**Retrospektíva šprintu**

**4.Šprint**

Úlohy, ktoré sme si pridelili do druhého šprintu sú nasledovné:

|  |  |  |  |
| --- | --- | --- | --- |
| **Číslo úlohy** | **Zhrnutie úlohy** | **Riešiteľ** | **Zodpovedný** |
| ROBOCUPTP-77 | Dokončenie ZMP a testovanie | Metod Rybár | Metod Rybár |
| ROBOCUPTP-78 | Automatické testy - Bamboo | Michal Segeč | Michal Segeč |
| ROBOCUPTP-79 | Oddelenie testovania | Peter Filípek | Miroslav Wolf |
| ROBOCUPTP-80 | Odstraňovanie nepoužívaných časti kódu | Juraj Šimek | Juraj Šimek |
| ROBOCUPTP-81 | Analýza Code Detectoru | Martin Vrabec, Peter Filípek, Miroslav Wolf | Martin Vrabec |
| ROBOCUPTP-83 | Odstraňovanie warning hlásení | Michal Segeč, Miroslav Wolf, Martin Vrabec | Miroslav Wolf |

Vo štvrtom šprinte sme sa zamerali na odstraňovanie nepotrebných častí hráča a rôznych chýb v kóde, ktoré by mohli v budúcnosti spôsobovať problémy a boli aj vývojárskym prostredím označené ako varovania. Zároveň sme nasadzovali automatické testy a dokončovali sme implementáciu Zero moment point, ktorá sa v predchádzajúcom šprinte ukázala komplikovanejšia ako sme plánovali.

ROBOCUPTP-77- Dokončenie ZMP a testovanie

Úloha bola pridelená Metodovi Rybárovi, ktorý na implementácií Zero moment point pracoval aj v predchádzajúcom šprinte. Po tom ako sa implementovali hlavné výpočtové funkcie sa v tomto šprinte tvoril nový HighSkill, ktorý Zero moment point využíval. Implementácia bola úspešná, ale ukázalo sa, že pri prechode z Ruby na Javu sa zmenili niektoré veci, kvôli ktorým v novom HighSkille napríklad hráč automaticky nevstal po tom ako spadol. Hráčova chôdza bola však očividne stabilnejšia, a preto v budúcnosti treba HighSkill upraviť tak, aby fungoval v Javovskej implementácií a zároveň sa Zero moment point môže využívať aj pri iných HighSkilloch.

**ROBOCUPTP-78- Automatické testy - Bamboo**

Úloha bola pridelená Michalovi Segečovi. Nástroj bol úspešne nasadený, avšak pri refraktoringu kódu boli niektoré testy zakomentované a po ich odkomentovaní automatické testy prestali fungovať. Je preto nutné v ďalšom šprinte tento problém odstrániť.

**ROBOCUPTP-79- Oddelenie testovania**

Tejto úlohe sa venoval Peter Filípek. Testy presunul do balíčku tests a zá- roveň aj upravil classpath. Juraj Šimek našiel jeden test, ktorý však nebol Unit test a teda nebol ani presunutý. Zároveň sa pri tom odhalil problém so zakomentovanými testami spomínaný v ROBOCUPTP-78- Automatické testy – Bamboo

**ROBOCUPTP-80- Odstraňovanie nepoužívaných časti kódu**

Úlohe sa venoval Juraj Šimek a v rámci nej odstránil najmä nevyužívane HighSkilly. Zároveň upravil niektoré triedy, v ktorých CodeDetector našiel problém a odstránil duplicity na výpočet vzdialenosti vektora. Kód celkovo zreduloval o približne 2000 riadkov kódu. V rámci úlohy vytvoril dokumentáciu k HighSkillom.

**ROBOCUPTP-81** – Analýza Code Detectoru

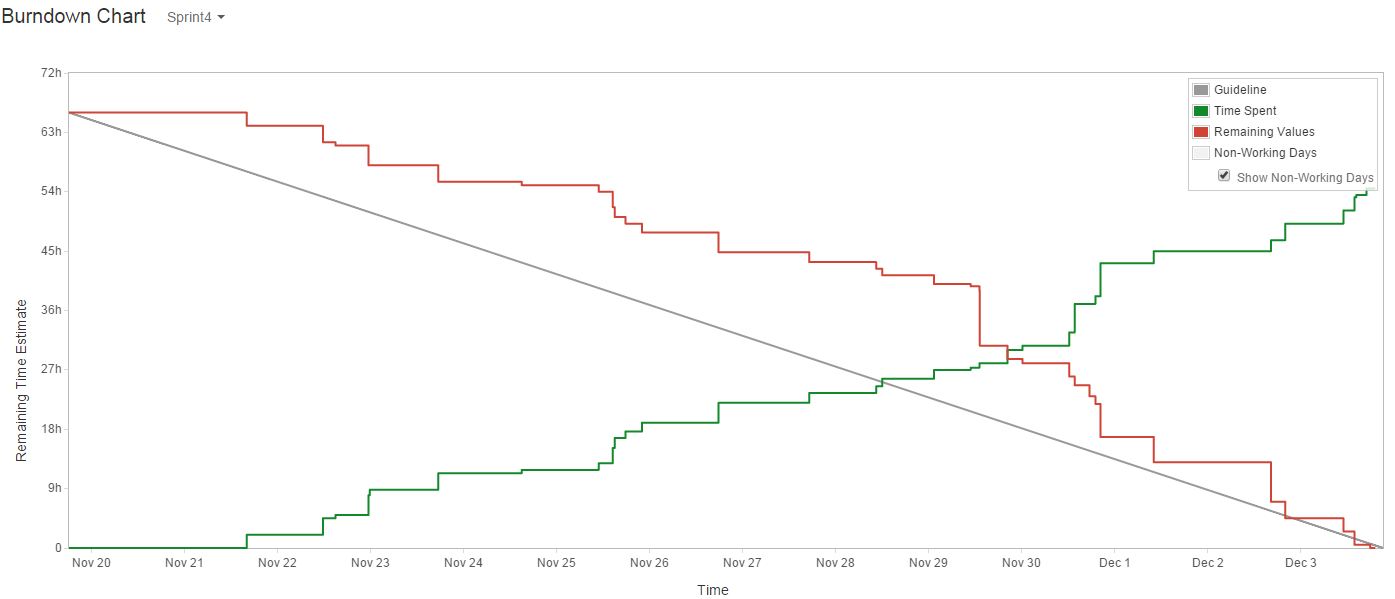
Úlohu mal na starosti Martin Vrabec a venovali sa jej Martin Vrabec, Peter Filípek a Miroslav Wolf. Najväčšiu časť odstránil Miroslav Wolf a bola 18 splnená rýchlejšie ako sa predpokladalo. Kód sa zanalyzoval pomocou pluginu CodeDetector do IDE Eclipse, ktorý slúži na alalýzu kódu a odhaľuje napríklad nevolané metódy. Tieto metódy sa skontrolovali a označili sa buď na odstránenie alebo ponechanie. Pri nerozhodných sa diskutovalo s tímom a rovnako sa o nich rozhodli. Následne sa nepotrebný kód odstránil.

**ROBOCUPTP-83-Odstraňovanie warning hlásení**

Úlohu mal na starosti Miroslav Wolf a pracovali na nej Michal Segeč, Miroslav Wolf a Martin Vrabec. Na základe warningov detekovaným IDE Eclipse sa vykonala analýza ku a odstraňovali sa warningy, ktorá by v budúcnosti mohli spôsobovať problémy. Najčastejšie problémy boli nepoužité metódy alebo premenná, prípadne ich viditeľnosť. V agentovi sa odstránili všetky warningy, RoboCup Library a TestFramework boli nezmenené, kedže sa na nich ešte nepracovalo a môžu sa podstatne upravovať. Zredukovaný počet warningov bol z 936 na 582.

**Zhodnotenie štvrtého šprintu**

V rámci šprintu sa nám podarilo podstatne zredukovať počet riadkov kódu odstránením nepotrebných častí, implementovať Zero moment point, nasadiť automatické testovanie a rovnako sme zredukovali warningy a tým šancu budúcich problémov. Ukázalo sa, že sme príliš podcenili implementáciu Zero moment point, a to najmä kvôli nášmu nepresnému odhadu urobených zmien pri prechode z Ruby do Javy minulý rok. Rovnako sa vyskytli problémy s pri presúvaní testov a automatickom testovaní, ktoré sa budú musieť v budúcnosti odstrániť.



Graf 1: Burndown chart 4.šprintu