


# System Wieloagentowy

Dokumentacja wstępna

Wojciech Rokicki  
Krzysztof Pierczyk



## 1. Wstęp

Aplikacja będzie realizować system symulujący sieć transportową przedsiębiorstwa. Obsługa symulatora odbywać się będzie poprzez przeglądarkę internetową, zaś sam program zostanie zaimplementowany w strukturze klient-serwer, przy czym część serwerowa i kliencka będą mogły zostać uruchomione na oddzielnych maszynach. Komunikacja między obiema częściami odbywać się będzie poprzez protokół HTTP.

## 2. Część kliencka

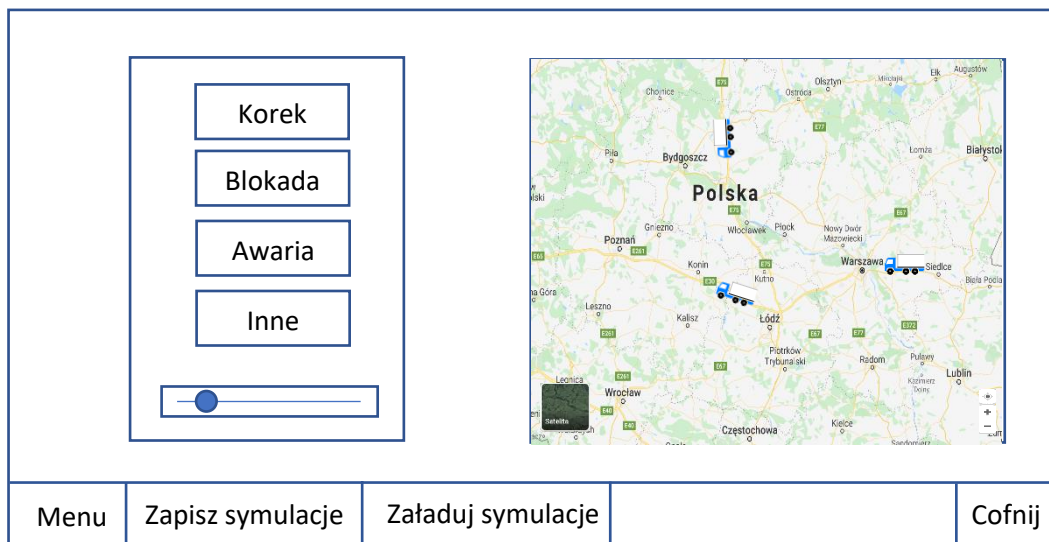
Część kliencka będzie umożliwiać pełne skonfigurowanie oraz przeprowadzenie symulacji. Po ustaleniu parametrów ich lista zostanie przesłana do serwera, który przeprowadzi symulację i odeśle do klienta jej wynik, który będzie mógł obejrzeć jej efekt w czasie pseudo-rzeczywistym. Interakcje związane z modyfikacją warunków symulacji w trakcie jej wizualizacji, jak na przykład wymuszenie zablokowania drogi, przeładowania transportu, skutkować będzie ponownym wysłaniem zapytania do serwera, który po wykonaniu obliczeń prześle do klienta zaktualizowany fragment symulacji. Użytkownik będzie miał m.in możliwość:

- Podglądu położenia i stanu pojazdów w czasie symulacji
- Śledzenia aktualnego stanu ruchu na kluczowych drogach
- Wprowadzania do symulacji nowych zdarzeń w trakcie jej trwania, m. in. zamykania ulic, powodowania wypadków, wymuszania przerw u kierowców, ...
- Kontrolowania prędkości symulacji oraz jej wstrzymywania
- Zapisania symulacji do pliku, który będzie możliwy do wczytania w późniejszym czasie

Miasta	Agent	Param mapy	Param czas	Inne
<input type="radio"/> Warszawa	I. poj. <input type="text"/>	I. kier. <input type="text"/>		
<input checked="" type="radio"/> Kraków	<input type="text"/>	<input type="text"/>		
<input type="radio"/> Gdańsk	<input type="text"/>	<input type="text"/>		
...				
<input type="radio"/> Łódź	<input type="text"/>	<input type="text"/>		

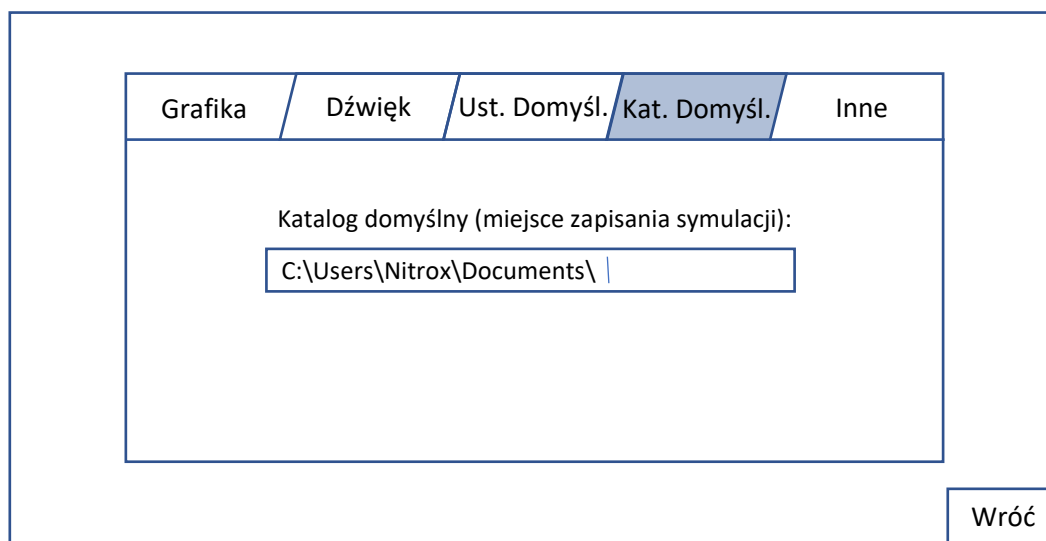
Symuluj	Zapisz Param.	Załaduj Param.		Wróć
---------	---------------	----------------	--	------

Rysunek 1. Konfiguracja parametrów symulacji - wstępny koncept



Rysunek 2. Okno symulacji - wstępny koncept

Główne menu umożliwi także konfigurowanie parametrów działania aplikacji, jak na przykład ustawienie wartości domyślnych dla przeprowadzanych symulacji w celu uniknięcia każdorazowego ich wpisywania, czy ustalenie domyślnych katalogów, do których zapisywane będą symulacji i zestawy parametrów.



Rysunek 3. Główne menu opcji - wstępny koncept

## Wykorzystane technologie

Komunikacja między klientem a serwerem zostanie zaimplementowana przy użyciu języka Python. Oprawa graficzna zostanie przygotowana w języku HTML5 wraz z Kaskadowymi Arkuszami Stylów (CSS). Za warstwę interakcyjną aplikacji odpowiedzialna będzie biblioteka jQuery lub jej odpowiednik w języku Python – pyquery. Obsługa mapy wykorzysta jedno z API oferowanych przez Google – odpowiednio Maps JavaScript API lub Maps Embed API.

### 3. Część serwerowa

Część serwerowa odpowiedzialna jest za rozdzielenie otrzymanych informacji od Klienta na poszczególnych agentów symulacji (odrębne wątki), zasymulowanie całego procesu dostaw i zwrócenie wyniku w postaci zbioru próbek stanów symulacji do bazy danych, a końcowo do klienta.

Agenci po otrzymaniu dotyczących ich danych wyznaczają najkorzystniejszą trasę, przy użyciu odpowiedniego algorytmu trasowania, po czym odsyłają ją do serwera w formie pewnej ilości próbek. Serwer dodatkowo optymalizuje odesłane przez agentów informacje zwrotne z uwzględnieniem zależności między agentami (np. przecinanie się dróg – możliwość zamiany kierowców, wypadek drogowy itp.), po czym przesyła wynik do bazy danych, który potem otrzymuje klient.

W trakcie wizualizacji serwer może otrzymać prośbę o aktualizację symulacji dotyczącą zaistniałych sytuacji na drogach, wprowadzonych przez użytkownika aplikacji. W takim przypadku serwer decyduje czy przeliczyć całą symulację, czy też fragment z uwzględnieniem tylko wybranych agentów.

Serwer może ponownie wywołać agenta do zaplanowania trasy (albo jej części) jeżeli na etapie optymalizacji stwierdzi, że jest to wymagane ze względu na zmiany wprowadzone u innych agentów.

#### Wykorzystane technologie

Serwer HTTP zostanie zrealizowany przy użyciu biblioteki Boost Beast w standardzie C++11, a format danych wymienianych pomiędzy Klientem a Serwerem przy pomocy JSON. Baza danych zostanie zrealizowana przy pomocy MySQL.

# Lista prac

Opis zadania	Liczba godzin
Interfejs graficzny	
Statyczne strony wykorzystywane w menu aplikacji (HTML)	4
Statyczne strony wykorzystywane w interfejsie symulacji (HTML)	3
Obsługa Maps JavaScript API/Maps Embed API	5
Mechanizmy odpowiedzialne za interaktywne elementy GUI (Python/JavaScript)	3
Mechanika części klienckiej	
Komunikacja klient-serwer poprzez protokół HTTP	4
Format zapytań i parametrów przesyłanych do serwera	2
Globalne opcje aplikacji	5
Mechanizm wprowadzania parametrów symulacji	4
Wizualizacja symulacji na podstawie danych odesłanych przez serwer	9
Mechanizm modyfikowania symulacji w czasie rzeczywistym	8
Możliwość zapisywania i wczytywania przeprowadzonych symulacji	2
Możliwość zapisywania i wczytywania zestawów parametrów symulacji	2
Możliwość zapisywania i wczytywania symulacji/parametrów z serwera	2
Dodatkowe motywy graficzne dla aplikacji*	4
Oprawa dźwiękowa*	4
Część serwerowa	
Format przesyłania symulacji i parametrów do klienta	2
Zarządzanie agentami reprezentującymi pojazdy	3
Algorytmy wyznaczania tras przez agentów	9
Komunikacja agent-serwer	3
Zbiorowa optymalizacja tras wyznaczonych przez agentów	6
Obsługa zapytań o modyfikację parametrów symulacji w trakcie jej trwania	8
Obsługa warunkowego trasowania przez agentów	3
Implementacja bazy danych przechowywanej na serwerze	4
Wprowadzenie wielowątkowości	4
Inne	
Napisanie skryptów budujących (SCons/CMake)	2
Przetestowanie aplikacji na systemach Windows i Ubuntu	1
Napisanie dokumentacji dla użytkownika	2
Napisanie dokumentacji dla dewelopera	3
<b>Łącznie</b>	<b>108</b>

\* - funkcjonalność dodatkowa, której zaimplementowanie zależne będzie od pracochłonności projektu