

FIIT grid (Dokumentácia k riadeniu)

Autori: Bc. Ján Kalmár, Bc. Juraj Petrík, Bc. Juraj Vincúr,
Bc. Martin Tibenský Bc. Pavol Pidanič, Bc. Radoslav Zápach

Kontakt: tp-12@googlegroups.com

Akademický rok: 2013/2014

Vedúci práce: Ing. Peter Lacko, PhD.

História zmien dokumentu

Dátum	Verzia	Zhrnutie zmien	Autor
19.11.2013	V 1.0	Pridanie metodík 1-6	M.Tibenský
20.11.2013	V 1.1	Pridanie zápisov zo stretnutí	M.Tibenský
21.11.2013	V 1.2	Úprava formátovania	M.Tibenský
12.12.2013	V 1.3	Pridanie zápisov zo stretnutí 8 - 10	P. Pidanič

Tabuľka 1: História dokumentu

Obsah

1 Úvod.....	3
2 Predstavenie tímu.....	3
2.1 Kompetencie členov tímu.....	3
2.2 Rozdelenie manažérskych úloh.....	5
3 Metodika práce k tvorbe technickej dokumentácie.....	6
3.1 Úvod.....	6
3.1.1 Cieľ dokumentu.....	6
3.1.2 Určenie metodiky.....	6
3.1.3 Slovník pojmov.....	6
3.1.4 Zoznam súvisiacich metodík.....	6
3.1.5 Roly a zodpovednosti.....	6
3.2 Všeobecné zásady.....	6
3.3 Technická dokumentácia.....	7
3.3.1 Členenie dokumentu.....	7
3.3.2 História zmien dokumentu.....	7
3.3.3 Členenie kapitol šprintu.....	8
3.3.4 Opis šprintu.....	8
3.3.5 Záznam vykonanej práce.....	8
3.3.6 Vytvorenie záznamu.....	9
3.4 Zápisnice zo stretnutí.....	9
3.4.1 Členenie zápisnice.....	9
3.4.2 Vytvorenie zápisnice zo stretnutia.....	9
Príloha 1. Šablóna zápisnice.....	11
4 Štábná kultúra písania kódu pre tím č. 12.....	12
4.1.1 Cieľ dokumentu.....	12
4.1.2 Určenie metodiky.....	12
4.1.3 Slovník pojmov.....	12
4.1.4 Všeobecné ustanovenia.....	12
4.2 Pokyny pre písanie zdrojových súborov.....	12
4.2.1 Formátovanie kódu.....	12
4.2.2 Organizácia súborov.....	12
4.2.3 Formálna stránka kódu.....	13
4.2.4 Exceptions.....	15
4.2.5 Písanie komentárov.....	17
4.2.6 Chyby a varovania.....	18
4.2.7 Iné odporúčania.....	18
4.3 Podporné nástroje.....	18
4.3.1 Findbugs.....	18
4.3.2 Checkstyle.....	19
5 Testovanie zdrojového kódu.....	20
5.1.1 Cieľ dokumentu.....	20
5.1.2 Určenie metodiky.....	20
5.1.3 Slovník pojmov.....	20
5.1.4 Všeobecné ustanovenia.....	20
5.1.5 Roly a zodpovednosti.....	20
5.2 Pokyny pre vytváranie testov.....	21
5.2.1 Kedy vytvárať testy a kto.....	21
5.2.2 Čo testovať.....	21

5.2.3	Umiestnenie.....	21
5.2.4	Štruktúra.....	23
5.2.5	Reprezentácia výsledkov.....	23
5.3	Pokyny pre testovanie.....	24
5.3.1	Kedy a čo testovať.....	24
5.3.2	Výstupy.....	24
5.4	Pokyny pre vytvorenie testovacieho protokolu.....	24
5.4.1	Umiestnenie.....	24
5.4.2	Štruktúra.....	24
6	Metodika pre písanie používateľskej príručky.....	26
6.1.1	Cieľ a obsah dokumentu.....	26
6.1.2	Určenie metodiky.....	26
6.1.3	Definícia pojmov.....	26
6.1.4	Zoznam súvisiacich metodík.....	26
6.1.5	Roly a zodpovednosti.....	26
6.2	Formátovanie, štruktúra a rozsah príručky.....	27
6.2.1	Vytvorenie záznamu z technickej dokumentácie.....	27
6.2.2	Úprava analytickej a implementačnej časti a výsledná štruktúra záznamu.....	28
6.2.3	Úprava inštalačnej časti.....	28
6.3	Dokumentová časť príručky.....	28
6.4	Wiki projektu.....	29
	Príloha 1 – Odporúčaný obsah jednotlivých kapitol.....	30
7	Metodika práce k verzionovaniu.....	31
7.1.1	Cieľ dokumentu.....	31
7.1.2	Určenie metodiky.....	31
7.1.3	Slovník pojmov.....	31
7.1.4	Roly a zodpovednosti.....	31
7.1.5	Všeobecné pokyny.....	31
7.2	Využívanie verzionovacieho systému git v IDE Eclipse.....	31
7.2.1	Inštalácia pluginu EGit do Eclipse.....	31
7.2.2	Generovanie a vloženie kľúča do FIIT Gitu.....	32
7.2.3	Import projektu (projektov) z GIT repozitára.....	32
7.2.4	Commit a push do repozitára.....	33
7.2.5	Stiahnutie aktuálnych dát z repozitára.....	34
7.2.6	Riešenie konfliktov.....	34
8	Príloha A: Zápis z stretnutí za zimný semester.....	35

1 Úvod

2 Predstavenie tímu

2.1 Kompetencie členov tímu

Pavol Pidanic

Vyštudoval odbor Informatika na FIIT. Pocas štúdia sa venoval hlavne jazyku Java. Taktiež C, Asembler, UML, SQL, XML. Ako softvérový inžinier pracuje na projekte využívajúcom hlavne Java EE technológie (JPA/Hibernate, JSF, JAXB), ďalej Spring, Maven, Oracle DB. Je držiteľom certifikátu Oracle Certified Professional, Java SE 6 Programmer.

Juraj Petrik

Ukončil bakalárske štúdium na FIIT STU v odbore Informatika. V súčasnosti pracuje ako vývojár pre správu služieb na platforme Linux.

Vo svojej bakalárskej práci sa venoval reimplementácii systému na evidenciu výpočtovej techniky pre FIIT STU, využívajúc programovací jazyk JAVA, knižnicu Hibernate, aplikacný server Apache Tomcat a databázového serveru PostgreSQL.

Martin Tibenský

Vyštudoval odbor Aplikovaná informatika na FPV UKF. Pocas štúdia sa zaoberal bunkovými automatmi, umelým životom a P2P prekryvnými sietami (hlavne BitTorrent protokol). Pracoval hlavne s programovacím jazykom Java, menej s C, SQL. Ovláda základy UML a dokumentácie projektov.

Juraj Vincúr

Absolvent 1. stupňa FIIT STU, INFO. V súčasnej dobe sa venuje vývoju aplikácií na platforme OSX a iOS (Python, Objective-C, AppleScript). 3 roky na pozícii script developer v 6-člennom tíme v oddelení automatizácie testovania softvéru (hodnotenie kvality, testovanie softvéru, IronPython/.Net). V záverečnej práci sa venoval vyhľadávaniu v zdrojových kódoch s prihliadaním na kontext a temporálnu dynamiku (C# .Net + VisualStudio Extensibility).

Ján Kalmár

Vyštudoval 1. stupeň na FIIT STU. Zaujíma sa o open source technológie, najviac GTK a Qt. Prevažne pracuje pod linuxom, nielen ako používateľ, ale aj ako programátor a správca. V minulosti robil správu linuxových serverov pre malú firmu poskytujúcu web, game a virtual server hosting. Najviac programuje v jazyku C++, ovláda aj C, Javu a C#. V záverečnej práci sa venoval významovej ekvivalencii, ktorú implementoval v C#/Mono.

Ondrej Jurcák

Momentálne sa venuje vývoju mobilných aplikácií pre platformu Android. Pracoval na analýze a návrhu multiplatformových aplikácií, primárne pre mobilné zariadenia (Android, iOS, Symbian) a viedol vývojový tím, ktorý realizoval tieto projekty. Taktiež rok pracoval na vývoji gui componentov pre platformu Windows mobile a Window CE v .Net Compact Frameworku.

Radoslav Zapach

Vyštudoval odbor Informatika na FIIT STU. Vo svojej bakalárskej práci sa venoval prehľadávaniu a získavaniu informácií z webu, za použitia crawlerov a programovacieho jazyka Java. Ďalej ovláda iné technológie a jazyky ako C, XML, SQL, UML a iné.

2.2 Rozdelenie manažérskych úloh

Meno	Pozícia
Pavol Pidanič	Manažér komunikácie
Juraj Vincúr	Manažér rozvrhu a plánovania
Radoslav Zápach	Manažér kvality
Ján Kalmár	Manažér dokumentácie
Martin Tibenský	Manažér rizík
Juraj Petrik	Manažér podpory vývoja

3 Metodika práce k tvorbe technickej dokumentácie

3.1 Úvod

3.1.1 Cieľ dokumentu

Dokument opisuje zásady použité pri tvorbe technickej dokumentácie. Opisuje ako dokumentáciu udržiavať prehľadnú a konzistentnú, takisto opisuje zásady dokumentovania práce na projekte (tvorby kapitol technickej dokumentácie/šprintov). Súčasťou tejto metodiky sú aj zásady pre tvorbu zápisníc zo stretnutí.

3.1.2 Určenie metodiky

Metodika je určená pre každého člena tímu, ktorý bude tvoriť technickú dokumentáciu k svojej práci. Zvlášť je určená pre vývojárov a dokumentaristov. Za dodržiavanie metodiky (formálnu stránku) a prípadne korekcie zodpovedá manažér dokumentácie. Za obsahovú stránku dokumentácie zodpovedá manažér kvality.

3.1.3 Slovník pojmov

Záznam – zdokumentovaný postup, výsledok alebo práca na danej úlohe

Úloha – zadanie práce tak, ako sa dohodlo na tímovom stretnutí

3.1.4 Zoznam súvisiacich metodík

1. Metodika pre vytváranie používateľskej príručky – ďalej len „metodika 1“
2. Štábná kultúra písania kódu pre tím č.12 – ďalej len „metodika 2“

3.1.5 Roly a zodpovednosti

Rola	Zodpovednosť
Dokumentarista	Formálna stránka a štruktúra dokumentácii, kontrola a úprava čiastkových dokumentácii prevzatých od ostatných členov tímu, pridávanie prevzatých dokumentácii do technickej dokumentácie. Správa zápisníc.
Programátor	Zmysluplné komentovanie zdrojových kódov v zmysle metodiky 2. Dokumentovanie vykonanej práce v zmysle odseku 3.2.2 tejto metodiky.
Správca systému Boinc	Zaznamenávanie potrebných krokov potrebných pre inštaláciu a správu systému Boinc a ich dokumentovanie v zmysle odseku 3.2.2 tejto metodiky.
Manažér kvality	Kontrola obsahovej stránky technickej dokumentácie.
Vedúci	Schvaľovanie technickej dokumentácie.

3.2 Všeobecné zásady

- Tvorba dokumentácie prebieha v Slovenskom jazyku.
- Dodržiava sa diakritika a pravidlá slovenského pravopisu.

- Používajú sa výstižné, krátke vety, pričom v jednej vete sa vyjadruje jedná myšlienka.
- Opisuje sa vecne ale výstižne.
- Zaužívané anglické pojmy sa neprekladajú.
- Všeobecné známe pojmy sa nepíšu do slovníka pojmov.
- Špecifické pojmy sa zapíšu do slovníku pojmov, kde sa aj stručne a vecne opíšu jednou alebo dvoma vetami.

3.3 Technická dokumentácia

Obsahuje všetky záznamy vykonaných prác na projekte. Za vytváranie technickej dokumentácie zodpovedá manažér dokumentácie. Manažér dokumentácie avšak nevytvára záznamy, ale ich len prijíma od ostatných členov tímu a následne ich upravuje a pripája k technickej dokumentácie.

3.3.1 Členenie dokumentu

Technická dokumentácia má nasledovné logické členenie:

- Titulná strana
- História zmien dokumentu
- Zoznam použitých skratiek
- Zoznam pojmov
- Obsah
- Úvod
- Šprinty
 - Opis šprintu
 - Záznamy vykonaných prác
- Zhrnutie
- Prílohy
 - Používateľská príručka (vytvorená podľa metodiky 1)
 - Inštaláčna príručka

3.3.2 História zmien dokumentu

Tabuľka, v ktorej sa zapisujú zápisy zmien dokumentu, štruktúra je nasledovná:

Dátum	Verzia	Zhrnutie zmien	Autor
6.11.2013	1.0	Vytvorenie dokumentu, formátovanie a štýly, vytvorenie kapitol úvod a prvý šprint	Ján Kalmár
7.11.2013	2.0	Vytvorenie kapitol druhý šprint	Ján Kalmár
7.11.2013	3.0	Vytvorenie kapitol tretí šprint	Ján Kalmár

V prípade zmeny v dokumente treba túto zmenu vždy zapísať do tabuľky zmien dokumentu, spolu so zhrnutím, dátumom a autorom zmeny. Verzie technickej dokumentácie sa riadia číslom šprintu, pričom prvá úroveň zodpovedá číslu šprintu a druhá úroveň zodpovedá prípadným úpravám

v dokumente v čase daného šprintu.

3.3.3 Členenie kapitol šprinty

Hlavnou časťou dokumentácie sú kapitoly zodpovedajúce jednotlivým šprintom v takom poradí ako za sebou reálne nasledovali. Šprinty sa dokumentujú vždy po skončení. Ich logická štruktúra je nasledovná:

3.3.4 Opis šprintu

- Jeden odstavec s opisom šprintu, jeho hlavné zameranie.
- Zoznam s hlavnými cieľmi stanovenými na stretnutiach prislúchajúcich k danému šprintu (ku každému šprintu sa konajú minimálne 2 stretnutia tímu).
- Jeden odstavec s opisom výsledku šprintu, kde sa stručne zanalyzuje či sa podarilo dosiahnuť stanovené ciele v šprinte.

3.3.5 Záznam vykonanej práce

V prípade, ak je úloha implementačná, tak je štruktúra nasledovná:

- Úloha – zadanie
 - Typ úlohy – implementačná
- Analýza
- Návrh riešenia
 - Opis vstupu
 - Postupnosť krokov riešenia
 - Opis predpokladaného výstup
- Implementácia
 - UML diagram – nepovinný, odporúčaný
- Testovanie

V prípade, ak je úloha analytická a nie implementačná, tak je štruktúra nasledovná:

- Úloha – zadanie
 - Typ úlohy - analytická
- Analýza daného problému
- Zhrnutie analýzy v kontexte zadania

V prípade, ak je úloha nasadiť určitý produkt na dané miesto, tak je štruktúra nasledovná:

- Úloha – zadanie
 - Typ úlohy - inštalačná
- Inštalácia

- Spustenie
- Testovanie

3.3.6 Vytvorenie záznamu

Záznam musí dokumentovať celú vykonanú prácu na úlohe.

KTO: Záznam vytvára osoba, ktorej bola pridelená daná úloha.

KEDY: Záznam sa vytvára priebežne počas práce na úlohe z dôvodu, aby aj menšie problémy boli zdokumentované a aby sa predišlo možným neskorším nezrovnalostiam so skutočne vykonanou prácou na úlohe.

Záznam, vytváraný podľa tejto metodiky obsahuje opis technického riešenia a práce vykonanej na danej úlohe. V zázname sa treba zamerať na dokumentovanie celej vykonanej práce relevantnej pre úlohu. Všeobecne známe postupy sa do záznamu nezapisujú. Problémy, s ktorými sa autor stretol počas práce na úlohe je odporúčené zdokumentovať a to:

- v prípade implementačnej úlohy v časti implementácia;
- v prípade inštaláčnej úlohy v časti inštalácia alebo spustenie, podľa toho, v ktorej fáze problém nastal.

3.4 Zápisnice zo stretnutí

Zápisnice patria k dokumentácii. Ich úloha je zdokumentovať priebeh tímových stretnutí. Do zápisnice sa zapíše priebeh stretnutia, téma stretnutia, výsledok stretnutia ale aj ktorý členovia tímu boli prítomný na stretnutí.

3.4.1 Členenie zápisnice

Zápisnica má nasledovnú štruktúru:

- Hlavička
 - Dátum
 - Čas
 - Miesto
 - Účastníci
 - Vypracoval
- Téma stretnutia
- Priebeh stretnutia
- Zadané úlohy do budúcnosti
- Revízia minulých úloh
- Zhodnotenie stretnutia

3.4.2 Vytvorenie zápisnice zo stretnutia

Zápisnica musí obsahovať vyplnenú hlavičku, priebeh stretnutia v kľúčových bodoch a úlohy.

KTO: Na začiatku stretnutia sa vyberie jeden účastník stretnutia (člen tímu), ktorý bude priebeh stretnutia zapisovať.

KEDY: Zápisnica sa vytvára po skončení stretnutia.

Zapisovateľ si počas stretnutia píše poznámky, zaznamenáva priebeh diskusie, zapisuje dohodnuté úlohy spolu s tým pre koho sú určené a zaznamenáva si ako sa pokročilo na úlohách z minulého stretnutia. Následne sa po stretnutí z poznámok vytvorí priebeh stretnutia, z úloh sa vytvoria zadané úlohy do budúcnosti a revízie minulých úloh. Nakoniec sa doplní téma stretnutia, zhodnotenie a vypíše sa hlavička zápisnice.

Príloha 1. Šablóna zápisnice

Distribúované počítanie na FIIT (FIIT grid)

747

Fakulta informatiky a informačných technológií STU

Boinc

Tím č. 12

Tímový projekt 2013/2014

Zápisnica z tímového stretnutia č. 1

Dátum:

Čas:

Miesto:

Účastníci

Vedúci projektu:

Ing. Peter Lacko, PhD.

Členovia tímu:

Bc. Ján Kalmár

Bc. Juraj Petrík

Bc. Juraj Vincúr

Bc. Martin Tibenský

Bc. Pavol Pidanič

Bc. Radoslav Zápach

Vypracoval:

Téma stretnutia:

V jednej vete o čom bolo celé stretnutie.

Priebeh stretnutia:

- V bodoch ako prebiehalo stretnutie.

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy

Zhodnotenie stretnutia:

Ako stretnutie dopadlo.

4 Štábna kultúra písania kódu pre tím č. 12

4.1.1 Cieľ dokumentu

Opisuje zásady a prístupy ako správane písať, udržiavať prehľadnosť a konzistentnosť zdrojových súborov.

4.1.2 Určenie metodiky

Táto metodika je určená pre každého člena vývojového tímu. Za dodržiavanie týchto ustanovení zodpovedá manažér kvality.

4.1.3 Slovník pojmov

- Plugin - Zásuvný alebo prídavný modul, ktorý rozširuje funkcie iného programu alebo ho dopĺňa.

4.1.4 Všeobecné ustanovenia

- Metodika vychádza z dokumentu Java Coding Conventions¹, ktorý rozširuje a dopĺňa o požiadavky pre tento projekt.
- Programuje sa v anglickom jazyku, vrátane písania komentárov.

4.2 Pokyny pre písanie zdrojových súborov

4.2.1 Formátovanie kódu

Každý odoslaný kód do repozitára (špecifikuje dokument Metodika verzionovania zdrojových súborov) v musí byť naformátovaný. Súbor `formatter.xml` obsahujúci šablónu formátu sa nachádza na stránke tímu. Manažér kvality zodpovedá za aktuálnosť súboru.

Formátovanie nastavíme vo vývojom prostredí Eclipse:

- *Windows -> Preferences -> Java -> Code Style -> Formatter*
- *Import -> vyberieme súbor -> Apply*

Zdrojový súbor formátuje klávesovou skratkou Ctrl + Shift + F.

Druhá možnosť je nastavenie automatického formátovania kódu v Eclipse:

- *Windows -> Preferences -> Java -> Editor -> Save Actions*
- Zaškrtneme
 - *Format source code + Format all lines*
 - *Organize imports*
- *Apply*

4.2.2 Organizácia súborov

Každá trieda musí byť v nejakom balíčku. Balíček zoskupuje zdrojové súbory, ktoré spolu logicky súvisia alebo je medzi nimi veľká previazanosť. Každý balíček pre tento projekt začína týmto prefixom:

1 <http://www.oracle.com/technetwork/java/javase/documentation/codeconvtoc-136057.html>

- sk.stuba.fiit.grid.

```
package sk.stuba.fiit.grid.utils;

public class MyClass {

    public void doSomething() {
        // do interesting stuff
    }
}
```

4.2.3 Formálna stránka kódu

Jednoznakové názvy premenných je zakázané používať. Výnimka platí, ak ide o súradnice, alebo názvy iterátorov.

Ak v cykloch alebo podmienkach sa nachádza jeden riadok, musí byť v množinových zátvorkách.

```
if (condition) {
    // do interesting stuff
} else {
    // do more interesting stuff
}
```

V prípade, ak meno rozhrania je podstatné meno, jeho implementácia obsahuje koncovku Impl.

```
public interface Document {
    void print();
}

public class DocumentImpl implements Document {

    @Override
    public void print() {
    }
}
```

Namiesto magických čísel sa používajú konštanty.

```
if (count > 10) {  
    // wrong, what does 10 mean  
}  
  
if (count > MAX_CAR_CAPACITY) {  
    // correct  
}
```

Používame foreach pred obyčajným for-cyklom. For cyklus sa použije v prípade, že sa iteruje iným spôsobom ako po jednom. V tom prípade for cyklus musí mať nasledujúcu formu:

```
for (initialization; condition; update) {  
}
```

Použiť číselníky, ak zdrojový súbor obsahuje číselné konštanty, ktoré slúžia ako vymenovanie rôznych možností. Udržiavanie číselných konštánt spôsobuje väčšie nároky na údržbu kódu (hlavne písanie podmienok a cyklov) a neposkytuje kontrolu typu ako číselníky.

```
public class Colors {  
    public static final int WHITE = 0;  
    public static final int BLACK = 1;  
    public static final int RED = 2;  
    public static final int GREEN = 3;  
    // inappropriate design of constants  
}  
  
public enum Colors {  
    WHITE, BLACK, RED, GREEN  
}
```

Číselníky môžu v sebe definovať aj hodnoty.


```

public enum Colors {
    WHITE(0xFFFFFFFF), BLACK(0x000000),
    RED(0xFF0000), GREEN(0x00FF00);

    private final int hexCode;

    private Colors(int hexCode) {
        this.hexCode = hexCode;
    }

    public int getHexCode() {
        return hexCode;
    }
}

```

4.2.4 Exceptions

Nie je povolené odchytať výnimky s prázdny telom bloku. Každá výnimka sa musí zalogovať a blok musí obsahovať akciu, ako naložiť s výnimkou. Aj vypísanie zásobníka na štandardný výstup je akcia.

```

try {
    // error might occur
} catch (Exception e) {
    // wrong
}

try {
    // error might occur
} catch (Exception e) {
    Logger.log(e.getMessage());
}

```

Výnimka platí, ak je jednoznačne jasné, že Exception nikdy nenastane, aj keď ju kód vyhadzuje. Blok s výnimkou musí obsahovať komentár s odôvodnením, prečo nenasleduje žiadna akcia.

```
private static final String TIME = "31.12.9999 23:59:59,999";
private static final String PATTERN = "dd.MM.yyyy HH:mm:ss,SSS";
private static final SimpleDateFormat FORMATTER = new SimpleDateFormat(PATTERN);

static Date getLastSecondsOfWorld() {
    Date result = null;
    try {
        result = FORMATTER.parse(TIME);
    } catch (ParseException e) {
        // this exception never occurs
        // all values are set as constants and parsing is under control
    }
    return result;
}
```

4.2.5 Písanie komentárov

Každá trieda a metóda musí byť zdokumentovaná použitím JavaDoc² komentárov. Pre lepšiu čitateľnosť sa používajú značky HTML.

```
/**
 * <p>
 * Implementation of Deep Thought computer based on the novel of Douglas
 * Adams The Hitchhiker's Guide to the Galaxy.
 * </p>
 * <p>
 * The computer was built to provide Answer to the Ultimate Question of
 * Life, the Universe, and Everything
 * </p>
 *
 * @author {author's name}
 *
 */
public class Hitchhiker {

    /**
     * <p>
     * Through archival recordings this method calculates Answer to the
     * Ultimate Question of Life, the Universe, and Everything.
     * </p>
     * <p>
     * This answer is incomprehensible because the beings didn't know what
     * they were asking.
     * </p>
     *
     * @return ultimate answer.
     *
     */
    public int getAnswerOfLife() {
        return 42;
    }
}
```

Povinné značky JavaDoc komentárov.

Pre triedy

- **@author** Autor(i) zdrojového súboru.
- **@param** Popis parametru, ak ide o generickú triedu.

Pre metódy

- **@param** Popis vstupného parametru metódy.
- **@return** Popis návratovej hodnoty metódy.
- **@throws** Popis výnimky, ktorá môže nastať vo vykonávaní metódy.

2 <http://www.oracle.com/technetwork/java/javase/documentation/index-137868.html>

4.2.6 Chyby a varovania

Odoslaný súbor nesmie obsahovať chyby ani varovania. V prípade, ak nie je iná možnosť a súbor obsahuje chyby, odoslať súbor so zakomentovanou chybou a popisom, ktorý začína značkou `// FIXME`. Následne sa vytvorí záznam v Redmine so značkou Bug. Zodpovedná osoba je manažér kvality. Ten na základe popisu rozhodne o ďalšom postupe, komu bude opravenie chyby pridelené.

Ak programátor nedokáže odstrániť warning, odošle súbor do repozitára s komentárom, warning sa nezakomentováva. Vytvorí záznam v Redmine. Manažér kvality rozhodne v rovnakom procese, ako pri chybe. Warning je možné odstrániť použitím anotácie `@SuppressWarnings`. Anotácia musí obsahovať komentár s odôvodením použitia anotácie.

4.2.7 Iné odporúčania

Odporúča sa bloky kódov vykonávajúce podobnú funkcionality oddeliť prázdny riadkom pre lepšiu čitateľnosť.

Odporúča sa nemať metódy a konštruktory s viac ako 3 vstupnými premennými.

4.3 Podporné nástroje

Pre lepšiu kvalitu napísaného kódu sa vyžaduje nainštalovanie a používanie pluginov vo vývojovom prostredí Eclipse. Vyžadované sú FindBugs³ a Checkstyle⁴.

Plugin nainštalujeme nasledovne:

- *Help -> Install New Software*
- Klikneme tlačidlo *Add*.
 - Uvedieme názov pluginu to kolonky *Name*.
 - Uvedieme url adresu pluginu do *Location*.
- *OK*.
- *Next -> Next -> Finish*.
- Reštartujeme Eclipse.

4.3.1 Findbugs

Používa sa na odhaľovanie chýb v kóde statickou analýzou kódu hľadaním tzv. bug patterns.

Url adresa pre plugin: <http://findbugs.cs.umd.edu/eclipse>.

Pred odoslaním zdrojových súborov do repozitára, každý vývojár nechá zvalidovať kód a to nasledovne:

- Pravým tlačidlom myši klikne na súbor/balíček/projekt, ktorý chceme nechať skontrolovať.
- *Find Bugs -> Find Bugs*

Otvorí sa okno s potencionálnymi chybami. Programátor na základe popisu odstráni chybu. Ak to nie je možné, vytvorí záznam v Redmine pre manažéra kvality. Ten rozhodne, ako sa odstráni chyba. Nie vždy je každú chybu možné odstrániť.

3 <http://findbugs.sourceforge.net/>

4 <http://checkstyle.sourceforge.net/>

4.3.2 Checkstyle

Nástroj, ktorý pomáha písať kód v Jave dodržiavaním všeobecných štandardov pre písanie kódu. Adresa pre plugin: <http://eclipse-cs.sourceforge.net/update/>.

Po nastavení automaticky kontroluje kód po každej kompilácii zdrojového súboru. Checkstyle nastavíme nasledovne:

- *Window -> Preferences -> Checkstyle -> tlačidlo New*
- *Type -> External Configuration File*
- Zvolíme názov
- Vyberieme cestu k súboru `checkstyle.xml` z disku.
- *OK*

Nastavenie automatickej kontroly.

- Pravé tlačidlo myši na projekt -> *Checkstyle -> Activate Checkstyle*

Checkstyle vyznačuje žltou farbou riadky, ktoré nespĺňajú podmienky definované v súbore `checkstyle.xml`. Súbor je možné stiahnuť zo stránky tímu. Každý programátor robí všetko preto, aby kód neobsahoval takéto značky. Všetky nezrovnalosti sa zaznamenávajú do Redmine a sú priradené manažérovi kvality. Ten zodpovedá aj za aktuálnosť súboru `checkstyle.xml`.

5 Testovanie zdrojového kódu

5.1.1 Cieľ dokumentu

Cieľom dokumentu je definovať prístupy a zásady, ktorých dodržiavanie vedie k tvorbe konzistentných testov jednotlivých častí implementácií projektu a výsledných testovacích protokolov. Dokument taktiež opisuje samotný proces testovania.

5.1.2 Určenie metodiky

Metodika Testovanie je určená pre každého člena vývojového tímu, ktorý zasahuje do zdrojových kódov projektu. Za dodržiavanie týchto ustanovení zodpovedá manažér kvality a manažér testovania.

5.1.3 Slovník pojmov

- *Testovacia sada* (angl. Test Suite) – súbor testovacích prípadov a testovacích sád
- *Testovací prípad* (angl. Test Case) – sada podmienok a premenných, ktoré určujú správnosť implementácie
- *Testovací protokol* – správa o výsledkoch testovania
- *Git* – distribuovaný systém na správu verzií

5.1.4 Všeobecné ustanovenia

- Názvy jednotlivých testovacích prípadov, sád a popisy sú písane v anglickom jazyku.
- Metodika je určená výlučne pre písanie testov pre implementácie v jazyku Java.

5.1.5 Roly a zodpovednosti

Rola	Zodpovednosť
Programátor	Vytváranie nových testov, vykonávanie existujúcich pri zásahoch do implementácie projektu.
Manažér testovania	Kontrola správnosti, štruktúry a organizácie testov a testovacích protokolov. Dohliada taktiež na správnu následnosť podprocesov procesu testovania.
Manažér kvality	Kontrola rozsahu pokrytia jednotlivých testov a výsledkov protokolov.

5.2 Pokyny pre vytváranie testov

5.2.1 Kedy vytvárať testy a kto

Testovacie prípady sa vytvárajú:

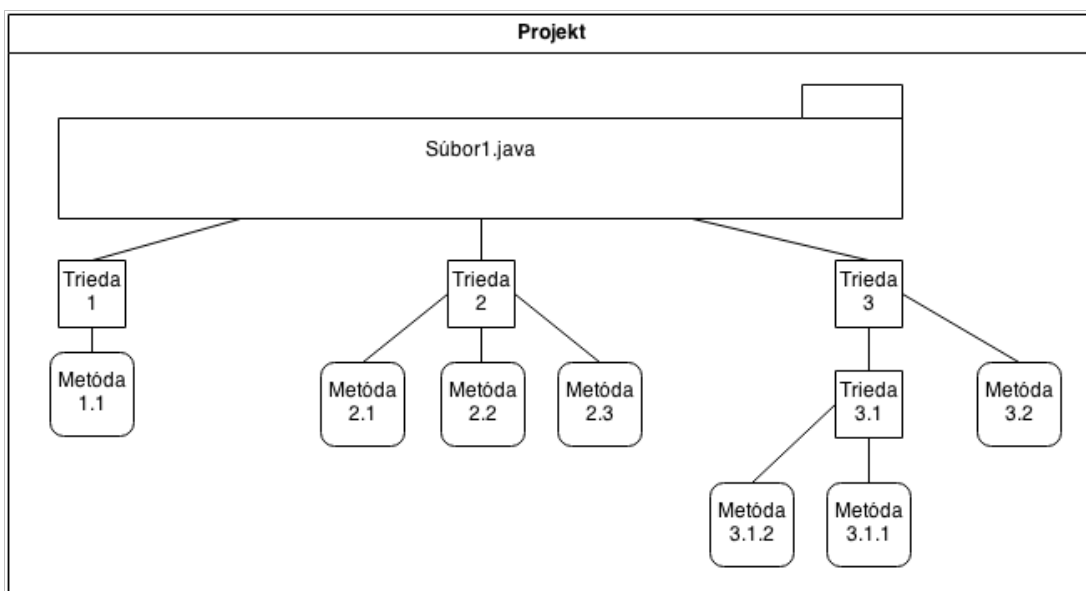
- pri pridávaní funkcionalít do implementácie, test vytvára ten, čo funkciu pridal
- pri odhalení chyby v metóde pre ktorú test doposiaľ nebol vytvorený, test vytvára autor chybnej metódy, ak to nie je možné, tak ten čo chybu objavil

5.2.2 Čo testovať

Testovacie prípady sú vytvárané len pre metódy, ktoré implementujú logickú alebo výpočtovú, nie triviálnu funkcionalitu. Tým je myslené, že daná metóda musí mať nejaké vstupy pre ktoré generuje výstupy.

5.2.3 Umiestnenie

Testovací prípad je v kóde implementovaný ako privátna statická metóda. Táto statická metóda testuje vždy práve jednu metódu implementovanú v súbore v ktorom sa nachádza. Jej názov je odvodený z názvu testovacej metódy pridaním prefixu TC_. Pokiaľ je testovaná metóda členom triedy, je táto metóda umiestnená do privátnej statickej triedy s rovnakým názvom, s pridaným prefixom TS_, ktorá reprezentuje testovaciu sadu. Celá implementácia sa vždy nachádza na konci súboru, za metódou main, ktorá volá všetky testovacie prípady.



Obrázok 1: príklad organizácie v projekte

Metóda 1.1	Testovací prípad 1.1	<i>Testovacia sada 1</i>	
Metóda 2.1	Testovací prípad 2.1	<i>Testovacia sada 2</i>	
Metóda 2.2	Testovací prípad 2.2		
Metóda 2.3	Testovací prípad 2.3		
Metóda 3.2	Testovací prípad 3.2		<i>Testovacia sada 3</i>
Metóda 3.1.1	Testovací prípad 3.1.1	<i>Testovacia sada 3.1</i>	
Metóda 3.1.2	Testovací prípad 3.1.2		

Tabuľka 2: zodpovedajúca kategorizácia k Obrázku 1

```
//IMPLEMENTATION
//...
public static void main(String[] args) {
    boolean overall = true;
    boolean ts_result = true;
    boolean tc_result;
    long currTime = System.nanoTime();
    tc_result = TS_class1name.TC_method1name(1, 2);
    // print result and duration time after each TC
    ts_result &= tc_result;
    tc_result = TS_class1name.TC_method2name(1, 3);
    ts_result &= tc_result;
    // print result and duration time after each TS
    overall &= ts_result;
    tc_result = TS_class2name.TC_method1name(1, 4);
    ts_result &= tc_result;
    tc_result = TS_class2name.TC_method2name(1, 5);
    ts_result &= tc_result;
    overall &= ts_result;
    // print overall result and duration time
}
private static class TS_class1name {
    private static boolean TC_method1name(final int in, final int out) {
        // here goes the logic
        return true; // If success else false
    }
    private static boolean TC_method2name(final int in, final int out) {
        // here goes the logic
        return false;
    }
}
private static class TS_class2name {
    private static boolean TC_method1name(final int in, final int out) {
        // here goes the logic
        return true;
    }
    private static boolean TC_method2name(final int in, final int out) {
        // here goes the logic
        return true;
    }
}
} //...
```

Obsah 1: jednoduchý príklad kategorizácie v kóde

5.2.4 Štruktúra

Testovací prípad v prvom rade obsahuje popis vo forme komentáru, ktorý sa skladá z týchto polí:

- *author* - meno autora
- *description* - stručný opis prípadu
- *in* - stručný opis vstupného parametra
- *out* - stručný opis výstupného parametra
- *return* - stručný opis návratovej hodnoty

Hlavnými atribútmi testovacieho prípadu je vstup pre testovanú metódu a jej očakávaný výstup, ktoré sú reprezentované ako jeho parametre. Vstupy pri testovaní musia byť také, aby vykonanie metódy netrvalo príliš dlho. Akceptovateľné sú rádovo sekundy. Očakávané výstupy musia byť určené manuálne alebo nástrojom, ktorého správnosť je overená. Za nimi nasleduje samotná logika vyhodnocovania. Ak daný test zbehol podľa tejto logiky úspešne, návratová hodnota je true, v opačnom prípade false.

```
/**
 * @author name
 * @description description
 * @param in
 * @param out
 * @return
 */
private static boolean TC_method1name(final int in, final int out) {
    return method1name(in) == out;
}
```

Obsah 2: jednoduchý príklad kategorizácie v kóde

5.2.5 Reprezentácia výsledkov

Výsledok testovania je nutné poslať na štandardný výstup. Pozostáva z týchto elementov:

- názov testovacej sady alebo prípadu
- reťazca FAIL ak metóda vrátila false, PASS ak true
- čas trvania

Na konci výsledkov musí byť celkové zhodnotenie testu a čas aký tento test trval. Odsadenie jednotlivých čiastočných výsledkov zodpovedá hĺbke v strome volaní. Výsledky pre jednu hĺbku musia byť v jednom stĺpci.

Konzola	
TS_class1name:	FAIL 0.10
TC_method1name:	PASS 0.01
TC_method2name:	PASS 0.02
TS_subclass1name:	FAIL 0.06
TC_method1name:	PASS 0.02
TC_method2name:	FAIL 0.04
OVERAL TEST RESULT:	FAIL 0.30 ms

Tabuľka 3: príklad výpisu na štandardný výstup

5.3 Pokyny pre testovanie

5.3.1 Kedy a čo testovať

Každý člen tímu je povinný spustiť pred každým commit-om testy vo všetkých zmenou ovplyvnených súboroch.

5.3.2 Výstupy

Pokiaľ test neprejde, člen tímu nemôže commit-núť zmeny a teda nevytvára žiadny výstup testovania. Špeciálnym prípadom je objavenie chyby v predchádzajúcom teste. Po odsúhlasení manažérom testovania môže vykonať commit, avšak do opisu musí pridať prečo a kým bola výnimka udelená a stručne opísať chybu predchádzajúceho testu.

Ak test prejde, člen tímu môže vykonať commit, ktorého správa obsahuje štandardný protokol.

5.4 Pokyny pre vytvorenie testovacieho protokolu

5.4.1 Umiestnenie

Testovací protokol sa vkladá na koniec štandardného opisu git commit-u, od ktorého je oddelený prázdny riadkom a následne nadpisom TESTING PROTOCOL.

5.4.2 Štruktúra

Protokol obsahuje údaje:

- kópiu reprezentácie výsledkov zo štandardného výstupu
- v prípade neúspechu nadpis EXCEPTION a doplňujúce informácie spomínané v kapitole 5.3.2 Výstupy

Commit správa

Štandardný opis commitu

TESTING PROTOCOL

```
TS_class1name:      FAIL 0.10
  TC_method1name:   PASS 0.01
  TC_method2name:   PASS 0.02
  TS_subclass1name: FAIL 0.06
    TC_method1name: PASS 0.02
    TC_method2name: FAIL 0.04
```

OVERAL TEST RESULT: FAIL 0.30 ms

EXCEPTION

by Pavol Pidanič
bug in test TS_subclass1name.TC_method2name
incorrect evaluation of bitboard

Tabuľka 4: príklad protokolu

6 Metodika pre písanie používateľskej príručky

6.1.1 Cieľ a obsah dokumentu

Cieľom dokumentu je opísať zásady a postupy záväzné pre vytváranie používateľskej príručky k systému Boinc. Používateľská príručka je určená pre študentov a výskumníkov, ktorí budú chcieť nasadiť, upraviť a udržiavať systém Boinc a programovať preň aplikácie na riešenie svojich problémov.

Tento dokument zahŕňa opis:

1. Rôl a zodpovedností
2. Formátovania, štruktúry a rozsahu príručky
3. Vytvorenia záznamu z dokumentácie k projektu
4. Vkladania a kompletizácie záznamov do dokumentovej časti používateľskej príručky
5. Informácie o vkladani a úprave záznamov pre wiki projektu
6. Príloha 1 – odporúčaný rozsah a obsah kapitol

6.1.2 Určenie metodiky

Metodika je určená pre každého člena tímu, ktorý sa zúčastní na inštalácii, udržiavaní, programovaní a dokumentácii projektu.

6.1.3 Definícia pojmov

Táto interpretácia pojmov je záväzná len pre tento dokument.

Záznam – budúca potencionálna časť používateľskej príručky, môže ho vytvoriť ktokoľvek z tímu.

Screenshot – grafické zachytenie aktuálneho stavu obrazovky. Po vyhotovení je potrebné ho orezať tak, aby zachytával len objekt záujmu. Primárne sa používa formát PNG:

Wiki – online encyklopédia vytvorená pre projekt. Nachádza sa na adrese <http://team12-13.ucebne.fiit.stuba.sk/wiki>.

6.1.4 Zoznam súvisiacich metodík

1. Metodika práce k tvorbe technickej dokumentácie – ďalej v dokumente ako metodika 1
2. Štátna kultúra písania kódu pre tím č.12 – ďalej v dokumente ako metodika 2

6.1.5 Roly a zodpovednosti

Roly a zodpovednosti vzhľadom na tvorbu používateľskej príručky.

Rola	Zodpovednosť
Dokumentarista	Nesie zodpovednosť za dokumentáciu k produktu, z ktorej bude príručka do veľkej miery čerpať informácie. Jeho úlohy a vlastnosti tohto dokumentu sú bližšie popísané v metodika 1.
Programátor	Zmysluplné a vyčerpávajúce komentovanie zdrojových kódov,

	ktoré sa neskôr použijú ako vzorové príklady. Formát a vlastnosti komentárov sú bližšie popísané v metodika 2.
Tvorca používateľskej príručky	Kontrola a úprava záznamov. Zodpovedá za formát a obsah používateľskej príručky.
Administrátor systému Boinc	Zodpovedá za inštaláciu a prevádzku systému, zaznamenávanie príkazov, činností a screenshotov, ktoré sa použijú v príručke.
Vedúci tímu	Kontroluje a schvaľuje jednotlivé verzie používateľskej príručky.

6.2 Formátovanie, štruktúra a rozsah príručky

V procese vytvárania záznamu a dokumentovej časti sa riadime rovnakým formátovaním ako pri písaní technickej dokumentácie, opísaním v metodike 1.

Príručka musí byť napísaná po slovensky a anglické termíny, ktoré sú zaužívané, alebo pre ne neexistuje vhodný preklad, musia byť opísané.

Príručka sa skladá z dvoch častí:

1. Dokumentová časť – súvislý dokument, vo finálnej verzii vo formáte PDF.
2. Wiki projektu – internetová encyklopédia.

Štruktúra dokumentovej časti zachytáva aj rozsah príručky všeobecne:

1. Titulná strana
2. Abstrakt
3. Obsah
4. Zoznam zmien – pomôcka len v pracovnej verzii
5. Úvod
6. Inštalácia
7. Správa a nastavenie systému
8. Programovanie v Boinc systéme
9. Nasadenie projektu
10. Slovník pojmov a skratiek

Odporúčania pre rozsah a obsah jednotlivých kapitol sa nachádzajú v prílohe č.1 tohto dokumentu.

6.2.1 Vytvorenie záznamu z technickej dokumentácie

Používateľská príručka sa do veľkej miery prekrýva s technickou dokumentáciou. Preto bude proces jej písania pozostávať z výberu a úpravy kapitol technickej dokumentácie.

KTO: Záznam vytvára tvorca používateľskej príručky, nemusí sa zhodovať s tvorcom pôvodnej kapitoly.

KEDY: Záznam sa vytvára zásadne vždy až po vytvorení a schválení prislúchajúcej kapitoly z technickej dokumentácie.

Jednotlivé kapitoly a podkapitoly z technickej dokumentácie sa líšia podľa typu úlohy, ktorú opisujú na analytické, implementačné a zachytávajúce proces nasadenia alebo inštalácie.

V používateľskej príručke sa analytická a implementačná časť zlučuje.

6.2.2 Úprava analytickej a implementačnej časti a výsledná štruktúra záznamu

Analytická časť sa upraví do formy stručného opisu problému. Nasleduje návrh riešenia z implementačnej časti, upravený do formy konečného riešenia, vrátane UML diagramov. Implementácia bude opísaná pomocou zdrojových kódov tých častí programu, ktoré využívajú metódy alebo funkcie špecifické alebo nutné pre aplikácie v Boinc systéme (Boinc API). Zdrojové kódy sú vyčistené od poznámok vytvorených pomocou nástroja na review kódu. Ak obsahujú dôležité informácie o niektorej časti kódu, prepíšu sa do formy štandardných komentárov. Testovanie sa uvádza len v prípade, že pri ňom boli použité funkcie systému Boinc.

Štruktúra záznamu:

1. Úloha – opis problému
2. Riešenie problému – UML diagramy a ich opis pre celé riešenie
3. Implementácia pre Boinc systém – zdrojové kódy pre časť riešenia špecifickú pre Boinc systém
4. Testovanie – v prípade, že testovanie prebieha v prostredí Boinc systému
5. Ďalšie odporúčania a komentáre – voliteľné, iba ak autor záznamu chce doplniť nejaké postrehy užitočné pre budúceho čitateľa

6.2.3 Úprava inštalačnej časti

Proces inštalácie alebo nasadenia musí obsahovať popis všetkých technológií a ich verzie. Každý významnejší krok musí byť zdokumentovaný aj pomocou screenshotu obrazovky, zachytávajúceho aktuálny stav obrazovky. Použité príkazy musia byť uvedené v textovej kopírovateľnej forme. V prípade, že sú použité knižnice z Boinc systému, musí k nim byť uvedená relatívna cesta v textovej kopírovateľnej forme.

Štruktúra záznamu:

1. Opis riešenej úlohy
2. Zoznam použitých technológií s ich presnou verzou (Apache – nesprávne, Apache 2.0.65 - správne)
3. Popis procesu v presných, jednoznačných krokoch

6.3 Dokumentová časť príručky

KTO: Tvorca používateľskej príručky

KEDY: Priebežne po vytváraní záznamov. Finálna verzia po uzavretí posledného šprintu a kompletizácii technickej dokumentácie.

Dokumentová časť tvorí jeden súvislý dokument, ktorý bude po kompletizácii distribuovaný v PDF formáte. Za kompletizáciu, úpravu a výsledné formátovanie zodpovedá Tvorca používateľskej príručky a nie autori jednotlivých záznamov. Dokument je súčasťou produktu, preto ho okrem vedúceho tímu schvaľuje priebežne aj produktový vlastník. Štruktúra je popísaná v časti *Formátovanie, štruktúra a rozsah príručky*. Odporúčaný rozsah a obsah kapitol je uvedený v prílohe 1.

Postup pre vkladanie záznamu do dokumentovej časti:

1. Vybraný záznam pridáme do pracovnej verzie dokumentu. Pôvodný záznam zachováваме v

nezmenenej podobe.

2.Upravíme jeho formátovanie.

3.Vykonané zmeny, dátum, autora pôvodnej verzie záznamu a autora zmien zaznamenáme do zoznamu zmien.

Štruktúra tabuľky, v ktorej sa nachádza zoznam zmien je nasledovná:

Dátum	Popis zmien	Autor pôvodného záznamu	Zmeny vykonal
12.5.1986	Pridaná kapitola o inštalácii webserveru Z kapitoly o inštalácii webserveru odstránené meno autora	Paľo	Riško

Vo finálnej verzii príručky tabuľku so zoznamom zmien odstránime. Archivuje sa posledná pracovná verzia príručky. V tejto verzii je posledný záznam v tabuľke *Schválenie finálnej verzie*, ktorý pridá vedúci tímu.

6.4 Wiki projektu

KTO: Všetci členovia tímu.

KEDY: Priebežne pri vytváraní záznamov. Wiki projektu sa po skončení projektu neuzatvára, prispievať do nej môžu aj neskorší používatelia systému.

Pri vkladaní obsahu sa riadime rozsiahlou dokumentáciou k wiki systému dostupnou na: <http://meta.wikimedia.org/wiki/Help:Contents>.

Zásady platné pre vkladanie záznamu do wiki:

- 1.Štruktúra zostáva rovnaká ako pri zázname.
- 2.Každý záznam sa nachádza na samostatnej stránke wiki.
- 3.Každý termín musí byť prepojený so svojou hlavnou stránkou pomocou wikilink.
- 4.V prípade odkazovania sa na oficiálnu dokumentáciu alebo iné stránky sa používajú referencie.
- 5.Obrázky sa vkladajú primárne vo formáte PNG.
- 6.Vzorcie sa vkladajú ako obrázky vo formáte PNG.

Príloha 1 – Odporúčany obsah jednotlivých kapitol

Príručka musí obsahovať nasledujúce kapitoly, určujúce jej štruktúru:

1. Titulná strana – obsahuje aj číslo verzie príručky
2. Úvod – všeobecné informácie o Boinc systéme, predstavenie schémy a pojmov. Na vyššej úrovni (menej podrobne) uviesť ako medzi sebou komunikujú jednotlivé súčasti systému – hlavne assimilator, transitioner, validator, scheduler, feeder, work generator, task server, data server.
3. Inštalácia – popísať krok po kroku ako sme postupovali pri inštalácii serverovej časti Boincu do nami vybraného prostredia (Debian, Apache2, MySQL). Uvádzať vždy konkrétne príkazy aj s vysvetlením čo robia. Zaznamenávať čo najpresnejšie verzie systémov, s ktorými sme pracovali. Vždy keď je to vhodné uviesť screenshot z obrazovky – ako by mala v tom momente vyzeráť (ale neuvádzať potrebné príkazy len takto, ale aj vo forme aby sa dali kopírovať ako text).
4. Nastavenie jednotlivých súčastí – podrobne opísať všetky vlastnosti a atribúty jednotlivých súčastí vymenovaných v úvode príručky. Každá časť spracovaná v samostatnej podkapitole. V prípade, že niektorú časť treba naprogramovať alebo modifikovať, treba uviesť aj časti programového kódu vo formáte, ktorý je uvedený v metodike 2. Kód treba vyčistiť od nevhodných poznámok vytvorených vrámci review kódu, vhodné poznámky treba prepísať do formy štandardných komentárov.
5. Programovanie v Boinc systéme – v úvode popísať Boinc API. Špecifiká pre jazyky Java a C uviesť do samostatných podkapitol. Uviesť hlavne príklady vo forme kódu.
6. Nasadenie projektu – pomocou vzorového projektu uviesť ako nasadiť projekt v Boinc systéme.
7. Slovník pojmov a skratiek – jednou vetou opísať objekt a ak je to možné, odkázať na strany príručky, kde sa nachádza. Použiť abecedné poradie.

7 Metodika práce k verzionovaniu

7.1.1 Cieľ dokumentu

Tento dokument opisuje základné princípy práce so systémom na verzionovanie zdrojového kódu, konkrétne systému git. Ďalej tento dokument opisuje prácu s git-om v IDE Eclipse s použitím rozšírenia Egit.

7.1.2 Určenie metodiky

Metodika je určená pre každého člena tímu, ktorý sa podieľa na vývoji.

7.1.3 Slovník pojmov

IDE – z angl. integrated development environment, integrované vývojové prostredie,

Git – distribuovaný systém na správu verzií,

Egit – rozšírenie pre IDE Eclipse, ktoré uľahčuje prácu s git-om.

Redmine – aplikácia na manažovanie projektov,

Repozitár – úložisko údajov,

7.1.4 Roly a zodpovednosti

Manažér podpory vývoja	<ul style="list-style-type: none">• Vytvorenie repozitára.• Správa repozitára.
Developer	<ul style="list-style-type: none">• Klonovanie repozitára.• Písanie kódu.• Riešenie kolízie.• Pridávanie kódu do repozitára.

7.1.5 Všeobecné pokyny

- Commit (aktualizácia súborov) sa vykonáva hneď, ako je to možné, t.j. zmeny sú už hotové.
- Program musí ísť vždy spustiť, aby ste nebránili v práci ostatným členom tímu.

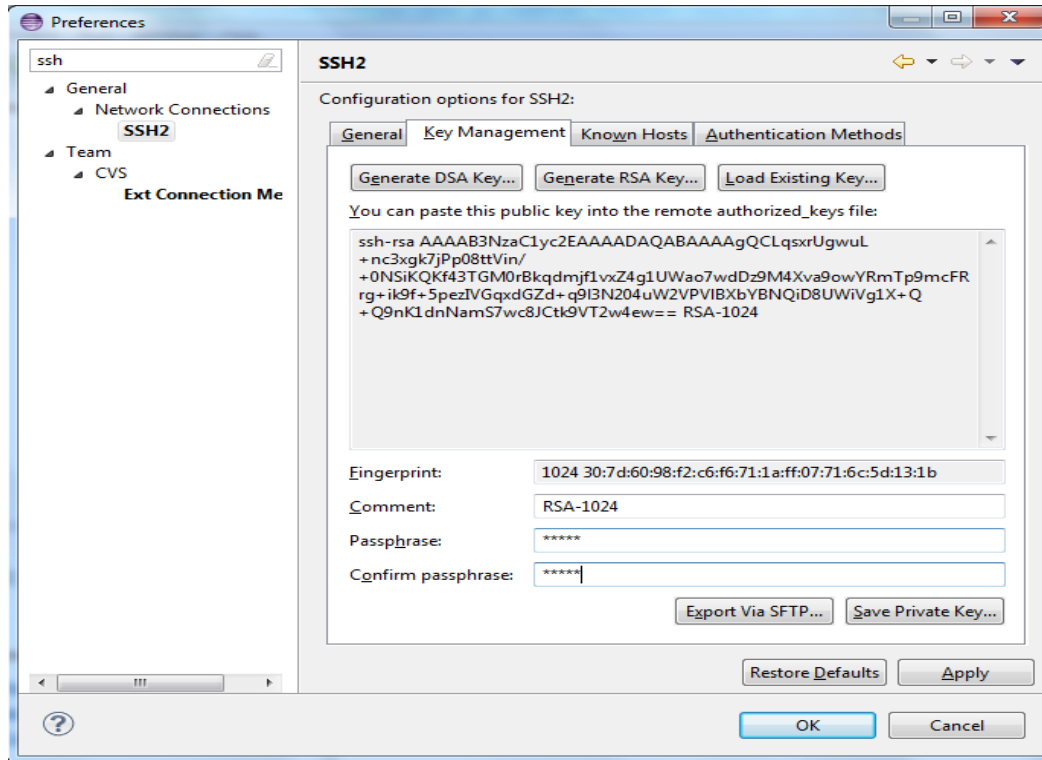
7.2 Využívanie verzionovacieho systému git v IDE Eclipse

7.2.1 Inštalácia pluginu EGit do Eclipse

- 1) Kliknúť v menu postupne na Help->Eclipse Marketplace
- 2) Do Find napísať egit.
- 3) Kliknúť na install pri Egit-e.
- 4) Potvrdiť potrebné veci a nainštalovať.

7.2.2 Generovanie a vloženie kľúča do FIIT Gitu

- 1) Kliknúť v menu postupne na Window->Preferences
- 2) Vybrať General->Network Connections->SSH2
- 3) Vybrať tab Key Management
- 4) Vyplniť heslo, ktoré sa neskôr použije pri importe
- 5) Kliknúť na Generate RSA Key...

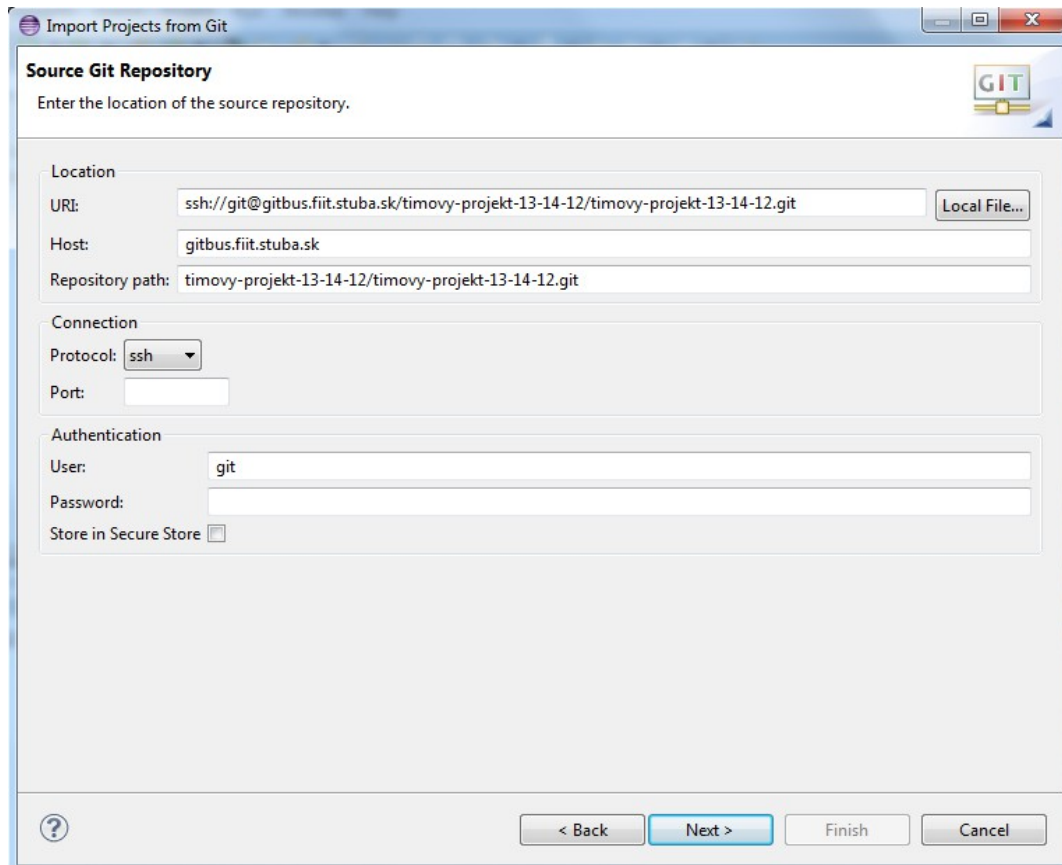


- 6) Kliknúť na Save Private Key... a potvrdiť
- 7) Kliknúť na Apply
- 8) Skopírovať public key do clipboardu,
- 9) Reštartnúť Eclipse
- 10) Prihlásiť sa do <http://gitbus.fii.stuba.sk/>, meno a heslo sú rovnaké ako do AIS,
- 11) Kliknúť na Manage SSH keys
- 12) Kliknúť na Add SSH key
- 13) Vložiť public key a kliknúť na save
- 14) Po reloadede stránky by mal byť key ready

7.2.3 Import projektu (projektov) z GIT repozitára

- 1) Kliknúť v menu na File->Import
- 2) Vybrať Git->Projects from Git
- 3) Kliknúť na Next

4) Vybrať Clone URI a kliknúť na Next



- 5) Vložiť do URI ssh://git@gitbus.fiit.stuba.sk/timovy-projekt-13-14-12/timovy-projekt-13-14-12.git
- 6) Kliknúť na Next
- 7) Vybrať master a kliknúť na Next
- 8) Kliknúť na Next
- 9) Vybrať Import existing projects a kliknúť na Next
- 10) Vybrať projekty a kliknúť na Finish

7.2.4 Commit' a push do repozitára

Slúži na uloženie zmien do repozitára

- 1) Kliknúť pravým tlačidlom myši v Project Exploreri na projekt.
- 2) Vybrať postupne Team->Commit
- 3) Vyplniť Commit message (najlepšie niečo zmysluplné :))
- 4) A kliknúť na Commit and Push

Pri každom commite je potrebné vyplniť správu o vykonaných zmenách. Táto správa dodržiava tieto zásady:

- Na začiatku správy musí byť uvedené meno a priezvisko v hranatých zátvorkách. Napr.

[Juraj Petrik].

- Nasleduje prázdny riadok a za ním samotná správa.
- V správe sú uvedené vykonané zmeny, nie je potrebný podrobný opis, sntčí stručné a zrozumiteľné zhrnutie.
- V správe je možné sa odvolávať na tickety zo systému Redmine.
- Nepoužívame diakritiku.

[Juraj Petrik]

Opravené generovanie tahov v triede Generator v metode generate.

Issue #6315

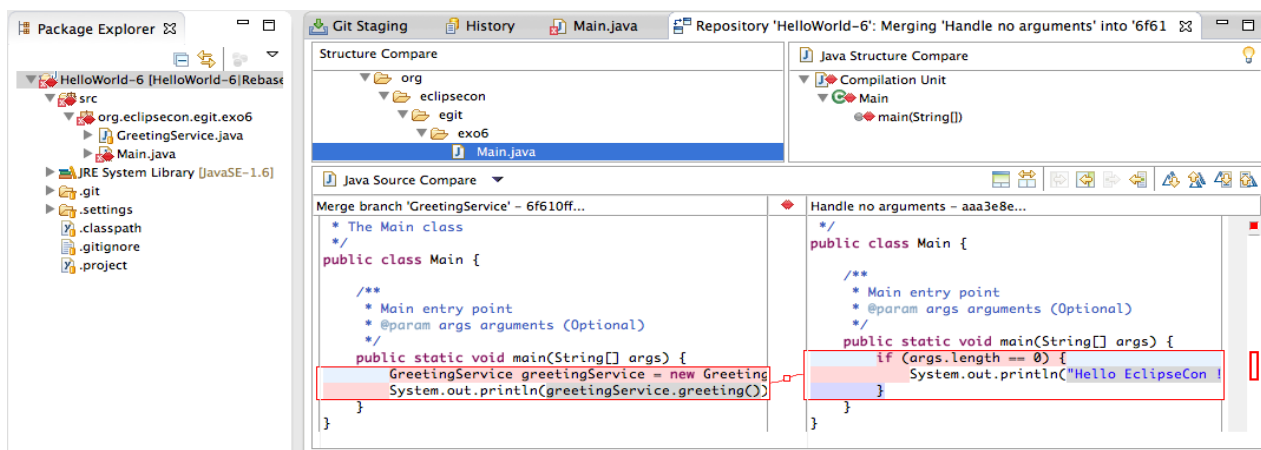
7.2.5 Stiahnutie aktuálnych dát z repozitára

Vždy pred začatím práce na projekte je treba toto vykonať, inak môžu vzniknúť rôzne konflikty.

- 1) Kliknúť pravým tlačidlom myši v Project Exploreri na projekt.
- 2) Vybrať postupne Team->Pull

7.2.6 Riešenie konfliktov

Na riešenie konfliktov používame „merge tool“, ktorý je priamo integrovaný v plugine Egit.



Kliknúť pravým tlačidlom myši v Project Exploreri na projekt.

- 1) Vybrať postupne Team->Merge Tool.

Nikdy neprepisujte cudzí kód, ktorý nevíete na čo slúži.

Po vyriešení konfliktov je treba overiť funkčnosť programu, konkrétne, či funkcionality ostala zachovaná.

8 Príloha A: Zápis z stretnutí za zimný semester

Zápisnica z tímového stretnutia č. 1

Dátum: 3.10.2013
Čas: 12:00 – 13:45
Miesto: Jobsovo softvérové štúdio
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrík
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Ján Kalmár

Téma stretnutia:

Zoznámenie sa s podrobnosťami a organizačnými vecami ohľadom projektu.

Priebeh stretnutia:

- Zoznámenie sa s vedúcim
- Diskutovanie ohľadom projektu
- Pýtali sme sa na organizačné veci, ohľadom virtualného servera, čo nás čaká a neminie, aké sú naše predstavy a predstavy vedúceho
- Spravili sme návrhy tém na počítanie
 - zapojenie sa do bitcoin – toto asi skôr nie ako áno
 - prelomenie hesla na pdf
 - DNA
 - neuronové siete
- Dostali sme inštrukcie k webu, mala by to byť nejaká statická stránka, nie dynamická, niečo ako iba HTML + CSS
- Rozprávali sme sa o Git-e, spustenie, nastavenie a údržbu sme zverili do rúk (prstov) Jurajovi Petrikovi
- Správa serverov, inštalácia, nastavovanie a ďalšie tým súvisiace veci prešli na plecia Janovi Kalmárovi
- V tíme sme si prideliť roly:
 - Pidanič - vedúci, zástava roly manažment komunikácie, manažment ľudských zdrojov, manažment rozvrhu
 - Tibenský - manažment rizík, monitorovanie projektu spojené aj s plánovaním testovania
 - Kalmár - manažment dokumentácie, manažment kvality, manažment integrácie

projektu

- Zápach - manažment dokumentácie, manažment kvality
 - Petrik - manažment podpory vývoja, manažment rozsahu projektu
 - Vincúr - manažment rozvrhu, monitorovanie projektu spojené aj s plánovaním testovania, manažment rozsahu projektu
- Rozdelili sme si úlohy do budúcnosti

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Vypracovať plagát	Pavol Pidanič	6.10.2013
Spraviť šablónu pre zápisnice	Ján Kalmár	5.10.2013
Rozbehať niekde Git a spraviť KRÁTKU príručku k jeho používaniu	Juraj Petrik	TBD
Zistiť čosi viac o TPCuPe + Prihláška	Martin Tibenský	10.10.2013
Nájsť vhodné nástroje na vytvorenie webového sídla	Juraj Vincúr	10.10.2013
Porozmýšľať nad ďalšími úlohami, ktoré by sme mohli riešiť distribuovane	Všetci	10.10.2013

Zhodnotenie stretnutia:

Bolo to naše prvé stretnutie s vedúcim, kde sme sa dozvedeli čosi viac k projektu a zdelili sme si úlohy. Takisto sme si rozdelili roly, ktoré budeme zastávať v tíme.

Zápisnica z tímového stretnutia č. 2

Dátum: 10. 10. 2013

Čas: 12:00 – 13:30

Miesto: Jobsovo softvérové štúdio

Účastníci

Vedúci projektu:	<input checked="" type="checkbox"/> Ing. Peter Lacko, PhD.	
Členovia tímu:	<input checked="" type="checkbox"/> Bc. Ján Kalmár	<input checked="" type="checkbox"/> Bc. Juraj Petrík
	<input checked="" type="checkbox"/> Bc. Juraj Vincúr	<input checked="" type="checkbox"/> Bc. Martin Tibenský
	<input checked="" type="checkbox"/> Bc. Pavol Pidanič	<input checked="" type="checkbox"/> Bc. Radoslav Zápach

Vypracoval: Pavol Pidanič

Téma stretnutia:

Diskusia ohľadom výpočtových úloh a zahájenie 1. šprintu.

Priebeh stretnutia:

- Zahájenie stretnutia
- Zhodnotenie stavu úloh z predchádzajúceho stretnutia.
- Diskusia ohľadom výpočtových úloh. Diskutované možnosti:
 - vypočítať π na niekoľko miliónov desatinných miest
 - počítanie veľkých prvočísel
 - problém obchodného cestujúceho
 - návrh spraviť na inej platforme ako Boinc
 - hackovanie pdf dokumentov
 - pomôcť SAV s nejakým zložitým výpočtom
 - prehľadávať herný strom hry – hra Reversi (Othello), strom riešenia pre rozlohu hracieho poľa 8×8 nebol vypočítaný.
- Rozdelenie úloh členom tímu s termínom vypracovania.
- Pokúsiť sa nainštalovať Boinc klient na čo najviac počítačov. Možnosť použiť školské počítače, fakultný cluster, oznámiť známym.
- Návrh na krátke neformálne stretnutia buď pred alebo po prednáške Tímový projekt.
- Ukončenie stretnutia

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Vytvoriť zdieľaný priečinok tímu	Ján Kalmár	13. 10. 2013
Vytvoriť dokument s popisom prístupu na tímový server	Ján Kalmár	13. 10. 2013
Nasadiť stránku tímu	Juraj Vincúr	13. 10. 2013
Príručka a testovací projekt pre Git	Juraj Petrík	13. 10. 2013

Podieľať sa niečím na stránke	Všetci	17. 10. 2013
Vytvoriť projekt v Redmine	Pavol Pidanič	13. 10. 2013
Plugin do Redmine pre Kanban	Pavol Pidanič	13. 10. 2013
Urobiť backlog	Pavol Pidanič	17. 10. 2013
Naštudovať princíp Boinc	Všetci	17. 10. 2013
Spísať príručku pre Boinc	Martin Tibenský	17. 10. 2013
Technika programovania v Boinc	Juraj Vincúr	17. 10. 2013
Naštudovať princíp hry Reversi (Othello)	Všetci	17. 10. 2013
Spísanie základného stromu riešenia hry Reversi	Radoslav Zápach	17. 10. 2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Vypracovať plagát	Pavol Pidanič	Hotová
Spraviť šablónu pre zápisnice	Ján Kalmár	Hotová
Rozbehať Git a spraviť krátku príručku k jeho používaniu	Juraj Petrík	Rozpracovaná
Zistiť čosi viac o TPCuPe + Prihláška	Martin Tibenský	Rozpracovaná
Nájsť vhodné nástroje na vytvorenie webového sídla	Juraj Vincúr	Hotová
Porozmýšľať nad ďalšími úlohami, ktoré by sme mohli riešiť distribuovane	Všetci	Rozpracovaná

Zhodnotenie stretnutia:

Stretnutie slúžilo ako príprava na 1. šprint. Nasledoval tím event v Lander Café.

Zápisnica z tímového stretnutia č. 3

Dátum: 17.10.2013
Čas: 12:00 – 13:45
Miesto: Jobsovo softvérové štúdio
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrik
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Martin Tibenský

Téma stretnutia:

Kontrola splnenia úloh dohodnutých na minulom stretnutí. Pokračovanie diskusie s riešeným paralelným problémom.

Priebeh stretnutia:

- Zahájenie stretnutia diskusiou o problémoch, ktoré sa vyskytli počas týždňa.
- Overenie zapojenia sa všetkých členov do prostredí, ktoré budeme používať pri programovaní a organizovaní práce – RedMine (ešte čakáme aj na nainštalovanie KanBan pluginu), FIIT GitBus
- Ján Kalmár podal správu o stave nasadenia BOINC – systém bol úspešne nasadený, celý tím sa s ním začal zoznamovať priamo na serveri
- Ďalšia diskusia ohľadom paralelných problémov
 - možnosti riešenia hry Reversi o veľkosti 8x8 (úlohou je lepšie analyzovať vhodné algoritmy, nie iba Alfa-Beta osekávanie)
 - DNA – úlohou je preskúmať súčasné BOINC projekty zaoberajúce sa touto problematikou, zistiť formát dát, s ktorým môžeme pracovať
- Diskusia o možnosti používania JAVY ako programovacieho jazyka úloh, počas študovania systému sme zistili, že by to mohlo byť problematické, keďže JAVA wrapper pre BOINC neposkytuje väčšinu API funkcionality
- Z vyššie uvedeného bodu vyvstáva potencionálna nutnosť používať C a teda by sme nemohli používať Perkoníka- popasovať sa s týmto problémom
- Správa serverov, inštalácia, nastavovanie a ďalšie stým súvisiace veci prešli na plecía Janovi Kalmárovi
- Postupne sme si zadelili úlohy na ďalší týždeň
- Počas týždňa sme mali viacero neformálnych stretnutí, väčšinou len časti tímu

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Preskúmať možnosti systému BOINC, čo sa týkajú	Pavol Pidanič	24.10.2013

programovania v Java		
Naštudovať algoritmy pre hru Reversi (táke, ktoré budú zahadzovať čo najviac nepotrebných stavov)	Radoslav Zápach Juraj Petrík	24.10.2013
Analyzovať existujúce riešenia pre BOINC DNA	Martin Tibenský	24.10.2013
Napísať draft prihlášky na TP CUP	Martin Tibenský	24.10.2013
Zistiť možnosti BOINC API	Juraj Vincúr	24.10.2013
Pohrať sa s backlog pluginom	Pavol Pidanič	24.10.2013
Správa BOINC serveru (+ workunity, otestovanie Hello World)	Ján Kalmár	24.10.2013
Vytvoriť aplikáciu, ktorá bude obsahovať checkpoint a pošle nejaký result serveru	Juraj Vincúr (C) Juraj Petrík (Java)	24.10.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Vytvoriť zdieľaný priečinok tímu	Ján Kalmár	Hotová
Vytvoriť dokument s popisom prístupu na tímový server	Ján Kalmár	Hotová
Nasadiť stránku tímu	Juraj Vincúr	Hotová
Príručka a testovací projekt pre Git	Juraj Petrík	Hotová
Podieľať sa niečím na stránke	Všetci	Hotová
Vytvoriť projekt v Redmine	Pavol Pidanič	Hotová
Plugin do Redmine pre Kanban	Pavol Pidanič	Rozpracovaná
Urobiť backlog	Pavol Pidanič	Hotová
Naštudovať princíp Boinc	Všetci	Hotová
Spísať príručku pre Boinc	Martin Tibenský	Hotová
Technika programovania v Boinc	Juraj Vincúr	Hotová
Naštudovať princíp hry Reversi (Othello)	Všetci	Hotová
Spísanie základného stromu riešenia hry Reversi	Radoslav Zápach	Hotová

Zhodnotenie stretnutia:

Od minulého týždňa sme sa všetci zoznámili so softvérovými pomôckami, fungovaním Boincu a môžeme pokračovať v hlbšom riešení nastolených problémov.

Zápisnica z tímového stretnutia č. 4

Dátum: 24.10.2013
Čas: 12:00 – 14:30
Miesto: Jobsovo softvérové štúdio
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrík
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Juraj Vincúr

Téma stretnutia:

Uzavretie a revízia úloh aktuálneho šprintu, naplánovanie budúceho. Debata o správe serverovej časti a o možných i aktuálnych problémoch spojených s implementáciou prototypu aplikácie v jazyku JAVA.

Priebeh stretnutia:

- Zahájenie stretnutia diskusiou o problémoch, ktoré sa vyskytli počas týždňa.
- Podanie správ členov tímu k minulým úlohám a uzavretie sprintu.
- Ladenie nastavení pluginu backlog.
- Diskusia k možnostiam BOINC API a k možným problémom spojených s použitím wrappera pre JAVU.
- Navrhnutie použitia JNI pre používanie funkcií BOINC API z JAVY.
- Skúmanie možných používateľských nastavení BOINC klienta.
- Navrhnutie vytvorenia WIKI pre používateľov.
- Krátka diskusia k problematike DNA.
- Experimentovanie s prototypom hry reverzi.
- Definovanie funkcionálnej a štrukturálnej stránky generátora úloh.
- Krátka diskusia k možným reprezentáciám stavov. String vs Int.
- Výber optimalizačných algoritmov prehľadávania stavového priestoru pre budúce prototypy.
- Ohodnotenie identifikovaných budúcich úloh action point-mi a ich rozdelenie.

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Wiki	Ján Kalmár	07.11.2013
Generátor úloh	Ján Kalmár Juraj Vincúr	29.10.2013
Implementácia prototypu prehľadávania stavov s použitím alfa-beta osekávania	Pavol Pidanič Martin Tibenský	29.10.2013
Implementácia prototypu prehľadávania stavov	Juraj Petřík	29.10.2013

stavov s použitím MTD(f)	Radoslav Zápach	
Inštalácia a vyskúšanie nástroja PERCONIK	Všetci	7.11.2013
Preskúmať možnosti JNI	Nepridelená	7.11.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Preskúmať možnosti systému BOINC, čo sa týkajú programovania v Java	Pavol Pidanič	Hotová
Naštudovať algoritmy pre hru Reversi (táke, ktoré budú zahadzovať čo najviac nepotrebných stavov)	Radoslav Zápach Juraj Petrík	Hotová
Analyzovať existujúce riešenia pre BOINC DNA	Martin Tibenský	Hotová
Napísať draft prihlášky na TP CUP	Martin Tibenský	Hotová
Zistiť možnosti BOINC API	Juraj Vincúr	Hotová
Pohrať sa s backlog pluginom	Pavol Pidanič	Hotová
Správa BOINC serveru (+ workunity, otestovanie Hello World)	Ján Kalmár	Hotová
Vytvoriť aplikáciu, ktorá bude obsahovať checkpoint a pošle nejaký result serveru	Juraj Vincúr (C) Juraj Petrík (Java)	Hotová

Zhodnotenie stretnutia:

Úspešné zavŕšenie prvého a plánovanie druhého šprintu trvalo dlhšie ako sme očakávali. Počas stretnutia sme identifikovali niekoľko problémov, no pre všetky sme dokázali navrhnúť adekvátne riešenia.

Zápisnica z tímového stretnutia č. 5

Dátum: 4.11.2013
Čas: 14:00 – 15:30
Miesto: Jobsovo softvérové štúdio
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrík
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Radoslav Zápach

Téma stretnutia:

Kontrola splnenia pridelených úloh, diskusia k ich riešeniu. Pokračovanie diskusie s riešením paralelným systémom.

Priebeh stretnutia:

- Názov projektu na serveri (grid.fiit.stuba.sk)
- Krátka diskusia k daemonovi na generovanie workunitov na serveri - vytvorenie vlastného.
- Diskusia ku generovaniu ťahov - bitové operácie.
- Podanie správ členov tímu k minulým úlohám.
- Diskusia k spracovávaní výstupných súborov klienta na serveri - ich názvov.

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Generátor úloh	Ján Kalmár Juraj Vincúr	7.11.2013
Implementácia prototypu prehľadávania stromu stavov s použitím MTD(f)	Juraj Petrík Radoslav Zápach	7.11.2013
Inštalácia a vyskúšanie nástroja PERCONIK	Všetci	7.11.2013
Preskúmať možnosti JNI	Nepridelená	7.11.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Wiki	Ján Kalmár	Hotová
Generator uloh	Ján Kalmár Juraj Vincúr	Rozpracovaná
Implementácia prototypu prehľadávania stromu stavov s použitím alfa-beta osekávania	Pavol Pidanič Martin Tibenský	Hotová
Implementácia prototypu prehľadávania stromu stavov s použitím MTD(f)	Juraj Petrík Radoslav Zápach	Rozpracovaná

Inštalácia a vyskúšanie nástroja PERCONIK	Všetci	Rozpracovaná
Preskúmať možnosti JNI	Nepridelená	Rozpracovaná

Zhodnotenie stretnutia:

Oboznámili sme sa s progressom na zadaných úlohách a prediskutovali postup na ich ďalšom riešení.

Zápisnica z tímového stretnutia č. 6

Dátum: 7.11.2013
Čas: 12:00
Miesto: Jobsovo softvérové štúdio
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrík
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Juraj Petrík

Téma stretnutia:

Nasadzovanie prototypu pre hru reversi 6x6.

Priebeh stretnutia:

- Generátor – vygenerované sú súbory pre hĺbku 8, zaberajú 207MB.
- Bol vytvorený na serveri projekt reversi, do ktorého sa budú pridávať aplikácie na počítanie.
- Mtdf – bola zistená neefektivita pri veľmi veľa prehľadaných stavoch, kvôli transpozičnej tabuľke, riešením sa zdá byť ukladať do tabuľky iba uzly z určitých hĺbok.
- Perconik – ide bez problémov na najnovšom eclipse.
- JNI – zdá sa byť jednoduchá implementácia. Nutné prekompilovanie knižníc, pokiaľ ich chceme použiť.
- Diskusia ohľadom použitia stringov a bitboardov pri generovaní ťahov – bitboardy sa zdajú byť niekoľkokrát rýchlejšie ako implementácia pomocou stringu.

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Asimilátor	Pavol Pidanič Martin Tibenský	21.11.2013
Scheduler	Ján Kalmár	21.11.2013
Nasadenie prototypu	Juraj Petrík	14.11.2013
Optimalizácie prototypu	Juraj Vincúr	14.11.2013
Možnosti optimalizácií pre hru reversi	Radoslav Zápach	14.11.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Generátor úloh	Ján Kalmár Juraj Vincúr	Hotová
Implementácia prototypu prehľadávania stavov s použitím MTD(f)	Juraj Petrík Radoslav Zápach	Hotová

Inštalácia a vyskúšanie nástroja PERCONIK	Všetci	Hotová
---	--------	--------

Zhodnotenie stretnutia:

Sme pripravení na nasadenie prototypu.

Zápisnica z tímového stretnutia č. 7

Dátum: 14.11.2013

Čas: 12.00

Miesto: Softvérové štúdio

Účastníci

Vedúci projektu:	<input checked="" type="checkbox"/> Ing. Peter Lacko, PhD.	
Členovia tímu:	<input checked="" type="checkbox"/> Bc. Ján Kalmár	<input checked="" type="checkbox"/> Bc. Juraj Petrík
	<input checked="" type="checkbox"/> Bc. Juraj Vincúr	<input checked="" type="checkbox"/> Bc. Martin Tibenský
	<input checked="" type="checkbox"/> Bc. Pavol Pidanič	<input checked="" type="checkbox"/> Bc. Radoslav Zápach

Vypracoval: Ján Kalmár

Téma stretnutia:

Revízia úloh minulého stretnutia,

Priebeh stretnutia:

- Prešli sme si úlohy z minulého stretnutia
- Zreferovali sme progres
- Riešila sa optimalizácia prototypu pre riešenie hry reversi
- Dohodli sme sa na nasadení projektu

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Asimilátor	Pavol Pidanič Martin Tibenský	21.11.2013
Scheduler	Ján Kalmár	21.11.2013
Praca na prototypu	Juraj Petrík	21.11.2013
Vytvoriť nový Boinc projekt s nejakým lepším názvom (niečo ako grid)	Ján Kalmár Juraj Petrík	21.11.2013
Odtestovať heuristiky	Radoslav Zápach	21.11.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Asimilátor	Pavol Pidanič Martin Tibenský	Rozpracovaná
Scheduler	Ján Kalmár	Rozpracovaná
Nasadenie prototypu	Juraj Petrík	Rozpracovaná
Optimalizácie prototypu	Juraj Vincúr	Rozpracovaná
Možnosti optimalizácií pre hru reversi	Radoslav Zápach	Hotová

Zhodnotenie stretnutia:

Na stretnutí sme prešli aktuálny stav šprintu.

Zápisnica z tímového stretnutia č. 8 Šprint č. 4

Dátum: 21. 11. 2013

Čas: 10:00 – 13:00

Miesto: Jobsovo softvérové štúdio

Účastníci

Vedúci projektu:	<input type="checkbox"/> Ing. Peter Lacko, PhD.	
Členovia tímu:	<input checked="" type="checkbox"/> Bc. Ján Kalmár	<input checked="" type="checkbox"/> Bc. Juraj Petrík
	<input checked="" type="checkbox"/> Bc. Juraj Vincúr	<input checked="" type="checkbox"/> Bc. Martin Tibenský
	<input checked="" type="checkbox"/> Bc. Pavol Pidanič	<input checked="" type="checkbox"/> Bc. Radoslav Zápach

Vypracoval: Pavol Pidanič

Téma stretnutia:

Cieľom ôsmeho stretnutia bolo ukončiť a zhodnotiť šprint č. 3 a zahájiť šprint č. 4 definovaním príbehov.

Priebeh stretnutia:

- Na začiatku stretnutia sme zhodnotili výstupy 3. šprintu:
 - Prototyp hry Reversi 6×6 bol nasadený na tímový server. Je dostupný pod linkou team12-13.ucebne.fiit.stuba.sk/reversi
 - Vygenerovali sme celkovo 52 127 výpočtových úloh.
 - Posielanie úloh funguje.
 - Všetci členovia tímu (vrátane vedúceho projektu) sa pripojili k projektu a začali počítať jednotlivé úlohy.
 - K prototypu bol implementovaný asimilátor výsledkov. Zatiaľ nefunguje ako daemon proces.
 - Do začiatku stretnutia asimilátor vyhodnotil, že z celkového množstva úloh bolo vrátených 121 výsledkov.
 - Aplikáciám pre klientom momentálne chýba niekoľko vlastností, ktoré by ju činili používateľsky prístupnou. Nie je možné prerušiť výpočet ukončením aplikácie alebo vypnutím počítača a následné pokračovanie od posledne vypočítaného stavu.
- Dokumentácia k prvým trom šprintom a k riadeniu je kompletná, zverejnená na stránke tímu. Na stretnutí nebol vedúci projektu Ing. Peter Lacko, PhD. Vytlačená verzia dokumentu mu bude odovzdaná na najbližšom stretnutí, príp. v inom dohodnutom čase.
- Ďalej stretnutie pokračovalo diskusiou o programovacom jazyku pre pokračovanie projektu. Diskusia sa niesla v znamení nevýhod jazyka Java oproti jazyku C/C++.
 - Jazyk Java nie je priamo podporovaný platformou BOINC, čo značne komplikuje vývoj.
 - Aktuálna klientská aplikácia napísaná v Jave je pamäťovo náročná. Riešenie spomaľuje aj Garbage Collector Java Virtual Machine.
- Otázka zmeny programovacieho jazyka je stále otvorená. K návrhu sa musí vyjadriť aj vedúci projektu, ktorý žiaľ nebol prítomný.
- Na odstránenie pamäťových nárokov budeme musieť obmedziť činnosť Garbage Collectora.
 - Aplikácia využíva rekurzívne volanie funkcií.

- Bez rekurzívnych funkcií by sme vedeli obmedziť automatické čistenie pamäte JVM využitím statického poľa, na ktorý nemá Garbage Collector dosah počas behu programu.
- Juraj Vincúr navrhol použitie aplikácie Edax⁵. Je to vyvíjaná aplikácia napísaná v jazyku C pre hru Reversi 8×8.

Pre šprint č. 4 sme definovali 3 body, na ktoré sa zameriame:

- Optimalizovanie existujúceho prototypu v Java - Pokúsime sa prepísať klientskú aplikáciu odstránením rekurzívnych volaní funkcií a znížiť celkové pamäťové nároky výpočtu.
- Vytvoriť prototyp v jazyku C/C++ - implementujeme klientskú aplikáciu v natívne podporovanom jazyku pre platformu BOINC a porovnáme časovú a pamäťovú náročnosť výpočtu.
- Napísanie príručiek pre používanie BOINC – návody zverejníme na wiki stránke tímu <http://team12-13.ucebne.fiit.stuba.sk/wiki/> a dokument pribudne na tímovej web stránke.

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Upraviť Edax pre veľkosť hracieho poľa 6×6	Juraj Vincúr	5. 12. 2013
Porovnať nároky C/C++ aplikácie oproti Java	Juraj Vincúr	5. 12. 2013
Napísať aplikáciu v C/C++ pre klientov	Ján Kalmár	5. 12. 2013
Vytvoriť štruktúru a napísať návody pre BOINC na wiki	Ján Kalmár	5. 12. 2013
Používateľská príručka – dokument	Martin Tibenský	5. 12. 2013
Prepísať aktuálny prototyp odstránením rekurzie	Juraj Petřík Radoslav Zápach	5. 12. 2013
Doplňovanie unit testov pre prototyp	Pavol Pidanič	5. 12. 2013
Prispieť na wiki stránku	Všetci	5. 12. 2013
Implementovanie checkpoint-u pre klientské aplikácie	Úloha bude pridelená až po úprave aplikácie odstránením rekurzie	
Optimalizovať pamäťové nároky Java aplikácie	Úloha bude pridelená až po úprave aplikácie odstránením rekurzie	

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Asimilátor	Pavol Pidanič Martin Tibenský	Hotová
Scheduler	Ján Kalmár	Rozpracovaná
Nasadenie prototypu	Juraj Petřík	Hotová

⁵ <http://code.google.com/p/edax-reversi/>

Optimalizácie prototypu	Juraj Vincúr	Hotová
Možnosti optimalizácií pre hru reversi	Radoslav Zápach	Hotová

Zhodnotenie stretnutia:

Uzavreli sme šprint č. 3 a zahájili šprint č. 4.

Zápisnica z tímového stretnutia č. 9 Šprint č. 4

Dátum: 28.11.2013
Čas: 12:00 – 17:00
Miesto: Jobsovo softvérové štúdio – tam kde sa každý prekríkuje a nič nefunguje
Účastníci Vedúci projektu: Ing. Peter Lacko, PhD.
Členovia tímu: Bc. Ján Kalmár Bc. Juraj Petrík
 Bc. Juraj Vincúr Bc. Martin Tibenský
 Bc. Pavol Pidanič Bc. Radoslav Zápach

Vypracoval: Juraj Vincúr

Téma stretnutia:

Stretnutie č. 9 sa prevažne zamieravalo na navrhnutie finálnej podoby prototypu pre hru reversi s hracou plochou veľkosti 6x6. Počas tohto stretnutia sme taktiež zaviedli otvorené diskusie k možnostiam riešenia jednotlivých úloh, ktoré sme definovali.

Priebeh stretnutia:

- Zhodnotenie súčasného stavu.
- Identifikácia problému prázdnej Wiki.
- Poukázanie na nutnosť značkovania zdrojového kódu aplikácie Edax.
- Diskusia k blížiacim sa udalostiam a termínom (ScrumPanel, MSI riadenie).
- Zmena termínu budúceho stretnutia na 11:00.
- Identifikovanie výhod a nevýhod z hľadiska efektivity pri spúšťaní jednej aplikácie multithread-ovo, alebo viacerých nezávislých aplikácií paralelne.
- Zavedenie dodatočných diskusií k implementačným problémom zadaných úloh.
- Diskusia k asimilátoru a generátoru. Navrhnutie viacerých prístupov implementácie, zváženie kladov, záporov a zložitosti jednotlivých návrhov. Následné vyhodnotenie diskusie.

Zadané úlohy do budúca:

Úloha	Pridelená	Termín vypracovania
8x8 benchmark hĺbok	Juraj Vincúr	6.12.2013
Generátor pre Edax	Ján Kalmár	9.12.2013
Prezentácia na ScrumPanel	Pavol Pidanič	2.12.2013
CodeReview Edax 6x6	Všetci	9.12.2013
Asimilátor s postupným vyhodnocovaním a mazaním nepotrebných workunit-ov	Pavol Pidanič Juraj Petrík Radoslav Zápach	9.12.2013
Definovať action points	Všetci	5.12.2013
Identifikovať vhodný plugin do eclipse pre code review (jazyk C)	Martin Tibenský	6.12.2013

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Upraviť Edax pre veľkosť hracieho poľa 6×6	Juraj Vincúr	Rozpracovaná
Porovnať nároky C/C++ aplikácie oproti Java	Juraj Vincúr	Rozpracovaná
Napísať aplikáciu v C/C++ pre klientov	Ján Kalmár	Rozpracovaná
Vytvoriť štruktúru a napísať návody pre BOINC na wiki	Ján Kalmár	Rozpracovaná
Používateľská príručka – dokument	Martin Tibenský	Rozpracovaná
Prepísať aktuálny prototyp odstránením rekurzie	Juraj Petřík Radoslav Zápach	Rozpracovaná
Doplňanie unit testov pre prototyp	Pavol Pidanič	Rozpracovaná
Prispieť na wiki stránku	Všetci	Neprebieha
Implementovanie checkpoint-u pre klientské aplikácie	Úloha bude pridelená až po úprave aplikácie odstránením rekurzie	Neprebieha
Optimalizovať pamäťové nároky Java aplikácie	Úloha bude pridelená až po úprave aplikácie odstránením rekurzie	Neprebieha

Zhodnotenie stretnutia:

Zavedenie diskusií k zadaným úlohám sa osvedčilo. Členom tímu bolo hneď po stretnutí jasné akým spôsobom majú pri riešení úloh postupovať. Jedinou nevýhodou tohto prístupu je podstatné predĺženie stretnutí.

Zápisnica z tímového stretnutia č. 10 Šprint č. 5

Dátum: 5.12.2013

Čas: 11:00

Miesto: Jobsovo softvérové štúdio

Účastníci

Vedúci projektu:	<input checked="" type="checkbox"/> Ing. Peter Lacko, PhD.	
Členovia tímu:	<input checked="" type="checkbox"/> Bc. Ján Kalmár	<input checked="" type="checkbox"/> Bc. Juraj Petrík
	<input checked="" type="checkbox"/> Bc. Juraj Vincúr	<input checked="" type="checkbox"/> Bc. Martin Tibenský
	<input checked="" type="checkbox"/> Bc. Pavol Pidanič	<input checked="" type="checkbox"/> Bc. Radoslav Zápach

Vypracoval: Martin Tibenský

Téma stretnutia:

Počas stretnutia sme preberali stav zadaných úloh, vykonali retrospektívu minulého šprintu a navrhli riešenia problémov, ktoré sme počas nej identifikovali.

Priebeh stretnutia:

- Retrospektíva – na konci zápisnice
- Zhodnotenie súčasného stavu riešených úloh:
 - Diskusia o stave prototypu, možnosť využitia Boinc API priamo v Edax klientskej aplikácii.
 - Diskusia ohľadom toho, či sa stihne vytvoriť generátor a asimilátor do termínu nasadenia prototypu
- Diskusia o nasledovných témach:
 - Uvoľnilo sa miesto na fakultnom clustri, možnosť použiť ho na testovanie aplikácií
 - Zváženie, či by nebolo vhodnejšie vzhľadom na charakter ukladaných dát vrátených od klientov používať NoSQL databázu
 - Dohodli sme sa, že budeme ukladať aj ťahy perfektnej hry
 - Návrh na využitie techniky extreme programming – pair programming
 - Príprava na prezentáciu k riadeniu – utorok o 9:00 v Jobsovom softvérovom štúdiu

Zadané úlohy do budúcnosti:

Úloha	Pridelená	Termín vypracovania
Aspiration search v Edaxe	Juraj Petrík	12.12.2013
Získať prístup ku clustru	Juraj Vincúr	12.12.2013
Prezentácia k riadeniu	všetci	11.12.2013
Generátor pre Edax	Ján Kalmár	10.12.2013
Asimilátor pre Edax	Radoslav Zápach	10.12.2013
Nasadenie prototypu Edax	Juraj Petrík Juraj Vincúr	12.12.2013

Používateľská príručka – dokument Používateľská príručka – wiki	Martin Tibenský	12.12.2013
--	-----------------	------------

Revízia minulých úloh:

Úloha	Pridelená	Stav úlohy
Generátor pre Edax	Ján Kalmár	rozpracovaná
Asimlátor	Radoslav Zápach	rozpracovaná
Používateľská príručka – dokument Používateľská príručka – wiki	Martin Tibenský	rozpracovaná
Doplňovanie unit testov pre prototyp	Pavol Pidanič	hotová
Identifikovať vhodný plugin do eclipse pre code review (jazyk C)	Martin Tibenský	rozpracovaná

Zhodnotenie stretnutia:

Na stretnutí sme sa dohodli na termíne definitívneho nasadenia prototypu aplikácie, analyzovali sme retrospektívu k minulému šprintu a navrhli riešenia ako odstrániť nedostatky.

Retrospektíva:

Start doing

- napísať zápisnicu zo stretnutia do 24 hodín
- zabezpečiť fyzickú tabuľku pre tasky
- zabezpečiť lepšie formy komunikácie
- zlepšiť delenie úloh

Stop doing

- prestať chodiť neskoro na stretnutia
- prestať jesť, piť, odbehovať a inak prerušovať stretnutia
- na oficiálnych stretnutiach sa nezaoberať vecami, čo sa netýkajú tímového projektu

Continue doing

- redmine
- verzionovanie
- pokračovať v neoficiálnych stretnutiach, aj keď nie v kompletnej zostave

Action points do budúceho šprintu:

1. Trest za nedodržiavanie disciplíny – pridať stránku na wikipediu
2. Zabezpečiť real-time komunikáciu pomocou skype, vymeniť si telefónne kontakty v prípade nutnosti riešenia urgentného problému
3. Robiť retrospektívy a sledovať, ktoré AP sme nedodržali a prečo
4. Zlepšiť granularitu a definovanie pridelených úloh, zaviesť rozsiahlejšiu diskusiu k úlohám
5. Vyskúšať techniky extreme programming, konkrétne pair programming