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 - PRZEWODNIK

PRZEMYSŁAW DĄBEK, ROMAN JANUSZ
TOMASZ KOWAL, MAŁGORZATA WIELGUS

OPIEKUN:
dr inż. Wojciech Turek

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

1.1. Opis problemu

Problem: Duża ilość zdalnych urządzeń rozmieszczonych w terenie. Urządzenia komunikują się przez zawodną sieć. Na różnych urządzeniach działają różne wersje aplikacji. Zachodzi potrzeba aktualizacji oprogramowania na grupie urządzeń. Rozwiązanie: Platforma umożliwi monitorowanie oraz aktualizację wersji aplikacji na grupach urządzeń.

1.2. Opis użytkownika i zewnętrznych podsystemów

Użytkownikiem systemu jest osoba odpowiedzialna za oprogramowanie w systemie składającym się z urządzeń zdalnych. System składa się z dużej liczby tych urządzeń, które mogą mieć mocno ograniczone zasoby sprzętowe (np. *Beagleboard*). Urządzenia mogą mieć różne architektury. Urządzenia będą łączyć się z serwerem głównym, do którego przesyłają zebrane dane. Komunikacja odbywa się za pomocą zawodnego połączenia (np. *GSM*).

1.3. Opis produktu

Produkt składa się z serwera zawierającego repozytorium oprogramowania. Serwer ma za zadanie monitorować jakie wersje aplikacji znajdują się na konkretnych urządzeniach i grupach urządzeń. Udostępniono interfejs webowy, dzięki któremu można wybrać urządzenia (lub ich grupy), na których chcemy przeprowadzić aktualizację i sprawdzić jakie aplikacje są zainstalowane. Ponieważ nie zawsze możliwe jest połączenie z wybranym urządzeniem, serwer przechowuje jego stan. W momencie uzyskania połączenia z urządzeniem serwer wykonuje zaplanowane aktualizacje.

2. Zakres funkcfonalności

2.1. Wymagania funkcjonalne

Platforma:

- monitoruje, czy dane urządzenie jest dostępne
- monitoruje zainstalowane aplikacje oraz ich wersje na urządzeniach
- rejestruje urządzenia
- pozwala na definiowanie grup urządzeń
- umożliwia aktualizację i instalację aplikacji na pojedynczych urządzeniach
- umożliwia aktualizację i instalację aplikacji na grupach urządzeń
- umożliwia aktualizację i instalację za pomocą systemowych narzędzi takich jak apt

- webowy interfejs użytkownika
- umożliwia tworzenie paczek aplikacji dedykowanych dla platformy wraz ze skryptami instalacyjnymi

2.2. Wymagania niefunkcjonalne

- Obsługa między 1000 - 10000 urządzeń
- Obsługa różnych architektur i systemów operacyjnych
- Prawidłowe działanie w przypadku zrywających się połączeń
- Wymagania stawiane dokumentacji:
  - Podręcznik użytkownika
  - Podręcznik instalacji i konfiguracji.
  - Dokumentacja techniczna – dokumentacja kodu, opis testów.

3. Wybrane aspekty realizacji

Właściwie wszystkie podsystemy zostały wykonane w Erlangu, aby zachować homogeniczne środowisko:

- baza danych - *mnesia*
- serwer http - *mochiweb*
- backend
- aplikacja kliencka

Jedynym elementem, który używa innych technologii (*ErlyDTL*, *JavaScript*, *jQuery*, *CSS*) jest webowy interfejs użytkownika.

Komunikacja między urządzeniem klienckim i serwerem odbywa się przez własny protokół nad TCP. Jest on w łatwy sposób rozszerzalny, gdyby trzeba było dodać nowe funkcjonalności oraz zapewnia możliwość komunikacji z urządzeniami znajdującymi się za NATem.

4. Organizacja pracy

W trakcie prac podzieliliśmy się na dwa zespoły:

- zespół zajmujący się backendem w składzie Roman Janusz oraz Tomasz Kowal
- zespół zajmujący się frontendem oraz bazą danych w składzie Przemysław Dąbek oraz Małgorzata Wielgus

Prace zostały podzielone na dwa etapy. W pierwszym dokonaliśmy przeglądu potrzebnych technologii oraz zbudowaliśmy prototypy. W drugim zaimplementowaliśmy właściwą funkcjonalność. Pierwszy etap zakończył się w czerwcu 2011 roku. Pracę nad drugim zaczęliśmy w trakcie wakacji 2011 roku.

5. Wyniki projektu

Wynikiem projektu jest działające oprogramowanie wraz z dokumentacją. System został przetestowany na urządzeniu beagleboard. W dalszej kolejności należy przetestować działanie systemu na większej liczbie urządzeń. Można spróbować dodać obsługę innych menedżerów pakietów (*yum*, *port*). Należy również popracować nad bezpieczeństwem systemu. W tej chwili pozwala on podłączać się do systemu każdemu urządzeniu, które zostanie poprawnie skonfigurowane (brak autoryzacji). Pozwala również na aktualizację dowolnej aplikacji działającej na urządzeniu klienckim, a nie tylko tych zarządzanych za pomocą systemu.