



AGH

AKADEMIA GÓRNICZO-HUTNICZA IM. STANISŁAWA STASZICA W KRAKOWIE

**WYDZIAŁ ELEKTROTECHNIKI, AUTOMATYKI,
INFORMATYKI I INŻYNIERII BIOMEDYCZNEJ**

KATEDRA INFORMATYKI STOSOWANEJ

Programowanie współbieżne i rozproszone

Symulacja inteligentnego domu

Autor:

Paweł Gałka, Michał Jasica

Kierunek studiów:

Informatyka

Opiekun pracy:

dr inż. Wojciech Szmuc

Kraków, 2020

Spis treści

1. Tytuł programu	3
2. Autorzy	3
3. Data oddania programu	3
4. Cel programu	4
5. Informacje o stosowanych pakietach zewnętrznych	4
6. Diagram	5
7. Informacje o zastosowaniu specyficznych metod rozwiązania problemu	6
8. Krótka instrukcja obsługi	7
9. Przykłady	8
10. Możliwe rozszerzenia programu	11
11. Ograniczenia programu	11

1. Tytuł programu

Symulacja inteligentnego domu w języku Erlang.

2. Autorzy

- Paweł Gałka, Informatyka rok III, nr indeksu 297203, grupa dziekanatowa 2a
- Michał Jasica, Informatyka rok III, nr indeksu 297212, grupa dziekanatowa 2a

3. Data oddania programu

Program oddany został 22.01.2020r.

4. Cel programu

Celem programu jest wykonanie symulacji zachowania inteligentnego domu na doznane bodźce zewnętrzne oraz wpływ zachowań domu na aktualne jego parametry (m. in. wpływ działania systemu klimatyzacji na temperaturę) z wykorzystaniem paradygmatu programowania rozproszonego. Jest on realizowany poprzez delegację odpowiednich obowiązków między oddzielne procesy komunikujące się ze sobą i wysyłające między sobą dane. Całość została oparta o GUI pozwalające w czytelny sposób wyświetlić aktualny stan domu oraz, zedytować wybrane parametry, wywołać pewne akcje.

Budowa domu opiera się o zestaw sensorów i zestaw odbiorników oraz logikę delegującą odpowiednie sygnały między odpowiednie odbiorniki. Sensory dostarczają kontrolerowi danych real-time, dzięki którym zostają podejmowane odpowiednie operacje. Ważniejsze akcje zostają zapisane przez mechanizm logujący co pozwala w dowolnym czasie użytkownikowi sprawdzić jakie akcje zaszły w jakim momencie. Istnieje możliwość zwiększenia liczby czujników włamania, przez co użytkownik dostanie informację z którego sensora pochodzi odczyt jeśli dojdzie do włamania oraz możliwość regulacji zakresu temperatur działania klimatyzacji.

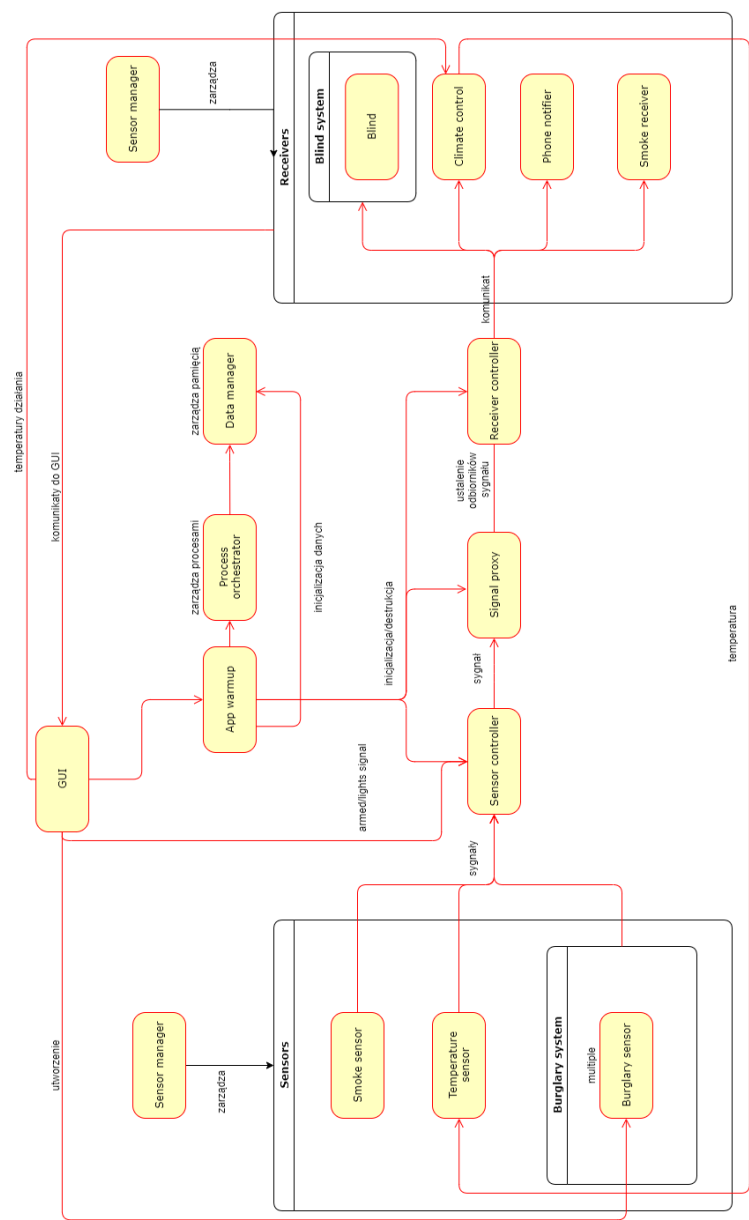
5. Informacje o stosowanych pakietach zewnętrznych

W projekcie starano ograniczyć się pakiety zewnętrzne do możliwego minimum w celu maksymalnego wykorzystania natywnego Erlanga

Lista zastosowanych pakietów z ich użyciem:

- io - do operacji IO, wypisywania danych na konsolę,
- file - do operacji na plikach (zapisu logów, odczytu/zapisu ustawień domyślnych)
- string - do manipulowania ciągami znaków i ich mapowania na typy liczbowe
- ets - wykorzystywany do przechowywania koniecznych stanów odbiorników
- wx - użyty do stworzenia GUI

6. Diagram



Rys. 6.1. Diagram zależności

7. Informacje o zastosowaniu specyficznych metod rozwiązania problemu

W komponencie *receiver_controller* zastosowano nowy proces do obsługi sygnału przychodzącego w celu zmniejszenia obciążenia procesu i wyeliminowaniu tzw. *bottlenecks* i zwiększeniu przepustowości tego procesu

GUI:

W GUI zastosowano bibliotekę wx i dziedziczenie komponentów do obsługi GUI

- wxButton
- wxChoice

Zastosowano pewnego rodzaju proxy które na zadany sygnał wybiera z reguł jaki jest cel danego sygnału (modyfikuje ścieżkę sygnału). Pozwala to na przesyłaniu sygnału nie w formacie one-one (sensor-odbiornik).

Receive controller posiada reguły obsługi danych sygnałów i ich propagacji. Np. włączony sygnał ARMED zablokuje wysyłanie sygnału do rolet antywłamaniowych które i tak są opuszczone na dół. Pozwala to odciążyć poszczególne odbiorniki od niepotrzebnego nakładu obliczeń.

W celu ulepszenia zastosowano zapis do pliku wartości domyślnych dla temperatur przez co możliwe jest odtworzenie stanu sprzed zatrzymania symulacji.

8. Krótka instrukcja obsługi

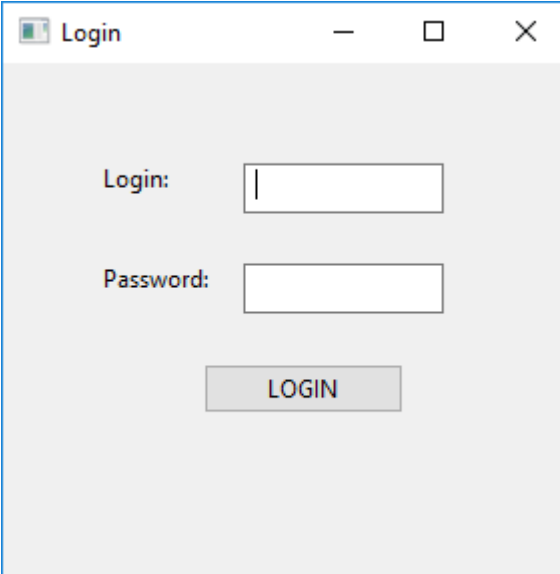
W rozdziale Przykłady zawarte są przykładowe grafiki pozwalające zrozumieć opisane tu działania.

1. Otworzyć projekt.
2. W katalogu /src uruchomić komendę `erl -make`, która skompiluje kod.
3. Przejść do katalogu /ebin.
4. Uruchomić shell Erlanga.
5. Uruchomić aplikację komendą `app_warmup:start_gui()`.
6. W ekranie logowania wpisać login `admin`", hasło `admin`".
7. Po udanej autoryzacji nastąpi przeniesienie do ekranu głównego aplikacji
8. Uruchomić symulację przyciskiem START.
9. Obserwować zmiany stanów poszczególnych odbiorników.
10. Klikając Switch lights aktywujemy Outlet.
11. Klikając Arm/Unarm uzbrajamy/odzbrajamy alarm.
12. Wybierając minimal/maximum temperature regulujemy zakres działania klimatyzacji.
13. Sensory alarmu można zwiększać przyciskiem ADD ALARM.
14. Kończymy symulację przyciskiem STOP.
15. Kończymy działanie programu przyciskiem CLOSE.

9. Przykłady

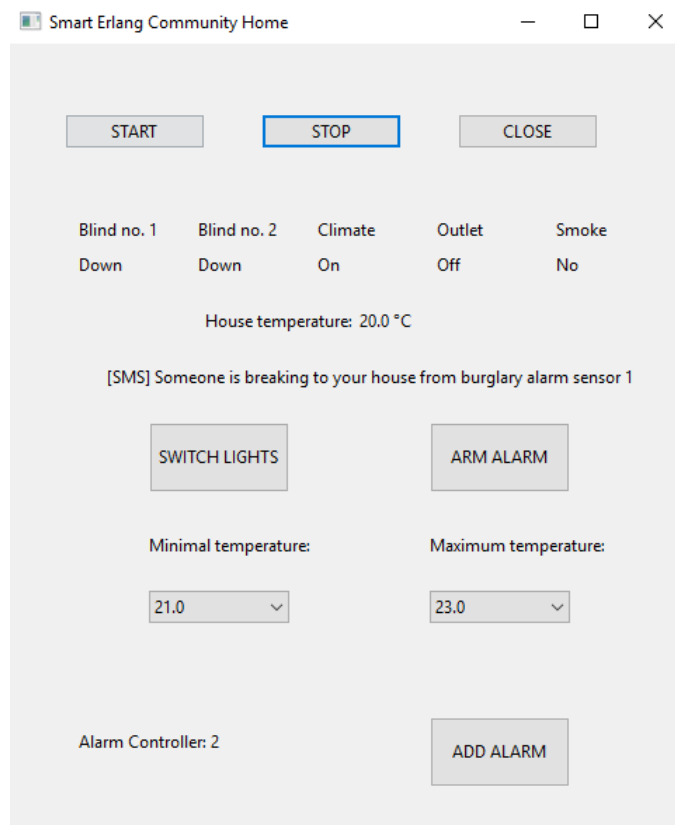
```
[receiver_controller] : 19:54:18 : Handling request finished  
[phone_notifier] : 19:54:18 : [SMS] Someone is breaking to your house  
[blind_receiver] : 19:54:18 : blind_receiver_listener1 going down  
[blind_receiver] : 19:54:18 : blind_receiver_listener2 going down  
[receiver_controller] : 19:54:24 : Receiver controller is handling request  
[receiver_controller] : 19:54:24 : Handling request finished  
[climate_control_receiver] : 19:54:24 : Climate control sensor received climate control off information!  
[receiver_controller] : 19:54:24 : Receiver controller is handling request  
[receiver_controller] : 19:54:24 : Handling request finished  
[receiver_controller] : 19:54:24 : Receiver controller is handling request
```

Rys. 9.1. Przykładowe logi programu

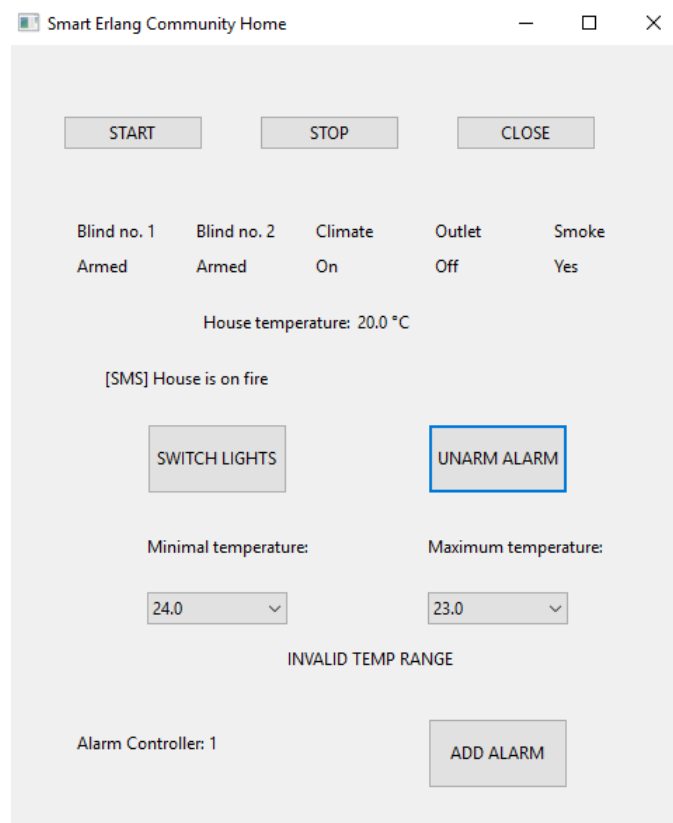


The image shows a graphical user interface for a login screen. It has a title bar with the text 'Login' and standard window control buttons (minimize, maximize, close). The main area is light gray and contains two labels, 'Login:' and 'Password:', each followed by a white text input field. Below these fields is a gray button with the text 'LOGIN' in white capital letters.

Rys. 9.2. Ekran logowania



Rys. 9.3. Ekran główny



Rys. 9.4. Ekran główny wersja 2

10. Możliwe rozszerzenia programu

Możliwe rozszerzenia programu mogą obejmować dodanie nowych komponentów, dynamiczną zmianę liczby sensorów, generyczne nadawanie ich nazw.

W dalszym procesie rozwoju aplikacji możnaby dodać lepsze przechowywanie danych domyślnych tak aby niepowołane osoby nie mogły ich edytować i ewentualnie zniszczyć.

11. Ograniczenia programu

Na obecną chwilę aplikacja jest w stanie z poziomu tego samego GUI po zakończeniu jednej symulacji rozpocząć kolejną. Po zamknięciu aplikacji w tym samym shellu jest możliwość uruchomienia jej ponownie bez żadnych konfliktów.

Ograniczeniem jest tylko jeden sensor temperatury i jeden sensor dymu, możliwość wykonywania jednej symulacji w jednym shellu i z góry ustalona hierarchia sensorów/odbiorników.