedry informatiky a výpočetní techniky za spolupráci.

A v neposlední řadě mé rodině a přátelům za neutichající podporu během celého studia.

Obsah

1 Úvod

Tato diplomová práce se věnuje tématu podpory vývoje v projektu Space Traffic, studentského projektu na Katedře informatiky a výpočetní techniky Fakulty aplikovaných věd Západočeské univerzity v Plzni[1], nástroji Microsoft Visual Studio Team Foundation Server (TFS), komerčním produktem od firmy Microsoft spadající do rodiny nástrojů pro podporu správy životního cyklu (ALM – Application Lifecycle Management). Tato práce navazuje na diplomovou práci Petra Vogla s názvem *Podpora vývoje webové hry pro více hráčů* [3], která byla obhájena na téže katedře roku 2012.

Práce si klade za cíl prakticky ověřit navrženou metodiku vývoje Petrem Voglem, navrhnout její případné úpravy a navržené vylepšení prakticky ověřit. Dále prozkoumat možnosti TFS pro plánování a sledování postupů prací a prakticky ověřit nasazení TFS do reálného univerzitního projektu. Závěrem práce shrnuje dosažené výsledky a přináší doporučení pro další vývoj.

V úvodu práce je popsán studentský projekt Space Traffic, který si klade za cíl zviditelnit FAV a ovlivnit studenty středních škol ke studiu na této fakultě prostřednictvím vyvíjené webové hry, poskytnout prostor pro studenty ke zlepšení jejich týmových vlastností, pochopení metodik vývoje a nástrojů pro podporu vývoje na reálném projektu.

Dále je uvedena charakteristika dosavadní vývojové metodiky navržené Petrem Voglem, její zhodnocení pří použití v zimním semestru na plánování a vedení prací, a zkušenosti vedoucích projektu z plánování a vedení prací na projektu.

Následující kapitola popisuje životní cyklus aplikace, oblasti pokrytí ALM nástrojů, možnosti TFS pro plánování a sledování postupů prací a porovnání s možnostmi nástroje Redmine.

Další kapitola popisuje upravenou metodiku vývoje pomocí získaných zkušeností ze zimního semestru a použití TFS pro univerzitní prostředí.

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

Závěrem práce přináší praktické ověření úprav postupů a nástrojů v letním semestru. Porovnává obě metodiky vývoje, použití nástrojů pro podporu vývoje, konkrétně TFS a Redmine, a přináší doporučení pro další vývoj.

2 Projekt Space Traffic

Tato kapitola seznamuje čtenáře se studentským projektem Space Traffic, popisuje jeho architekturu a používané technologie. Dále popisuje jeho stav, nástroje pro vývoj, plánování, řízení projektu a cíle v akademickém roce 2012/2013.

2.1 O projektu

V rámci studentského projektu Space Traffic je vytvářena webová obchodní strategie typu MMORTS (Massively multiplayer online real-time strategy) s programovatelnými prvky jako součást ovládání. Projekt je kompletně řízen a vyvíjen samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací. Hra je vyvíjena pod záštitou Katedry informatiky a výpočetní techniky Fakulty aplikovaných věd Západočeské univerzity v Plzni, která se jejím prostřednictvím snaží rozšířit programování mezi středoškolské studenty a tím je ovlivnit v rozhodování o dalším studiu.

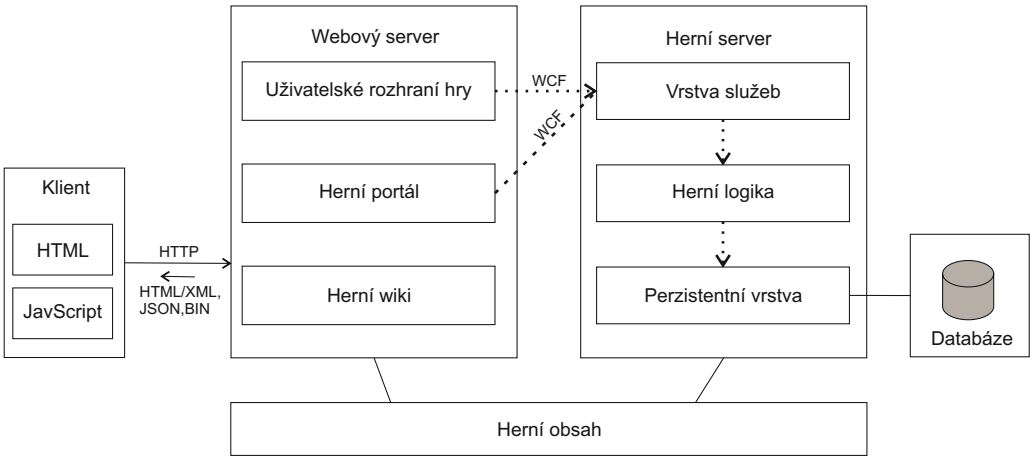
V samotné hře má hráč za cíl být nejlepším obchodníkem ve vesmíru, levně nakupovat a draze prodávat. k tomu má k dispozici množství přepravních obchodních lodí, pomocí kterých může převážet zboží mezi planetami. Planety jsou umístěny ve hvězdných soustavách, mezi kterými vedou červí díry pro rychlý přesun z jedné soustavy do druhé. Jednotlivé herní akce se svými loděmi může hráč vykonávat, jednak prostřednictvím grafického rozhraní, ale také pomocí programovacího jazyka Starship Basic. Vykonáváním herních akcí hráč získává zkušenostní body a dosahuje úspěchů (achievements). Pro potřeby produkce vlastního zboží může hráč nakupovat pozemky na planetách a stavět své vlastní továrny.

Původní analýza a návrh projektu je popsán v diplomové práci Zbyňka Neuderta [1]. Historie, plánování a vedení projektu je popsáno v diplomových pracích Richarda Kocmana [2] a Petra Vogla [3]. Návrh herních principů je

v diplomové práci Martina Štěpánka [4].

2.2 Architektura a technologie

Architektura hry vychází z architektonického vzoru Model-view-controller (MVC). z celkového náhledu architektury na obrázku 2.2.1 je vidět rozdělení do pěti modulů.



Obrázek 2.2.1: Diagram architektury aplikace Space Traffic.

Klient je klientská část webové aplikace běžící v internetovém prohlížeči napsaná v HTML5 s využitím Less (dotles) pro formátování stránek. Pro interaktivní prvky je využit JavaScript s frameworkem jQuery.

Webový server IIS 7 obsluhuje tři nezávislé webové aplikace: **Uživatelské rozhraní hry** postavené na frameworku ASP.NET MVC 3 zprostředkovává komunikaci mezi herním serverem a klientem, **Herní portál** jako systém pro zprávu obsahu pro prezentaci projektu a **Herní wiki** sloužící jako nápověda ve hře.

Herní server jako jádro celého systému hry je implementován v .NET frameworku 4 jazykem C# a je složen ze tří vrstev: **Vrstva služeb** publikuje veřejné služby poskytované serverem technologii WCF (Microsoft Windows Communication Foundation) pro komunikaci mezi serverem a uživatelským rozhraním. **Herní logika** simuluje herní svět a **Perzistentní vrstva** zpro-

středkovává persistenci herních objektů. Přístup do MS SQL databáze je realizován pomocí ADO.NET Entity frameworku.

Databáze běžící na MS SQL serveru 2008 slouží jako záloha dat při výpadku serveru pro následnou obnovu a pro uchování nepotřebných dat v herním světě.

Herní obsah je uložiště statických datových souborů obsahující konfiguraci herního světa, doplňkové texty, audiovizuální zdroje a další digitální zdroje, které lze přímo posílat klientovi a využít mechanismu cachování obsahu.

2.3 Stav na začátku ak. roku 2012/2013

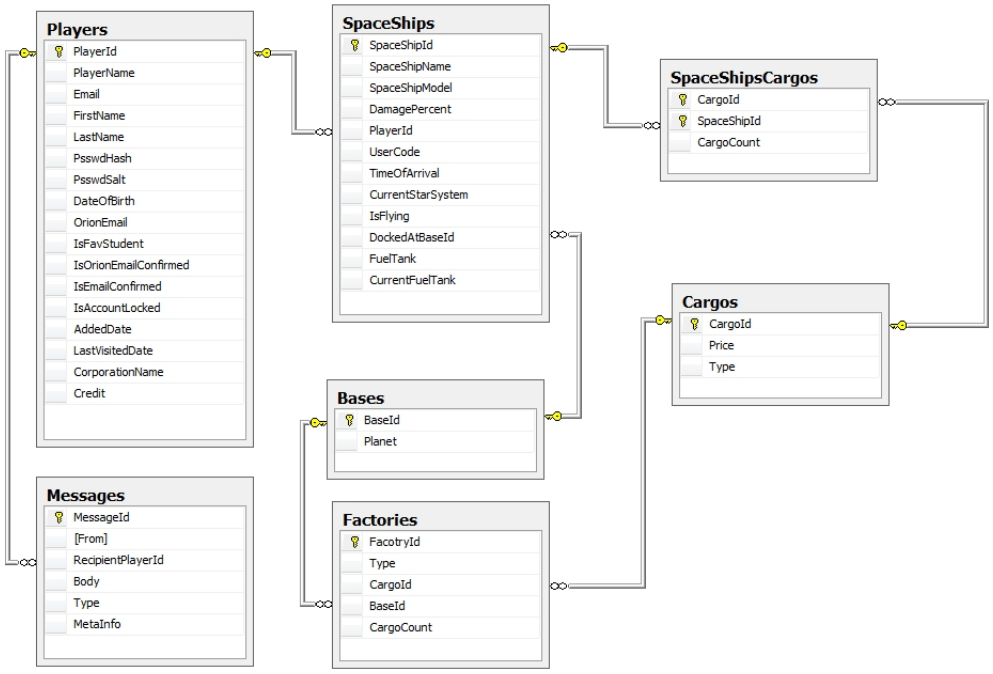
Na začátku akademického roku 2012/2013 byl kompletně navržen game design, hotová architektura na nejvyšší úrovni (GameServer, Webové Ui, JS Engine), hvězdná mapa (zpracování XML, vykreslení SVG, překreslování SVG JavaScriptem), podpora skriptů v C#. Byla vytvořena DAO vrstva pro databázi a základní databázové schéma zobrazeno na obrázku 2.3.1.

Započalo se s psaním testů a vytvářením testovacích dat (assety). Dále bylo rozpracováno přihlášení, registrace, nákup a pohyb lodí. Pro podporu projektu byla vytvořena projektová wiki, návody a dokumentace. Celý stav projektu je rozepsán v příloze a Stav projektu léto 2012.

2.4 Nástroje

Jako hlavní vývojový nástroj byl používán Microsoft Visual Studio 2010/2012 s pluginem pro připojení verzovacího uložiště AnkhSVN, které připojuje zdrojové kódy z Subversion (SVN)[1]. Pro externí přístup do SVN byl využíván TortoiseSVN. Plánování projektu a řízení týmů bylo prováděno s podporou nástroje Redmine[2]. Publikování vývoje projektu probíhalo pomocí Tiki Wiki.

[1]URL k SVN: svn+ssh://students.kiv.zcu.cz/home/subversion/diplomka/spacetraffic.
[2]Projekt je přístupný z URL: http://students.kiv.zcu.cz:3000/projects/space-traffic.



Obrázek 2.3.1: ERA model základní databáze.

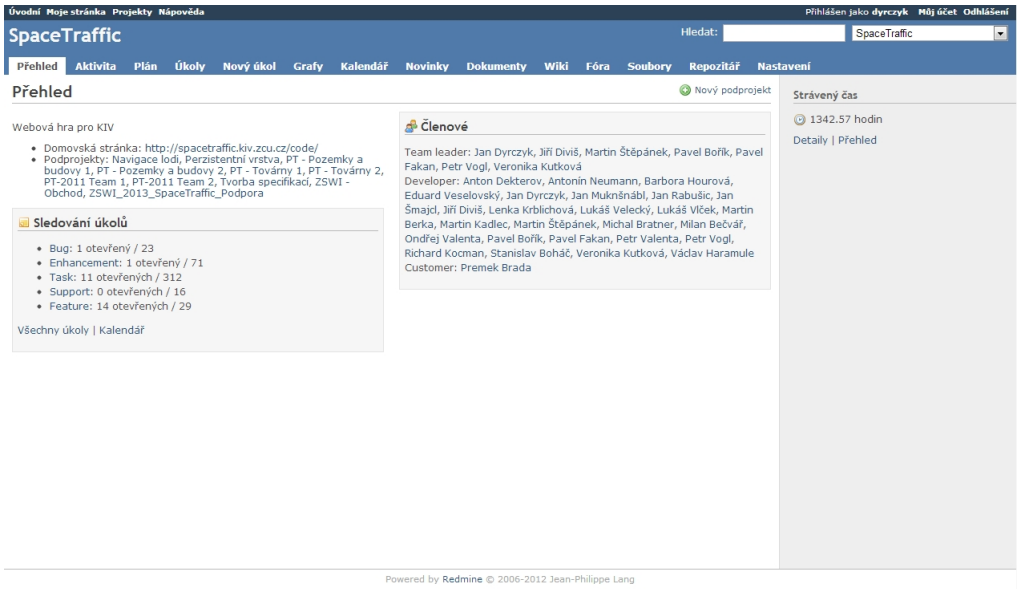
2.4.1 Redmine

Pro podporu řízení projektu byl využíván nástroj Redmine, provozovaný a spravovaný KIV. Redmine je webová aplikace pro flexibilní řízení softwarových projektů, která je využívaná studenty KIV při řešení jejich týmových semestrálních prací. Na obrázku 2.4.1 je náhled aplikace s přehledem projektu, kde jsou dostupné podprojekty, seznamy úkolů a členů.

Tento nástroj umožňuje:

- správu více projektů v hierarchickém uspořádání,
- flexibilní správu přístupu uživatelů podle jejich rolí,
- flexibilní systém sledování úkolů (vytvoření, naplánování, členění do logických hierarchii, sledování a ukončení úkolu),
- vytvářet Ganttovy diagramy,

- kalendář zobrazující naplánované úkoly,
- oznamování událostí emailem,
- prostor pro novinky, dokumentaci a projektové soubory,
- wiki a fórum pro každý projekt,
- sledování času stráveného na daném úkolu (projektu) daným uživatelem,
- integraci verzovacích nástrojů
- a podporu více jazyků (i češtiny)[5].



Obrázek 2.4.1: Screenshot aplikace Redmine - přehled projektu.

2.5 Cíle projektu pro ak. rok 2012/2013

Pro akademický rok 2012/2013 bylo hlavním cílem dokončit první hratelnou verzi hry, nasazení demoverze na server a rozšíření povědomí o projektu. Dále

zhodnocení dosavadní metodiky vývoje, navržení případných úprav a navržené vylepšení prakticky ověřit. Prakticky ověřit použití TFS pro podporu vývoje projektu. Zajistit kvalitu zdrojových kódů pomocí jednotkových a funkčních testů. Dostupnou funkčnost ověřit herními testy a testy použitelnosti. a připravit plány pro beta testy celé aplikace. Podrobný seznam cílů je přiložen v příloze B Cíle pro ak. rok 2012/2013. Splnění cílů projektu je zhodnoceno v závěru práce.

3 Zhodnocení zkušeností dosavadní metodiky vývoje, plánování a řízení projektu

Tato kapitola popisuje metodiku vývoje projektu navrženou Petrem Voglem [3], její praktické použití a zhodnocení. Dále jsou zde uvedeny zkušenosti vedoucích projektu ze zimního semestru s vývojem projektu a s prací s Redmine.

3.1 Metodika vývoje projektu

Metodika vývoje projektu vzešla s ohledem na specifické univerzitní prostředí ze spojení několika známých metodik. Postupy vývoje byly inspirovány metodikou Scrum, ze které byl využit backlog - zásobník pro scénáře, funkční vlastnosti a úkoly, které mají být v budoucnu realizovány. a také daily scrum schůzky, jen s ohledem na prostředí byla jejich perioda změněna na čtrnáct dní. z metodiky Feature Driven Development byla využita koncepce podle užitných vlastností, dílčí vývojové týmy tak byly zodpovědné za jimi vytvořené přírůstky funkcionality a jejich výsledky vykazovaly viditelný pokrok.

Pro požadavky speciálního univerzitního prostředí byl zvolen iterativní vývoj, který je schopen flexibilně reagovat na měnící se prostředí, změny studentů, změny vedení a s tím spojené změny v plánování nových funkčností. Iterativní přístup byl zvolen s time-boxovanými iteracemi[1] trvající jeden měsíc.

Akademický rok je rozdělen na dva semestry. v každém semestru jsou do projektu vázáni jiní studenti, v rámci svých semestrálních prací, kteří na konci semestru odevzdávají své řešení ke kontrole a ohodnocení. Na konci každého semestru je přirozený milník ve vývoji, kdy je možné sledovat nový přírůstek hry, který je otestovaný a zdokumentovaný.

[1] Angl. time-boxed iterations - na začátku projektu je stanovena fixní délka iterace a ta je po dobu realizace projektu stejná pro všechny iterace.

Obrázek 3.1.1: Fáze vývojového cyklu během ak. roku podle Petra Vogla[3].

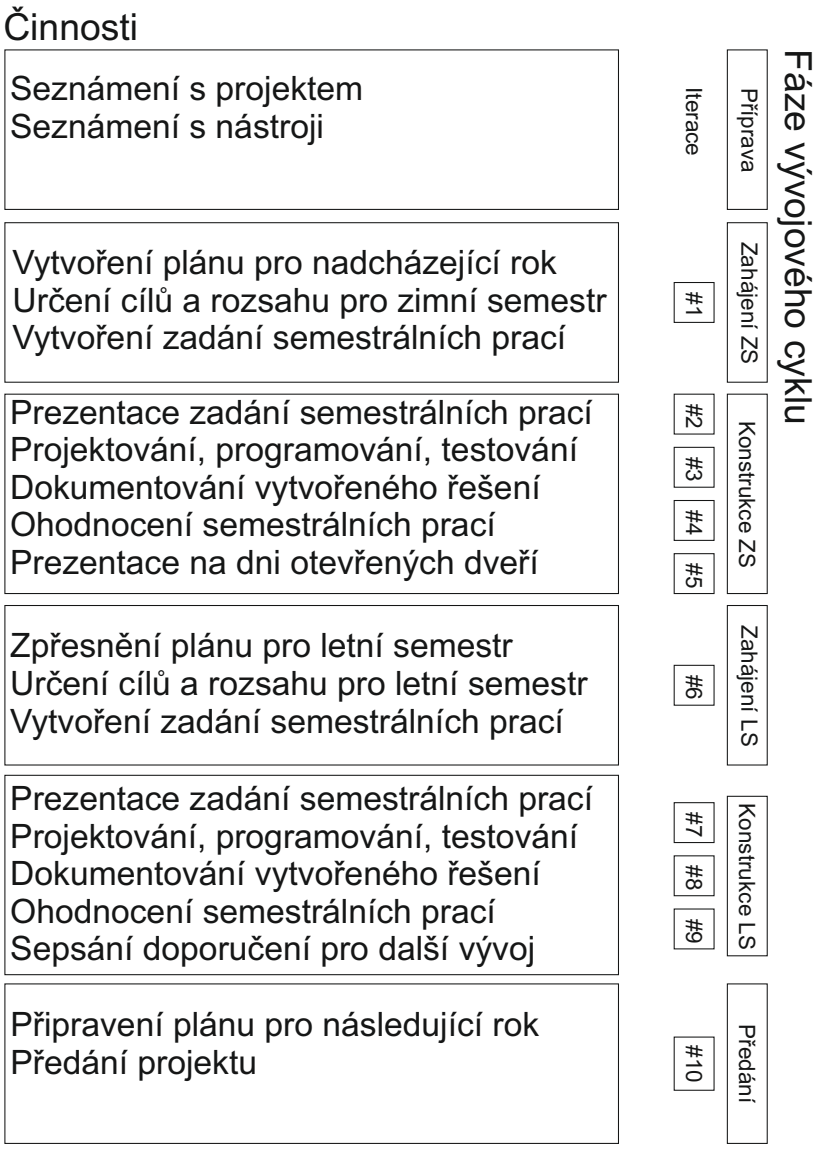
Celý akademický rok je rozdělen do několika fází podle obrázku 3.1.1, každá fáze je striktně vymezena počtem iterací. Vykonávané činnosti v průběhu iterací se liší podle fáze, ve které se projekt nachází. v následujícím seznamu jsou jednotlivé fáze popsány a obrázek 3.1.2 ukazuje přehled vykonávaných činností v jednotlivých fázích.

- **Příprava** – probíhá již od předání projektu předchozím vedením. Slouží především k seznámení nového vedení s aktuálním stavem projektu, zjištění cílů a účelu hry, nastudování způsobu řízení projektu a prozkoumání používaných nástrojů a služeb.

- **Zahájení ZS** (zimního semestru) – fáze probíhá jednu iteraci v měsíci září. Hlavní náplní je určení plánu pro nadcházející akademický rok, stanovení cílů pro zimní semestr a vytvoření zadání semestrálních prací pro splnění stanovených cílů.

- **Konstrukce ZS** – tato fáze probíhá následující čtyři iterace, jejím účelem je prezentace vytvořených zadání semestrálních prací na příslušných předmětech, složení vývojových týmů. Řízení návrhu, implementace, testování a dokumentování plánované funkcionality. Kontrola a ohodnocení výsledků dílčích týmů. Prezentace projektu na dni otevřených dveří Fakulty aplikovaných věd.

- **Zahájení LS** (letního semestru) – fáze probíhá jednu iteraci v měsíci únoru a odpovídá fázi Zahájení ZS, jen se změnou vybíraných předmětů. v letním semestru se obecně mohou vytvářet složitější zadání semestrálních prací pro větší týmy, které se již řídí sami a spolupráce probíhá formou zakázky, kdy je pro celý tým zadán úkol a termín jeho



Obrázek 3.1.2: Činnosti prováděné během vývojového cyklu podle Petra Vogla [3].

splnění. Hlavní náplní této fáze je zpřesnění plánu projektu, stanovení cílů pro letní semestr a vytvoření zadání semestrálních prací pro splnění stanovených cílů.

- **Konstrukce LS** – fáze probíhá následující tři iterace a odpovídá fázi ze zimního semestru, jejím účelem je prezentace vytvořených zadání se-

mestrálních prací na příslušných předmětech, složení vývojových týmů. Řízení návrhu, implementace, testování a dokumentování plánované funkcionality. Kontrola a ohodnocení výsledků dílčích týmů. Sepsání doporučení pro další vývoj.

- **Předání** – této fázi odpovídá jedna iterace v měsíci červnu. Jejím účelem je předání projektu nastupujícímu vedení, které povede projekt následující akademický rok.

3.2 Praktické použití metodiky vývoje

Navržená metodika byla použita v dané formě pro zimním semestr a její praktické použití je popsané v následující části. v letním semestru byly provedeny změny plánování iterací a v jednom týmu použit Team Foundation Server pro podporu plánování a řízení projektu viz kapitola 6.

Plánování iterací projektu proběhlo podle obrázku 3.1.1. v **předávací fázi** v měsíci červnu došlo k předání projektu od stávajícího vedení směrem k vedení novému. Petr Vogl a Martin Štěpánek předávali projekt Pavlu Boříkovi a autorovi této práce Janu Dyrczykovi, byla nastíněna vize a cíl, popsané používané nástroje, předána přístupová jména a hesla k používaným nástrojům (službám). Byla vysvětlena architektura a používané technologie. Popsán současný stav projektu a plán na následující rok. v neposlední řadě byla předána doporučení a zkušenosti z plánování a vedení projektu. Pro lepší pochopení zdrojových kódů byly připraveny Oborové projekty pro nové vedení.

V **přípravné fázi**, která trvala neintenzivní formou od července do srpna, nové vedení zpracovávalo své Oborové projekty, pomocí kterých lépe pochopilo zdrojové kódy a mělo jednodušší práci s navržením dalšího vývoje.

V **zahajovací fázi**, před začátkem semestru, byl vytvořen konkrétní plán projektu na celý následující akademický rok 2012/2013 a připraveny odpovídající zadání semestrálních prací. Jednotlivá zadání jsou přiložena v příloze E Zadání pro předměty v ak. roce 2012/2013. Pro zimní semestr byla

připravena tři zadání pro předmět Programovací techniky (KIV/PT), jedno zadání pro předmět Formální jazyky a překladače (KIV/FJP), jedno zadání pro předmět Teoretická informatika (KIV/TI) a dvě zadání pro Projekt 5 (KIV/PRJ5) s následným pokračováním v bakalářské práci. Jednotlivá zadání byla prezentována po zahájení semestru na prvních hodinách tížených předmětů, jednalo se především o cvičení z předmětu Programovací techniky.

Ve **fázi konstrukce** byly složeny vývojové týmy. Podařilo se vytvořit čtyři týmy pracující vždy paralelně na jednom zadání z předmětu Programovací techniky. Ostatní zadání byla obsazena vždy po jednom studentovi. Výsledný počet studentů pracujících na projektu v zimním semestru byl 11, plus dva členové ve vedení. Jednotlivé týmy vytvořili návrh, implementaci, testování a dokumentaci své funkcionality. Fáze byla rozdělena na čtyři iterace, kde v první bylo zahájení dílčích projektů, druhá a třetí obsahovala produktivní práci týmů. Koncem třetí iterace byly ukončeny jednotlivé projekty a ve čtvrté došlo ke zhodnocení výsledků a ke spojení projektů do hlavní vývojové větve (trunku). v měsíci lednu byly výsledky práce studentů prezentovány na dni otevřených dveří.

3.3 Zhodnocení dosavadní metodiky vývoje

Následuje kritické zhodnocení použití dosavadní metodiky vývoje v závislosti na získaných zkušenostech vedoucími projektu v zimním semestru.

Specifické požadavky, scénáře, funkční vlastnosti a úkoly byly ukládány do backlogu. z jednotlivých položek v backlogu vznikaly v průběhu plánování projektu úkoly, které byly přiřazovány do iterací a svým řešitelům. Využití backlogu pro uložení zásobníků požadavků bylo tím nejlepším řešením.

Vývojové týmy se scházely se zadavateli každých čtrnáct dní na pravidelných schůzkách, vždy se musel dostavit celý tým, který prezentoval svůj postup. Již v průběhu druhé iterace byl časový odstup schůzek uznán za příliš dlouhý a počínaje třetí iterací se perioda schůzek zkrátila na jeden týden. Schůzky s řešiteli jsou nepostradatelné a musí se vykonávat co možná

nejčastěji, v univerzitním prostředí dostačuje perioda jedou týdně.

Jednotlivé týmy pracovaly na svých zadáních, která přinášela do projektu viditelné přírůstky funkcionality, za kterou byly jednotlivé týmy zodpovědné. Tento postup byl hodnocen velmi kladně, týmy viděly svůj reálný přínos pro projekt a byly za něj odpovědny. Všechny nalezené chyby opravoval vždy řešitelský tým dané chybné oblasti, ostatní řešitelé tuto chybu jen nahlásili.

Iterativní vývoj je jasnou volbou, jen požití time-boxovaných iterací trvajících jeden měsíc je velmi nepraktické. Daleko lepší volbou je flexibilní nastavení délky iterací při plánování projektu podle časové náročnosti daných plánovaných úkolů pro iteraci viz kapitola 5.

Činnosti ve fázi konstrukce jsou dvojího druhu. Ve druhé a třetí iteraci pracují jednotlivé týmy na svých semestrálních pracích. Ve čtvrté a páté iteraci dochází k ohodnocení semestrálních prací, integraci výsledků do hlavní vývojové větve, přípravě na den otevřených dveří a k samotné prezentaci projektu na dni otevřených dveří. Jelikož se tyto činnosti značně liší bude fáze konstrukce rozdělena viz kapitola 5.

3.4 Zkušenosti vedoucích projektu

Následuje popis zkušeností vedoucích z vedení projektu v zimním semestru s duplicitním zadáním, rolí vedoucích, ukončením podprojektů a neúspěšným ukončením podprojektů.

3.4.1 Duplicitní zadání

Vytvoření čtyř paralelních týmů v sobě neslo nutnost vyhodnocení lepšího řešení a to přidat do hlavní vývojové větve. v jednom případě bylo jednoduše vybráno kvalitnější vypracování a to použito. v druhém případě byla zvolena možnost spojení obou řešení, kdy z jednoho řešení bylo ponecháno grafické zpracování a z druhého řídící logika. Následné spojení se ukázalo jako velmi časově náročné a pro další vývoj tento přístup vedení nemůže doporučit.

Zadání paralelního podprojektu v sobě nese nepříjemnost v podání vytvoření dvou vývojových větví, které se během zimního semestru mohou velmi

odchýlit od hlavní větve a následné spojení může být velmi bolestivé. Pro další vývoj je doporučeno volit spíše jednodušší zadání na implementaci, především pro studenty druhých ročníků, důsledně dodržovat práci s repositářem a pracovat přímo v hlavní vývojové větvi.

3.4.2 Role vedoucích projektu

Vedoucí projektu v zimním semestru vystupovali jako týmoví vedoucí, kde plánovali a řídili jednotlivé dílčí činnosti v průběhu semestru ve všech týmech. Plánovali týmové schůzky, kde se řešilo, co jednotlivý člen týmu udělal, co bude dělat v dalším týdnu a jestli má nějaké problémy. Tyto schůzky jsou ve studentském prostředí naprostou nutností, především s přihlédnutím na studentský syndrom, kdy studenti vše odkládají na poslední chvíli. Pomocí týdenních schůzek týmy opravdu pracují celý semestr, jelikož jejich dílčí mezní termíny jsou s každou naplánovanou schůzkou a je vždy nutné prokázat svojí činnost.

3.4.3 Ukončení podprojektů

Jak se ukázalo, není dobré stanovovat mezní termín projektu na konec semestru, jednak jsou studenti vytíženi ostatními povinnostmi, ale především již není možnost termín posunout a hrozí neúspěšné či nekvalitní ukončení projektu. Je dobré mezní termín posunout o týden až dva, následně termín prodloužit a získat kvalitní zpracování.

Na konci semestru je velmi vhodné jako součást dokumentace požadovat od vývojářů zpětnou vazbu, kde studenti popíšou svůj pohled na projekt, své případné nápady na vylepšení, ať už z pohledu samotné hry nebo plánování a řízení projektu. Tyto informace jsou velmi cenné pro další rozvoj projektu a samotných vedoucích. Po úspěšném dokončení podprojektu je velmi výhodné zůstat s řešiteli v kontaktu a zapojit je do vývoje Space Traffic projektu i v dalších semestrech, studenti jsou již s projektem seznámeni a odpadá přípravná fáze seznamování s projektem.

3.4.4 Neúspěšné podprojekty

V zimním semestru došlo bohužel k několika neúspěšným podprojektům. Hlavním problémem se jevilo jednak nedodržování pravidelných schůzek, kdy vedoucí nemohou přesně sledovat průběh práce a ke konci semestru může být již pozdě na případnou korekci prací. Jako další zásadní problém je špatná komunikace ze strany studentů, kteří i přes snahu vedoucích vůbec nekomunikují, nebo komunikují jen sporadicky. Bez komunikace v projektu se může velmi snadno stát, že studenti špatně pochopí zadání a na konci semestru odevzdávají chybné vypracování.

3.5 Zkušenosti z praktického použití Redmine

Pro vedoucí projektu byla práce s Redmine velmi přínosná a ulehčovala jim celý proces plánování a řízení projektu. Pro každý tým byl vytvořen podřízený projekt, který si kladl za úkol splnění jednoho ze stanovených cílů pro daný semestr. Podřízený projekt byl rozdělen na dvě produktivní iterace, kdy vždy na konci iterace docházelo ke zhodnocení výsledků práce. Do každé iterace byly přidány úkoly směřující tým k cíli, kdy úkoly byly plánovány spíše s pesimistickým odhadem časové náročnosti. v zimním semestru byly úkoly zadávány vždy týmu jako celku a nebyli rozlišováni jednotliví členové, toto rozdělení bylo ponecháno v rukách samotných řešitelů. Při řízení projektu vedoucí kontrolovali průběh práce pomocí výpisu úkolů pro danou iteraci a jejich stavu plnění. Správné konvergování k cíli bylo vidět také na Ganttově diagramu.

Pro jednotlivé členy týmu poskytoval Redmine přehled o jejich aktuální činnosti, měli možnost vidět všechny úkoly, které museli splnit do konce iterace. Viděli jejich časové odhady, a tak si mohli udělat svůj vnitřní rozvrh práce. Přesně věděli jak si v daný okamžik stojí a jak se jejich projekt ubírá k cíli. Pro správnou funkčnost nástroje bylo samozřejmě důležité k práci přistupovat důsledně a vyplňovat jednotlivé úkoly. Studenti by měli ke každém commitu práce na SVN uvést do komentáře číslo úkolu a u daného úkolu změnit procentuální míru splnění práce. Po dokončení úkolu ho nezapomenout

potvrdit jako dokončený, aby vedení bylo správě informováno o dokončení úkolu a mohlo případně danou skutečnost ověřit.

Při práci studentů na projektu se ukazovala jejich nezkušenost s prací v týmu a s používáním nástroje pro správu projektu. Bylo nutné je často upozorňovat na korektní vyplňování přidělených úkolů a i přesto docházelo k neúplnému či k chybnému vyplňování. Ze získaných údajů je vidět rozdílná úroveň studentů. Vedoucí nemohou přesně dělat odhady časů, když studenty neznají a přichází s nimi do kontaktu až v době vytváření odhadů. Proto je na místě dělat odhady spíše pesimistické a nechávat pro studenty více času na práci, případně jim připravit další rozšiřující možnosti v zadání, nad rámec jejich standardního plnění.

Z údajů získaných z Redmine na obrázku 3.5.1 jasně vyplývá, že tým *Pozemky a budovy 1*, pro který byl udělán odhad náročnosti práce na 32,5 hodiny, byl zkušenější a zvládl svoji práci již za 21,7 hodin. Oproti tomu tým *Továrny 1*, který měl odhadovaný čas náročnosti na 27 hodin tento čas značně přesáhl až na 43,4 hodin, jak je vidět na obrázku 3.5.2. Tento rozdíl byl vytvořen jednak menšími zkušenostmi týmu, ale také objevením složitější implementace databázové části a nutnosti důsledného testování. Tým *Pozemky a budovy 2* bohužel svůj projekt nedokončil a po 22 strávených hodinách byl ukončen. Tým *Továrny 2* nebyl schopen přes důsledné apelování údaje do nástroje doplnit a bohužel od něj nebyly získány žádné odpracované časové údaje. Ostatní řešitelé pracovali bez použití nástroje Redmine. Do výsledků musíme samozřejmě zahrnout nepřesné zanášení časů studentů, ale i přesto nám získané údaje názorně ukazují náročnost odhadů pro neznáme studenty.

Úkol	2012	Celkem
Support #1490: Seznámení se s nástroji a projektem	1.00	1.00
Task #1491: Návrh GUI pro nákup pozemků a stavbu budov	1.00	1.00
Task #1492: Implementace pozemků na základnách	2.00	2.00
Task #1493: Implementace GUI pro nákup pozemků	2.00	2.00
Task #1494: Implementace objektu budova	0.50	0.50
Task #1495: Implementace rozšíření pozemků	2.00	2.00
Task #1496: Implementace zmenšení pozemků	3.00	3.00
Task #1497: Implementace výstavby budov na pozemku	3.00	3.00
Task #1498: Implementace demolice budov	1.20	1.20
Task #1499: Uložení pozemků a budov do DB	2.00	2.00
Task #1501: Dokumentace	3.00	3.00
Bug #1569: Úprava GUI	1.00	1.00
Celkem	**21.70**	**21.70**

Obrázek 3.5.1: Přehled stráveného času týmu Pozemky a budovy 1.

Úkol	2012	Celkem
Support #1516: Seznámení se s nástroji a projektem	2.00	2.00
Task #1518: Návrh schématu pro zboží	6.00	6.00
Task #1519: Návrh schématu pro továrny	3.00	3.00
Task #1520: Implementace načtení schématu pro zboží	2.00	2.00
Task #1521: Implementace načtení schématu pro továrny	5.50	5.50
Task #1522: Implementace objektu továrna	1.25	1.25
Task #1523: Implementace objektu sklad	0.30	0.30
Task #1524: Implementace přesunu zboží ze skladu do továrny	2.50	2.50
Task #1525: Implementace přesunu zboží z továrny do skladu	1.50	1.50
Task #1526: Implementace výroby zboží	4.20	4.20
Task #1527: Uložení stavu skladu do DB	0.20	0.20
Task #1528: Testování funkčnosti	11.00	11.00
Task #1529: Dokumentace	4.00	4.00
Celkem	**43.45**	**43.45**

Obrázek 3.5.2: Přehled stráveného času týmu Továrny 1.

4 Možnosti platformy TFS pro plánování a sledování postupu prací

Tato kapitola nastiňuje obecný úvod do nástrojů správy životního cyklu aplikace (ALM – Application Lifecycle Management), popisuje možnosti platformy TFS pro plánování a sledování postupu prací. Dále porovnává TFS a Redmine z pohledu plánování a sledování postupů prací a porovnává verzovací nástroje TFS a SVN pro použití v projektu Space Traffic.

4.1 Životní cyklus aplikace

Životní cyklus vývoje aplikace se skládá z pěti základních fází, které se v závislosti na zvolené vývojové metodice mohou různě prolínat.

- **Specifikace** – tato fáze obsahuje intenzivní komunikaci se zadavatelem, specifikují se požadavky na aplikaci, vytváří se plán projektu a stanovují se mezní termíny pro nasazení aplikace.
- **Návrh řešení** – v této fázi se stále intenzivně komunikuje se zadavatelem, do hry vstupují vývojářské týmy. Upravuje se specifikace aplikace a požadavky na aplikaci. Navrhuje se řešení jednotlivých požadavků, které je konzultováno se zadavatelem, a vytváří se odhady časové náročnosti požadavků. Upravuje se plán projektu podle odhadnuté náročnosti požadavků a mezní termín pro nasazení aplikace.
- **Implementace** – podle zvolené metodiky vývoje je prováděna implementace navrženého řešení vývojářskými týmy. Jednotlivé funkční přírůstky aplikace mohou být konzultovány se zadavatelem. Případné chyby v aplikaci jsou hlášeny, jsou navrhovaná řešení a probíhá oprava.
- **Testování** – tato fáze může být spojena již s předchozí implementační fází, kde si vývojáři sami píší jednotkové testy na svůj kód. v testovací fázi probíhají integrační testy celé aplikace a mohou být využívány automatické testy pro opakující se testovací případy. Rozsáhlejší aplikační

funkčnosti, např. grafické rozhraní, se můžou testovat manuálně. Nalezené chyby ze všech testů jsou hlášeny vývojářům, ti navrhují řešení a opravují je.

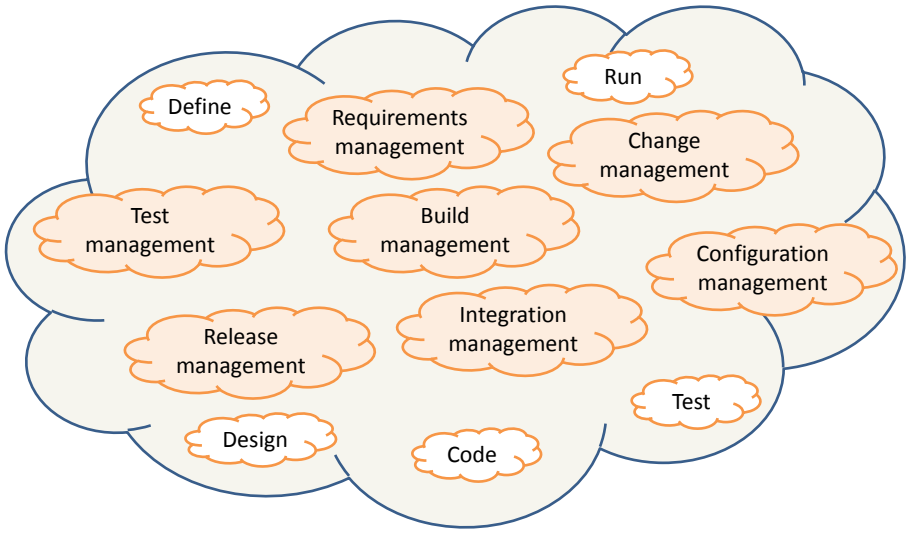
- **Nasazení** – může probíhat automaticky, kdy po commitování hotové implementace a po úspěšném průchodu testovací fází se může spustit automatické nasazení na cílový stroj, kde mohou proběhnout systémové testy. Fáze nasazení může probíhat také manuálně, kdy dojde k nasazení na cílový stroj manuální instalací.

4.2 ALM nástroje

ALM je spojení podnikového řízení a softwarové inženýrství pomocí nástrojů, které usnadňují a integrují činnosti z životního cyklu aplikace. Tyto činnosti zobrazuje Michael Hüttermann [6] na obrázku 4.2.1. Mezi činnosti z životního cyklu aplikace patří :

- správa uživatelů a jejich rolí,
- dokumentování – prostor pro uložení a sdílení specifikací, návrhů, dokumentací, manuálů a jiných potřebných dokumentů,
- správa požadavků – správa produktového backlogu,
- plánování a modelování vývoje – zanesení požadavků a termínů do kalendáře, požití vývojové metodologie,
- správa úkolů – schůzky týmu, správa týmového backlogu,
- správa týmů – práce v jedné či více vývojových větvích,
- správa a verzování zdrojových kódů,
- implementace – implementace přiřazených úkolů, ohlašování stavu zpracování úkolů, hlášení chyb ve vývoji,
- testování – testování aplikace, hlášení chyb vývojovému týmu,

- sledování postupů práce pro týmové vedoucí, projektové manažery a zadavatele
- a nasazení aplikace.



Obrázek 4.2.1: ALM spojuje vývojové disciplíny a fáze definice požadavků, designu, kódování, testování a provozu [6].

Každá role ve vývojovém cyklu má jiné požadavky na ALM nástroje:

- **Zadavatel** – naplnění projektového backlogu, specifikace akceptačních kritérií, stanovení mezních termínů, přehled o dokončení funkčností a možnost testovat již hotové funkčnosti.
- **Manažer** – přehled o aktuálním stavu projektu a průběhu prací, přehled o hotových, rozpracovaných nebo naplánovaných funkčnostech.
- **Týmový vedoucí** – přehled o aktuálním stavu svého týmu, přehled o přiřazených funkčnostech týmu a jejich aktuálním stavu a přehled o přiřazených úkolech daným členům týmu a jejich stav.

- **Vývojář** – přehled o přiřazených úkolech a termínech dokončení, možnost hlášení stavu svých úkolů.
- **Tester** – přehled o dokončených funkčnostech, které jsou připraveny k testování, možnost hlášení nalezených chyb.

Následuje seznam několika ALM nástrojů se stručnou charakteristikou.

Team Foundation Server od společnosti Microsoft je komerční nástroj nabízející zdarma cloudové řešení s plnou licencí pro pět uživatelů. s neomezeným počtem projektů, správcem zdrojových kódů, správou úkolů, nástroji pro agilní plánování, správou uživatelů, automatickým sestavováním a testováním projektu [8]. Podrobný popis nástroje je uveden později v této kapitole.

JIRA od společnosti Atlassian nabízí zdarma licenci pro veřejné a akademické projekty. Nástroj obsahuje správu úkolů, agilní vývoj (Scrum a agilní tabule), více projektů, správce zdrojových kódů (Git, SVN nebo CVS), správu sestavení a nasazení aplikace. Dovoluje import dat z nástrojů Redmine, Bugzilla, Microsoft Excel a dalších. Zdrojové kódy nástroje jsou otevřené a tím je možné nástroj libovolně upravovat [9].

Assembla nabízí zdarma licenci pro veřejné projekty nebo pro týmy do tří členů s jedním projektem, jedním správcem zdrojových kódů, s prostorem pro data 500MB a emailovou podporou. Nástroj nabízí správu úkolů, správce zdrojových kódů (Git, SVN nebo Subversion), sledování postupu prací a hlášení chyb ve vývoji, sdílení souborů, nástroje pro sestavení a nasazení aplikace [10].

VersionOne poskytuje zdarma licenci pro deset uživatelů se základními funkcemi a jedním projektem. Nástroj dovoluje plánovat vývoj, spravovat úkoly, využívat agilní tabule a grafy vývoje práce [11].

Rally software nabízí zdarma licenci pro deset uživatelů v jednom týmu. Poskytuje správu požadavků, agilní plánování, hlášení chyb, správu úkolů a agilní tabule [12].

Mingle od společnosti ThroughtWorks poskytuje zdarma licenci pro pět uživatelů. Nástroj podporuje agilní vývoj, správu požadavků, úkolů a chyb, agilní tabule a grafy [13].

Největší ALM nástroje v komerční sféře nabízí společnosti HP, IBM a SAP. Tyto nástroje pokrývají uceleně celý ALM. Pro univerzitní prostředí se jedná o zbytečně velké a drahé nástroje.

4.3 Kolekce a týmové projekty v TFS

Plánování v TFS většinou začíná navržením infrastruktury (definováním fyzických a logických zařízení), naplánováním struktury týmových projektů (Team Project) a kolekcí (Team Project Collection) pro dosažení efektivního a škálovatelného ALM systému pro všechny zainteresované strany.

Každý TFS může obsahovat jednu nebo více kolekcí a každá z těchto kolekcí obsahuje jeden nebo více týmových projektů. Kolekce je základní jednotka pro zálohování a obnovu TFS. (Z pohledu zálohování a obnovy je kolekce podobná SharePoint Site Collection.) Týmový projekt nemůže být samostatně zálohován a obnoven, pro účely zálohování a obnovení slouží jen kolekce. Pro použití granulární zálohy a obnovení musí být týmový projekt uložen do oddělené kolekce.

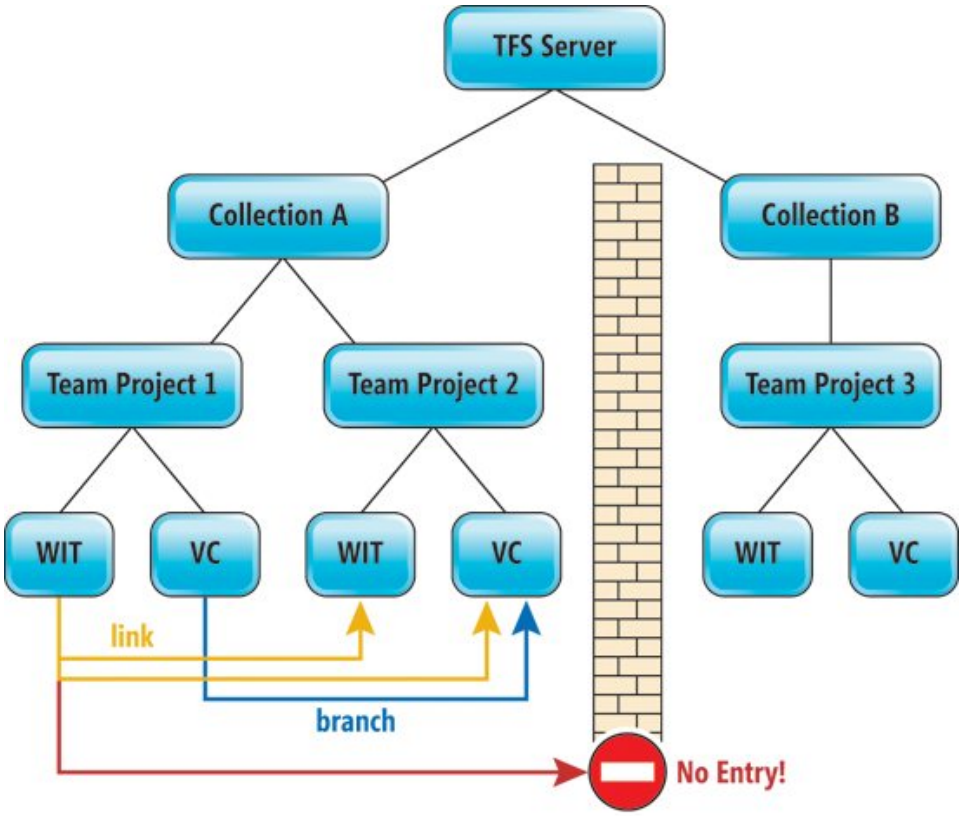
Škálovatelnost TFS je podpořena schopností vyrovnávat zatížení kolekcí přes fyzické SQL servery a virtuální SQL server instance, čímž TFS přebírá výhody škálovatelnosti a vyrovnávání zatížení infrastruktury, která je typická pro databázové prostředí. Při možnosti vyrovnávání zatížení přes SQL servery TFS těží z rozdělení týmových projektů přes více kolekcí.
Týmový projekt je kontejner artefaktů obsahující pracovní položky (WIT – Work Item Type), zdrojové kódy ve správci zdrojů (VC – Version Control) organizované do složek a vývojových větví (branch), jednu nebo více Visual Studio Solution a další konfigurační soubory.

Visual Studio Solution obsahuje jeden nebo více Visual Studio projektů, definuje projektové závislosti a proces sestavení aplikace.

Týmový projekt nesmí být zaměňovaný za Visual Studio projekt, který obsahuje popis sestavení a konfigurační nastavení pro generování spustitelného projektu.

Jednotlivé týmové projekty v jedné kolekci jsou navzájem viditelné a mo-

hou mezi sebou sdílet své pracovní položky a zdrojové kódy, oproti tomu projekty v jiných kolekcích jsou od sebe navzájem izolované, jak je vidět na obrázku 4.3.1. Jestliže dva týmy mezi sebou chtějí sdílet pracovní položky, musí být ve stejné kolekci.



Obrázek 4.3.1: Sdílení a izolace v kolekcích [7].

4.3.1 Tvorba kolekce a týmového projektu v TFS

Vytvoření nové kolekce probíhá v *Team Foundation Server Administration Console* přímo na serveru, kde běží TFS. Pro práci s kolekcemi slouží formulář *"název-serveru"/Application Tier/Team Project Collections* a pro vytvoření nové kolekce odkaz *Create Collection*, který otevře průvodce vytvořením kolekce. v průvodci se nastavuje jméno, popis a SQL server pro uložení dat. Možnost vybrání SQL serveru umožňuje vyrovnávat zátěž TFS. Po ověření dostupnosti SQL serveru je kolekce vytvořena.

Nový týmový projekt je vytvářen podle procesní šablony obsahující pracovní postupy, pracovní položky, zprávy a dotazy. Procesní šablona může být vybraná jedna z připravených: Microsoft Visual Studio Scrum, MSF for Agile Software Develoment, MFS for CMMI Process Improvement, nebo může být vytvořena vlastní úpravou stávající šablony.

Vytvoření nového týmového projektu probíhá přímo ve Visual Studiu (*File/New/Team project...*), kde je po připojení na TFS a vybrání kolekce spuštěn průvodce vytvořením týmového projektu. v průvodci se nastavuje jméno týmového projektu, popis práce, vybírá se procesní šablona a nastavuje správce zdrojů (Source Control). Správce zdrojů může obsahovat buď nový prázdný prostor, nebo novou vývojovou větev (branch) ze stávajícího správce zdrojů ve stejné kolekci.

Do týmového projektu lze dále přidávat Visual Studio Solution a Visual Studio projekty.

4.4 Licence TFS

TFS je komerční nástroj distribuovaný ve třech licencích:

- **Limitovaná** – umožňuje pracovat s iteracemi, backlogem, úkoly (task) a testovacími případy (test case),
- **Standardní** – navíc oproti limitované licenci podporuje standardní funkce a agilní tabuli,
- **Plná** – navíc oproti standardní licenci podporuje nástroje pro plánování sprintů a správu požadavků.

Pro univerzitní účely je TFS distribuovaný přes DreamSpark (dříve MSDN AA – the Microsoft Developer Network Academic Alliance) v limitované licenci. Pro účely testování TFS v této práci byla použita verze *Microsoft Visual Studio Team Foundation Server Express 2012 with Update 2 32/64-bit (English)*. TFS je dostupný také v cloudovém řešení, kde je k dispozici zdarma plná licence pro pětičlenný tým [8]. Toto omezení znemožňuje použití

cloudového řešení pro vývoj v projektu Space Traffic, kde se počítá s týmem větším než pět členů.

4.5 Možnosti TFS limitované licence

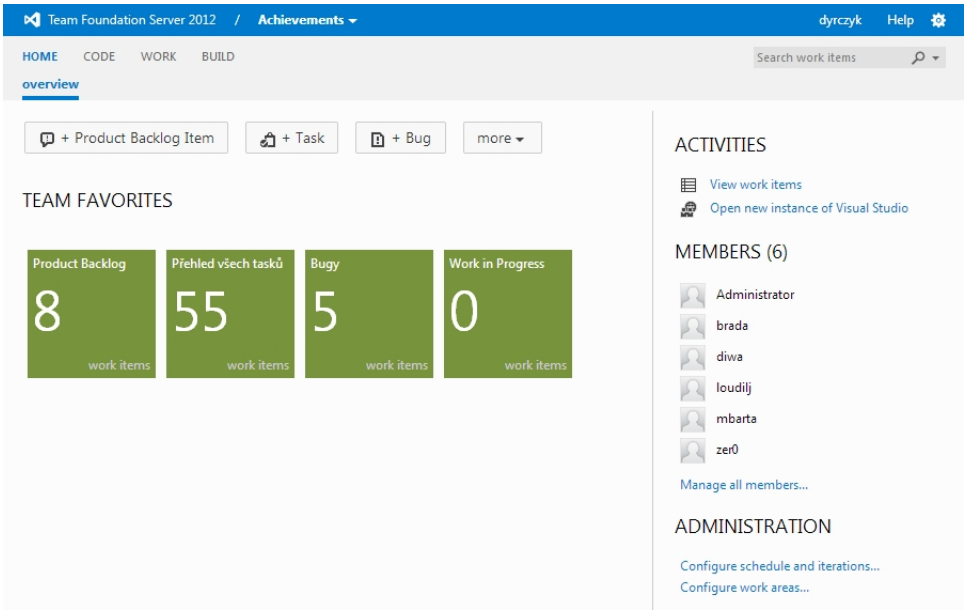
Limitovaná licence nabízí uživatelům jen základní funkčnosti pro plánování a sledování postupů. Uživatelé mohou vytvářet kolekce a přiřazovat je k příslušným SQL serverům. Vytvářet týmové projekty a zařazovat je do kolekcí. Dále je přístupná správa uživatelů a jejich rolí. v neposlední řadě plánování iterací a správa úkolů. Uživatelé mohou pro práci s TFS využívat webové rozhraní, Visual Studio nebo Eclipse.

4.5.1 Možnosti webového rozhraní

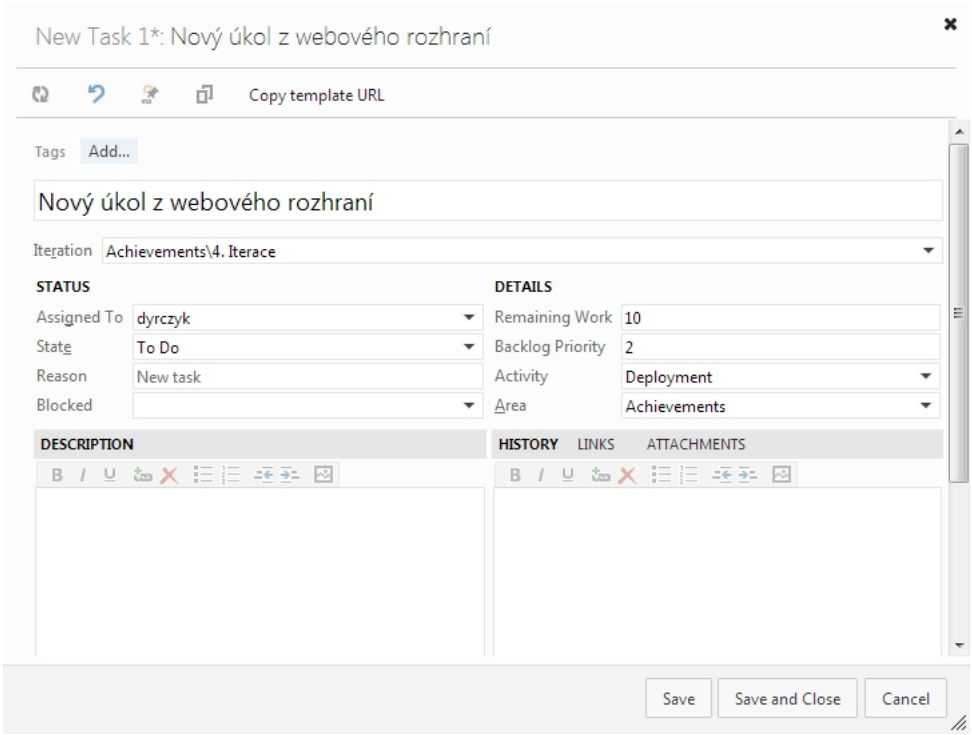
Webové rozhraní TFS je nastavitelné z TFS konzole a je obsluhované webovým serverem IIS na nasazeném serveru. Pro práci s webovým rozhraním lze použít libovolný webový prohlížeč. Internet Explorer provádí automatickou autentizaci přihlášeného uživatel do Windows. Pro ostatní prohlížeče je nutné autentizaci provést zadáním jména a hesla TFS uživatele.

Domovská stránka projektu umožňuje přidávat nové položky do Backlogu, přidávat nové úkoly (Task), chyby (Bug), překážky ve vývoji (Impediment) a testovací případy (Test Case). Zobrazuje týmové oblíbené záložky v podobě interaktivních dlaždic s informacemi z TFS, jako jsou např. počet položek v Backlogu, přehled všech úkolů, chyby nebo nedokončené úkoly, jak je vidět na obrázku 4.5.1. Položky lze libovolně vytvářet pomocí dotazovacího jazyka a přidávat na domovskou stránku. Pomocí jednotlivých položek lze zobrazit detailní výpis.

Dále lze zobrazit úkoly přiřazené přihlášenému uživateli a spustit novou instance Visual Studia. v neposlední řadě domovská stránka zobrazuje výpis členů týmu s možností jejich správy, kde lze uživatele přidávat a odebírat z týmu. Dále možnost plánování iterací, kde lze podle zvolené šablony týmového projektu definovat iterace a sprinty s počátečním a koncovým termínem. a možnost nastavení oblastí, kde lze vytvořit několik oblastní práce,



Obrázek 4.5.1: Domovská stránka projektu v TFS.



Obrázek 4.5.2: Formulář pro přidání nového úkolu z webového rozhraní.

kde každá oblast má svůj vlastní Backlog a seznam úkolů.

Přidání nové pracovní položky, např. úkolu, probíhá po kliknutí na příslušný odkaz, kdy se otevře formulář pro přidání nového úkolu, který je k vidění na obrázku 4.5.2. Formulář obsahuje vstupní pole pro název, přiřazení k příslušné iteraci, přiřazení svému řešiteli, pole pro stav úkolu, odhadovaný čas náročnosti, prioritu, typ aktivity (specifikace, návrh, vývoj, testování, dokumentování nebo nasazení) a oblast do které úkol spadá. Vstupní pole pro popis a historii úkolu umožňují základní formátování textu, vkládání odkazů, seznamů a obrázků. k úkolu může být přidán libovolný počet souborů jako příloha.

Jednotlivé pracovní položky mohou být strukturované do hierarchie pomocí přiřazovaných vazeb. Lze vytvářet závislé položky typu rodič-dítě nebo předchůdce-následovník. Položky se mohou vázat ke konkrétním změnám ve zdrojových kódech, mohou mít sdílené společné kroky, mohou se párovat s testy a být ve více verzích.

Další oblastí webového rozhraní je správa zdrojových kódů, kde jsou vidět zdrojové kódy, provedené commity s komentáři a možností zobrazení změn mezi verzemi.

Dále je zde sekce s úkoly, kde lze zobrazit pomocí dotazovacího jazyka informace z TFS. Vždy je vidět výsledek dotazu v podobě seznamu nalezených položek a detailní informace jednotlivých položek. s vybranými položka lze pracovat a upravovat je.

Poslední oblastí webového rozhraní je sestavovací oblast, kde je možné sledovat automatické sestavování pomocí TFS.

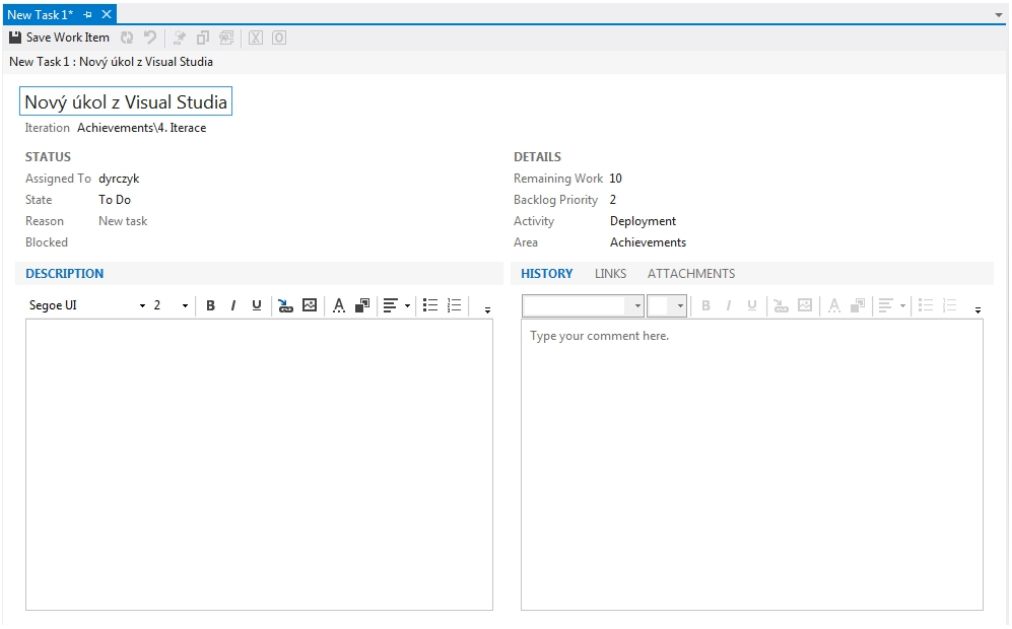
Ze standardního webového rozhraní se lze přepnout do kontrolního webového rozhraní, kde lze provádět základní kontrolní činnosti. Lze spravovat kolekce, nastavovat jejich bezpečnost a členství uživatelů. Dále spravovat týmové projekty, nastavovat bezpečnost, členství ve skupinách a vytvářet nové týmy.

4.5.2 Integrace TFS do Visual studia

TFS je spjatý s vývojovým prostředím Visual Studio a je do něj ve velké míře zaintegrován. Nástroje pro práci s TFS přes Visual Studio podporují stejné funkce jako webové rozhraní, jen jsou přizpůsobeny práci přímo ve Visual Studiu.

Pro práci s týmovými nástroji slouží ve Visual Studiu záložka *Team*, pomocí které se lze připojit do TFS. Při připojování se vybírá adresa serveru, kde je TFS nasazen, týmová projektová kolekce a týmový projekt. Při autentizaci na TFS lze využít lokální profil uživatele nebo jméno a heslo TFS uživatele zadat ručně. Po přihlášení do TFS je zobrazena úvodní obrazovka v Exploreru, která slouží jako rozchodník do dalších částí, jako jsou správa zdrojových kódů, pracovní úkoly, sestavovací sekce, odkaz na webové rozhraní a nastavení.

Formulář pro přidání nového úkolu z Visual Studia je vidět na obrázku 4.5.3, který ilustruje podobnost s webovým rozhraním.



Obrázek 4.5.3: Formulář pro přidání nového úkolu z Visual Studia.

Sekce zdrojových kódů spravuje lokální zdrojové kódy, kódy ve správci zdrojových kódů na TFS a jejich změnu. Při commitování upravených zdro-

jových kódů lze vybrat jaké změny se budou opravdu odesílat a jaké jsou uchovány pouze jako lokální kopie (neodesílají se). Ke každému commitu lze vložit komentář a ID úkolu, ke kterému se vztahuje.

V **sekci pracovních úkolů** je možnost zakládat nové položky v Backlogu, úkoly, chyby, překážky a testovací případy. Tato sekce také obsahuje možnost vytvoření nové položky pomocí nástroje Microsoft Excel. Na úvodní obrazovce je dále seznam týmových oblíbených položek, které zobrazují stejné informace jako ve webovém rozhraní jen s jiným designem. Sekce umožňuje zadávat dotazy na TFS a zobrazovat výsledky v podobě seznamů se zobrazením detailních informací, kde jsou jednotlivé položky editovatelné.

Sekce pro sestavování slouží k nastavení a sledování sestavování aplikace.

Poslední **sekce nastavení** slouží k nastavení TFS a obsahuje stejné možnosti jako **Kontrolní webové rozhraní**, tj. nastavení kolekcí a týmových projektů.

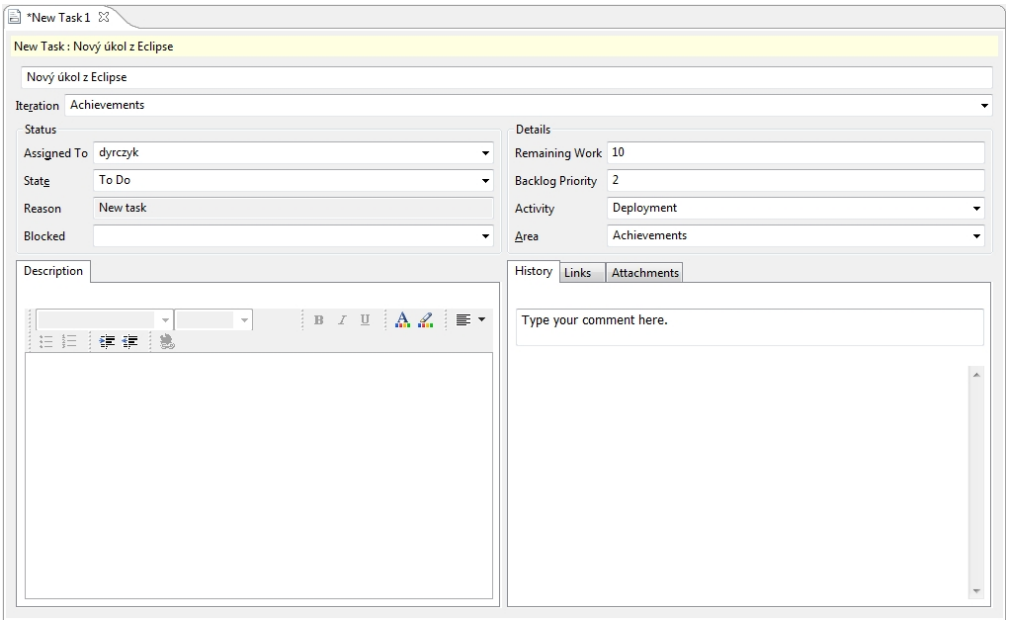
4.5.3 Integrace TFS do Eclipse – TFS modul pro Eclipse

Microsoft vydává oficiální TFS modul pro Eclipse[1], který přináší do Eclipse Team Explorer s napojením na TFS.

Připojení do TFS probíhá pomocí modulu stejně jako ve Visual Studiu. Následná práce v Eclipse je stejná jako v Team Exploreru ve Visual Studiu, jen s malými designovými odlišnostmi. Na domovské obrazovce Team Exploreru je opět rozchodník se sekcemi zdrojových kódů, pracovních položek, sestavování, webového přístupu a nastavení. Prováděné akce v sekcích jsou stejné jako ve Visual Studiu popsané v předchozí kapitole.

Formulář pro přidání nového úkolu z Eclipse je vidět na obrázku 4.5.4, který ilustruje podobnost s rozhraním ve Visual Studiu a webu.

[1]Dostupný z URL http://marketplace.eclipse.org/content/tfs-plug-eclipse



Obrázek 4.5.4: Formulář pro přidání nového úkolu z Eclipse.

4.6 TFS správce zdrojů

TFS obsahuje vlastního správce zdrojů pro správu a verzování zdrojových kódů. Správce poskytuje všechny obvyklé funkce verzovacích nástrojů: commitování změn na server, komentování commitů, spojování commitů s úkoly, uchovávání historie změn, spojování konfliktních změn mezi verzemi, stahování jednotlivých verzí ze serveru a vytváření vývojových větví. Správce je přístupný přímo ve Visual Studiu nebo s použitím modulu pro připojení TFS v Eclipse. Webové rozhraní umožňuje pouze náhled do správce zdrojů.

4.7 Porovnání možností TFS a Redmine v rámci vývoje, plánování a řízení projektů

Následuje porovnání TFS v limitované licenci s Redmine nasazenou na univerzitním serveru z pohledu vývojových etap a pohledu zúčastněných stran.

4.7.1 Porovnání z pohledu vývojových etap

Specifikace a návrh řešení – oba nástroje obsahují prostor pro uložení a sdílení souborů (specifikací, dokumentací či návodů). v obou nástrojích lze naplánovat celý projekt, rozdělit ho do jednotlivých iterací a sprintů. Iterace (sprinty) jsou časově omezeny, je nastaven počáteční a mezní termín. Specifikované funkčnosti lze v obou nástrojích nahrát do projektového backlogu. v obou nástrojích lze definovat jednotlivé úkoly s časovým odhadem náročnosti práce, přiřadit je řešiteli a nastavit vazbu s ostatními úkoly.

Implementace a testování – v obou nástrojích jsou úkoly přiřazeny vývojářům. Vývojáři vidí probíhající iteraci (sprint), její mezní termín a seznam svých úkolů v této iteraci. Redmine navíc obsahuje kalendář s vizualizací úkolů. Vývojáři mohou měnit stav průběhu práce svých úkolů, nastavovat, že úkol přijali, mají ho rozpracovaný nebo dokončený. Nastavovat reálný strávený čas na úkolu a v Redmine navíc nastavovat procentuální dokončení úkolu. Dané změny stavu mohou být ohlašovány pomocí emailu vedoucímu týmu. Hotovou práci mohou vývojáři commitovat do správce zdrojových kódů a svazovat se souvisejícími úkoly.

V Redmine je správce zdrojových kódů připojen externě a všechny činnosti se zdrojovými kódy, jako je verzovaní, commitování, checkoutování, branching a marging jsou prováděny externě. Redmine pouze obsahuje náhled do repozitáře a umožňuje svazovat commity s úkoly. TFS používá svého vlastního správce zdrojových kódů, při použití externího správce zdrojových kódu (např. SVN) TFS nepodporuje tohoto správce a není možné jednotlivé commity svazovat se souvisejícími úkoly.

Nasazení – v Redmine není fáze nasazení podporována, po dokončení iterace může být proveden tag v externím správci zdrojových kódů a aplikace může být manuálně čí s použitím externích nástrojů nasazena. TFS podporuje fázi nasazení, lze nakonfigurovat sestavovací stroj, který bude provádět automatické sestavení aplikace, následné nasazení probíhá opět pomocí externího nástroje nebo manuálně.

4.7.2 Porovnání z pohledu zúčastněných stran

Pro každou zúčastněnou stranu lze přiřadit v obou nástrojích uživatelskou roli. v Redmine jsou to role týmového vedoucího, vývojáře, zadavatele a mentora. Uživatel v TFS zastupuje roli podle nastavených oprávnění na serveru, na serveru je množství připravených rolí nebo lze vytvářet role vlastní.

Zadavatele – uložení specifických funkčností aplikace do backlogu a sdílení souborů je podporováno oběma nástroji. Možnost sledování postupů práce s ohledem na stav jednotlivých úkolů je v obou nástrojích. v Redmine se zadavatel navíc dozví procentuální splnění úkolů a může si zobrazit grafy průběhu práce. Pro zadavatele je nejzajímavější burndown graf.

Manažer a týmový vedoucí – přehled o aktuálním stavu průběhu prací je lépe zobrazeno v Redmine, kde manažer vidí procentuální míru splnění úkolů, strávený (zbývající) čas na úkolech a grafy mapující průběh práce. Oproti tomu v TFS manažer vidí jen stav úkolů v aktuální iteraci a jejich zbývající časovou náročnost. Týmový vedoucí má možnost zobrazení přiřazených úkolů pro jednotlivé členy v obou nástrojích.

Vývojář a tester – vývojář vidí své aktuální úkoly v dané iteraci a hlásí stav svých úkolů pro oba nástroje. v TFS jen označuje úkol za započatý a dokončený, datum se přiřazuje automaticky podle aktuálního data na serveru. v Redmine lze úkol označit za přijatý, započatý, dokončený, verifikovaný nebo ukončený. Míra dokončení práce lze definovat v Redmine procentuálně a v obou nástrojích lze nastavovat strávený čas. Vývojář nebo tester může v obou nástrojích vytvářet nové úkoly a hlásit nalezené chyby.

5 Úpravy postupů a nástrojů pro další vývoj

Kapitola popisuje upravenou metodiku vývoje podle zkušeností s použitím metodiky navržené Petrem Voglem z kapitoly 3 v zimním semestru a použití TFS pro podpoření plánování a sledování postupů.

5.1 Upravená metodika vývoje

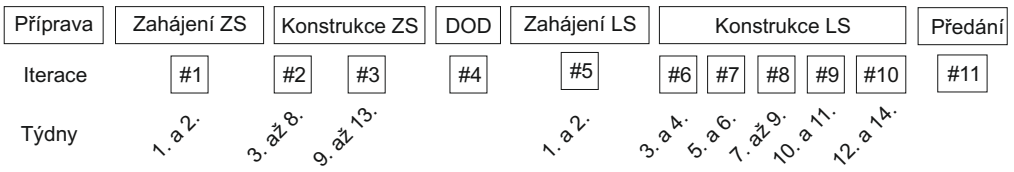
Došlo k upravení metodiky vývoje od Petra Vogla [3] a k větší flexibilitě, přesnému nastavení metodiky vývoje na speciální univerzitní prostředí. Jelikož vývoj probíhá především prostřednictvím semestrálních prací, metodika vývoje musí respektovat náplň předmětů, v rámci kterých jsou práce zpracovávané.

Z metodiky Scrum je zachován backlog pro uložení scénářů, funkčních vlastností a úkolů pro budoucí realizaci. a také daily scrum schůzky, u kterých je časová perioda změněna na jeden týden. z metodiky Feature Driven Development je použita koncepce podle užitých vlastností, kde dílčí týmy zodpovídají za své vytvořené funkční přírůstky, které vykazují viditelný pokrok v projektu.

Plánování projektu zůstává u iterativního vývoje, který dokáže flexibilně reagovat na měnící se univerzitní prostředí. Time-boxované iterace jsou zrušeny a přechází se na iterace s proměnou délkou trvání, kdy délky iterací jsou nastavovány vždy na začátku semestru v závislosti na plánovaných funkčnostech.

Nově plánované funkčnosti jsou rozkládané na jednotlivé úkoly, které jsou jednoduše zpracovatelné studenty v maximální časové náročnosti jednoho dne (8 hodin). Jednotlivé úkoly jsou kontrolovány a až po dokončení jednoho úkolu je studentům přiřazen úkol další. Tímto postupem se škáluje práce studentů, získává se kvalitní řešení jednotlivých úkolů a konečného řízeného celku.

Akademický rok je rozdělen na dva semestry. Na konci každého semestru je přirozený milník, kdy je vidět funkční přírůstek sestávající se z výsledků odevzdaných semestrálních prací spolupracujících studentů.



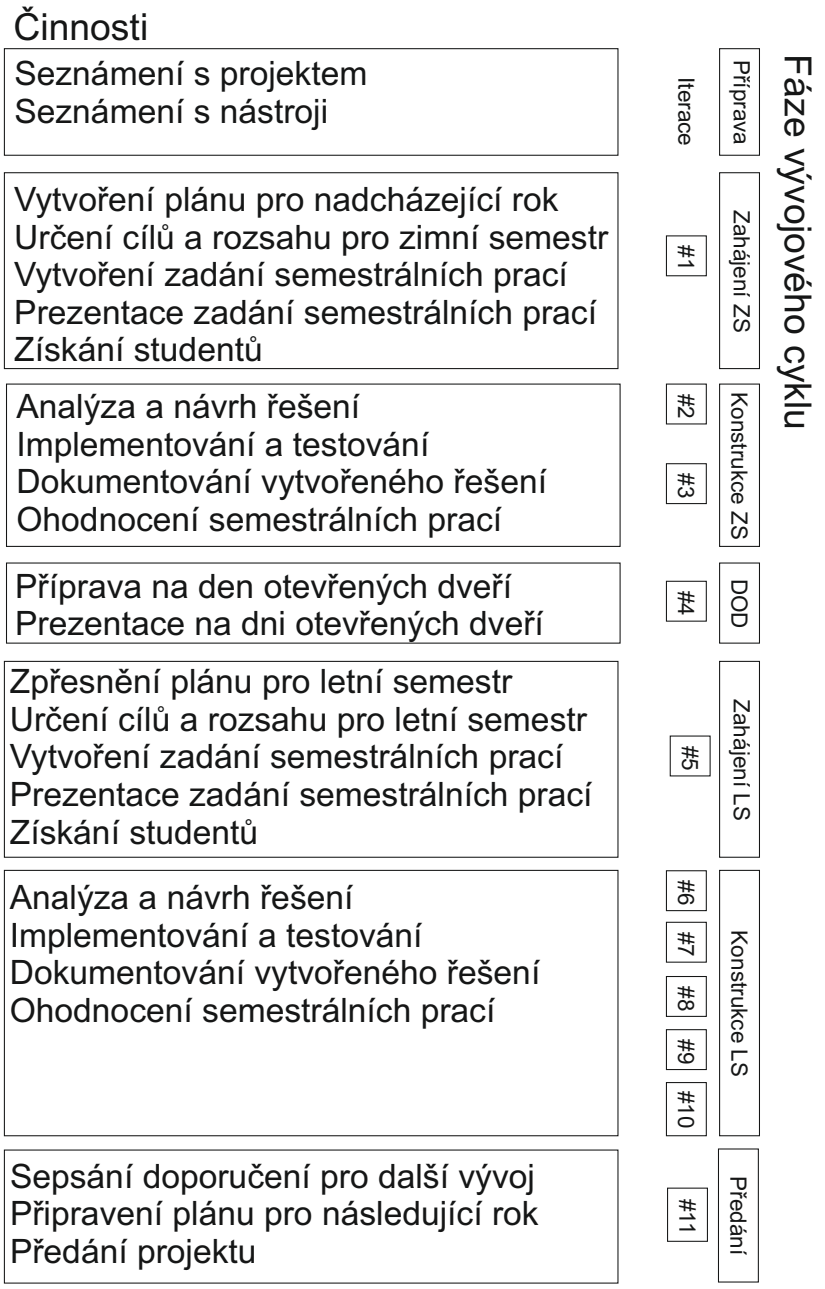
Obrázek 5.1.1: Fáze vývojového cyklu během ak. roku.

Celý akademický rok je rozdělen do několika fází podle obrázku 5.1.1, každá fáze je vymezena počtem iterací. Daný počet a délky iterací se mohou měnit v závislosti na plánované funkcionalitě a aktuální potřebě vývoje, jak je vidět ve fázi konstrukce ZS (zimní semestr) a konstrukce LS (letní semestr). v následujícím seznamu jsou jednotlivé fáze popsané a obrázek 5.1.2 zobrazuje přehled vykonávaných činnosti v jednotlivých fázích. Připravený plán metodiky vývoje pro projekt v akademickém roce 2012/2013 je v příloze C Plán ak. rok 2012/2013.

- **Příprava** – probíhá již od předání projektu předchozím vedením od června. Slouží především k seznámení nového vedení s aktuálním stavem projektu, zjištění cílů a účelu hry, nastudování způsobu řízení projektu a prozkoumání používaných nástrojů a služeb.

- **Zahájení ZS** – fáze probíhá jednu iteraci a navazuje na přípravnou fázi, ukončení iterace je plánováno na první dva týdny zimního semestru. Hlavní náplní je určení plánu pro nadcházející akademický rok, stanovení cílů pro zimní semestr, vytvoření zadání semestrálních prací pro splnění stanovených cílů, prezentace vytvořených zadání semestrálních prací na příslušných předmětech a získání studentů pro konstrukční fázi.

- **Konstrukce ZS** – tato fáze probíhá následující dvě iterace. Druhá iterace trvá po dobu šesti týdnů a třetí po dobu pěti týdnů. Jejím účelem je složení vývojových týmů. Řízení návrhu, implementace, testování



Obrázek 5.1.2: Činnosti prováděné během vývojového cyklu.

a dokumentování plánované funkcionality. Na závěr probíhá kontrola a ohodnocení výsledků dílčích týmů.

- **Den otevřených dveří** – tato fáze probíhá následující jednu iteraci o

délce šesti týdnů. Hlavní náplní fáze je příprava a samotná prezentace projektu na dni otevřených dveří FAV.

- **Zahájení LS** – fáze probíhá jednu iteraci po dobu tři týdnů a odpovídá fázi Zahájení ZS, jen se změnou vybíraných předmětů. v letním semestru se obecně mohou vytvářet složitější zadání semestrálních prací pro větší týmy, které se již řídí sami a spolupráce probíhá formou zakázky, kdy je pro celý tým zadán úkol a termín jeho splnění. Hlavní náplní této fáze je zpřesnění plánu projektu, stanovení cílů pro letní semestr, vytvoření zadání semestrálních prací pro splnění stanovených cílů, prezentování vytvořených zadání na příslušných předmětech a získání studentů pro konstrukční fázi.

- **Konstrukce LS** – fáze probíhá následujících pět iterací, kde celkově šestá iterace v semestru trvá dva týdny, slouží pro specifikaci zadání a je zakončena vytvořením dokumentu specifikace. Sedmá iterace trvá také dva týdny, slouží pro analyzování a návrh zadání. Následující osmá iterace trvá tři týdny a slouží pro samostatné implementování. Dokončení implementace a testování probíhá v iteraci devět. Poslední desátá iterace trvá tři týdny a slouží pro ukončení projektu a nasazení. v poslední iteraci dochází ke zhodnocení práce.

- **Předání** – této fázi odpovídá jedna iterace v měsíci květnu a červnu. Jejím účelem je sepsání doporučení pro další vývoj a předání projektu nastupujícímu vedení, které povede projekt následující akademický rok.

5.2 Nástroje pro podporu vývoje

Pro podporu plánování a sledování postupu prací je nadále využíván nástroj Redmine v závislosti na předchozích skutečnostech při porovnávání Redmine a TFS. Při případném přechodu na TFS by bylo nutné vyexportovat všechna data z Redmine (iterace s úkoly) a naimportovat je do TFS, jinak by došlo ke ztrátě historie práce na projektu. Jelikož TFS nebude v budoucnu využíván,

nebyl tento postup testován. Praktické ověření a porovnání obou nástrojů je uvedeno v následující kapitole.

Při rozhodování o přechodu ze stávajícího SVN na TFS správce zdrojů byly brány v potaz následující argumenty:

1. SVN je plně spravované KIV a je zaručena bezpečnost (přihlášení pomocí orion loginu) a záloha dat. Oproti tomu TFS je plně v rukách studentů. Může dojít k nesprávnému nastavení serveru, k bezpečnostním chybám, ke ztrátě či poškození dat.

2. Při přechodu z SVN na TFS by bylo nutné provést migraci celé historie projektu z jednoho nástroje na druhý. v dnešní době existuje několik migračních nástrojů třetích stran, které by mohly daný problém vyřešit, ale kvalita této migrace není zaručena.

V závislosti na skutečnosti možnosti využití jen limitované licence TFS a na argumentech uvedených výše byl vývoj projektu ponechán na SVN a nebyly testovány žádné migrační nástroje.

6 Praktické ověření úprav na vedení projektu

Kapitola popisuje praktické ověření úprav na vedení projektu, praktické porovnání TFS a Redmine pro podpoření plánování a sledování postupů a praktické zkušenosti vedoucích z letního semestru.

6.1 Praktické použití nové metodiky

Nová metodiky byla prakticky použita pro letní semestr.

V **zahajovací fázi** proběhlo naplánování iterací podle obrázku 5.1.1, byl upraven plán projektu pro následující semestr a připraveny zadání semestrálních prací. Pro letní semestr bylo připraveno pět zadání. Dvě zadání pro předmět Základy softwarového inženýrství (KIV/ZSWI) a vždy po jednom zadání pro předměty Pokročilé softwarové inženýrství (KIV/ASWI), Mobilní komunikace a zařízení (KIV/MKZ), Programování v prostředí .NET (KIV/-NET) a Programování Internetových aplikací (KIV/PIA). Jednotlivá zadání byla prezentována na přednáškách daných předmětů a jsou přiloženy v příloze E Zadání pro předměty v ak. roce 2012/2013.

Ve **fázi konstrukce** byly složeny tři vývojové týmy, pro každé zadání jeden. ASWI tým pracoval s nástrojem TFS, oba ZSWI týmy využívaly Redmine a všechny tři týmy používaly pro správu zdrojových kódů SVN. Pomocí těchto tří týmů bylo prováděno srovnání obou nástrojů v rámci plánování a sledování postupu prací v reálném provozu. Výsledný počet studentů pracujících na projektu v letním semestru byl 13 plus dva členové vedení. Jednotlivé týmy vytvořily návrh, implementaci, testování a dokumentaci své funkcionality. Závěrem fáze proběhlo zhodnocení výsledků jednotlivých týmů.

Ve **fázi předání** proběhlo sepsání doporučení vedoucích pro další vývoj, připravení plánu pro následující rok a samotné předání projektu novému vedení.

6.2 Zhodnocení nové metodiky

Následuje kritické zhodnocení nové metodiky vývoje v závislosti na získaných zkušenostech vedoucími projektu během prací v letním semestru.

Využití backlogu pro uložení zásobníku požadavků zůstávalo stále tím nejlepším řešením ať už byl využit backlog v Redmine nebo v TFS. Pravidelné schůzky s periodou jednoho týdne byly pro vedení týmu naprosto dostačující, zvyšovaly efektivitu a kvalitu práce. v letním semestru se na schůzky již nemusel dostavovat celý tým, ale vždy jen vedoucí týmu, který musel vědět co všichni jeho týmoví kolegové dělali v uplynulém týdnu, co budou dělat v týdnu následujícím a jestli se neobjevila překážka ve vývoji. Případné překážky ve vývoji řešil vždy její řešitel na nejbližší schůzce nebo akutně přes elektronickou komunikaci.

Jednotlivá zadání opět přinášela do projektu viditelné přírůstky, za které byly plně zodpovědné řešitelské týmy. Tento přístup je hodnocen velmi pozitivně a mělo by se v něm pokračovat i nadále.

Iterativní vývoj s plánováním délek iterací podle časové náročnosti požadavků na začátku semestru je pro univerzitní prostředí lepším řešením než použití iterací se stejným časovým trváním.

Oddělení fáze konstrukce od zbytku projektu je jasným krokem, kdy v dané fázi probíhá opravdu jen konstrukce v rámci řešitelských týmů.

6.3 Praktické porovnání TFS a Redmine v rámci vývoje, plánování a řízení projektu

V letním semestru byl nasazen TFS pro podporu vývoje v jednom ASWI týmu, v ostatních ZSWI týmech byla ponechána podpora vývoje v Redmine a tím došlo k porovnání obou nástrojů při praktickém použití na projektu.

TFS byl v ASWI týmu použit pro plánování a sledování prací. Byla vytvořena jedna kolekce, jeden týmový projekt a jeden tým. Naplánovalo se pět iterací s mezními termíny a naplnil se backlog. Jednotlivé požadavky z bac-

klogu byly přiřazovány do iterací a jednotlivým řešitelům. Sledování postupů prací probíhalo pomocí výpisů aktuálních úkolů a jejich stavu dokončení.

Redmine byl využit u zbylých ZSWI týmů pro plánování a sledování prací. Byly vytvořeny dva podřízené projekty. v každém projektu bylo vytvořeno pět iterací s mezním termínem a naplněn backlog. Jednotlivé požadavky z backlogu byly přiřazovány do iterací a jednotlivým řešitelům. Každý požadavek měl nastaven odhadovanou časovou náročnost a řešitelé v průběhu práce vyplňovali jeho procentuální míru dokončení a odpracovaný čas. Sledování postupu prací probíhalo pomocí výpisů aktuálních úkolů a jejich procentuální míry dokončení. Pro vizualizaci průběhu práce byly využívány přehledy a grafy sestavované v závislosti na časaových údajích.

S ohledem na porovnání TFS a Redmine v kapitole 4 a praktickému porovnání je jasné, že bez zpřístupnění pokročilých funkcí a nepoužití TFS správce zdrojových kódů je Redmine pro plánování a sledování postupu prací vhodnějším řešením. TFS nepřináší žádnou přidanou hodnotu oproti Redmine a nemá smysl ho nadále využívat v projektu.

6.4 Zkušenosti vedoucích projektu

Následuje popis zkušeností z vedení projektu v letním semestru s prací v jedné vývojové větvi a s dodržováním mezních termínů.

6.4.1 Zkušenosti z vedení projektu v celém roce

V zimním semestru probíhá vývoj především v předmětu PT, kde zpracovávají semestrální práci studenti druhých ročníků po dvojicích. Studenti nemají zkušenosti s prací v týmech, ani ve dvojicích, jejich programovací zkušenosti a plánování vlastního času je na minimální úrovni a s tím se musí v rámci metodiky vývoje počítat. Semestrální práce z předmětu PT je rozdělena na dvě poloviny, kde uprostřed semestru je hodnocení dílčí části a na závěr semestru probíhá odevzdávání a hodnocení závěrečné, finální části a dokumentace.

Pro vývojovou metodiku to znamená naplánování dvou produktivních

iterací pro PT studenty. Plánování více iterací je s ohledem na nezkušenost studentů a na předpokládaný malý funkční přírůstek zbytečný. Jestliže se počet iterací zmenšil, a tím se zvětšil časový odstup od hodnocení práce, frekvence schůzek se zvýšil na každý týden, tento fakt je velmi důležitý pro správné vedení studentů a rychlé odhalení případných nedorozumění a chyb ve vývoji. Na schůzce se řeší především základní otázky kterými jsou: "Co jednotliví členové týmu dělali od minulé schůzky?", "Byly nějaké problémy s prací?"a "Co budou jednotliví členové týmu dělat do další schůzky?". Následně schůzka většinou pokračuje diskuzí a řešením případných překážek ve vývoji.

Důležitou skutečností také bývá zjištění aktuálního stavu projektu na schůzce, zkontrolování správných odhadů a zbývající práce, tzn. zjištění stavu projektu vůči meznímu termínu a fakt, jestli se práce provádí podle plánu a vše se stihne dokončit. v případě, že je z vývoje projektu jasné, že se práce nestíhá je nutné provést revizi plánu, aby se v mezní termín odevzdávala kompletní kvalitní práce. Důležitým faktorem v týmu je celková komunikace, která musí probíhat nejen v rámci schůzek, ale i při jakékoli nejasnosti a problému. Vedení týmu musí být neustále schopné reagovat na případné dotazy, a tím rychle řešit překážky ve vývoji.

V letním semestru jsou pro vývoj projektu vybírány především předměty ZSWI a ASWI, kde studenti pracují ve větších týmech a očekává se od nich větší funkční přírůstek. v rámci ZSWI jde především o studenty druhých ročníků, kteří nemají s prací v týmu velké zkušenosti, většinou žádné. Je velice výhodné pokračovat se studenty z PT předmětů další semestr v rámci ZSWI, kde odpadá celá úvodní část seznámení s projektem a tým se seznamuje jen se svým zadáním a se svojí oblastí práce. v rámci ZSWI předmětu mají studenti povinnost pravidelně se scházet se svým mentorem a odevzdávat v průběhu vývoje artefakty vývoje.

Tato metodika je v rámci Space Traffiku pro studenty ZSWI zachována a plně akceptována. Týmy jsou obvykle pětičlenné a studenti si v rámci týmu sami rozdělují své role. Pro vedení projektu tím odpadá nutnost řízení celého týmu, vedení se přesouvá z role vedoucích do role projektových manažerů.

Manažeři nadále plánují práci celého týmu a sledují postup vývoje. Schůzky jsou nadále každý týden, jen se charakter schůzek liší. Na schůzku již nemusí docházet celý tým, ale pouze vedoucí týmu, se kterým probíhá většina komunikace, a člen týmu s případnou překážkou ve vývoji.

V rámci ASWI probíhá spolupráce se studenty čtvrtého ročníku, kde se již počítá s velkým funkčním přírůstkem. Tito studenti mají již velké zkušenosti s vývojem softwarů a s prací v týmu. Členové týmu jsou opět rozděleny do rolí. Vedení Space Traffiku opět vystupuje v roli manažerů a spolupráce probíhá především s vedoucím týmu. v rámci ASWI probíhá vývoj iterativně v pěti iteracích. Důležitost týdenních schůzek přetrvává i pro ASWI a jejich náplň je stejná jako u ZSWI.

6.4.2 Práce v jedné vývojové větvi

V letním semestru nebyla navržena žádná duplicitní zadání ani experimentální funkčnosti a proto mohl vývoj probíhat pouze v hlavní vývojové větvi. Všechny tři týmy, plus vedoucí projektu, pracovaly v hlavní vývojové větvi, což s sebou neslo několik problémů. Bylo nutné všechny členy týmů na tuto skutečnost důsledně upozornit a předejít případným problémům. Členové týmu museli vždy před commitováním dovést projekt do sestavitelného a spustitelného stavu, kdy prošly všechny stávající testy. Tato činnost se bohužel nedařila ZSWI týmu a několikrát se jim podařilo dostat hlavní vývojovou větev do nepřeložitelného stavu. Dělo se tak především z důvodu malých zkušeností s vývojem s použitím správce zdrojových kódů. Tento problém byl intenzivně řešen a podařilo se ho vyřešit.

Pro další vývoj v hlavní vývojové větvi je velmi důležité vysvětlit všem členům týmu závažnost chybných commitů. Každý člen týmu musí být schopen svůj lokální projekt sestavit, spustit a musí projít všechny stávající testy. Až po úspěšném doběhnutí všech testů smí vývojář commitovat svoji práci. Commitování musí být co nejčastější, přírůstky musí být tak malé, aby v nich bylo jednoduché nalézt a opravit chybu. Ke commitu musí být vždy přidán komentář, vysvětlující změny zdrojových kódů, a vazba na plněný úkol, připsáním ID úkolu do komentáře.

6.4.3 Plánování a dodržování mezních termínů

Díky zkušenostem ze zimního semestru, kdy studenti nestíhali odevzdat práci ve stanoveném termínu a museli projekt natahovat, byl termín odevzdání všech semestrálních prací posunut o dva týdny dopředu a vytvořen projektový buffer pro dodatečné dokončení práce. Projektový buffer je dobrým nástrojem, i když se správě daří škálovat práci na projektu, případný časový prostor může sloužit ke kvalitnímu dokončení prací při nalezených problémech v závěru projektů.

Jak se ukázalo v průběhu semestru, ASWI tým neměl se stanoveným termínem a celým plánem projektu problém a vše probíhalo v naprostém pořádku. Oba ZSWI týmy byly posunutým termínem v závěru projektu velmi překvapeni a snažily se cílit svůj projekt na mezní termín odevzdání svých výsledků semestrálních prací na daných předmětech. Díky důslednému řízení týmů jeden ZSWI tým odevzdal práci ve stanoveném termínu bez větších komplikací. Druhý ZSWI tým bohužel narazil v závěru implementační části na překážky a nebyl schopen projekt odevzdat do stanoveného termínu, ani do termínu o čtrnáct dní později. Jelikož ve Space Traffiku je požadována kvalitní práce, byl termín posunut o další měsíc a projekt úspěšně dokončen.

6.5 Splnění cílů pro ak. rok 2012/2013

Všechny hlavní cíle projektu pro akademický rok 2012/2013 byly splněny. Byla vytvořena první hratelná verze hry, která byla nasazena na server[1]. Rozšíření o povědomí projektu bylo prováděno především prezentací semestrálních prací, prezentací projektu na Dni otevřených dveří FAV a prezentací projektu na středních školách.

Byl vytvořen portál hry umožňující prezentaci projektu, zobrazení projektové wiki, registraci a přihlášení do hry. Byla kompletně předělána projektová wiki[2] na technologii MediaWiki.

Do hry byly přidány funkčnosti základen, výstavby budov, generování

[1]Dostupné z http://spacetraffic.kiv.zcu.cz/demo.
[2]Dostupná z http://spacetraffic.kiv.zcu.cz/mediawiki

zboží v továrnách, obchodu, aukce, NPC obchodníka, programování lodí, podpory achievementů a levelování.

Probíhalo testování hry pomocí jednotkových a funkčních testů. Celá funkčnost byla testována herními testy a testy použitelnosti, jak uvádí kolega Pavel Bořík [14].

Uživatelské grafické rozhraní bylo napojeno na herní server a umožňuje použití všech naimplementovaných funkčností. Na mapu hvězdné soustavy se podařilo vykreslit dráhy lodí odchýlených od hvězdy.

Byla připravena jednoduchá mobilní aplikace pro platformu Android podporující základní funkčnosti hry.

Aktuální stav projektu po akademickém roce 2012/2013 je uveden v příloze C Stav projektu léto 2013.

6.6 Doporučení

Závěrem práce je uvedeno doporučení vedoucích projektu, autora této práce a Pavla Bořika, v závislosti na získaných zkušenostech během vedení projektu v letošním roce a v závislosti na zkušenostech jejich předchůdců, především Petra Vogla a Martina Štěpánka.

1. Jako hlavní bod doporučení je dobrá **komunikace** v celém projektu, jakýkoli problém se musí řešit okamžitě ve chvíli objevení, ať už se jedná o chybu ve zdrojových kódech, nepochopení zadání nebo nedodržení plánu. Objevený problém je nejlepší řešit ihned při jeho odhalení a neoddalovat čas jeho nahlášení.

2. Příprava před začátkem každého semestru je kritická, jelikož v prvních týdnech semestru musí být naplánovaný projekt, připravené zadání semestrálních prací a prezentace na příslušné předměty.

3. v rámci dokumentace zaznamenávat všechny často kladené dotazy, jedná se především o dotazy z oblastí předání projektu a začátku semestru s novými studenty.

4. Snažit se pokračovat ve spolupráci se studenty, kteří již na projektu spolupracovali.

5. Důsledně dodržovat pravidelné týdenní schůzky a kontrolovat průběh práce.

6. Pro další vývoj se doporučuje používat SVN, Redmine a nová upravená metodika vývoje z kapitoly 5.

6.6.1 Cíle pro ak. rok 2013/2014

Po minulých dvou letech plných implementování nových funkčností je třeba provést kontrolu zdrojových kódů a celkový refactoring projektu. Do dalšího akademického roku se neplánuje velký funkční přírůstek. Jako hlavní cíl je stanoveno vyladění dosavadních funkčností a dokončení rozdělaných prací. Je potřeba dopsat všechny testy na již hotové funkčnosti, a to nejen jednotkové testy na jednotlivé dílčí činnosti, ale také integrační testy přes celou aplikaci, od grafického uživatelského rozhraní, přes vrstvu služeb, vrstvu herního světa, perzistentní vrstvu a uložení do databáze.

Nadále zůstává důležitým cílem vytvoření komunity kolem projektu. Zvážení založení a udržování projektového blogu a aktivní účast v sociálních sítích. Důležitým faktorem je prezentace projektu své cílové skupině, a to středoškolským studentům. Projekt musí být prezentován na Dni otevřených dveří FAV a pokud možno i přímo na středních školách.

Seznam cílů pro akademický rok 2013/2014 je uveden v příloze D Cíle pro ak. rok 2013/2014.

7 Závěr

Tato diplomová práce se zabývá podporou vývoje v projektu Space Traffic nástroji Microsoft Visual Studio Team Foundation Server. Cílem této práce bylo zhodnocení dosavadní metodiky vývoje, prostudování možností TFS, navržení případných úprav vývoje projektu a praktické ověření těchto vylepšení.

Na základě analýzy aktuálního stavu studentského projektu Space Traffic, charakteristických vlastností univerzitního prostředí jeho realizace, existujících rizik a používané vývojové metodiky navržené Petrem Voglem došlo ke kritickému zhodnocení celého vývojového procesu při použití dané vývojové metodiky v zimním semestru.

Byly popsány oblasti pokrývající nástroje pro správu životního cyklu aplikací, prozkoumány možnosti zástupce nástrojů správy životního cyklu TFS pro plánování a sledování postupů prací a došlo k porovnání s nástrojem Redmine.

Došlo k upravení a praktickému otestování vývojové metodiky a použití TFS pro podporu vývoje projektu v letním semestru, získané zkušenosti byly zaznamenány v kapitole 6.

Výsledkem této práce je upravení metodiky vývoje a její značné dotažení k dokonalosti pro potřeby vývoje rozsáhlého projektu v univerzitním prostředí. Otestování možností TFS v limitované licenci a doporučení nadále využívat pro podporu vývoje projektu Redmine s uložením zdrojových kódů na SVN. Upravená metodika vývoje a excelentní vedení projektu vedlo k oboustranné spokojenosti, jak vedoucích projektu, tak všech úspěšných spolupracovníků na projektu. Všechny hlavní cíle naplánované na akademický rok 2012/2013 byly splněny, dokonce se podařilo vypracovat i několik dalších funkčních požadavků a několik požadavků rozpracovat. Projekt vývoje hry Space Traffic se značně přiblížil ke svým cílům, byla vydána první hratelná demoverze. Díky těmto faktům došlo ke splnění všech cílů této práce.

Literatura

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

[2] Richard Kocman: *Řízení projektu webové hry*. Diplomová práce. Západočeská univerzita v Plzni, Fakulta Aplikovaných věd, Katedra informatiky a výpočetní techniky, Plzeň, 2012.

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

[4] Martin Štěpánek: *Architektura a implementace webové hry pro více hráčů*. Diplomová práce. Západočeská univerzita v Plzni, Fakulta Aplikovaných věd, Katedra informatiky a výpočetní techniky, Plzeň, 2012.

[5] *Redmine*. (červen 2013).
`http://www.redmine.org/`

[6] Michael Hüttermann: *Agile ALM: Lightweight tools and Agile strategies*. Manning Publications, Shelter Island, 2012. ISBN 978-1-935182-63-4.

[7] *MSDN Magazine: Visual Studio TFS Team Project and Collection Guidance*. (červen 2013).
`http://msdn.microsoft.com/en-us/magazine/gg983486.aspx`

[8] *TFS cloudové řešení*. (červen 2013).
`http://tfs.visualstudio.com/`

[9] *JIRA*. (červen 2013).
`https://www.atlassian.com/software/jira`

[10] *Assembla*. (červen 2013).
`https://www.assembla.com/`

[11] *VersionOne*. (červen 2013).
`http://www.versionone.com/`

[12] *Rally software*. (červen 2013).
`http://www.rallydev.com/`

[13] *Mingle*. (červen 2013).
`http://www.thoughtworks.com/products/mingle-agile-project-management/`

[14] Pavel Bořík: *Zajištění kvality webové hry Space Traffic*. Diplomová práce. Západočeská univerzita v Plzni, Fakulta Aplikovaných věd, Katedra informatiky a výpočetní techniky, Plzeň, 2013.

Seznam používaných pojmů a zkratek

ALM	Application Lifecycle Management
ASWI	Předmět Pokročilé softwarové inženýrství
backlog	zásobník pro scénáře, funkční vlastnosti a úkoly
Bug	chyba v aplikaci
CSS	Cascade Style Sheets
DAO	Data Access Objects
DreamSpark	univerzitní distribuce nástrojů společnosti Microsoft
FAV	Fakulta aplikovaných věd
Feature Driven Development	iterativní inkrementální vývojový proces
HTML	HyperText Markup Language
IIS	Internet Information Services
Impediment	překážka ve vývoji
KIV	Katedra informatiky a výpočetní techniky
LS	letní semestr
MMORTS	Massively multiplayer online real-time strategy
MVC	Model-view-controller
NPC	Non-player character
PT	Předmět Programovací techniky
Redmine	webový nástroj pro podporu řízení projektů.
Scrum	iterativní inkrementální agilní vývojový proces
SQL	Structured Query Language
SVG	Scalable Vector Graphics
SVN	Subversion

Task	úkol
Test case	testovací úkol
TFS	Microsoft Visual Studio Team Foundation Server
URL	Uniform resource locator
VC	Version Control
WIT	Work Item Type
XML	Extensible Markup Language
ZČU	Západočeská univerzita v Plzni
ZS	zimní semestr
ZSWI	Předmět Základny softwarového inženýrství

A Stav projektu léto 2012

- Game design na několik let dopředu.
- Architektura na nejvyšší úrovni - GameServer, Webové UI, JavaScriptový 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.
- JavaScriptový 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 a částečně naplněna wiki projektu.
- Návody a dokumentace včetně diplomových prací (Štěpánek [4] a Vogl [3], popř. Kocman [2] a Neudert [1]).

B Cíle pro ak. rok 2012/2013

1. Dokončení integrace výsledků práce z letního semestru 2011-2012 [priorita: vysoká].
2. Implementace programování lodí [priorita: vysoká].
3. Dokončení GUI pro ovládání lodí [priorita: vysoká].
4. Minimální potřebná implementace základen [priorita: vysoká].
5. Implementace obchodu (pouze trh) [priorita: vysoká].
6. NPC obchodník (základní funkce) [priorita: vysoká].
7. Účast na akci Den otevřených dveří na FAV [priorita: vysoká].
8. Vytvoření komunity projektu (portál, vývojářský blog, sociální sítě) [priorita: vysoká].
9. Vytvoření webového portálu hry [priorita: vysoká].
10. Vylepšení konfigurace projektové wiki [priorita: vysoká].
11. Implementace podpory achievementů a levelování [priorita: střední].
12. Implementace misí [priorita: střední].
13. Tvorba herního obsahu [priorita: střední].
14. Playtesting [priorita: střední].
15. Balancování [priorita: nízká].
16. Základní integrace wiki pro hráče do portálu hry [priorita: nízká].

C Plán ak. rok 2012/2013

Datum	Týden	Iterace	Fáze	Činnosti	
od předání projektu do začátku zimního semestru		0.	Příprava	Seznámení s projektem	
				Seznámení s nástroji	
		1.	Zahájení ZS	Vytvoření plánu pro nadcházející rok	
				Určení cílů a rozsahu práce pro zimní semestr	
				Vytvoření zadání semestrálních prací	
24.9.2012	1.			Prezentace zadání semestrálních prací	
1.10.2012	2.			Získání studentů	
8.10.2012	3.	2.	Konstrukce ZS	První schůzky se studenty	
15.10.2012	4.			Analýza, návrh a začátek implementace	
22.10.2012	5.				
29.10.2012	6.				
5.11.2012	7.				
12.11.2012	8.			Buffer, hodnocení práce	
19.11.2012	9.	3.		Implementace, testování, dokumentování a nasazení.	
26.11.2012	10.				
3.12.2012	11.				
10.12.2012	12.				
17.12.2012	13.			Buffer, hodnocení práce	
24.12.2012		4.	Den otevřených dveřích	Prostor pro dokončení projetů ze zimního semestru	
31.12.2012				Integrace výsledků	
7.1.2013				Příprava na den otevřených dveří	
14.1.2013					
21.1.2013					
28.1.2013	DOD			Účast na dni otevřených dveří	
4.2.2013		5.	Zahájeni LS	Zpřesnění plánu pro letní semestr	
				Určení cílů a rozsahu pro letní semestr	
				Vytvoření zadání semestrálních prací	
11.2.2013	1.			Prezentace zadání semestrálních prací	
18.2.2013	2.			Získání studentů	
25.2.2013	3.	6.	Konstrukce LS	Specifikace	LCO
4.3.2013	4.				
11.3.2013	5.	7.		Analýza a návrh	LCA
18.3.2013	6.				
25.3.2013	7.	8.		Implementace	
1.4.2013	8.				
8.4.2013	9.				
15.4.2013	10.	9.		Dokončování, testování a dokumentování	IOC
22.4.2013	11.				
29.4.2013	12.	10.		Ukončení a nasazení	REL
6.5.2013	13.			Buffer, hodnocení práce	
13.5.2013	14.				
20.5.2013		11.	Předání	Integrace výsledků	
květen a červen				Dokončení restů	
				Sepsání doporučení pro další vývoj	
				Příprava plánu na další rok	
				Předání projektu	

C Stav projektu léto 2013 (nové funkčnosti)

- Definice XML formátů pro zboží, budovy, achievementy a levely (včetně načítání a zpracování).
- Rozšíření databáze o tabulky pro reportáže, obchodníky, základny, pozemky, budovy (sklady, doky a továrny), achievementy a statistiku hráče.
- Vykreslení drah lodí na hvězdné mapě automaticky odkloněné od hvězdy.
- Portál s možností registrace, přihlášení, prezentace projektu a odkazu na projektovou wiki.
- Dokončení nákupu a pohybu lodí.
- Nákup pozemků a budov na základnách, generování zboží v továrnách.
- Obchod – nakupování zboží na základnách do lodí, aukce, NPC obchodníci a NPC přepravci.
- Testy na hotové funkčnosti.
- Vytvoření nové MediaWiki (místo stávající TikiWiki).
- Jednoduchá mobilní aplikace pro android podporující základní funkčnosti hry.
- Rozpracovaná customizace lodí (definice modelů, trupů a modulů lodí).
- Programování lodí (hotová gramatika, zpracování zadaného programu – stačí napojit na příslušné funkce).

D Cíle pro ak. rok 2013/2014

1. Kontrola a refactoring zdrojových kódů [priorita: vysoká].
2. Vyladění stávajících funkčností a dokončení rozdělaných prací [priorita: vysoká].
3. Dopsání všech jednotkových testů [priorita: vysoká].
4. Rozšíření komunitu kolem projektu (blog, sociální sítě, prezentace na Dni otevřených dveří a prezentace na středních školách) [priorita: vysoká].
5. Playtesting další verze hry [priorita: vysoká].
6. Integrační testy přes celou aplikaci od grafického uživatelského rozhraní, přes vrstvu služeb, vrstvu herního světa, perzistentní vrstvu až po uložení do databáze [priorita: střední].
7. Podpora pro vícejazyčnou verzi hry [priorita: střední].
8. Implementace miniher [priorita: střední].

E Zadání pro předměty v ak. roce 2012/2013

Zadání pro předmět ASWI - Achievementy a úrovně ve hře Space Traffic

Zadavatel: Jan Dyrczyk dyrczyk@students.zcu.cz

Téma je součástí studentského projektu webové MMORTS hry SpaceTraffic (http://spacetraffic.kiv.zcu.cz/code/). Úkolem týmu bude ve spolupráci s členy projektu hry navrhnout konkrétní achievementy, levelové stupně a odměňováni zkušenostními body za plnění herních aktivit. Navržené prvky pak bude třeba implementovat a zpřístupnit v uživatelském rozhraní. Práce nabízí poměrně velký prostor pro řešitelsky tým a jeho fantazii.

Technologie: C# (ASP.NET, HTML, JavaScript)

Tool: Team Foundation Server

Zadání pro předmět PT – Space Traffic – rozšíření základen na planetách – pozemky a budovy

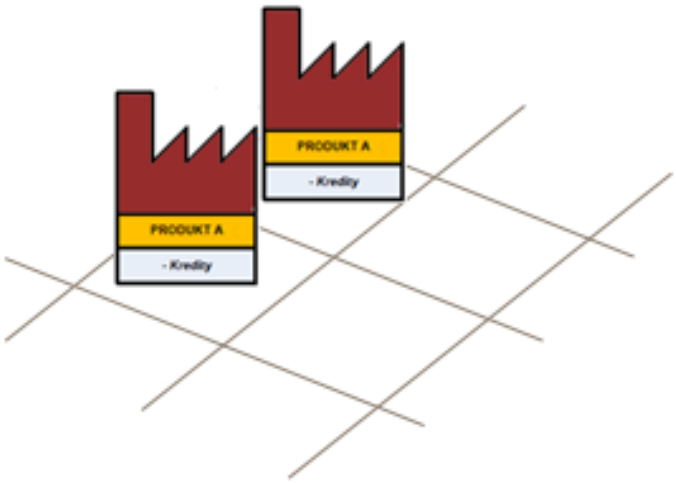
Co je Space Traffic?

Space Traffic je webová MMORTS (Massively-Multiplayer Online Real-Time Strategy) se zaměřením na obchodování ve vesmíru obsahující programovatelné prvky jako součást ovládání. Hra má za cíl reprezentovat KIV a přilákat středoškolské studenty ke studiu na FAV. Hra je vyvíjena samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací.

V případě dalšího zájmu je možné v projektu pokračovat v rámci PRJ5, bakalářské práce potažmo diplomové práce.

Zadání

1. Implementovat pozemky na základnách. Pozemek je reprezentován jako plocha čtvercových polí, na které lze stavět budovy. Jednotlivá pole lze nakupovat po skupinách, vždy tak aby vznikla obdélníková plocha, až do maximální velikosti 8x8 polí.
2. Implementovat budovy. Budova může být trojího typu (dok, sklad a továrna) s rozdílnou velikostí do max. 3x3 polí. Budova bude implementována pouze jako třída s atributy id, typ a velikost.



Obr. 1: Grafické přiblížení pozemků a továren.

3. Implementovat GUI pro nákup pozemků a výstavbu budov:
    a. Nákup pozemků;
    b. Rozšíření již stávajícího pozemku;
    c. Zmenšení pozemku;
    d. Výstavba budov na pozemku;
    e. Demolice budov.

Zadání pro předmět PT – Space Traffic – rozšíření základen na planetách – továrny

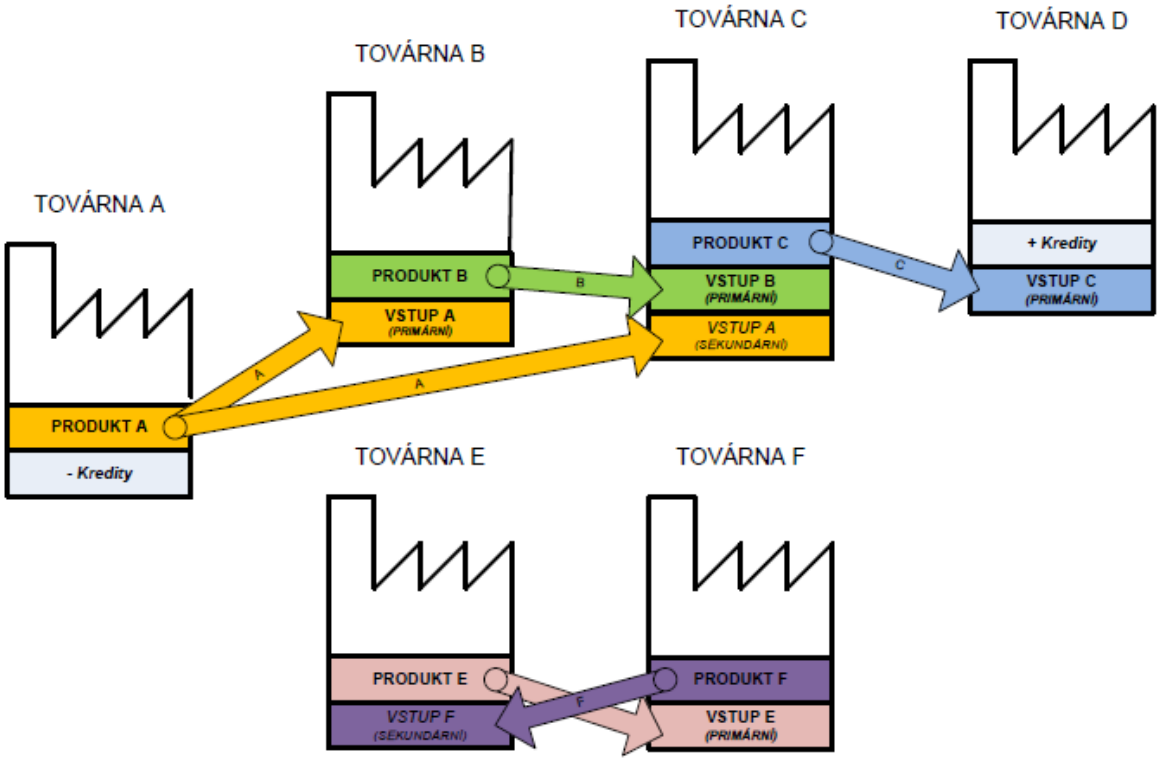
Co je Space Traffic?

Space Traffic je webová MMORTS (Massively-Multiplayer Online Real-Time Strategy) se zaměřením na obchodování ve vesmíru obsahující programovatelné prvky jako součást ovládání. Hra má za cíl reprezentovat KIV a přilákat středoškolské studenty ke studiu na FAV. Hra je vyvíjena samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací.

V případě dalšího zájmu je možné v projektu pokračovat v rámci PRJ5, bakalářské práce potažmo diplomové práce.

Zadání

1. Navrhnout xml schéma pro definici zboží a továren.
    a. Zboží jako je voda, vzduch, železo, plazma, atd. Jeho velikost, váha, atd.
    b. Továrny typu A, B, C, D, viz obr. 1.



Obr. 1: Diagram vzájemných zásobovacích vztahů továren.

2. Implementovat továrny. Továrna podle svého typu (A,B,C,D) bude spotřebovávat suroviny a vytvářet produkty.
3. Implementovat základní funkci skladu. Sklad uchovává suroviny pro chod továren a vytvořené produkty.

Zadání pro předmět MKZ - Space Traffic – Mobilní aplikace pro Android

Navrhněte a implementujte mobilní aplikaci pro platformu Android pokrývající základní funkčnost hry Space Traffic. Jako předloha může sloužit webová aplikace dostupná na http://spacetraffic.kiv.zcu.cz/demo.

Jednotlivé prvky plně funkční ve webové aplikaci:

- Zobrazení základních informací o hře
- Registrace
- Přihlášení
- Zobrazení hráčova stavu
  - Kredity
  - Lodě
  - Pozemky a budovy
- Zobrazení mapy hvězdného systému
  - Ve webové aplikaci řešeno pomocí SVG a Javascriptu, nutno přepracovat na malé mobilní rozlišení
- Základní interaktivita mapy
  - Po kliknutí na planetu zobrazení informací o planetě
    - Nákup lodí
    - Nákup zboží
    - Nákup pozemků a stavění budov
    - Poslání lodě na cestu
  - Po kliknutí na červí díru zobrazení daného hvězdného systému

Zadání pro předmět NET - Space Traffic - Customizace lodí

Cílem je vytvoření aplikace, která umožní navrhování nových modelů a komponentů vesmírných lodí a následně umožní vytváření lodí z těchto navržených modelů a komponentů.

Vize je taková, že bude vytvořeno několik základních modelů lodí obsahujících několik základních komponentů. Hráč si bude moci lodě kupovat a jejich komponenty následně přidávat, měnit či vylepšovat.

1. Jádro celé aplikace

Navrhněte vhodné datové struktury pro jednotlivé prvky programu:

- model lodi
- komponent lodi
- vesmírná loď

Modely a komponenty lodí budou ukládány do XML souborů. Složená vesmírná loď bude ukládána do textového souboru (později v 3. úloze do databáze).

2. Uživatelské rozhraní

Vytvořte nad první úlohou grafické uživatelské rozhraní, které umožní:

- načíst již hotové modely a komponenty z XML
- vytvořit nový model, komponent
- vytvořit nový model (komponent) z již stávajícího modelu (komponentu)
- upravit stávající model, komponent
- smazat stávající model, komponent
- sestavit novou lodě z modelu a několika komponentů
- upravit loď: přidat, odebrat, změnit, vylepšit komponenty, změnit model

3. Webové rozhraní

Vytvořte nad první úlohou (s využitím kódu z druhé aplikace) grafické webové uživatelské rozhraní, které zaintegrujete do projektu Space Traffic.

Webové rozhraní umožní:

- vybrat a koupit loď
- upravit loď: přidat, odebrat, změnit, vylepšit komponenty, změnit model
- prodat loď

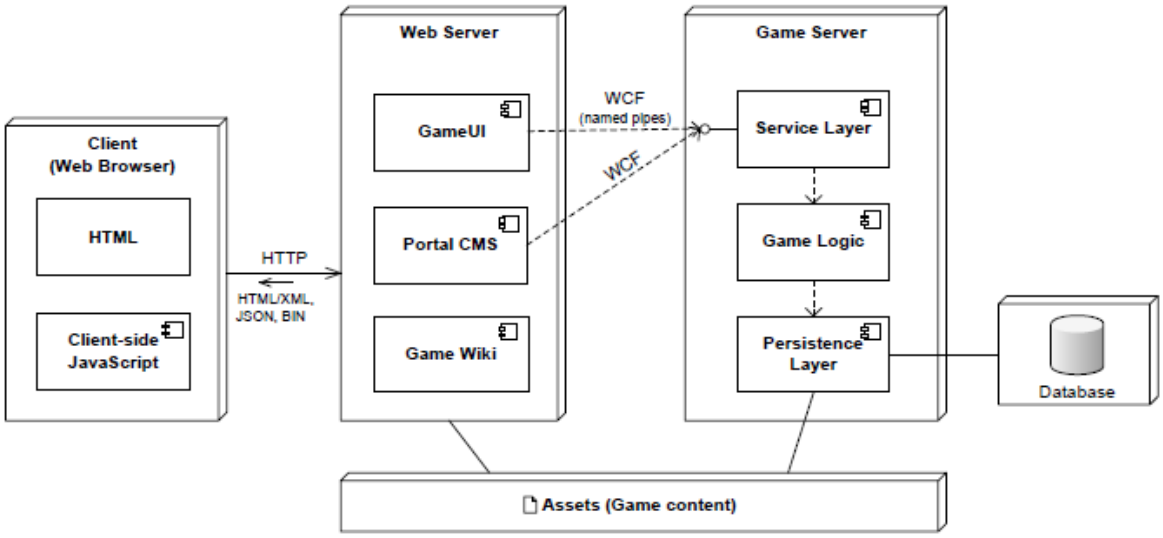
Aplikace načte XML soubory s hotovými modely a komponenty a zpřístupní je Space Traffic serveru, vytvořená loď bude ukládána do MSSQL databáze projektu.

Zadání pro předmět PIA - Space Traffic – Nový front-end webové hry

Co je Space Traffic?

Space Traffic je webová MMORTS (Massively-Multiplayer Online Real-Time Strategy) se zaměřením na obchodování ve vesmíru obsahující programovatelné prvky jako součást ovládání. Hra má za cíl reprezentovat KIV a přilákat středoškolské studenty ke studiu na FAV. Hra je vyvíjena samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací.

Architektura hry vychází z architektonického vzoru MVC. Celkový pohled na systém je znázorněn na obrázku 1.



Obr. 1: Diagram architektury aplikace Space Traffic.

Zadání

1. Navrhnout a implementovat nový front-end
2. Zobrazení základních informací o hře, screenshots, popis hry (pouze prezentace)
3. Výpis informací ze hry
    a. seznamy nejlepších hráčů (aliancí) možné seřadit (filtrovat) podle zadaných parametrů
4. Výpis novinek přímo z herního světa (CMS): započal nový věk, nové featury, překonávání rekordů, spuštění nových achievementů atd.
5. Akce s uživatelskými účty: registrace/login/logout/edit (standardní zadání PIA)
6. Čtyři uživatelské role:
    a. Host může články jen číst.
    b. Přihlášený uživatel může články číst a komentovat.
    c. Administrátor může články číst, komentovat, přidávat, editovat a mazat.
    d. auto_writer - automatický výpis z herního světa: někdo splnil velmi těžký achievement, právě se registroval 1000. hráč, začala cenová válka na planetě Zemi - ceny strmě padají

Zadání pro předmět ZSWI – Space Traffic – Obchod

Co je Space Traffic?

Space Traffic je webová MMORTS (Massively-Multiplayer Online Real-Time Strategy) se zaměřením na obchodování ve vesmíru obsahující programovatelné prvky jako součást ovládání. Hra má za cíl reprezentovat KIV a přilákat středoškolské studenty ke studiu na FAV. Hra je vyvíjena samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací.

V případě dalšího zájmu je možné v projektu pokračovat v rámci PRJ5, bakalářské práce potažmo diplomové práce.

Zadání

Seznamte se s předchozí prací na problematice obchodu v projektu webové hry Space Traffic.

- Navrhněte a implementujte ekonomický systém, podle kterého se budou automaticky generovat NPC obchodníci.
- Implementujte NPC obchodníka, sídlícího na každé planetě, který nabízí a poptává zboží různého druhu s měnícími se cenami podle ekonomického systému z bodu 1.
- Implementujte hráčského NPC obchodníka spojeného se skladem.
- Implementujte NPC přepravce (počítačem řízené lodě létající po daných obchodních trasách, přepravující a prodávající zboží na volném trhu).

Rozšíření

- Implementujte NPC korporace (hrou poskytované základní služby, výrobci lodí, výrobci zboží, zadavatelé kontraktů).
- Implementujte kontrakty (úkoly pro hráče automaticky zadávané korporacemi).
- Implementujte NPC frakce (populace planety, politická uskupení atd).

U všech bodů dbejte na perzistentní uložení stavu hry a důsledné testování.

Požadavky

- Implementace v jazyku C#.
- Kvalitní dokumentace
- Dokumentace v kódu v angličtině.
- Dodržování štábní kultury projektu.
- Průběžné konzultace a výsledky práce s vedoucími projektu.
- Používání nástrojů Redmine a SVN.

Kontakty

Jan Dyrczyk, dyrczyk@students.zcu.cz
Pavel Bořík, pborik@students.zcu.cz

Zadání pro předmět ZSWI – Space Traffic – Podpora projektu

Co je Space Traffic?

Space Traffic je webová MMORTS (Massively-Multiplayer Online Real-Time Strategy) se zaměřením na obchodování ve vesmíru obsahující programovatelné prvky jako součást ovládání. Hra má za cíl reprezentovat KIV a přilákat středoškolské studenty ke studiu na FAV. Hra je vyvíjena samotnými studenty v rámci jejich semestrálních, bakalářských a diplomových prací.

V případě dalšího zájmu je možné v projektu pokračovat v rámci PRJ5, bakalářské práce potažmo diplomové práce.

Zadání

Úkolem týmu bude prostudovat dosavadní práci na projektu, nastudovat dokumentace, pochopit celý projekt a jeho vizi.

Na základně získaných znalostí a spolu s konzultací s vedoucími navrhnout a vytvořit programátorskou wikipedii a uživatelský manuál.

Navrhnout herní obsah. Při návrhu obsahu spolupracovat s týmem na ASWI, který má na starosti Achievementy.

V další fázi navrhnout propagaci projektu, ať už prostřednictvím blogu či sociálních sítí. Pro lepší komunikaci s komunitou připravit projektové fórum.

Požadavky

- Průběžné konzultace a výsledky práce s vedoucími projektu.
- Používání nástroje Redmine.

Kontakty

Jan Dyrczyk, dyrczyk@students.zcu.cz
Pavel Bořík, pborik@students.zcu.cz