Inżynieria oprogramowania II

Universal Computational Cluster

Dokument startowy

Albert Wolant
Katarzyna Kuzawska
Mateusz Jabłoński
Stanislav Skuratovich

2.03.2014

Technologia:

.NET i c++
Wybór wynika z łatwości zaimplementowania komunikacji za pomocą xml z wykorzystaniem technologi .NET oraz jej dobrej znajomości . Drugi argument był również powodem wyboru c++ do implementacji. Planowana jest dowolność w wyborze języka, aby ułatwić pracę członkom zespołu. Nie będzie stanowić to problemu, ponieważ większość zadań będzie się sprowadzała do stworzenia biblioteki dll dla odpowiedniej funkcjonalności.

Metodyka:

Spiralna
Podział rozwoju oprogramowania na prototypy, odpowiadające kolejnym etapom w kalendarzu przedmiotu.
Z wyboru metodyki wynika fakt, że dokładnie rozplanowany jest jedynie 1 etap. Przydział zadań na kolejne etapy, aż do momentu skończenia projektu, będzie wyznaczany na podstawie efektów poprzedzającej pracy i dokładnym przeanalizowaniu nadchodzących wymagań.
Koncepcja podziału zadań w związku z tym może zmienić się, jeśli zespół uzna to za konieczne – priorytetem jest równomierne obciążenie członków zespołu pracą.

Podział zadań:

Etapy wynikające z zastosowania metodyki spiralnej.
*Czas przewidywany z oczywistych względów jest mocno orientacyjny. Po zrealizowaniu zadania będzie wpisywany dokładny czas w godzinach w odrębnej kolumnie.

etap 1 (od 03.03 do 24.03)
Komunikacja.
(implementacja samego mechanizmu komunikacji, utworzenie modułów systemu z funkcjonalnością ograniczoną jedynie do komunikacji)
A,S: stworzenie funkcjonalności wspólnych dla wszystkich modułów.
K,M: stworzenie klas dla modułów systemu, z metodami do komunikacji.

Albert Wolant (A) – obsługa komunikatów,
Stanislav Skuratovich (S) – obsługa sieci,
Mateusz Jabłoński (M) – Communications Server,
Katarzyna Kuzawska (K) – Task Manager, Computational Node, Computational Client.

Tydzień 1. - 03.03 – 10.03

Nr zadania	Przypisanie	Opis	Czas przewidywany*	Czas spędzony*
0	K	Założenie repozytorium na serwerze gamma i udostępnienie odpowiednim użytkownikom. Organizacja struktury katalogów projektu	10m	10m

1	A	Szkielet biblioteki. Klasy bazowe wiadomości i assemblerów. Klasy instancji problemów	4h	4h
2	M	Ustawianie portu nasłuchu i czasu timeout. Rejestracja Task Managera i Computational Node i uaktualnianie ich stanów. Usuwanie nieaktywnych komponentów	4h	4h
3	S	Nawiązywanie połaczenia tcp/ip. Klasa pozwalająca na nasłuchiwanie na konkretnym porcie , łączenie z tym portem oraz utrzymywanie połączenia. Dostosowanie protokołu do wysyłania wiadomości w postaci xml.	4h	4h
4	K	Stworzenie prototypów klas Task Managera i Computational Node z publicznym api do wprowadzania danych na start systemu. Obsługa zapytań Computational Clienta	4h	4h

Tydzień 2. - 10.03 – 17.03

Zmiana w organizacji projektu. Repo jest przeniesione na github-a. Wszytkie zebrane zmiany z tygodnia beda commitowane w kazdy poniedzialek na repo serwerze gamma. Zmiana była potzrebna ze względu na ciągłe problemy z połączeniem z gammą.

Nr zadania	Przypisanie	Opis	Czas przewidywany	Czas spędzony
5	A	Klasy dla konkretnych komunikatów i ich assemblery. Metody budowania komunikatów z obiektów i obiektów z komunikatów.	8h	?
6	M	Odebranie problemu do rozwiązania, wysyłanie info o trwających obliczeniach do klienta, czekanie na komunikaty, kolejkowanie komunikatów, wysyłanie do odpowiednich modułów. Obsługa timeout. Kolejkowanie rozwiązania i czekanie na request.	8h	11h
7	S	Obsługa błędów, zerwanie połączenia.	8h	
8	K	Stworzenie prototypów klas Task Managera i Computational Node z publicznym api do wprowadzania danych na start systemu. Obsługa zapytań Computational Clienta	8h	4h
9	S,K	Testowanie, nanoszenie ewentualnych zmian lub	1h	4h

		dodawanie brakujących metod do obsługi połączenia tcp/ip.		

Tydzień 3. - 17.03 – 24.03

wszyscy: Poprawki i optymalizacje. Dodatkowe testy.

Nr zadania	Przypisanie	Opis	Czas przewidywany	Czas spędzony
10	A	Ewentualne dostosowanie publicznego api biblioteki do aktualnych potrzeb.	4h	
11	M	Obsługa błędów (crash w trakcie rozwiązywania i ponowne wysłanie). Wysyłanie rozwiązania/informacja o trwających obliczeniach.	4h	
12	wszyscy	Unit testy, dostosowanie implementowanych klas do korzystania CommunicationXML	2h	

etap 2 (od.24.03 do 28.04)

Algorytm.

(implementacja części związanej z rozwiązywaniem konkretnego problemu)
nie może być na razie rozplanowana, ponieważ nie jest znany jeszcze wspomniany typ problemu.

etap 3 (od 28.04 do 26.05)

Pełna funkcjonalność.

(połączenie wszystkich funkcjonalności)