Plan wykonania klastra obliczeniowego

Anna Zawadzka, Monika Żurkowska, Filip Turkot, Piotr Waszkiewicz

1 Metodyka

Projekt klastra obliczeniowego prowadzony przez naszą grupę realizowany będzie w metodyce Scrum. Jest to iteracyjna i przyrostowa metodyka prowadzenia projektów, zaliczana do metodyk zwinnych, zgodnych z manifestem Agile.

2 Plan działania

Ustaliliśmy, że realizacja postawionego nam zadania będzie odbywała się w tygodniowych sprintach. Codzienne spotkania (daily meeting) będą miały miejsce w miarę możliwości na uczelni bądź (w przypadku braku wspólnych zajęć) za pomocą aplikacji Skype. Nie ustaliliśmy podziału pracy pomiędzy osoby - obowiązywać będzie zasada "First - Best". Każda osoba która jako pierwsza zasiądzie do realizacji danej części będzie odpowiedzialna za jej skończenie.

3 Product backlog i user stories

Projekt klastra obliczeniowego składa się z kilku łatwo wyróżnialnych funkcjonalności które stanowią również backlog produktu. Jest on dołączony do PDFa w formie pliku o nazwie scrum_documentation.ods.

4 Podział pracy na sprinty

Postawione przed nami zadanie rozplanowaliśmy na 7 sprintów:

- Stworzenie uruchamialnej, reagującej na opcje z linii poleceń i posiadającej (opcjonalny) interfejs graficzny aplikacji dla każdego z modułów.

- Dodanie możliwości nawiązywania i utrzymywania połączeń między poszczególnymi komponentami systemu (wstępna implementacja protokołu łączności).
- Implementacja parsowania, tworzenia i reagowania na poszczególne typy wiadomości. Pełne wsparcie dla XML Schema.
- Wstępna implementacja algorytmu DVRP
- Kontynuacja implementacji algorytmu DVRP
- Obsługa błędów połączeń (utrata łączności) dla modułów podłączonych do głównego serwera. Ochrona przed uszkodzeniem danych w przypadku awarii.
- Pełne działanie serwera zapasowego.

Każdy sprint zaczynany będzie sporządzeniem sprint backlogu i zapisany w postaci dokumentu programu Excel.

5 Technologia

Cały projekt pisany będzie w języku programowania Java a kod będziemy trzymać w repozytorium git. Wraz z wyborem języka postanowiliśmy narzucić na siebie dodatkowe obostrzenia związane ze stylem i dokumentacją tworzonego oprogramowania. Biorąc pod uwagę jak ważnym i istotnym elementem tworzenia oprogramowania jest zapewnienie jego niezawodności przyjęliśmy, że wszelkie zmiany zawsze muszą przechodzić tworzone testy jednostkowe. Każda funckcja publiczna, chroniona oraz klasy muszą posiadać odpowiedni komentarz opisujący ich zadanie, argumenty i wartości zwracane. Zgodnie z dobrymi praktykami programistycznymi przyjęliśmy, że nazwy zmiennych i funkcji muszą być samotłumaczące się a długość funkcji nie powinna przekraczać 40 linijek (linie liczymy z pominięciem komentarzy i klamer). Dążymy również do jak największej hermetyzacji kodu wewnątrz klas, oraz logicznego podziału systemu na niezbędne części. Do znajdowania potencjalnych miejsc błędów korzystamy z wtycznki PMD dostępnej dla platformy Eclipse.