AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki
Katedra Informatyki

PROJEKT INŻYNIERSKI

KOMPONENTY MONITOROWANIA I WIZUALIZACJI PARAMETRÓW PLATFORMY OBLICZENIOWEJ

PAWEŁ KACZMARCZYK, JAKUB SMAJEK

OPIEKUN:
dr inż. Marek Kisiel-Dorohinicki

Kraków 2012

OŚWIADCZENIE AUTORA PRACY

OŚWIADCZAM, ŚWIADOMY(-A) ODPOWIEDZIALNOŚCI KARNEJ ZA POŚWIADCZENIE NIEPRAWDY, ŻE NINIEJSZY PROJEKT WYKONAŁEM(-AM) OSOBIŚCIE I SAMODZIELNIE W ZAKRESIE OPISANYM W DALSZEJ CZĘŚCI DOKUMENTU I ŻE NIE KORZYSTAŁEM(-AM) ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W DALSZEJ CZĘŚCI DOKUMENTU.

. . . . . .

PODPIS

1. Cel prac i wizja produktu

Celem naszego projektu było stworzenie aplikacji umożliwiającej wizualizację parametrów platformy obliczeniowej. Wizualizacja ta przeprowadzana może być poprzez wykresy (także dynamiczne), tworzone na podstawie dostarczonych danych. Zrealizowana przez nas aplikacja jest łatwo rozszerzalna. Dla programistów przygotowane zostały mechanizmy pozwalające na definiowanie nowych sposobów prezentacji danych, a także ich dostawców. Dzięki temu możliwa jest personalizacja aplikacji, w celu dostosowania jej do szczególnych potrzeb.

Przed przystąpieniem do realizacji projektu przewidywaliśmy zagrożenia, które mogły niekorzystnie wpłynąć na prace nad aplikacją. Jednym z nich była nieznajomość i niepewność wykorzystanych bibliotek. Mogła ona spowodować problemy z implementacją niektórych funkcjonalności lub wręcz uniemożliwić ich realizację. Innym zagrożeniem były niesprecyzowane wymagania funkcjonalne. Dzięki częstym kontaktom z klientem udało się jednak ustalić wszystkie wymagania, a następnie je zrealizować. Obawialiśmy się również o nieznajomość platformy obliczeniowej jAgE i dlatego zapoznaliśmy się z jej dokumentacją. Okazało się jednak, że znajomość tej platformy nie była konieczna.

2. Zakres funkcjonalności

2.1. Wymagania funkcjonalne

W celu zdefiniowania wymagań funkcjonalnych, użytkowników naszego systemu podzieliliśmy na 3 grupy. Przedstawiciele pierwszej z nich (zwykli użytkownicy) korzystają z podstawowych funkcjonalności oferowanych przez naszą aplikację. Ich celem jest wizualizacja parametrów platformy obliczeniowej. Zakładamy, że użytkownicy ci nie muszą posiadać zaawansowanej wiedzy informatycznej. Druga grupa (użytkownicy zaawansowani) oprócz wykorzystywania podstawowych funkcjonalności potrafi także rozszerzać naszą aplikację za pomocą specjalnie przygotowanych do tego mechanizmów. Ostatnia grupa (programiści) to zbiór użytkowników posiadających zaawansowaną wiedzę informatyczną oraz doświadczenie w pracy z naszą aplikacją. Celem tej grupy jest dalszy rozwój stworzonego przez nas systemu. Poniżej przedstawiamy zbiór wymagań funkcjonalnych wraz z podziałem ich na odpowiednich użytkowników.

- użytkownik - podstawowe funkcjonalności wymagane przez tę grupę to: stworzenie nowego wykresu, dodanie serii danych do wykresu, stworzenie nowej zakładki, zapisanie/odczytanie konfiguracji aplikacji, wydrukowanie/eksportowanie wykresu, zmiana języka aplikacji, zmiana opcji/przybliżenie wykresu, dodanie statystyki do wykresu
- zaawansowany użytkownik - użytkownik ten będzie rozszerzał naszą aplikację za pomocą zdefiniowanych mechanizmów. Umożliwiają one definiowanie nowych statystyk, tworzenie nowych zakładek w oknie do konfiguracji wykresu oraz dodawanie nowych dostawców danych
- programista - głównym zadaniem programisty, do którego przygotowana została nasza aplikacja, jest definowanie nowych sposobów wizualizacji danych

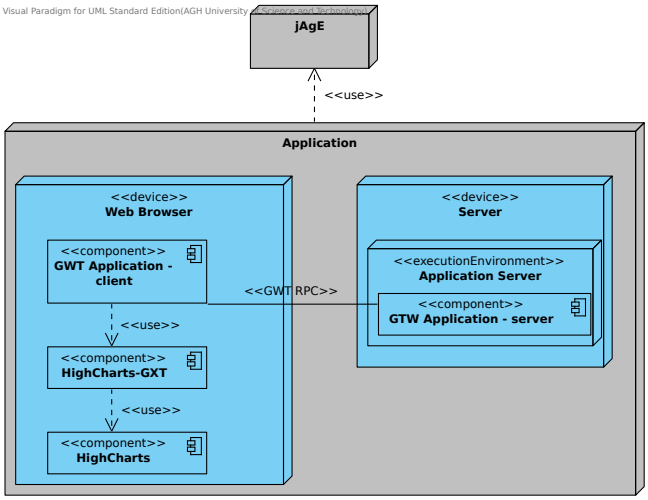
2.2. Wymagania niefunkcjonalne

Pierwszym, niefunkcjonalnym wymaganiem klienta odnośnie systemu była realizacji go jako aplikacji webowej. Dodatkowo stworzony komponent powinien być przenośny pomiędzy różnymi systamami i przeglądarkami. Bardzo ważnym aspektem dla klienta była rozszerzalność aplikacji. Zobowiązaliśmy się stworzyć mechanizmy, które umożliwią rozszerzanie aplikacji w prosty i wygodny sposób.

3. Wybrane aspekty realizacji

Zgodnie z wymaganiami komponent został zrealizowany jako aplikacja webowa przy użyciu framework'u GWT oraz jego rozszerzenia Ext GWT. Do wizualizacji danych zdecydowaliśmy się wykorzystać bibliotekę HighCharts. W celu połączenia GWT oraz HighCharts (w której funkcjonalności implementuje się w JavaScript) skorzystaliśmy z biblioteki HighCharts-GXT stworzonej przez Daniele Strollo w ramach projektu typu Open Source.

Stworzona przez nas aplikacja dzieli się na dwie części: kliencką i serwerową. Część kliencka odpowiedzialna jest za wizualizację danych oraz udostępnienie użytkownikowi wszelkich funkcjonalności. Część serwerowa składa się głównie z serwisów umożliwiających działanie aplikacji, poprzez realizację czynności niemożliwych do wykonania po stronie klienckiej. Na poniższym diagramie przedstawiona została budowa naszej aplikacji.

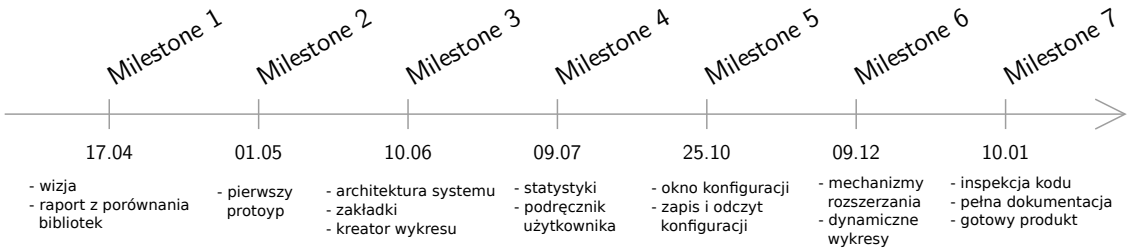


Na powyższym diagramie widzimy, że docelowo aplikacja współpracować będzie z platformą jAgE (platforma ta będzie dostarczać dane do wizualizacji). Klient może korzystać z naszej aplikacji za pomocą przeglądarki. Dodatkowo część kliencka GWT do wizualizacji danych wykorzystuje bibliotekę Highcharts (za pomocą łącznika, którym jest biblioteka HighCharts-GXT). Strona serwerowa jest natomiast umieszczona na serwerze aplikacyjnym. Mechanizmem, który umożliwia komunikację pomiędzy stroną kliencką i serwerową jest GWT RPC.

4. Organizacja pracy

Stworzona przez nas aplikacja powstawała w ramach procesu badawczego. Wymagania nie były do końca rozpoznane, dlatego zdecydowaliśmy się na wybór ewolucyjnej metodyki tworzenia

oprogramowania. W związku z tym, prace planowane były na bieżąco, gdy pojawiały się nowe pomysły oraz funkcjonalności, o które można było wzbogacić aplikację. Dzięki takiemu podejściu, powstało 7 etapów w trakcie których zrealizowane zostały założone wymagania. Przedstawione są one na poniższej osi:



Podział prac nad aplikacją przedstawia się następująco:

Paweł Kaczmarczyk:

- implementacja architektury systemu po stronie klienckiej
- implementacja funkcjonalności umożliwiającej organizowanie wykresów w zakładki oraz tworzenie wykresów za pomocą konfiguratora
- stworzenie funkcjonalności zwiększających konfigurowalność aplikacji
- projekt oraz implementacja mechanizmu pozwalającego na przybliżanie wykresów
- projekt oraz implementacja dynamicznej formy wizualizacji danych

Jakub Smajek:

- implementacja strony serwerowej oraz stworzenie odpowiednich serwisów
- implementacja podstawowych statystyk służących porównywaniu danych na wykresach
- stworzenie dodatkowej warstwy pomiędzy dostawcami danych a stroną kliencką, która umożliwia odpowiedni wybór danych
- projekt i implementacja mechanizmu zapisu oraz odczytu konfiguracji wizualizacji
- stworzenie mechanizmów pozwalających na rozszerzanie aplikacji po stronie klienckiej

Do zarządzania projektem wykorzystaliśmy platformę Trac, na której znajdowały się wszystkie dokumenty dotyczące realizacji aplikacji. Za jej pomocą rozdzielaliśmy również zadania do wykonania oraz zarządzaliśmy etapami tworzenia projektu. Później platforma ta zastąpiona została przez narzędzie Confluence.

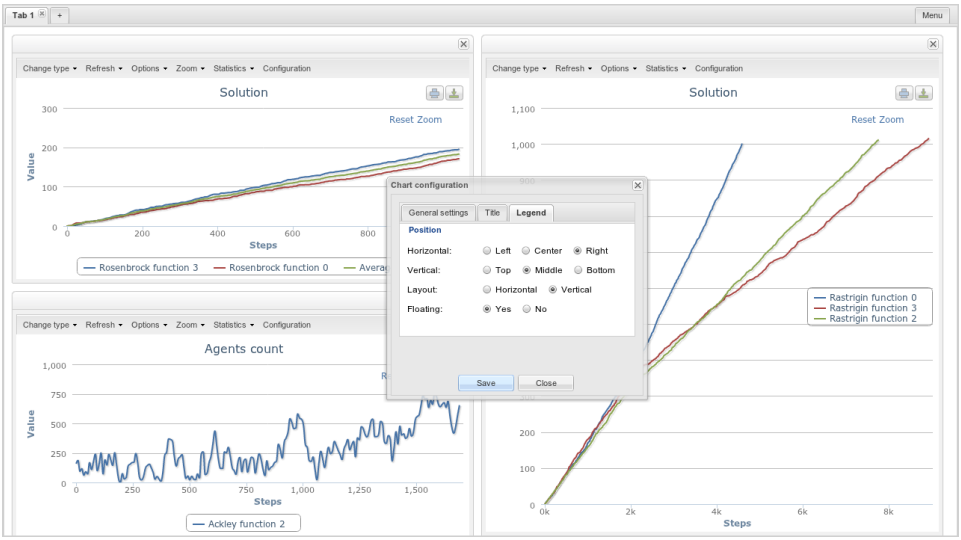
5. Wyniki projektu

Stworzona przez nas aplikacja spełnia wszystkie wymagania postawione w punkcie 2. Poprawność jej działania została zweryfikowana i potwierdzona zarówno przez klienta jak i menadżera projektu. Dokumentacja podzielona została na 3 główne części. Pierwszy fragment (dokumentacja użytkownika) prezentuje możliwości naszej aplikacji oraz sposób jej użycia. Dokumentacja techniczna opisuje budowę aplikacji, a także mechanizmy użyteczne do dalszego

jej rozwoju. Ostatni fragment dokumentacji poświęcony jest procesowi, w ramach którego powstała aplikacja.

Priorytetem podczas tworzenia aplikacji była jej rozszerzalność. W ramach prac przygotowaliśmy kilka mechanizmów umożliwiających wygodne dodawanie nowych funkcjonalności do aplikacji. Naszym zdaniem, bardzo ciekawym kierunkiem rozwoju byłoby dodanie do systemu nowych form wizualizacji. Jedną z nich mogłaby być np. wizualizacja danych w formie tabel.

Aplikacja przewiduje różne sposoby porównywania danych. Pierwszym z nich jest możliwość wyświetlania wielu serii danych na jednym wykresie (w przypadku tego samego rodzaju parametru). Dla porównywanych danych mogą zostać wyświetlone statystyki takie jak: maksimum, minimum, średnia czy odchylenie standardowe. W przypadku gdy chcemy przedstawić różne parametry danego obliczenia, możemy to zrobić na wykresach sąsiadujących ze sobą w ramach jednej zakładki. Przed wyświetleniem dane poddane zostają niezbędnym przekształceniom związanym z przejrzystością wykresu.



Dodatkowo w naszej aplikacji przewidzieliśmy szereg funkcjonalności takich jak: drukowanie wykresów, eksportowanie wykresów do popularnych formatów (pdf, jpg, png, svg), przybliżanie dla obu osi, możliwość tymczasowego ukrywania serii danych oraz organizowanie wykresów w zakładki czy zapisywanie i odczytywanie bieżących ustawień naszej aplikacji do pliku XML. Konfigurowanie parametrów odbywa się za pomocą wygodnego i rozszerzalnego wizarda.

Materiały źródłowe

[1] Ext gwt. `http://www.sencha.com/products/extgwt`.

[2] Google web toolkit. `http://code.google.com/intl/pl/webtoolkit`.

[3] Highcharts. `http://www.highcharts.com`.

[4] Java simple plugin framework. `http://code.google.com/p/jspf`.

[5] D. Strollo. Highcharts-gxt. `http://highcharts-gxt.sourceforge.net`, 2011.