IO2 - specyfikacja początkowa

Mateusz Augustyniak
Mateusz Bąkała
Jakub Bondyra
Paweł Duszak

3 marca 2016

Streszczenie

W niniejszym dokumencie zostaną przedstawione założenia naszej pracy nad projektem Computational Cluster. Obejmują one sposób prowadzenia pracy, harmonogram oraz używane technologie.

1 Specyfikacja systemu

Dostarczona nam specyfikacja zostaje przez nas zaakceptowana. Mamy jednakże kilka niewielkich pytań i zastrzeżeń zawartych poniżej:

- Jednym z głównych zadań projektu jest napisanie systemu w taki sposób, aby każdy komponent był kompatybilny z komponentami innych zespołów projektowych. Wobec tego newralgiczną częścią systemu jest wysyłanie wiadomości po TCP. Z tego co zrozumieliśmy, taka funkcjonalność odbywa się poprzez pisanie do strumienia binarnego danej wiadomości. Czy jest to obowiązujacy wszystkich standard?
- Z powodu braku diagramów klas w specyfikacji nasuwa się pytanie - czy panuje dowolność w hierarchii i konstrukcji klas wewnątrz komponentów?

2 Metodyka

Projekt zostanie przeprowadzony w metodyce **Scrum**. Z uwagi na przymus oddawania kolejnych funkcjonalności i komponentów systemu w kolejnych zdefiniowanych terminach zalecane jest przyrostowe tworzenie projektu. Długość poszczególnych sprintów uzależniona będzie od dat oddania kolejnych etapów. Zgodnie z metodykami rodziny *Agile* sprinty będą możliwie równej długości, około trzech tygodni. W ramach danego sprintu oszacujemy kolejne oczekiwania oraz dokonamy podziału obowiązków. Ponadto planujemy przeprowadzanie częstych cyklicznych spotkań (około *15 min*), na ogół poprzez internetowy komunikator *Hangouts* lub *Skype* z racji na brak możliwości częstych spotkań na żywo. Spotkania w świecie rzeczywistym planujemy robić stosunkowo rzadziej, za ich organizację będzie odpowiedzialny Scrum Master. Zadaniem tych spotkań jest doinformowanie zespołu o postępach prac każdego z członków oraz wyjaśnienie ewentualnych niejednoznaczności.

3 Technologia

Implementacja poszczególnych komponentów klastra obliczeniowego zostanie zrealizowana w technologii *.NET* (w wersji 4.6). Będziemy przy tym używać języka programowania *C#* (w wersji 6.0). Komunikacja pomiędzy wszystkimi składowymi systemu będzie oparta o protokół *TCP/IP*, zaś wiadomości wymianane pomiędzy poszczególnymi elementami systemu będą oparte na standardzie *XML*. Całość projektu będzie wersjonowana przy użyciu systemu kontroli wersji *git*.

4 Spotkanie projektowe

Z uwagi na wybraną metodologię pracy *Scrum* konieczne jest odbywanie regularnych spotkań konsultacyjnych w sprawie projektu. Wobec tego postanowiliśmy, że w ramach sprintu odbędziemy kilka spotkań i rozmów na temat funkcjonalności koniecznych do zaimplementowania w bieżącym sprincie. Prawdopodobnie nie będą to spotkania codzienne (tzw. daliy scrum), ale będą się odbywać kilka razy w tygodniu, zazwyczaj w formie rozmów z użyciem komunikatorów internetowych *Hangouts* lub *Skype*. Podczas takich spotkań, nadzorowanych przez Scrum Mastera, będziemy omawiali bieżące zadania do implementacji oraz przydzielali sobie kolejne funkcjonalności do wykonania. Ponadto będziemy przeprowadzali dyskusję na temat problemów, które każdy z nas napotkał podczas implementacji przydzielnych mu zadań. Podczas takich dyskusji wspólnie zastanowimy się nad rozwiązaniami tych problemów oferując sobie wzajemne wsparcie w implementacji.

5 Harmonogram

Na podstawie przydzielonych terminów, przygotowaliśmy harmonogram oddania prac, który przedstawia się następująco (terminy według laboratoriów czwartkowych o godz. 14 z przedmiotu):

- **laboratoria 4** Komunikacja - działający prototyp
- **laboratoria 5** Komunikacja - całość
- **laboratoria 8** Obliczenia - działający prototyp
- **laboratoria 9** Obliczenia - całość
- **laboratoria 12** Współpraca - działający prototyp
- **laboratoria 13** Współpraca - całość
- **laboratoria 14** Oddanie końcowej wersji projektu

Każdy *sprint* będzie dostosowany do terminów oddania. W ramach *sprintu* każdemu członkowi zespołu będą przydzielane zadania do wykonania w następujący sposób: dana osoba zostanie odpowiedzialna za jeden z komponentów systemu, np. serwer lub węzeł obliczeniowy. Dla danej funkcjonalności oczekiwanej w sprincie członek zespołu zobligowany będzie do implementacji w obrębie swojego komponentu. Z racji na fakt, że ilość pracy potrzebna przy każdym z komponentów jest nieproporcjonalna, osoby z mniejszą ilością zadań w danym sprincie po ukończeniu prac przed czasem powinny wesprzeć bardziej zaabsorbowane osoby. Umożliwi to lepszą znajomość całego systemu przez każdego z członków zespołu oraz zapobiegnie pracy w obrębie jednego z komponentów.

6 Struktura projektu i system kontroli wersji

W celu współpracy nad wspólnym kodem zostało założone repozytorium *git* na serwerze (*ccluster.mini.pw.edu.pl*) w laboratorium wydziałowym. Dostęp do repozytorium został ograniczony tylko do członków zespołu oraz osoby nadzorującej projekt.

Architektura projektu będzie wielowarstwowa. W ramach solucji wyznaczonych zostało wstępnie 7 projektów:

- **AlgorithmSolvers** - zawiera implementację Task Solvera oraz algorytmu DVRP
- **CommunicationUtils** - zbiór klas i interfejsów używanych przez pozostałe projekty w celu komunikacji między sobą
- **Server** - implementacja serwera oraz serwera zapasowego
- **Client** - aplikacja kliencka - tutaj zleca się problem do rozwiązania i oczekuje na wynik
- **ComputationalNode** - projekt zawierający implementację węzła obliczeniowego
- **TaskManager** - projekt zawierający implementację menadżera zadań
- **Tests** - zbiór testów jednostkowych funkcjonalności wszystkich kompenentów systemu