Karol Katerżawa
Artur Sawicki

Wstępna wstępna dokumentacja ZPR

prowadzący: Rafał Biedrzycki

1. Cel projektu

Tematem projektu jest implementacja gry Neuroshima Hex (http://neuroshimahex.pl). W ramach prac chcemy umożliwić grę na jednym komputerze dla paru osób, grę przeciwko graczom komputerowym oraz grę przez sieć. Aplikacja ma być przenośna między różnymi systemami operacyjnymi.

2. Wymagania funkcjonalne

l.p.	Nazwa	Opis (opcjonalny)	Priorytet*
1	Gra na jednym komputerze	Umożliwienie gry na jednym komputerze dla 2 – 4 graczy.	
1.1	Plansza do gry	Pole gry, przyciski, pionki, znaczniki, itp.	1
1.2	Logika gry	Implementacja zasad gry	1
1.3	Ekran menu głównego	Ekran startowy aplikacji, umożliwiający wybór między opcjami.	2
1.4	Przegląd statystyk	Zbieranie statystyk przeprowadzonych gier.	3
1.5	Tryb teatru	Przegląd historii gier z mozliwością ich odtworzenia.	4
1.6	Zapis stanu gry	Moliwość wznowienia gry od momentu zapisania.	3
2	Gra przeciwko komputerowi	Umożliwienie gry przeciwko sztucznej inteligencji.	
2.1	Komputer wykonujący ruchy		1
2.2	Wybór poziomu trudności		2
3	Gra sieciowa		
3.1	Łączenie z innymi graczami		1
3.2	Wybór gry mieszanej		2
4	Gra przez przeglądarkę		
4.1	Możliwość podłączenia się do gry	Klient przeglądarkowy.	1

*Główne wymagania posortowane według ważności. Priorytet w obrębie głównych wymagań,

3. Wymagania niefunkcjonalne

l.p.	Nazwa	Opis	Priorytet
1	Przenośność	Aplikacja działa pod systemami Windows i Linux.	1
2	Przepustowość serwera	Serwer umożliwia połączenie do dwudziestu graczy.	3
3	Bezpieczeństwo	Informacje użytkowników chronione	3

4	Sprzęt	Obsługa myszki i klawiatury	1
5	Środowisko graficzne	Umożliwiające wyświetlanie okien	1
6	Pojemność serwera	Serwer zdolny pomieścić 1GB danych	2

4. Plan prac

4.1 Metodyka zwinna
Pracę zorganizujemy zgodnie z podejściem zwinnym. Projekt podzielony będzie na tygodniowe sprinty w ramach których wytwarzana będzie zaplanowana część funkcjonalności.

4.2 Test Driven Development
Chcemy zastosować tę technikę programowania i zaczynać pisanie każdej nowej funkcjonalności od testów. Wzorować będziemy się na wskazówkach z książki Kenta Becka pt. 'Test driven development'.

4.3 Harmonogram
31 października - dokumentacja wstępna
14 listopada – możliwość gry na jednym komputerze
7 grudnia – możliwość gry przeciwko komputerowi

5. Technologie

5.1 Implementacja
- C/C++ (silnik)
- Qt 5.1.1 (GUI)
- jade, coffeescript, less, AngularJS (klient webowy)
- python (serwer)

5.2 System kontroli wersji
- git – najpopularniejsze narzędzie, łatwa integracja ze środowiskami developerskimi.

5.3 Zarządzanie projektem
- microsoft team foundation service – darmowy dla małych projektów, zapewnia duże możliwości I łatwą organizację pracy.

5.4 Testy
Nie zdecydowaliśmy się jeszcze na żadną konkretną bibliotekę. Nie mamy doświadczenia w pracy z żadną z poniżej podanych bibliotek, oferują one podobne możliwości, dlatego na wstępnym etapie prac zdecydujemy się na jedną z nich:
- boost
- qttest
- cppunit
- googletest

5.5 Środowisko developerskie
- qtCreator – wybór ze względu użycie biblioteki Qt oraz fakt, że mieliśmy sposobność pracy z tym środowiskiem.

- Eclipse – zapewnia dobre wsparcie dla bibliotek testowych i mamy z nim większe doświadczenie.

6. Styl kodowania

Przyjmiemy jeden wspólny styl kodowania, który zawrzemy w oddzielnym dokumencie. Będzie on zbliżony do ogólnie przyjętych standardów. Jednym z celów będzie utrzymywanie “czystego kodu” (zgodnie z zaleceniami zawartymi w “Clean Code” Roberta C. Martina) na każdym etapie prac co umożliwi łatwiejszą modyfikację, pielęgnację kodu oraz zwiększy jego czytelność. Chcemy także zastosować się do zaleceń programowania obiektowego i korzystać ze wzorców projektowych.