

**AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE**

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki
Katedra Informatyki



PROJEKT INŻYNIERSKI

**PLATFORMA DO AUTOMATYCZNYCH
AKTUALIZACJI OPROGRAMOWANIA NA
URZĄDZENIACH ZDALNYCH. -
DOKUMENTACJA PROCESU**

**PRZEMYSŁAW DĄBEK, ROMAN JANUSZ
TOMASZ KOWAL, MAŁGORZATA WIELGUS**

OPIEKUN:
dr inż. Wojciech Turek

Kraków 2012

OŚWIADCZENIE AUTORA PRACY

OŚWIADCZAM, ŚWIADOMY(-A) ODPOWIEDZIALNOŚCI KARNEJ ZA PO-
ŚWIADCZENIE 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. Podział na milestoney

Prace zostały podzielone na dwa milestoney. Zakończenie pierwszego miało na celu zapoznanie z technologiami, które można wykorzystać do rozwiązania problemu automatycznej aktualizacji oprogramowania. Celem Milestone 1 jest stworzenie wizji oraz koncepcji architektury systemu, a także przetestowanie branych pod uwagę technologii. Produktem końcowym tego etapu były:

- Prototyp komunikacji serwera z klientem
- Zapoznanie z działaniem menedżerów pakietów *apt* oraz *opkg*
- Wybór technologii do stworzenia graficznego interfejsu użytkownika
- Wybór bazy danych

Drugi milestone zakończył się dostarczeniem działającego produktu.

1.1. Milestone 1

Ticket	Summary	Type	Owner	Status	Created
#273	Wizja - określenie wymagań	task	tkowal	closed	06/05/11
#274	Wizja - wstępna analiza ryzyka	task	szemek	closed	06/05/11
#275	Wizja - ogólny opis produktu	task	tkowal	closed	06/05/11
#276	Przegląd technologii apt	task	szemek	closed	06/05/11
#277	Przegląd technologii bazodanowych.	task	szemek	closed	06/05/11
#278	Przegląd technologii do tworzenia GUI	task	szemek	closed	06/05/11
#279	Przegląd technologii do komunikacji ze zdalnymi urządzeniami	task	tkowal	closed	06/05/11
#280	Określenie architektury systemu.	task	malgorza	closed	06/05/11
#281	Przetestowanie komunikacji z użyciem modułu gen_tcp.	task	tkowal	closed	06/05/11
#282	Przetestowanie możliwości stworzenia GUI przy pomocy eHTML.	task	szemek	closed	06/05/11
#283	Połączenie erlanga z bazą danych - drivery MySQL i PostgreSQL	task	szemek	closed	06/05/11
#285	Sprawdzenie możliwości wykorzystania menedżera pakietów OPKG	task	tkowal	closed	06/20/11
#327	Rozważenie możliwości użycia i przetestowanie mochiweb	task	tkowal	closed	07/22/11
#284	Instalacja Debiana na beagleboardzie	task	roman	closed	06/20/11

1.2. Milestone 2

Ticket	Summary	Type	Owner	Status	Created
#334	GUI do przeglądania urządzeń i dodawania prostych zadań	enhancement	małgorza	closed	10/19/11
#333	Integracja ERLRC	task	roman	closed	10/19/11
#331	Zmniejszenie rozmiaru paczek z releasami generowanych przez rebara	enhancement	roman	closed	09/27/11
#328	Problem z uruchomieniem releasu wygenerowanego rebar-em	defect	tkowal	closed	09/27/11
#329	Implementacja infrastruktury sesji zarządzania urządzeniem na kliencie i serwerze	enhancement	roman	closed	09/27/11
#286	Zapoznanie się z OTP Design Principles (application, release) oraz narzędziem rebar	task	roman	closed	06/20/11
#330	Instalacja i integracja z systemem serwera FTP, implementacja prostego klienta FTP dla urządzenia.	enhancement	roman	closed	09/27/11
#332	Mechanizm logowania	enhancement	szemek	closed	10/18/11

2. Podział prac

W trakcie prac podzieliliśmy się na 2 podzespoły: Przemysław Dąbek i Małgorzata Wielgus zajmowali się front-endem, natomiast Roman Janusz i Tomasz Kowal zajmowali się backendem. Podział nie był sztywny i często pracowaliśmy wspólnie w czwórkę.

3. Spotkania z klientem

3.1. Treść notatki z 2011-06-07

Zapoznać się:

- RPC w Erlangu <http://erldocs.com/R14B02/kernel/rpc.html>
- rebar <https://bitbucket.org/basho/rebar/wiki/Home>
- release, update aplikacji erlangowych

Do zrobienia:

- prototyp technologiczny (repozytorium, urządzenie mobilne, serwer http, komunikacja pomiędzy składowymi systemu)
- opracowanie harmonogramu

Pomysły rozbudowy funkcjonalności:

- GPS, Google Maps
- VM i aplikacja uruchamiane przy starcie systemu na urządzeniu mobilnym

3.2. Treść notatki z 2011-09-29

W czasie spotkania klient ocenił nasze dotychczasowe postępy.

Do zrobienia:

- integracja poszczególnych części projektu
- integracja systemu z menedżerami pakietów

3.3. Treść notatki z 2011-10-28

W czasie spotkania odbyła się prezentacja prototypu

Klient zgłosił następujące uwagi:

- apt - dokończyć generator
- apt - integracja z systemem aktualizacji
- pobieranie pakietów z http - opcjonalnie
- interfejs — dokończenie
 - edycja grupowa
 - zakończone zadania
 - stan - uruchomione aplikacje, wersje
 - MAC - identyfikacja

4. Przegląd technologii do zastosowania w platformie

4.1. Technologia do stworzenia graficznego interfejsu użytkownika

Pod uwagę brano:

- Ruby on Rails z połączeniem Erlectricity
- Yaws
- Mochiweb

- Webmachine
- Nitrogen
- Zotonic – CMS i framework

RoR: Według przykładów najpierw uruchamiany był proces erlanga, który dopiero wywoływał program w Rubym i dopiero wtedy zachodziła komunikacja. Nie można jednym procesem erlangowym dopiąć się do jednej klasy aplikacji w Ruby on Rails.

Yaws: Serwer napisany całkowicie w Erlangu. Tworzenie interfejsu odbywa się w specjalnym dialekcie eHTML. Wydaje się być najlepszym rozwiązaniem.

Mochiweb: jest narzędziem do budowania własnych lekkich serwerów http. Zbudowanie własnego serwera od zera dodałoby niepotrzebny dodatkowy stopień do złożoności problemu.

Webmachine: Zestaw narzędzi do tworzenia web serwisów opartych o technologię REST. Prawdopodobnie nie będzie nam potrzebny.

Nitrogen: Jest to framework do tworzenia aplikacji webowych w erlangu. Do tego potrzebny byłby jeszcze serwer.

Zotonic: Framework i CMS w erlangu. Aby dopisać obsługę serwera trzeba napisać dodatkowe moduły i poznać jego strukturę.

4.2. Komunikacja między urządzeniem zdalnym, a głównym serwerem platformy:

- wywoływanie poleceń przez SSH
- komunikacja przez socket TCP
- komunikacja między węzłami sieci erlanga
- web service

SSH: Połączenie może być w każdej chwili zrywane, co może skutkować nieprzewidywalnym zachowaniem. Wymaga również zmian konfiguracji w systemie operacyjnym zdalnego urządzenia (użytkownik, klucz, plik sudoers).

TCP: Rozwiązanie o najmniejszym narzucie komunikacyjnym. Łatwe do obsłużenia w erlangu za pomocą `gen_tcp`. Wymagana samodzielna obsługa zrywanych połączeń, zaprojektowanie własnego formatu wiadomości. Wydaje się być najlepszym rozwiązaniem.

Węzły sieci erlanga: Po połączeniu z danym urządzeniem połączenie jest stale utrzymywane, co nie jest pożądane (ani nawet możliwe) w naszym przypadku.

Web Serwisy: Powodują duży narzut komunikacyjny, a ponieważ zakładamy dużą liczbę urządzeń, chcieliśmy tego uniknąć.

4.3. Język programowania do implementacji klienta oraz serwera

W związku z tym, że wybraliśmy komponenty, które są napisane w erlangu, to aby uniknąć problemów na styku różnych technologii, postanowiliśmy napisać zarówno klienta, jak i serwer w erlangu.

4.4. Wybór bazy danych

- mnesia
- MySQL
- ewentualnie inne relacyjne bazy danych