

# pRoman - process manager

Projekt TIN. Dokumentacja wstępna

---

Damian Bułak  
Michał Sypetkowski  
Marcin Waszak  
Ahata Valiukevich

19 kwietnia 2017

## TREŚĆ ZADANIA

W sieci jest zbiór zarządzanych węzłów, serwer zarządzający i stacja konsoli administratora. W węzłach pracują agenty zarządzające. Agent zarządzający może: załadować kod nowego procesu, usunąć kod procesu, uruchomić/zatrzymać/wznowić/zabić dany proces zgodnie z harmonogramem, wznowić proces nie raportujący swej żywotności, podać dane statystyczne serwerowi. System umożliwia administratorowi zarządzanie rozproszonymi procesami. System komunikacji powinien móc pracować w przestrzeni adresów *IPv4* i *IPv6*. Ponadto należy zaprojektować moduł do *Wireshark* umożliwiający wyświetlanie i analizę zdefiniowanych komunikatów.

## PRZYJĘTE ZAŁOŻENIA FUNKCJONALNE I NIEFUNKCJONALNE

### ZAŁOŻENIA FUNKCJONALNE

Implementowany system ma umożliwiać:

- wysyłanie obrazu procesu
- usuwanie obrazu procesu
- wznawianie procesu nie raportującego swej żywotności

- podawanie danych statystycznych serwerowi
- uruchamianie danego procesu zgodnie z harmonogramem
- zatrzymywanie danego procesu zgodnie z harmonogramem
- wznowianie danego procesu zgodnie z harmonogramem
- zabijanie danego procesu zgodnie z harmonogramem

## ZAŁOŻENIA NIEFUNKCJONALNE

### PODSTAWOWE PRZYPADKI UŻYCIA

Administrator - użytkownik zalogowany do konsoli administratorskiej

Tablica 1: Podstawowe przypadki użycia

NAZWA	AKTORZY	OPIS
Wysłanie obrazu procesu	Administrator	<ol style="list-style-type: none"> <li>1. Administrator wybiera proces do załadowania.</li> <li>2. Proces zostaje załadowany do systemu.</li> </ol>
Przeglądanie listy procesów	Administrator	<ol style="list-style-type: none"> <li>1. Administratorowi prezentowana jest lista załadowanych procesów.</li> <li>2. Obok każdego procesu administrator może zobaczyć stan procesu (uruchomiony bądź nie), jak i podstawowe informacje o procesie.</li> </ol>
Usuwanie obrazu procesu	Administrator	<ol style="list-style-type: none"> <li>1. [ Przeglądanie listy procesów ]</li> <li>2. Administrator usuwa wybrany z listy proces poleceniem 'Usuń'.</li> </ol>

*Kontynuacja na następnej stronie*

Tablica 1 – *Kontynuacja z poprzedniej strony*

NAZWA	AKTORZY	OPIS
Uruchomienie procesu	Administrator	<ol style="list-style-type: none"> <li>1. [ Przeglądanie listy procesów ]</li> <li>2. Administrator wybiera z listy proces i uruchamia go poleceniem <i>‘Uruchom’</i>.</li> <li>3. Następuje przetwarzanie procesu (opis w architekturze systemu). Administrator może zobaczyć postęp przetwarzania procesu w postaci procent zakończenia.</li> <li>4. Po zakończeniu przetwarzania administrator widzi rezultat przetwarzania.</li> </ol>
Zatrzymanie uruchomionego procesu	Administrator	<ol style="list-style-type: none"> <li>1. [ Przeglądanie listy procesów ]</li> <li>2. Administrator wybiera z listy uruchomiony proces.</li> <li>3. Administrator zatrzymuje proces poleceniem <i>‘Zatrzymaj’</i>.</li> </ol>
Planowanie czasu uruchomienia procesu	Administrator	<ol style="list-style-type: none"> <li>1. [ Przeglądanie listy procesów ]</li> <li>2. Administrator edytuje czas wykonania procesu po wpisaniu polecenia <i>‘Edytuj’</i>.</li> </ol>
Podgląd harmonogramu procesów	Administrator	<ol style="list-style-type: none"> <li>1. Administrator przegląda tabelaryczny harmonogram załadowanych procesów.</li> </ol>

*Kontynuacja na następnej stronie*

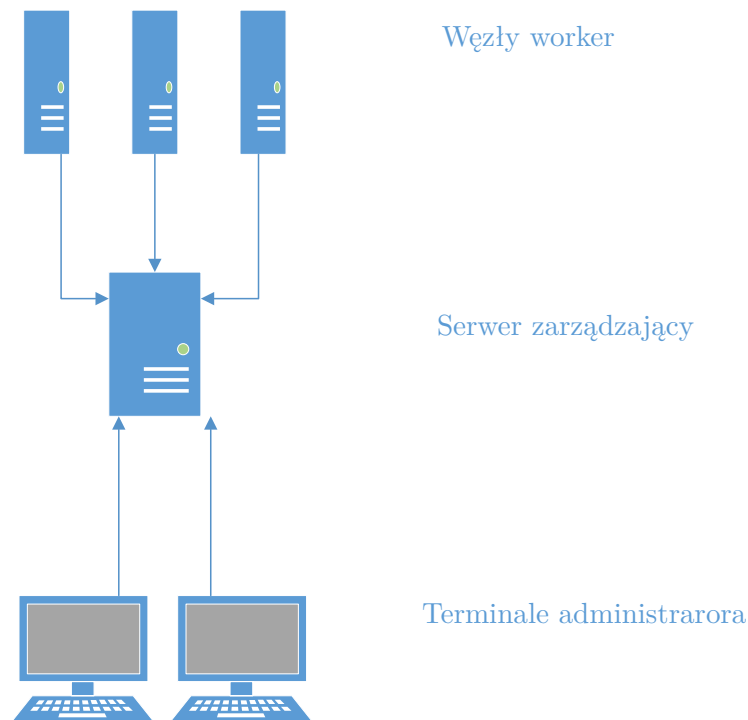
Tablica 1 – *Kontynuacja z poprzedniej strony*

NAZWA	AKTORZY	OPIS
Podgląd danych statystycznych	Administrator	1. Administrator przegląda dane statystyczne z informacjami o załadowanych, uruchomionych, zatrzymanych procesach oraz o obciążeniu poszczególnych węzłów.

## WYBRANE ŚRODOWISKO SPRZĘTOWO-PROGRAMOWE

1. Wspierany system operacyjny: *Linux*.
2. Język: *C++*.
3. Biblioteki: STL, boost/filesystem, boost/process, BSD Sockets API.
4. Testowanie: boost/test.
5. Debugowanie: wireshark.
6. Budowanie: cmake

## ARCHITEKTURA ROZWIĄZANIA



Rysunek 1: Ilustracja systemu

**Węzeł worker** - maszyna, która wykonuje procesy. Bezpośrednia kontrola procesów jest sprawowana przez procesy sieciowe klasy worker.

**Proces klasy worker** - specjalny proces sieciowy na maszynach węzłów worker. Jego zadaniem jest zarządzanie procesami na tych maszynach według rozkazów serwera zarządzającego.

**Watchdog** - specjalny proces obok procesu workera. Stanowi zabezpieczenie przed skutkami zawieszenia procesu workera.

**Serwer zarządzający** - maszyna, która wysyła rozkazy do agentów maszyn węzłowych. Może przeprowadzać zaplanowane wcześniej czynności. Serwer posiada proces sieciowy klasy server.

**Proces klasy server** - oczekuje na połączenia od węzłów worker i terminali administratora.

**Terminal administratora** - maszyna, dzięki której administrator ma możliwość zarządzać procesami na maszynach węzłowych. Każdy taki terminal posiada proces sieciowy klasy administrator.

Planowane aplikacje sieciowe do wykonania:

- aplikacja klasy worker

- aplikacja klasy server
- aplikacja klasy administrator

## API MODUŁÓW STANOWIĄCYCH BLOKI FUNKCJONALNE

### SPOSÓB TESTOWANIA

Planujemy użyć testów jednostkowych w poszczególnych aplikacjach naszego projektu, żeby sprawdzić poprawność działania osobnych części całego systemu. Poza testami jednostkowymi napisane będą również testy integracyjne. Przewidujemy również testowanie ręczne. Scenariusze testów akceptacyjnych są przedstawione poniżej.

### SCENARIUSZE TESTÓW AKCEPTACYJNYCH

#### PODZIAŁ PRAC W ZESPOLE

Damian:  
Michał:  
Marcin: team-leader  
Agata:

#### HARMONOGRAM PRAC

- **19.04** - oddanie dokumentacji wstępnej.
- **30.04** - szkielec programów: serwera, workera oraz administratora; rozpoczęcie prace nad skryptami testującymi.
- **8.05** - konsultacje #1: projekt z w pełni działającą komunikacją między administratorem a serwerem (nawiązywanie połączenia, działający mechanizm przesyłania pakietów i poleceń).
- **22.05** - konsultacje #2: działający i prawie skończony projekt, z ewentualnymi błędami niewpływającymi istotnie na działanie programu, rozpoczęcie pracy nad końcową dokumentacją poprojektową.
- **31.05** - oddanie projektu.

#### ADRES PROJEKTU NA SERWERZE KONTROLI WERSJI

<https://github.com/marcin-waszak/DistributedProcesses>

W zakładce *Issues* można przeglądać informacje o przydzielanych przez Marcina zadaniach:

- dla kogo zostało przydzielone
- kiedy zostało utworzone
- na jakim etapie realizacji obecnie jest