Akademia Górniczo-Hutnicza
im. Stanisława Staszica w Krakowie

Wydział Elektrotechniki, Automatyki, Informatyki i Elektroniki

KATEDRA AUTOMATYKI

PRACA MAGISTERSKA

ŁUKASZ ZIEŃKOWSKI

INTERAKTYWNA MAPA ŚWIATA Z DODATKOWĄ OSIĄ CZASU

PROMOTOR:
dr inż. Grzegorz Rogus

Kraków 2013

OŚWIADCZENIE AUTORA PRACY

OŚWIADCZAM, ŚWIADOMY ODPOWIEDZIALNOŚCI KARNEJ ZA POŚWIADCZENIE NIEPRAWDY, ŻE NINIEJSZĄ PRACĘ DYPLOMOWĄ WYKONAŁEM OSOBIŚCIE I SAMODZIELNIE, I NIE KORZYSTAŁEM ZE ŹRÓDEŁ INNYCH NIŻ WYMIENIONE W PRACY.

. . . . . .

PODPIS

AGH

University of Science and Technology in Krakow

Faculty of Electrical Engineering, Automatics, Computer Science and Electronics

DEPARTMENT OF AUTOMATICS

MASTER OF SCIENCE THESIS

ŁUKASZ ZIEŃKOWSKI

THESIS IN LATEX

SUPERVISOR:
Grzegorz Rogus Ph.D

Krakow 2013

Serdecznie dziękuję . . . tu ciąg dalszych podziękowań np. dla promotora, żony, sąsiada itp.

Spis treści

1. Wstęp

1.1. Temat pracy

Tematem pracy jest połączenie ogólnodostępnych źródeł informacji jakimi są mapy świata z możliwością zmiany czasu i co się z tym wiąże zmiany prezentowanych informacji. Tradycyjne podejście zakłada statyczną prezentację obrazu terenu. Są to często zdjęcia lotnicze które prezentują sytuację w konkretnym czasie. Poniższej pracy podjęto próby stworzenia aplikacji która bedzie rozszeżała tradycyjne podejście, doda czwarty wymiar, czas.

1.2. Geneza teamatu

Mapy są znane człowiekowi od dawna, ich rozwój sprawił że obecnie, szczególnie ich wirtualne wersje zawierają dużą ilość dodatkowych opcji które mają za zadanie dostarczyć jeszcze więcej informacji. Niestety żadko można spotkać funkcjonalność która pozwalałaby na spojżenie w różne okresy czasu, zazwyczaj na jednej mapie znajduje się maksymalnie wąski okres czasu.

Obecny gwałtowny rozwój miast,zmian terytorialnych tworzy potrzebę prezentacji zachodzących zmian. Dodatkowym zastosowaniem może być prezentacja zmian pogodowych takich jak przesuwania się frontów atmosferycznych.

1.3. Realizacja pracy

Praca ma na celu stworzenie aplikacji dodającej do tradycyjnie znanych map dodatkowego wymiaru, czasu i możliwośći jego kontroli. Zadanie to wymaga rozwiązania kilku aspektów, są nimi.

- Przechowywania i przysułu danych, docelowe zbiory danych mogą być duże. Należy zapewnić możliwość ich przechowywania w sposób przyjazny dla użytkownika.
- Wizualizacja danych, aplikacja powinna zapewniać estetyczną prezentację danych.
- Wiarygodność informacji, użytkownik powinien być pewien że informacje z których korzysta są poprawne i nie zmienione przez nieautoryzowane osoby.

Każdemu z nich poświęcony jest odzielny rozdział aby w pełni przeanalizować tematykę.

Zdecydowano się na stworzenie frameworku, programu który dostarczy intuicyjny interfers do jego obsługi. Aby jego konfiguracja nie zajmowała czasu zrezygnowano z korzystania tradycyjnych baz danych, konieczność ich konfiguracji zniechęcała by nowych użytkowników. Wymaga to wykorzystania innego rozwiązania, które pozwalałoby na przechowywanie danych po stronie klienta, najlepiej w przeglądarkce klienta.

W początkowej fazie projektu rozważano możliwość współpracy osi czasu z możliwośćia tworzenia kształtó dostarczana przez HTML5, niestey brak kompatybilności tego rozwiązania z wybranymi mapami wymusił odrzucenie tej koncepcji.

1.4. Zawartość pracy

W rozdziale 2 przedstawiono krótki rys historyczny związany z obecnością map w histori człowieka. Rozdział 3 zawiera omówienie teoretycznych aspektów które zą związane z tematem pracy. Rozdziały 4 i 5 szczgółowo opisują implementację, napotkane problemy podczas tego procesu i sposób ich rozwiązania. Omówiono dlaczego jaki czynniki zdecydowały na wybrane konkretnych rozwiązań. Rozdział 6 zawiera krótki przykład wykorzystania stworzonej aplikacji. Ostatni rozdział jest podsumowaniem całej pracy, co udało się wykonać, co stanowiło barirę nie do pokonania a także zawiera informację o ewentualnych ścieżkach rozwoju.

2. Rys historyczny

Pomimo że w obecnych czasach, mapy nie są niczym niezwykłym, i wydają się codziennością nie zawsze tak było. W histori człowieka można znaleść okresy czasu kiedy kartografia była nieznana,a jeśli była to bardzo niedokładna.

Pierwsze malowidła które można uznać za graficzną reprezentację otoczenia archeolodzy napotkali w okolicach Pavloc (Czechy), datowaną są one na 25 wiek przed naszą erą [Lam09] . Na 14 wiek p.n.e datowane są wykopaliska w Navarre(Hiszpania) obejmujące rysunki na piaskowcu [Cho09].

Jeszcze w XIV kartografia była bardzo ograniczona, mapy którymi posługiwano się nie były zbyt dokłądne. 12 października 1410 gdy Krzysztof Kolumb dotarł do brzegów wyspy San Salvador uznaj ją za jedną z wysp japońskich [Irv28]. Aby lepiej zrozumieć powód tej pomyłki wystarczy spojrzeć na mapę stworzoną w XV wieku przez kartografa Henricusa Martellusa 2.1. Widzimy na niej że m.in. obie ameryki nie były znane uwczesnym ludzion.

Rysunek 2.1: Mapa świata z 1489 roku.

3. Dostępne rozwiązania

Przed rozpoczęciem implementacji własnego rozwiązania postanowiono przeprowadzić badania mające na celu analizę aktualnie dostępnych rozwiązań spełniających przynajmniej w części założenia. Proces ten nie ma na celu znalezienie idealnego rozwiazania lecz poznanie różnych podejść do zagadnienia. Analiza różnych technologi które mogą być wykorzytane pozwoli na świadome wybranie najlepszych.

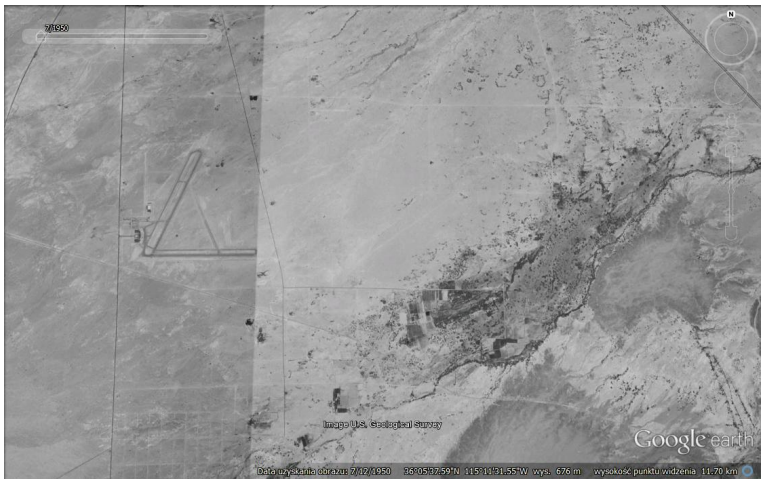
3.1. Google Earth

Funkcjonalnośc interaktywnych map można znaleść w aplikacji stworzonej przez firmę Google. Program ten jest napisany przy użyciu języka C++, nie służy on do tworznia aplikacji mobilnych, dlatego nie został on wybrany jako kandydat do naszej pracy.

Wykorzystuje ono statyczne obrazy będące zdjęciami satelitarnymi dla obrazów w dużej wysokośći lub zdjęciami wykonanymi z pokładów samolotów. Pomimo duzej atrakcyjności ma ono bardzo ograniczoną możliwośći zmiany oglądanych danych, ograniczone do ilości wykonanych zdjęć, dodatkowo większość punktów najstarsze dane ma z lat 50 XX w.

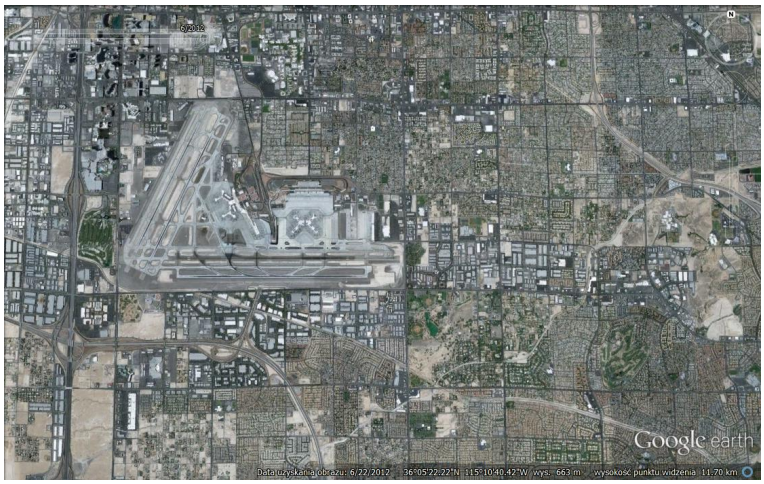
Przykład takiej sytuacji został przedstawiony na rysunku 3.1, obraz terenu na którym powstanie miasto Las Vegas w roku 1950. Jak teren ten wyglądał w roku 1977 widzimy na rysunku 3.2, pomimo widocznych zmian teren ten nadal w dużym stopniu jest pustynny,dopiero na rysunku 3.3 widzimy aktualny stan miasta.

Dzięki funkcji zmiany punktu i kąta patrzenia, pokazywania ciekawych miejsc czy chociażby włączania trybu w którym budynki nabierają formy przestrzennej, 3D, możemy poprzez zabawę i wirtualne wycieczki poszeżać naszą wiedzę o otaczającym nas świecie.



Rysunek 3.1: Las Vegas w 1950 roku.

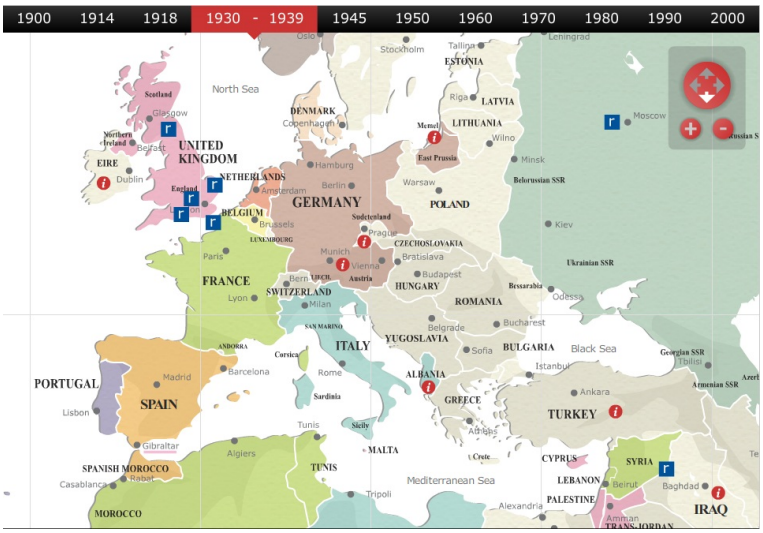
Rysunek 3.2: Las Vegas w 1950 roku.



Rysunek 3.3: Las Vegas w 1950 roku.

3.2. Aplikacje wykonane w technologi Flash

Podczas prac badawczych natrafiono na kilka aplikacji któych główna część aplikcji została wykonana w technologi flash. Przykładami są interaktywna mapa historii europy [wor], mapa prezentująca zmianę granic państw w XX wieku [flaa] przykład na rysunku 3.4. Oba przykłady zawierają bogatą szatę graficzną, ich wykonanie jest bardzo estetyczne. Zaletą wykorzystanie mocy obliczeniowj karty graficznej do wyświetlania obrazów(dostępne od wersji 10.1 [flab]), dzięki temu procesor jest może wykonywać inne zadania.



Rysunek 3.4: Europa w latach 30.

Wykorzystanie tej technologii wydaje się dobrym wyborem, niestety posiada kilka wad które ja eleiminują. Pierwszą z nich jest brak wsparcia technicznego na wszystkich urządzeniach mobilnych, przez co jej wykorzystanie ogranicza rynek docelowy. Wymagana jest instalacja detykowanego odtwarzacza który umożliwia na korzystanie z programów stworzonych w tym języku. Ponieważ tworzona aplikacja jest jednym komponentem wymagane jest jest ściągnięcie z serwera na lokalny komputer aby można było z niej korzystać. Ostatnim aspektem jest brak kompatybilności elementów flash-owych z głównymi wyszukiwarkami internetowymi, ich indeksacja jest utrudniona(możliwe wykorzystanie tagów) a ich zawartość całkowicie niewidoczna w procesie tym. Z uwagi na powyższe wady technologia nie została wykorzystana w omawianej pracy.

3.3. Wybrane technologie

Po analizie dostępnych rozwiązań zdecydowano się na wykorzystanie w pracy następujących technologi.

– HTML 5 - budulec każdej strony internetowej

– JavaScript - zapewnienie interakcji i zmiany konkentu bez konieczności odświeżania całej strony

– Less - agrgacja i uporządkowanie stylów CSS

4. Definicja problemu

Poniższy rozdział zawiera omówienie technologi które zostały wykorzystane w procesie tworzenia aplikacji, ze szczególnym uwzględnieniem funkcjonalności które okazały się najbardziej przydatne. Następnie zawarto szczegółowy opis sposóbów rozwiązania problemów które należało rozwiązać w trakcie tworzenia interaktywnej mapy.

4.1. Wykorzystane technologie

4.1.1. JavaScript

Wybór języka JavaScript to tworzenia aplikacji został podyktowany jego dużymi możliwościami na polu tworzenia projektów do użytku na przeglądarkach internetowych.Pozwala na łatwe tworzenie asynchronicznych aplikacji, oznacza to aktualizowanie jedynie wybranych danych na ekranie moniotra bez konieczności przeładowania całej strony. Jest to szczególnie istotne w omawianym przypadku, oznacza to brak potrzeby wczytywania i renderowania całego tła mapy za każdym razem gdy nastąpi nawet małą zmiana.

Spełnia warunki paradygmatu obiektowego, umożliwia korzystanie z obiektów które mogą odpowiadać pewnej klasie rzeczywistych przedmiotów lub rzeczy, imperatywnego, składa się z komend które zmieniają stan programu jak też funkcyjne, umożliwia tworzenie funkcji które w zależności od podanych argumentów wejściowych dostarczają odpowiednich danych wyjściowych.

Bardzo ważną cechą języka jest fakt jego wykonywania po stronie klienta. Oznacza to wykonywanie obliczeń i korzystanie z pamięci najczęściej na komputerze osoby która korzysta z aplikacji. Pozwala to na zmiejszenie wymagać w stosunku do serwera(wykonyje on mniej operacji i zapisuje mniej danych w swojej pamięci)

Obecnie powstają frameworki których zadaniem jest stworzenie małej, lekkiej aplikacji webowej wykorzystującej w maksymalny sposób omawiany język.[BG13] Przykładem takiego rozwiązania jest AngularJs, korzysta on z architekury MVC w której większość pracy została przeniesiona na stronę klienta. Kontroler wykonuje obliczenia w przeglądarce użytkownika, odciąża to główny serwer, pozwala na szybszą i płynniejszą pracę większej ilości osób.

Nie wątpliwą zaletą takiego podejścia jest wzrost opdorności na ataki typu Dos(en. Denial of Service), polega on na przeciążeniu aplikacji dostarczającej określone dane lub usługi do momentu gdy przestaje ona opdowiadać na jakiekolwiek zapytania. Aplikacje które wykonują większość operacji po stronie serwera(różnego typu obliczenia matematyczne, analizę danych) aby sprostać dużej ilośći klien-

tów która występuje w tego typu atakach nakłada duże wymagania w stosunku do wykorzystywanego sprzętu elektrycznego. Przeniesienie ciężaru z serera na końcowego klienta wykonywania większości operacji pozwala nie tylko na obsłużenie większej ilości użytkowników ale także może wpłynąć na zmiejszenie kosztów utrzyamnia serwera.

Prosty przykład na stronie http://angularjs.org/ prezentuje w jak prosty sposób można stworzyć prostą liste rzeczy do zrobienia. Poniżej zaprezentowano fragment odpowiedzialny za wyświetlenie zadań z dodatkowym polem określającym czy zostało już wykonane.

```
1
2       //fragment kodu html
3     <div ng-controller="TodoCtrl">
4       ...
5       <li ng-repeat="todo in todos">
6         <input type="checkbox" ng-model="todo.done">
7         <span >{{todo.text}}</span>
8       </li>
9       ...
10    </div>
11
12      //kontroler
13      function TodoCtrl($scope) {
14        $scope.todos = [
15          {text:'learn angular', done:true}];
16        ...
17      }
```

Listing 4.1: AngularJs

4.1.2. Możliwości HTML5

HyperText Markup Language, hipertekstowy język znaczników Pozwala na opisanie struktury informacji zawartych na stronie internetowej, to dzięki niemu przeglądarka moze rozróżnić takie elementy jak hiperłącze, akapit czy chociażby nagłówek.

Podobnie jak w przypadku XML, tak i tutaj wymagane jest aby wykorzystywane znaczniki umieszczane były w nawiasach ostrokątnych, a każdy z nich miał swoje domknięcie.

Poprawnym zapisem jest <p>Wiadomość<\p> który oznacza pojedyńczy akapit. Zapis <p>Wiadomość<p>, który różni się od poprzedniego brakiem znaku "\"w drugim znaczniku, czyni to ten zapis niepoprawnym. Istnieje możliwość aby wykorzystać pojedyńczy znacznik, przykładem jest <br \> określający wstawienie nowej lini w miejscu wystąpienia tagu.

Obecnie powszechnie używany standart HTML4 ma niestety wiele ograniczeń, z tego powodu pracowano nad jego następnikiem. 22 stycznia 2008 W3C opublikował HTML5, wtedy jeszcze jako jedynie szkic.

4.1.3. Less

Do stworzenia bardziej zaawansowanych arkuszy stylów CSS (en. Cascading Style Sheets) wykorzystaniu narzedzie które pozwala na łatwe tworzenie i utrzymywanie reguł. Głównymi zaletami są:

– Deklarowanie zmiennych.

Jeśli chcemy wykorzystać jedną wartość(przykładowo jeden kolor) w wielu miejscach wystarczy w głównym pliku zadeklarować zmienną i wykrzystywać ją w każdym miejscu gdzie chcemy użyć tej wartośći.

Przykładowo chcąc wykorzystać kolor czerwony wykorzystywany w kolorystyce uczelni AGH deklarujemy zmienną @aghRed, następnie w miejscu gdzie chcemy aby pojawił się ten kolor korzystamy z istniejącej zmiennej

```
1 @aghRed: #a71930;
2 bgcolor: @aghRed;
```

Listing 4.2: json

– Mixiny

Możemy tworzyć nowe klasy które będą posiadały właściwości innej, wcześniej zadeklarowanej.Pozwala to na tworzenie kodu bardziej czytelnego, niepowielanie go.

```
1 .RoundBorders {
2  border-radius: 5px;
3  -moz-border-radius: 5px;
4  -webkit-border-radius: 5px;
5 }
6 #menu {
7  color: gray;
8  .RoundBorders;
9 }
```

Listing 4.3: json

– Osadzanie elementów wedlug dziedziczenia

Tworząc zagnieżdzoną strukturę storny często elementy wewnętrzne zależą od elementów nadrzędnych. Wygląd komurki może się różnić w zależości od rodzaju tabeli. W tworzeniu tak zadnieżdżonych struktur pomocna okazuje się omawiana cecha.

4.2. Przechowywanie i transmisja danych

Rozdział ten zawiera informacje związane z analizą wybranych metod przechowywania danych a także możliwości ich transmisji.

4.2.1. GIS

GIS(Geographical Information Systems) jest to System Informacji Geograficznych, jest to zbiór wiedzy, informacji do pozyskiwania i analizowania danych przestrzennych. Dane są pobieranie w trakcie naziemnych pomiarów jak i zbierane przy użyciu systemu GPS. Informacje te są następnie zapisywane do określonych formatów,dzięki czemu możliwa jest ich dalsza analiza. http://www.gis-support.pl/co-to-jest-gis/

Wykorzystywany układ współrzędnych

Istnieje wiele układów współrzędnych któe służą do opisu pojedyńczego punktu na powierzchni ziemi.Na stronie http://www.spatialreference.org/ref/epsg/ (dostęp) znajduje się lista zawierająca przykłady zapisu różnych obszarów kuli ziemskiej przy użyciu różnych formatów. Zawierają one niezbędne infomacje w konkretnym zapisie.

Wersją która ma za zadanie być ogólnoświatowym formatem jest WGS(World Geodetic System). Jego ostatnia wersja WSG84 jest powszechnie używana w urządzeniach do nawigacji. Tradycyjny zapis oparty jest na wykorzystaniu stopni, minut i sekunk, jest on obecnie wypierany przez nowszy,łatwiejszy do obliczeń komputerówych. Poniżej zaprezentowano jak wygląda zapis w starszym i nowszym punktu określającego położenie budynku A-0 uczelni AGH.

– Pierwotny zapis

  50°03’52.2803", 019°55’23.7968"

– Zapis unowocześniony

  50.06452231874906, 19.923276901245117

Drugi zapis został wykorzystany do przechowywania danych w plikach, tak aby wymiana i wspólna praca była jak najprostsza. Wybór ten sprawił że nie występuje problem konwersji punktu do formatu czytelnego dla komputera, można go bez problemu odczaytać jako liczbę.

4.2.2. Storage

Aby stworzyć framework któy był by w stanie działać przy minimalnej konieczności konfiguracji dodatkowych środowisk postanowiono aby dane w pierwszej kolejności były przechowywane po stronie klienta. Do tego celu nadaje się funkcją stworzona w ostatniej wersji HTML którą jest Storage. Jest ona dokładniej omówiona w podręczniku do HTML5 [Pil10]. Pozwala on na przechowywanie danych w przeglądarce użytkownika. Różnicą w stosunku do ciasteczek które również potrafią przechowywać informacje o konretnym użytkonwniku jest:

– Większy rozmiar dostępnej pamięci m.in. Chrome 5MB , IE 10MB

– Informacje przechowywane są po stronie użytkownka, nie są przesyłane za każdym razem do serwera.

– Informacja może być przechowywana przez długi okres czasu.

Dodatkowo nie można pominąć faktu istnienia dwóch rodzaii tej pamięci.

– Session Storage Dane przechowywane są w kontekście sesji użytkownika, są one tracone w momencie zamknięcia okna przeglądarki.

– Local Storage Teoretycznie dane są przechowywane w nieskończoność, do momentu kiedy użytkonwik nie usunie ich. Zamknięcię sesji nie powoduje usunięcia danych.

Powodem który wymaga wykorzystania tego typu pamięci jest możliwość przechwowywania danych nad którymi użytkonwik pracuje w aktualnym czasie. Nie potrzebuje on informacji które były dla niego istotne podczas poprzedniej wizyty,sesji. Sytuacja ta jednoznacznie wskazuje że lepszym wyborem jest wybór pamięci sesyjnej(Session Storage).

Wadą tej pamięci jest jej interfejs. Obecnie przechowywany sposób danych to mapowanie w postaci napis->napis. Wymusza to aby każde dane które chcemy przechować muszą być w formie ciągu znaków. Przykład 4.2.2 przedstawia w jaki sposób możemy obiekt zawierający imię i nazwisko zapisać w pamięci. Linia 4 przedstawia obiekt w postaci której chcielibyśmy go przechować. Niestety zwykłe przypisanie do zmiennej w pamięci powoduje że jedynie typ instancji zostaje zapisany. Aby móc zapisać w poprawnej formie dane musimy doknać serializacji danych. Czynność tą możemy wykonać przy pomocy metody stringify z obiektu JSON, wynikiem jest ciąg znaków który możemy bez problemu zapisać w pamięci sesyjnej. Do odzyskania pierwotnego obiektu, odtworzenia go z zapisanego napisu wykorzystujemy metodę parse również z obiektu JSON.

```
1  uzytkownik={};
2  uzytkonwik.imie='Jan'
3  uzytkownik.nazwisko='Kowalski'
4  //uzytkownik : Object {imie: "Jan", nazwisko: "Kowalski"}
5
6  sessionStorage.u1 = uzytkownik
7  //sessionStorage.u1 : "[object Object]"
8
9  sessionStorage.u2 = JSON.stringify(uzytkownik)
10 //sessionStorage.u2 : "{"imie":"Jan","nazwisko":"Kowalski"}"
11
12 uzytkonwik2 = JSON.parse(sessionStorage.u2)
13 //uzytkonwik2 : Object {imie: "Jan", nazwisko: "Kowalski"}
```

Listing 4.4: json

Wsparcie dla pamięci Storage nie jest obecne zazwyczaj w nowszych wersach przeglądarek. Na stronie http://www.html5rocks.com/en/features/storage możemy sprawdzić aktualny stan większości przeglądarek.

4.2.3. Relacyjne bazy danych

Początkowo zakładano wykorzystanie relacyjnej bazy danych do przechowywania danych. Zdecydowaną się na MySqL(z uwagi na jej powrzechne wykorzystanie i brak opłat licencyjnych),jedną z table wypełniono testowymi danymi w ilości 1 miliona rekordów o strukturze zbioru punktów o określonym

czasie trwania zawierała kolumny takie jak data początkowa i końcowa występowania, dwie współrzędne. W celach optymalizacyjnych koszt zapytań zastosowano indeks na pola po któych wykonywane będzię wyszukiwanie danych. Stworzona została prosta aplikacja napisana w języku Python przy pomocy frameworka Django v.2.1, miałą ona na celu wysłanie wysłanie zapytania o konkretne punkty które zawierały się w określonym przedziale czasie(ich czas trwania zawierał się w szukanym zakresie) a następnie otrzymane punkty rysowała na ekranie. Na tak przygotowanym środowisku przeprowadzono testy sprawdzające czas potrzebny na pobranie i wyświetleie danych.

Zadowalające wyniki otrzymywano gdy ilość elementów nie przekraczała 10 tysięcy punktów. Gdy zwiększono szukany zakres dat, tym samym zwiększono ilość punktów do wyświetlenia czas potrzebny do wyświetlenia danych wynosił ponad jedną sekundę. Z uwagi na dużą ilość danych które towarzyszą informacjom mapowym(docelowa baza będzie zawierała więcej danych a ilość jednorazowo wyciąganych informacji może być większa) zdecydowano się nie kontynuować prób z wykorzystaniem tego typu baz danych.

4.2.4. NoSQL

W celu pernamentnego przechowywania i rozpowszechniania danych pomiędzy różnymi użytkownikami postanowiono wykorzystać nierelacyjną bazę danych jaką jest MongoDB. Posiada ona interfejsc służący do komunikacji z JavaScript-em, możliwe jest wykonywanie skryptów które bezpośrednio połączą się i uzyskają dane[mon]. W omawianym projekcie nie jest to możliwe, nawet przy użyciu zapytań Ajax-owych nie będzie możliwe zapewnienie bezpieczeństwa danym pozwalającym na połączenie a w efekcjie edycję danych bezpośrednio w bazie danych. Postanowiono aby aplikacji która będzie korzystała z tworzonego frawerork-a dostarczała metod zapewniających dostęp do bazy danych. Zachowanie takie spełnia założenia projektu, infromacje i wszelkie zmiany dokonane na nich są przechowywane w lokalnej pamięci z możliwością eksportu ich do pliku tekstowego bez konieczności ingerencji w aplikację nadrzęczną. W przypadku chęci posiadania błyskawicznej możliwości dzielenia się informacjami i dokonywanymi zmianami należy zapewnić interfejs do tego celu. Przykładową implementację takiego rozwiązania stworzono w języku PHP. Powstała strona która korzystała z opracowanego frameworku, dla celów komunikacji z bazą danych udostępniała adres ./resources/mongodb który na zapytanie przesłane w metodzie POST wykonywała zwracała dane dla podanych warunków. Adres ten dla odpowiednich danych potrafił dokonać zmian(edytować lub usunąć dane). Ponieżej zaprezntowano listening prezentujący minimalne obiekty które nalezy wysłać na przygotowany adres aby wykonać odpowiednie akcje. Pierwszy ma za zadanie wykonać akcję select pobierający wszystkie dane należące do mapy od numerze "123"które zaszły w podanym zakresie czasu. Drugi obiekt odnosi się do akcji aktualizacji danych, zmienia położenia punktu o id równym 234,jego tytuł a także poziomy na których będzie on widoczny, funkcjonalnośc ta została opisana w sekcji 4.3.

```
1 Object {method: "select", mapId: "123", fromDate: "2013-01-01 12:30", toDate: "
      2013-01-31"}
2
3
4 Object{}
```

```
5  YCord: "19.9232769"
6  mapId: "123"
7  pointId: "234"
8  method: "update"
9  showFrom: "1"
10 showTo: "8"
11 title: "Lepszy tytuł"
12 xCord: "50.0645223187"
```

Listing 4.5: caption2

DOM

4.3. Wizualizacja

Poniższa sekcja zawiera próbę analizy dostępnych metod prezentacji danych zarówno zbiorów punktów, zdjęć i animacji. Główną uwagę zwrócono na 3 główne sposoby wizualizacji grafik. Ostatnia część zawiera zbiór dostępnych funkcjonalności, stanowi przewodnik po dostępnych możliwościach i ukazuje w jaki sposób można personalizować dane.

4.3.1. Rodzaje map

Tworząc aplikację która ma dostarczać informacji korzystających z map należy zapoznać się dostępnymi źródłami. Z powodu szrokiego wyboru poniżej omówione zostaną jedynie aplikacje które dostarczają informacji ogólnoświatowych. Na polskim rynku dostępnych jest kilka rozwiązań, ich główną wadą jest ograniczenie do terytorium Polski, dodatkowo często nie dostarczają one obrazów satelitarnych, są to m.in. `http://zumi.pl` **??**

Google Maps

Rysunek 4.1 przedstawia obraz otrzymany w aplikacji Google Maps. Dodatkowo włączona opcja prezentacji natężenia ruchu jedynie potwierdza duże możliwości i łatwość obsługi. Przyjazny interfejs sprawia że praca jest prosta i pozwala na osiągnięcie bardzo dobrych wyników.

Rysunek 4.1: Google Maps.

Windows Maps

Rysunek 4.2 przedstawia obraz otrzymany wykorzystując Bing Maps. W tym konkretnym przykładzie widzimy znaczną różnicę kolorów, obecność chmur pomniejsza wartość tych zdjęć. Należy wspomnieć o znacznie uboższym interfejsie dostarczanym użytkownikowi, interakcja jest w znacznym stopniu uboższa.

Rysunek 4.2: Bing Maps.

Yahoo Maps

Kolejnym dostarczycielem danych kartograficznym jest Yahoo, przykład znajduje się na rysunku 4.3. Interfejs jest zbliżony do Bing Maps, jednak obszar na którym możemy przedlądać zdjęcia jest mniejszy, obszary oddalone od większych miast nie są w pełni uwzględnione. Z tego powodu nie stanowi w pełni akceptowalnej alterantywy.

Rysunek 4.3: Yahoo Maps.

Apple Maps

Kolejną dużą marką która dostarcza informację jest Apple. Niestety nie ma wersji która pozwalałaby na dostęp do tej usługi z powszechnie używanych komputerów stacjonarnych. Dodatkowo jakość dostarczanych informacji jest bardzo złej jakości.

4.3.2. Metody prezentacji

Poniższa sekcja zawiera opis dostępnych metod służących do prezentacji danych w przeglądarce internetowej. Przedstawia zalety jak i wady każdego z rozwiązań.

Flash

Wadą tego rozwiązania wynikająca ze specyfikacji całego języka jest bardzo słaba integracja z dokumentami HTML, nie możliwa jest pełna iterakcja z pozostałymi elementani na stronie. Problem ten nie dotyczy kolejnego podejścia **??**

Canvas

Bardzo ciekawym i wartym zainteresowania dodanym elementem w nowej wersji HTML jest obecność znaczniku canvas. Pozwala on na dynamiczne , skryptowe renderowanie kształtów i obrazów. Dzięki temu obiektowi możliwe stało się tworzenie animacji czy nawet gier działających w przeglądarce bez konieczności używania dodatkowych wtyczek czy programów.

`http://techtrendy.pl/title, Pierwsza-gra-3D-napisana-w-HTML5,wid,14102779,w`

Przykładem wielkich możliwości jakie dostarcza udoskonalony język jest fakt iż już w roku 2011 powstała pierwsza trójwymiarowa gra stworzona w całości przy użyciu HTML5. Przykład grafiki widoczny jest na rysunku 4.4.

Rysunek 4.4: Pierwsza gra 3D w html5.

Funkcjonalnością która wydaje się być szczególnie przydatna w stosunku do omawianego projektu jest możliwość rysowania kształtów geomeycznych. Przykładem wykorzystania tej technologi jest 4.5 na którym przy pomocy okręgu zaznaczono rynek główny w Krakowie i jego okolice, możliwość zmiany transparentości narysowanego kształtu pozwala aby obraz podnim był nadal widoczny.

Niestety rozwiązanie to nie może zostać wykorzystane wraz z wybranym modelem prezentacji danych geograficznych. Rysunek 4.5 prezentuje potencjlne możliwośći jednak nawet analiza listeningu 4.6 wskazuje podstawowe wymaganie wykorzystanie tego rozwiązania. Bazowy element w obrębie którego możliwa jest akcja musi być elementem canvas. Otrzymany efekt jest wynikiem wykorzystania

statycznego obrazu który został zaimportowany z pamięci komputera i wykorzystany jako tło. Zabieg ten pozwala nazaprezentowanie możliwości rozwiąznia ale uniemżliwia wykorzystanie funkcjonalności które dostarczane są przez Google Maps(zmiana punktu patrzenia, stopnia przybliżenia i inne). Z tego powodu zdecydowano się na zaprzestanie prac z tym konketnym rozwiązaniem i nie wykorzystanie go w docelowej aplikacji.

Rysunek 4.5: Zaznaczenie przez canvas

```
1
2     var canvas = document.getElementById('myCanvas');
3     var context = canvas.getContext('2d');
4     var imageObj = new Image();
5
6     imageObj.onload = function() {
7     context.drawImage(imageObj, 69, 50);
8   context.beginPath();
9     context.arc(canvas.width / 2, canvas.height / 2, 90, 0, 2 * Math.PI, false);
10    context.fillStyle = "rgba(255, 0, 0, 0.5)";
11    context.fill();
12    context.stroke();
13    };
14    imageObj.src = './gm_1.jpg';
15    .
16    .
17    .
18    <canvas id="myCanvas" width="200" height="100">
19    </canvas>
```

Listing 4.6: json

SVG

SVG(en. Scalable Vector Graphics) do prezentacji danych, tworzenia animacji i płynnych zmian kształtów. Jest to powszechnie dostępny format zapisu grafiki wektorowej który oprócz prostych kształtów geometrycznych pozwala na korzystanie z zaawansowanych filtrów i efektów.

Wykorzystywany jest wektorowy format zapisu danych, został on dokładniej omówiony w sekcji 5.1.1. W wyniku takiego zabiegu plik svg opisujący obraz graficzny ma znacznie mniejszy rozmiar od oryginalnego, jednocześnie niezależnie od wielkości jego jakość jest taka sama.

Autorzy artykułu Źastosowanie języka SVG do wizualizacji danych geoprzestrzennych"[JB] zwracają również uwagę na tekstwoy format zapisu, jest on mniej oszczędny niż binarny ale pozwala na edycję dowolnym edytorem tekstu co pozwala na przeszukiwanie zawartości np. wyszikiwarkom internetowym.

W celach porównawczych wykorzystano logo AGH 4.6 o rozmiarach 305x591 pikseli o gęstości 150 dpi (en. Dots per inch, określa ilość indywidualnych punktów na odcinu jednego cala, 2.54cm), po zapisie na dysku twardym zajmował on 75 KB pamięci. Następnie wykoano zapis grafiki w testowanym formacie zajmował on 3KB , efekt został przedstawiony w listingu 4.7.

Rysunek 4.6: Logo AGH

```
1
2 <g transform="translate(0.000000,591.000000) scale(0.100000,-0.100000)" fill="#1
    e1e1e" stroke="none">
3 <path d="M984 5863 c70 -97 112 -224 143 -429 15 -97 17 -289 20 -1761 14 -1653 99 0
    100 0 0 1543 c0 933 -4 1594 -10 1673 -12 158 -37 262 -89 369 -61 127 -136 208
    -265 285 l-38 23 36 -50z"/>
4 <path d="M1115 5878 c164 -131 256 -283 306 -503 123 -100 3 -1627 4 -1628 94 0 95 0 0
     1618 c-1 1317 -3 1631 -14 1692 -50 274 -208 455 -485 556 l-66 24 40 -32z"/>
5 <path d="M1317 5867 c208 -110 326 -253 395 -477 123 -75 3 -1647 2 -1648 100 0 100 0
    0 1609 c0 1744 2 1676 -54 1826 -62 163 -173 282 -343 368 -57 29 -193 71 -263 81
    -44 6 -44 5 37 -37z"/>
6 <path fill="#00693c" d="M52 3578 l3 -1433 23 -79 c76 -271 252 -444 522 -515 146 -38
    153 -36 65 15 -170 97 -261 190 -331 339 -84 180 -78 49 -81 1668 l-3 1437 -100 0
    -101 0 3 -1432z"/>
7 <path fill="#00693c" d="M350 3622 c0 -909 4 -1417 11 -1472 12 -101 54 -228 100 -307
    73 -124 220 -238 384 -298 170 -26 -35 27 c-61 47 -164 160 -199 218 -61 102 -100
    219 -121 361 -6 44 -10 594 -10 1478 l0 1407 -100 0 -100 0 0 -1388z"/>
8 <path fill="#00693c" d="M650 3607 c0 -906 4 -1433 10 -1487 35 -282 148 -465 368 -594
     8 -5 -3 14 -22 42 -74 103 -115 233 -141 452 -13 106 -15 346 -15 1558 l0 1432
    -100 0 -100 0 0 -1403z"/>
9 <path fill="#a71930" d="M2237 3518 c-3 -1478 -3 -1494 -25 -1604 -27 -139 -69 -258
    -121 -334 l-39 -60 40 23 c68 37 186 155 228 226 226 21 36 50 101 65 143 55 168 55
    152 55 1686 l0 1412 -100 0 -99 0 -4 -1492z"/>
```

```
10 <path fill="#a71930" d="M2537 3563 c-3 -1354 -4 -1453 -21 -1523 -50 -211 -144 -362
       -301 -488 l-40 -32 65 24 c255 93 402 249 472 501 123 80 3 1443 3 1442 -101 0
       -100 0 -3 -1447z"/>
11 <path fill="#a71930" d="M2838 3573 l-3 -1438 -23 -80 c-63 -228 -190 -384 -403 -498
       -83 -44 -75 -45 56 -12 292 74 468 241 547 521 123 79 3 1433 3 1432 -100 0 -100 0
        -3 -1437z"/>
12 <path d="M1480 925 c-307 -69 -464 -387 -331 -670 101 -216 388 -308 704 -224 138 10
       -3 225 -3 226 -33 29 c-32 28 -35 29 -142 29 l-111 0 23 -31 c21 -29 23 -42 26
       -199 l3 -169 -25 -7 c-58 -14 -155 23 -199 75 -48 58 -70 135 -71 251 -1 96 1 110
       27 162 31 64 71 102 136 131 60 26 195 21 260 -11 25 -12 46 -22 48 -22 2 0 3 40 3
        88 10 89 -37 12 c-60 20 -238 23 -313 6z"/>
13 <path d="M244 889 c14 -17 26 -39 26 -49 0 -10 -61 -197 -135 -416 -74 -219 -135 -400
       -135 -401 0 -2 38 -3 85 -3 184 0 34 108 35 107 150 3 150 3 37 -111 37 -110 128 0
        129 0 -29 83 c-97 284 -244 690 -265 732 -37 77 -58 85 -220 85 l-137 0 26 -31z
       m201 -369 c24 -71 41 -131 37 -134 -8 -9 -192 -7 -192 1 0 4 21 71 47 150 34 103
       50 138 56 127 5 -9 28 -74 52 -144z"/>
14 <path d="M2263 883 c22 -37 22 -45 25 -450 l3 -413 120 0 119 0 0 190 0 190 135 0 135
       0 0 -190 0 -190 120 0 120 0 0 450 0 450 -120 0 -120 0 0 -185 0 -185 -134 0 -134
       0 -4 148 c-3 158 -9 176 -59 206 -20 12 -53 16 -128 16 l-100 0 22 -37z"/>
15 </g>
16 </svg>
```

Listing 4.7: Logo AGH w zapisie SVG.

Google Maps Overlay

Wraz z wykorzystaniem Google Maps jako bazowym źródłem danych otrzymujemy API(en.Application programming interface) umożliwiające tworzenie różnego rodzaju elementów których wygląd można w dowolny sposób edytować i zmieniać według potrzeb użytkownika.W prosty sposób możliwe jest otrzymanie efektu który został stworzony przy pomocy canvas na rysunku 4.5 zachowując pełną swobodę w dalszej pracy.

Pakiet google.maps zawiera kilka klas których zadaniem jest generowanie adekwatnych obiektów. Klasy te zostały omówione poniżej.

Marker

Pojedyńczy punkt na mapie o określonych właściwościach które można dowolnie edytować, tak aby jego wygląd, zachowanie i położenie było zgodne z oczekiwaniami.

Polyline

Linia stworzona z dwóch lub więcej punktów. Tworzone odcinki mogą być liniami prostymi lub przedstawiać zakrzywienie ziemi, nie jest możliwe tworzenie bardziej skomplikowanych łuków.

Polygon

Twrzy ciąg połączonych ze sobą punktów w określonej kolejności które tworzą zamknięty obszar. Określają obszar terenu który znajduje się wewnątrz figury.

Circle

Na podstawie środka i jego promienia tworzy okrąg z obszarem wewnętrznym.

Rectangle

Tworzy prostokąt posiadający przeciwległe boki w określonych punktach. Podobnie jak Polygon i Circle posiada obszar wewnętrzy którego właściwości graficzne można dowolnie definiować.

OverlayView

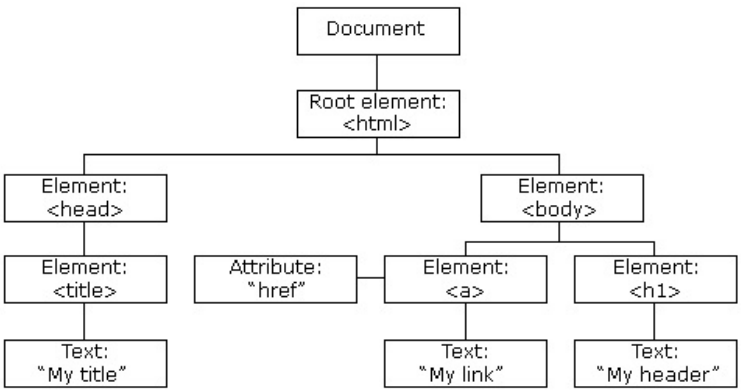
Specjalna klasa służąca do wyświetlania własnych wartsw graficznych, takich jak zewnętrzny plik graficzny. Pozwala na łatwe łączenie ogólnodostępych danych jakimi są mapy z własnymi zdjęciami, rysunkami itp.

Intuicyjna i szybka edycja danych jest zapewniona poprzez udostępnianie klasy "DrawingManager". Tworzy ona dodatkowy element na ekranie użytkownika służący do pracy z wartwami znajdujacymi się na ekranie. Komponent ten wymaga jedynie wybrania odpowiedniego elementu a następnie wyznaczenie wymaganych informacji,zazwyczaj są to jeden lub dwa punkty niezbędne do stworzenia elementu. Pozostałe właściwości mogą być edytowane w dalszym etapie prac według wymagać.

Aktualizacja struktury DOM

Wszystkie elementy znajdujące się na stronie internetowej tworzą struktutę drzewa DOM(en.Document Object Model). Określa ono wzajemne położenie poszczególnych elementów, przy wykorzystaniu JavaScript-u możliwe jest akutalizowanie poszczególnych fragmentów strony bez konieczności renderowania wszystkich elementów.

Na rysunku 4.7 zaprezentowano minimalną struktturę wymaganą do poprawego korzystania z tworzonego projektu. W części head dodane są niezbędne odwołania do skttyptów które są odpowiedzialne za prawidłowe funkcjownowanie frameworku, do elementu body dołączono elementy ktore zapewniają miejsce dla wizualnej części, zdefiniowany rozmiar pozwala na precyzyjne wkompnowanie.



Rysunek 4.7: Drzewo DOM

4.4. Bezpieczeństwo danych i aplikacji

Tworząc aplikację która będzie używana przez duża grupę obiorców, oprócz osób z pełnymi uprawnieniami do dowolnej akcji również osoby które nie powinny mieć możliwości zmiany danych należy

przewidzieć i przygotować odpowiedni zbiór ról i uprawnień. Dostęp do dostarczanych funkcjonalności należy zapewnić poprzez zbiór ról i uprawnień.

Do prawidłowego działania przewidziano kilka ról których obecność ma na celu zabezpieczenie systemu przed nieautoryzowanym dostęp, zmianą jego kodu źródłowego lub edycją danych do których użytkownik nie ma dostępu. Poniżej przedstawiono spis osób, ról które istnieją na końcowym etapie, właściwym działaniu na produkcyjnym środowisku.

– Administrator

  Osoba która posiada dostęp do serwera na krórym działa aplikacja korzystająca z omawianego frameworku. Ponieważ tworzony program jest jedynie narzędziem służącym do pracy z interaktywnymi mapami, a jednym z założeń jest jak największa adaptabliność i mozliwość działania na różnych środowiskach wymagana jest osoba która doda do działającej strony aplikację.

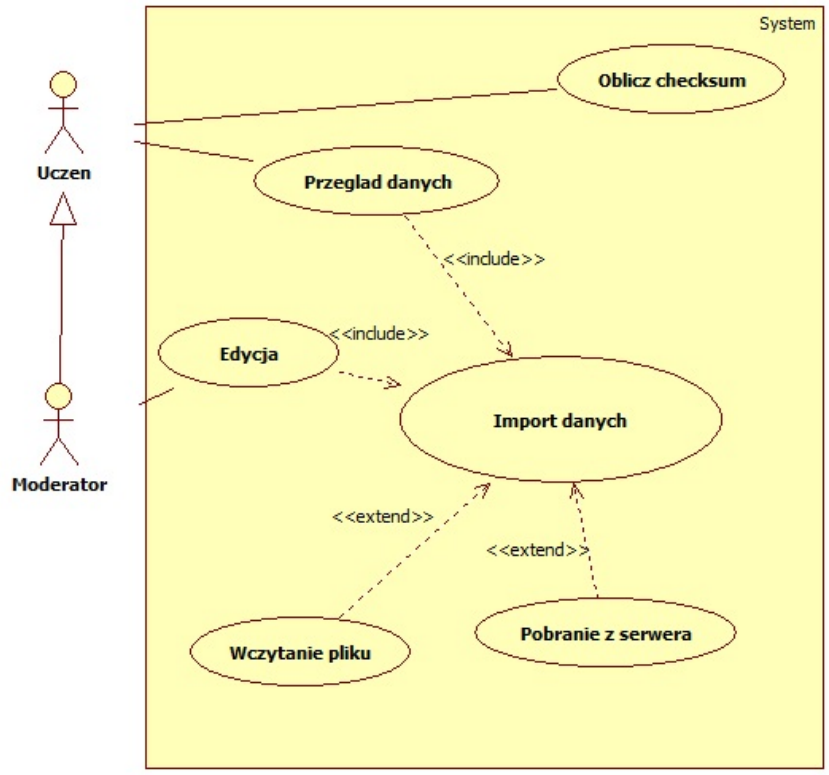
– Moderator

  Pzygotowanie i edycja map powinna być wykonana przez uprawnione osoby. Może to być na przykład osoba posiadającą dużą wiedzę na dany temat, posiada ona możliwość edycji mapy.

– Uczeń

  Użytkownik z najmniejszymy uprawnieniami, jedynie do przeglądania mapy.

Rysunek 4.8 przedstawia diagram Case Use,jeden z wielu w notacji UML. Przedstawia on podstawowe operacje dostępne dla poszcególnych użytkowników. Nie zamieszczono na nim administratora, jest on niezbędny do uruchomienia aplikacji, jednak nie należy on do docelowego systemu, nie spełnia on w nim żadnej roli.



Rysunek 4.8: Case use.

Niezwykle ważne jest aby pamiętać o strukturze aplikacji i ograniczeniach jakie niosą za sobą wykorzystane technologie. Każdy użytkownik uczeń według nomenklatury opisanej powyżej jest w posiadaniu pliku którego edycja powinna być zachowana jedynie dla moderatora. Niestety nie jest możliwe zapewnienie nieedytowalności pliku, z tego powodu nawet bez dostępu do uprawnień edycyjnych dostarczanych przez framework każdy może edytować plik w edytorze tektsowym, zmienić granice obszarów czy chociażby przedziały czasowe. Aby użytkownicy mieli pewność że plik z któego korzystają i mapa stworzona na jego podstawie jest wiarygodna i posiada poprawne dane proponowanym rozwiązaniem jest wykorzystanie jednego z algorytmu tworzących skrót z dowolnego ciągu danych, jednym z najczęściej wykorzystanym do takich celów jest MD5. Otrzymany wynik można zamieścić w widoczny miejscu obok pliku z danymi w miejscu gdzie jedynie moderator ma dostęp. Dzięki takiej operacji użytkownik będzie mógł porównać wygenerowany skrót pliku którego chce użyć z wartością na stronie, identyczne wartości potwierdzą autentyczność danych.

4.4.1. Funkcje haszujące

W pewnych sytuacjach nie potrzebujemy lub nie chcemy przechowywać oryginalnego pliku lub ciągu znaków. Częstym wykorzystaniem jest utworzenie sktótu hasła w module logowania do różnych aplikacji. Czynność ta utrudnia poznanie rzeczywistych wartości w momencie dostępu do nich przez nieuprawnione osoby. Jedną z cech dobrej funkcji haszującej jest spełnienie wymagań dla funkcji jednokierunkowej. Wymagane jest aby funkcja była łatwa do obliczenia ale trudna, lub wręcz niemożliwa

do odwrócenia. Do poznania wartości dla której został wygeneorany skrót można wykorzystać m.in. metodę słownikową, oznacza to wykonanie operacji haszującej dla dużego zbioru danych i odnalezienie szukanego sktótu w otrzymanych wynikach.

Innym częstym wykorzystaniem, zaproponowanym w poprzedniej sekcji, jest weryfikacja integralności pliku lub wiadomości. Umieszczenie w jedym miejscu pliku i jego skrótu umożliwia sprawdzenie czy plik który posiadamy jest identyczny z tym znajdującym się na stronie, w przypadku braku zgodności możliwe jest ściągnięci poprawnej wersji, w pozostałych sytuacjach posiadamy pewność autentyczności danych bez potrzeby każdorazowego ściągania pliku z serwera, oszczędzamy przez to czas poświęcony na ściągnięcie pliku, który może zajmować wiele megabajtów.

Istnieje wiele funkcji haszujących, wybór konkretnej z nich zależy od celu użycia najpopularniejszymy są MD5 i SHA-1.

– MD5 Opracowany w 1991 roku przez Rona Rivesta, z dowolnego ciągu danych generuje 128-bitowy sktót,pozwala to na utworzenie 32 znakowego ciągu w systemie szesnastkwoym. Ważną cechą jest zmiana skróty w przypadku najmniejszej edycji wejściowych danych. Tabela 4.1,stowrzona przy pomocy strony `http://www.sha1-online.com/`, prezentuje przykład zmiany jednej litery, wygenerowany skrót dla tej wartości jest zupełnie inny. Z uwagi na szybkość działania jest powszechnie używana, jednym z zastosowań jest ukrywanie haseł w bazie danych.

– SHA-1

  Przedstawiciel funckji z rodziny SHA, opublikowany w 1995 roku. Wprzeciwieństiw do MD5 do zapisu skrótu korzysta z 160-bitów, oznacza do 40 znaków w systemie heksadecymalnym.

– SHA-2 Określenie czterech wariantów zastępujących SHA-1. Wersją zapewniającą najwyższą obecnie niezawodność i bezpieczeństwo jest SHA-512, do zapisu wykorzystuje 512 bitów co pozwala na otrzymanie 128 znakowego skrótu.

Algytm	Wejściowy ciąg	Wygenerowany skrót
MD5	AGH Kraków	c9c16a90d4938da799b5aa7d6f37edce
MD5	AGH Krakow	fc6a83206673002588490eb05b89313f
SHA-1	AGH Kraków	6205880f638b3e584578125b14231c30c3ab9fea
SHA-1	AGH Krakow	a536aa59ed829801cd0f099c4ae3ba34ca57ca9f
SHA-512	AGH Kraków	49d9e2deb053a58dcb0a778758e5(...)61499145e36353
SHA-512	AGH Krakow	6b65c56ca2e5567a5f18dd0d7465(...)f46166c29d7105

Tablica 4.1: Rezulaty funkcji haszujących

http://eprint.iacr.org/2008/469.pdf

4.5. Optymalizacja rozwiązania

4.5.1. Funkcje haszujące

5. Opis rozwiązania

W poniższym rozdziale omówione zostaną kroki pracy.

5.1. Transmisja danych

Pierwszym aspektem który należy rozwiązać jest sposób przesyłania danych. Problem ten jest szczególnie istotny w omawianej pracy z uwagi na możliwość przesyłania dużej ilości informacji o granicach lub innych liniach przezentowanych na mapie. Do opisu kwadratowego obszaru wymagane jest przesłanie informacji o minimum 4 punktach. Jeżeli będziemy chcieli przekazać dokłądniejszy zarys obszaru, zaprezentować granicę państwa lub linię frotnu wojennego linia prosta w większości przypadków będzie zbyt ogólnym przybliżeniem, nie oddającym prawdziwej sytuacji.

Projekt zakłada korzystanie z lokalnej pamięci komputera użytkownika podczas pracy, jednak dostarczenie informacji, danych wejściowych nie powinno być ograniczone do wczytania ich z pliku tekstowego, należy pamiętać o możliwości przesłania ich z serwera aplikacji do użytkownika. W sytuacji takiej, gdzie w jednym momencie przesłane zostaja wszystkie informacje na któch będzie zachodzia praca, ilość informacji może być znaczna.

Z raportu Akamai wynika że śrenia przepływność łączy internetowych dla użytkowników korzystających z puli adresów IP przeznaczonych dla Polski w I kwartale 2012 r. wynosiła 5Mb/s `http://www.rp.pl/artykul/924483.html` (dostęp 13.04.2014). Jest to bardzo dobry wynik który plasuje Polskę w czołówce rankingu. Pomimo tego nie można pominąć faktu optymalizacji zapytać i danych przesyłanych, wymieniane dane pomiędzy użytkownikiem a serwerem powinny być jak najmniejsze. Duża popularność urządzeń mobilnych w których dostęp do internetu jest zapewniany często poprzez sieć bezprzewodową a dostęp do interentu nie jest jeszcze tak dogodny jak jest to w przypadku użytkowników stacjonarnych wymusza optymalizację.

Kolejnym powodem dla którego odpowiedzi serwera powinny być jak najlżesze jest koszt pracy samego serwera. Jest to szczególnie widoczne w dużych aplikacjach mających wiele urzytkowników, czas jaki jest przeznaczany dla pojedyńczego użytkownika jest mnożony przez ich ilość. Z tego powodu zawsze podczas zwiększania ilości użtkowników korzystających z aplikacji następuje czas w którym należy zacząć korzystać z dodatkowego serwera. Celem programisty tworzącego kod który będzie wykorzystywał zasoby serwera(zarówno czas jak i pamięć) jest dbanie aby moment w którym niezbędne będzie korzystanie z większej ilośći maszym nastąpił przy jak największej ilości użytkowników.

xml - 729 557 json - 695 400

5.1.1. Format zapisu

Istnieje wiele formatów używanyc w zależności od konkretnych potrzeb. Można dokonać ogólnego podziału na dwa rodzaje. Pierwszym są pliki zapisane rastrowo, jest to zapis punktów o określonym kolorze i położeniu na płaszczyźnie. Efektem takiego podejścia jest pogorszenie obrazu w momencie dużego powiększenia. Przykładem są formaty takie jak ADRG,ASC czy RGB. Porównanie litery a i jej 7-krotnego powiększenia w tym zapise zaprezentowano na rysunku 5.1

Rysunek 5.1: Zapis rastrowy

Innym podejściem jest zapis wektorowy. Jest to zapis oparty na formułach matematycznych m.in. proste i łuki. Dzięki takiemu podejściu obraz w dowolnej skali na identyczną jakość, widać to na rysunku 5.2. Przykładowymi formatami są GML(Geography Markup Language) otwarty standart XML dla wymiany danych GIS,Shapefile czy tradycyjny kartezjański układ współrzędnych.

Rysunek 5.2: Zapis wektorowy

Do celów projektu zdecydowano się skorzystać z zapisu wektorowego. Oprócz bez problemowej skalowalności, pozwala on również na szeroką i łatwą edycję. W prosty sposób można dodać informacje nie tylko o położeniu, ale inne takie jak słowny opis miejsca lub wydarzenia które miało miejsce w pobliskim rejonie.

Format KML jest używany do reprezentacji danych graficznych w aplikacjach firmy Google, takich jak Google Maps czy Google Earth. Oparty jest na standardzie XML, wrażliwy na wielkość liter, wymusza ścisłą poprawność danych z formatem. Oznacza to że każdy tag, element ma ścisle określone miejsce i nie może pojawić się nigdzie indziej.

Aby tworzona aplikacja miała możliwość współpracy ze stworzonymi wczesniej zbiorami danych, mapami mimo faktu iż informacje te nie będą w pełni wykorzystywały możliwości frameworku, stwo-

rzony został parser plików kml. Dzięki temu możliwy jest import danych, uzupełnienie ich o dodatkowe informacje i zapis do własnego formatu.

5.2. Parser plików

Założenia projektu zapewniają dwa źródła dostarczania informacji niezbędnych do poprawnej pracy frameworka. Pierwszym jest przesyłanie infromacji z serwera do urządzenia klienta przy wykorzystaniu formatu JSON, wiadomość taka jest zbudowana w odpowiednią strukturę aby bezpośrednio dokonać konwersji do jegnego ciągu znakowego. W ten sposób możliwe będzie zapewnienie w bardzo szybkim czasie danych pozwalających na pełne wykorzystanie wszystkich możliwości aplikacji.

Dodatkową opcją dostarczania informacji będzie wczytywanie pliku z informacjami opisującymi mapę. Do tego celu stworzono parser plików który pozwala na dokonanie analizy wczytanego pliku i konwersję danych do formatu używanego do przechowywania danych po stronie klienta.

5.3. Oś czasu

Tematem pracy oprócz wykorzystania i prezentacji danych na bazie informacji geograficznych jest ich połączenie z osią czasu, która pozwoliłaby na zmianę prezentowanych danych ze względu na interesujący nas okres czasu.

Ponieważ własna implementacja osi czasu która nieograniczałaby użytkownika, pozwala mu w pełni korzystać z możliwości pełnej aplikacji, która tylko w części składa się z możliwości manipulacji czasem postanowiono wykorzystać istniejące rozwiązanie.

Przeprowadzając badania rozwiązań głównymi aspektami na które zwrócono uwagę było:

– Typ licencji Niezwykle ważne jest aby końcowy projekt był całkowicie otwarty dla dalszego rozwoju, pozwalał użytkownikom na adaptację rozwiązań dla swoich potrzeb. Z uwagi na to licencja musi pozwalać na nieograniczoną modyfikację kodu źródłowego.

– Wykorzystywane technologie Pamiętając że efektem końcowym powinien być framework odpowiadający szerokiemu gronowi odbiorców, będący jednocześnie darmowym ważne aby języki programowania wykorzystane przy jego tworzeniu też taki były. Oznacza to nie korzystanie z płatnych, wymagających licencji oprogramowań. Używane standarty powinny być rozpropagowane i używane przez szerokie grono obiorców.

Po długich i wnikliwych poszukiwaniach zdecydowano skorzystać z widget-u o nazwie Timeline który został stworzony przej projekt SIMILE działający na uczelni MIT. Pozwala on na bardzo dużą konfigurowalność dzięki czemu jego modyfikacja, nawet bez ingerencji w kod źródłowy jest bardzo prosta. Projekt jest w stanie stworzyć kilka pasm któe będą określały interwały czasu. Pozwala to aby wynik końcowy był przejżysty bez względu czy prezentujemy wydarzenia które miały miejsce w okresie kilku minut czy setek lat.

Problemem który pojawił się było połączenie wybranego sposobu prezentacji osi czasu z mapą i elementami które powinny zmieniać się wraz z upływem czasu.

Rysunek 5.3 prezentuje możliwości tego projektu. Pasmo odpowiadają zakresom czasu, nie muszą on jednak być jednakowe w każdym miejscu. W zaprezentowanym przykładie dzień 23 listpoada uznany został za warty dokładniejszemu przyjżeniu,łatwo możemy z dokładnością co do godziny zmieniać zakres czasu.Natomiast kolejny miesiąc może być o wiele mniej ciekawy, a obszar przeznaczony dla niego być mniejszy niż ten posiadany przez wymieniony powyżej dzień.

Dla porównania 5.4 prezentuje o wiele prostszą konfigurację w której dolne pasmo podzielone jest przez miesiące, natomiast góre przez tygodnie.

Rysunek 5.3: Timeline

Rysunek 5.4: Timeline 2

5.4. Interferjs

Aby tworzone rozwiązanie mogło być używane przez szerokie grono odbiorców niezbędnym elementem jest stworznie interfejsu który pozwlałby na edycję danych wczytanych z pliku, lecz także pozwalał na stworzenie nowej mapy. W tym celu stworzono interfejs graficzny do edycji danych.

6. Implementacja

Działanie programu można podzielić na dwa główne etapy. Pierwszym jest dostarczenie danych zgodnych z formatem apliakacji, drugim jest reprezentacja danych.

6.1. Minimalna konfiguracja

Do rozpoczęcia korzystania z gotowej aplikacji należy zapewnić obecność konfiguracji niezbędnej do jej konfiguracji na docelowej stronie. Przykład minimalnej strony która spełnia wymagane został pokazany w listinu 6.3.

Każdy dokument stworzony w HTML sklada się z 3 części:

-lini zawierającej informację o używanej wersji HTML, w poniższym przykładzie dotyczy ona HTML5

-deklarację sekcji header, zawiera odwołania do zewnętrzych plików i delkaracje funcji, styli

-część właściwa strony która zawiera infromacje właściwe, widoczne dla użytkownika storny.

Linie 5-7 mają za zadanie wczytanie zewnętrzych plików zawierające delkaracje metod w JS. 8 wczytuje plik zawierający główną klasę frameworku, poniżej znajduje się wczytanie niezbędnych styli CSS dla działania osi czasu. 13-23 jest to opis metody która inicjuje działanie mapy, jest ona wykonywana po załadowaniu statycznych fragmentów strony.

```
1
2  <!DOCTYPE html>
3  <html>
4   <head>
5     <script type="text/javascript" src="https://maps.googleapis.com/maps/api/js?
          sensor=false&key={KEY_ID}">
6     <script type="text/javascript" src="./js/specialMap/lib/jquery-1.6.2.min.js"></
          script>
7     <script type="text/javascript" src="./js/specialMap/lib/mxn/mxn.js?(googlev3)"></
          script>
8     <script type="text/javascript" src="./js/specialMap/lib/timeline-1.2.js"></script
          >
9     <script type="text/javascript" src="./js/specialMap/src/timemap.js"></script>
10    <script type="text/javascript" src="./js/specialMap/src/utils.js"></script>
11
12    <link type="text/css" rel="stylesheet" href="./css/examples.css"/>
13
```

```
14    <script>
15    var sm;
16    $(function() {
17       tm = SpecialMap.init({
18          mapDivId: "mapId",
19          timeDivId: "timeId",
20          dataType: "sessionStorage",
21          bandIntervals: [
22            Timeline.DateTime.YEAR
23        ],
24       });
25    });
26  </head>
27  <body>
28       <div id="specialMap">
29          <div id="timeId"></div>
30          <div id="mapId"></div>
31       </div>
32  </body>
33 </html>
```

Listing 6.1: Minimalna konfiguracja.

/

6.2. Odczyt danych

Tak jak zostało omówione w sekcji **??** do przechowywania danych wybrany został format KML, jego duża popularność sprawia że stworzona aplikacja posiada dużą bazę danych z których może korzystać.

W celu odczytu i konwersji danych z pliku zapisu ich w pamięci sesyjnech przeglądarki omówionej w rozdziale 4.2.2 stworzona została klasa KmlParser. Posiada ona jednoargumentowy konstruktor przyjmujący ciąg znakowy.

Ze względów bezpieczeństwa JavaScript nie zezwala na bezpośredni dostęp do plików lokalnych uzytkownika. Jest to pozytywne zachowanie zapewniające że żaden kod nie jest w stanie bezwiednie zmieniać zawartość dysku.

Sytuacja taka wymusza korzystanie z event-ów które będą aktywowane w momecie kiedy plik zostanie wybrany. W momencie zakończenia procesu odczytu zawartość pliku przekazywana jest do parsera który zapisuje dane w pamięci przeglądarki.

```
1
2 var exml;
3 function readFile(file,start,stop){
4  var reader = new FileReader();
5    reader.onloadend = function(evt) {
6      if (evt.target.readyState == FileReader.DONE) {
7    exml = new KmlParser( evt.target.result);
8    exml.parse()
```

```
9       }
10    };
11
12   var blob = file.slice(start, stop);
13    reader.readAsBinaryString(blob);
14 }
```

Listing 6.2: Minimalna konfiguracja.

/

```
1
2  KmlParser.value = function(e) {
3   a = GXml.value(e);
4   a = a.replace(/^\s*/, "");
5   a = a.replace(/\s*$/, "");
6   return a;
7  }
8
9  KmlParser.prototype.createMarker = function(point,begin, end, name, desc, style,
       iconUrl) {
10   var icon = {};
11   var markeroptions = this.opts.markeroptions || {};
12   var icontype = this.opts.icontype || "style";
13   if (icontype == "style") {
14    if (!!this.styles[style]) {
15     icon = this.styles[style];
16    }
17   }
18   if (!markeroptions.icon) {
19    markeroptions.icon = icon;
20   }
21   var m = new google.maps.Marker(point, markeroptions);
22
23   if (this.opts.elabelclass) {
24    var l = new ELabel(point, name, this.opts.elabelclass,
25       this.opts.elabeloffset, this.elabelopacity, true);
26   }
27   this.gmarkers.push(m);
28   item = {};
29   item.start = begin;
30   item.end = begin;
31   item.title = name;
32   item.desc = desc;
33   item.iconUrl = iconUrl;
34   p = {};
35   p.lat = point.lat();
36   p.lon = point.lng();
37   item.point = p
38   return item;
39 }
```

```
40
41 KmlParser.prototype.createPolyline = function(points, color, width, opacity,
42    name, desc) {
43  var polylineoptions = this.opts.polylineoptions || {};
44  var p = new GPolyline(points, color, width, opacity, polylineoptions);
45  this.gpolylines.push(p);
46
47 KmlParser.prototype.createPolygon = function(points, id, pId, begin, end, color,
48    width, opacity, fillcolor, fillopacity, name, desc) {
49
50  pointsArray = []
51  item = {};
52  for ( var i = 0; i < points.length; i++) {
53    point = points[i];
54    p = {};
55    p.lat = point.lat();
56    p.lon = point.lng();
57    if (p.lon == null || p.lat == null || isNaN(p.lat) || isNaN(p.lon)) {
58      console.log("Null points");
59    } else {
60      pointsArray.push(p)
61    }
62  }
63  item.start = begin;
64  item.end = end;
65  item.polygon = pointsArray;
66  item.title = name;
67  item.mId = id;
68  item.pId = pId;
69  item.color = color;
70  item.width = width;
71  item.opacity = opacity;
72  item.fillcolor = fillcolor;
73  item.name = name;
74  item.desc = desc;
75
76  return item;
77 }
78
79 KmlParser.prototype.processing = function(doc) {
80  var that = this;
81  var xmlDoc = GXml.parse(doc)
82
83  var styles = xmlDoc.documentElement.getElementsByTagName("Style");
84  for ( var i = 0; i < styles.length; i++) {
85     //wczytanie styli
86  }
87
88  var placemarks = xmlDoc.documentElement.getElementsByTagName("Placemark");
```

```
items = [];
itemsMarkers = [];
for ( var i = 0; i < placemarks.length; i++) {
  var id = placemarks[i].getAttribute('id');
  var timeSpan = placemarks[i].getElementsByTagName("TimeSpan")[0];
  var begin = KmlParser.value(timeSpan.getElementsByTagName('begin')[0])
  var end = KmlParser.value(timeSpan.getElementsByTagName('end')[0])
  var name = KmlParser.value(placemarks[i].getElementsByTagName("name")[0]);
  var desc = KmlParser.value(placemarks[i]
     .getElementsByTagName("description")[0]);
  if (desc == "") {
    var desc = KmlParser
       .value(placemarks[i].getElementsByTagName("text")[0]);
    desc = desc.replace(/\$\[name\]/, name);
    desc = desc.replace(/\$\[geDirections\]/, "");
  }
  if (desc.match(/^http:\/\//i)) {
    desc = '<a href="' + desc + '">' + desc + '</a>';
  }
  if (desc.match(/^https:\/\//i)) {
    desc = '<a href="' + desc + '">' + desc + '</a>';
  }
  var style = KmlParser.value(placemarks[i]
     .getElementsByTagName("styleUrl")[0]);
  var coords = GXml.value(placemarks[i]
     .getElementsByTagName("coordinates")[0]);

  var path = coords.split(" ");

  multiGeometry = getChildrenTag(placemarks[i], 'MultiGeometry')[0];
  if(multiGeometry){
    allPolygons = multiGeometry.getElementsByTagName("Polygon")
  }else{
    allPolygons = {};
  }

  onePoint = getChildrenTag(placemarks[i], 'Point')[0];
  if (!!that.styles[style]) {
      var width = that.styles[style].width;
      var color = that.styles[style].color;
      var opacity = that.styles[style].opacity;
      var fillopacity = that.styles[style].fillopacity;
      var fillcolor = that.styles[style].fillcolor;
      var iconUrl = that.styles[style].iconUrl;
    } else {
      var width = 5;
      var color = "#0000ff";
      var opacity = 0.45;
      var fillopacity = 0.25;
```

```
138         var fillcolor = "#0055ff";
139       }
140
141     if(onePoint){
142       coords = GXml.value(onePoint.getElementsByTagName("coordinates")[0]);
143       var bits = coords.split(",");
144       var point = new google.maps.LatLng(parseFloat(bits[1]),
145           parseFloat(bits[0]));
146       that.bounds.extend(point);
147         itemsMarkers.push(that.createMarker(point ,begin,end, name, desc, style,
              iconUrl));
148     }
149
150     for ( var j = 0; j < allPolygons.length; j++) {
151       polygon = allPolygons[j];
152       coords = GXml.value(polygon.getElementsByTagName("coordinates")[0]);
153       path = coords.split(" ");
154
155       if (path.length > 1) {
156         var points = [];
157         var pId = polygon.getAttribute('id');
158         for ( var p = 0; p < path.length; p++) {
159           var bits = path[p].split(",");
160           if (parseFloat(bits[1]) == null
161               || parseFloat(bits[0]) == null
162               || isNaN(parseFloat(bits[1]))
163               || isNaN(parseFloat(bits[0]))) {
164           } else {
165             var point = new google.maps.LatLng(parseFloat(bits[1]),
166                 parseFloat(bits[0]));
167             points.push(point);
168           }
169         }
170         this.pointsCount = this.pointsCount + points.length;
171         var linestring = placemarks[i]
172             .getElementsByTagName("LineString");
173         if (linestring.length) {
174             that.createPolyline(points, color, width, opacity,
175                 name, desc);
176         }
177
178         var polygons = placemarks[i].getElementsByTagName("Polygon");
179         items.push(that.createPolygon(points, id, pId, begin, end,
180             color, width, opacity, fillcolor, fillopacity,
181             name, desc));
182       } else {
183         var bits = path[0].split(",");
184         var point = new google.maps.LatLng(parseFloat(bits[1]),
185             parseFloat(bits[0]));
```

```
186       that.bounds.extend(point);
187         that.createMarker(point, name, desc, style);
188       }
189     }
190   }
191   sessionStorage.polyline = JSON.stringify(items);
192   sessionStorage.marker = JSON.stringify(itemsMarkers);
193 }
```

Listing 6.3: Minimalna konfiguracja.

/

6.2.1. Wydajność

W celach sprawdzenia wydajności zaimplementowanego parsera przeprowadzono testy porównawcze. W pierwszej próbie wykorzystano dwa pliki zawierające dane w formacie który pozwolił na jego analizę, pierwszy "plik1.kmlśkładał się z jednego obszaru i jednego stylu określającego preferencje graficzne, jego dokładana zawartość została zawarta w załączniku A.1. Drugi plik "plik2.kmlźawierał informacje o granicy wszystkich stanów USA, na potrzeby badań on również zawierał informacje o jednym stylu. Wykonano proces wczytania ich zawartości, mierząc za każdym razem czas potrzebny na zakończenie procesu. Wyniki zamieszczono w zbiorczej tabeli 6.2.

Dokładne informacje dotyczące zawartości plików zamieszczono w tabeli 6.1

Nazwa pliku	Ilość punktów	Ilość poligonów
plik1.kml	13	1
plik2.kml	13697	133

Tablica 6.1: Pliki testowe

Test	13 punktów	12697 punktów
I	3	343
II	13	449

Tablica 6.2: Czas dostępu [ms]

6.3. Reprezentacja danych

Posiadając już informacje należało wybrać najlepszy sposób umożliwiający ich przedstawienie. Wykorzystanie w tym celu możliwości dostarczanych przez HTML5 wydawało się bardzo zachęcające. Przykład przedstawiony na rysunku 4.5 pokazuje w jak prosty sposób można zaznaczyć okrąg na gotowym obrazie. Niestety ze względu na brak kompatybilności tego rozwiązania z wybranym sposobem dostarczania informacji geograficznych sposób tworzenia kształtów został zmieniony inny.

6.3.1. Rysowanie kształtów

Aplikacja umożliwa tworzenie różnych kształtów i pojedyńczych elementów które można umieścić na mapie. Wykorzystywane są obiekty z pakiety google.maps, zawiera takie elementy jak:

- Marker Pozwala on na umieszczenie pojedyńczego punktu na mapie. Możliwe jest dokładne sprecyzowanie jego wyglądu dzięki czemu możliwe jest aby prezentował on określą ikonę lub symbol.
- Okrąg Pierwszy kształt, tworzy idealy okrąg.
- Poligon Jedna z bardziej przydatnych opcji. Dzięki niemu możemy stworzyć każdy kształt. To ten element został wykorzystany do stwrozenia granicy wyspy zaprzezentowany w załączniku A.3.
- Linia Elemnt pozwalający na tworzenie lini, której wygląd możemy swobodnie określać.
- Prostokąt Ostatni element tworzy prostokąty o kątach dokłanie 90 stopni. Kształt ten można próbować stworzyć przy pomocy poligonu, nie będzie on jednak tak dokładny jak przy pomocy tej opcji.

6.3.2. Dodawanie filtrów

Rozwinięciem podstawowej możliwości tworzenia prostych kształtów geometrycznych jest opcja tworzenia filtrów które mogą wpływać na prezentację danych.

Przykładowym użyciem jest możliwość stworzenia animacji płynnego przejścia z jednego kształtu w inny. Proces ten jest wykonany w następujących krokach:

- Przekazanie do aplikacji kształtu zawierającego informacje o stanie początkowym i końcowym.
- Dla każdej zmiany daty obliczenie aktualnego procentowego zakończenia procesu transformacji, gdzie pomiędzy startem a końcem okresu zmiany kształtu obecnie się znajdujemy.
- Oblicznie położenia nowego punktu, tak aby leżał pomiędzy dwoma w stosunku oblicoznym w poprzednim kroku.

Istnieje możliwość tworzenia nowych filtrów dla potrzeb konkretnych map lub danych. Nie stanowi dużym problemem imlementacja takiego rozwiąznaia które w pewnym momencie wyświetliłoby krótki film wideo dostarczający dodatkowych informacji o obserwowanym wydarzeniu.

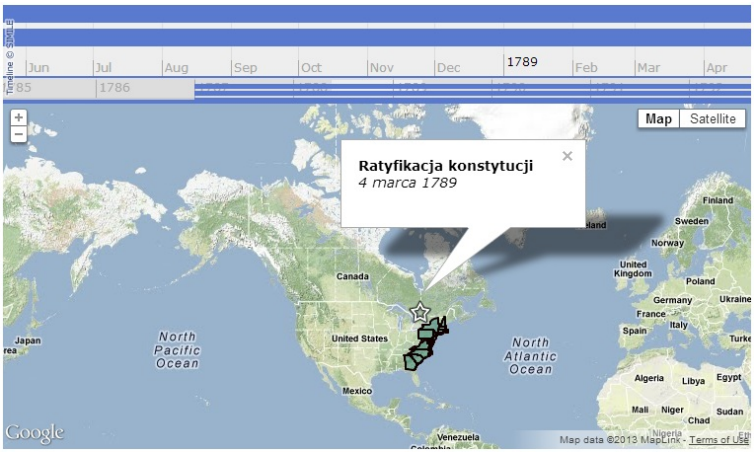
Google DrawingManager

7. Przykład użycia

10 Poniższy rozdział zawiera przykład użycia stworzonej aplikacji. Prezentuje jedynie niewielką część możliwości.

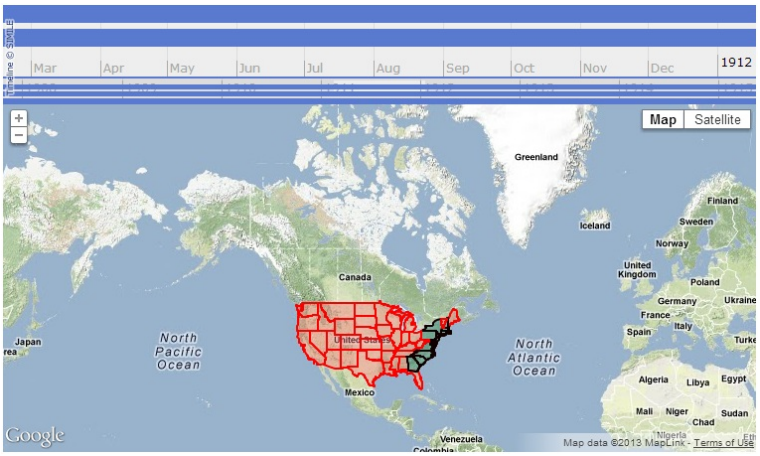
7.1. Historia USA

W celu zobrazowania powstawania USA poniżej zaprezentowano 3 stany jego rozwoju. Na rysunku 7.1 znajduje się obszar państwa w momencie ratyfikacji konstytucji.Kolorem czarnym zaznaczone są 13 pierwszych stanów które uznawane są za założycielskie. Pokazano również możliwość dodawania opisu do wydażeń dodanych do sceny, umożliwia to dodanie dodatkowych informacji które mogą wspomóc proces dydatktyczny.



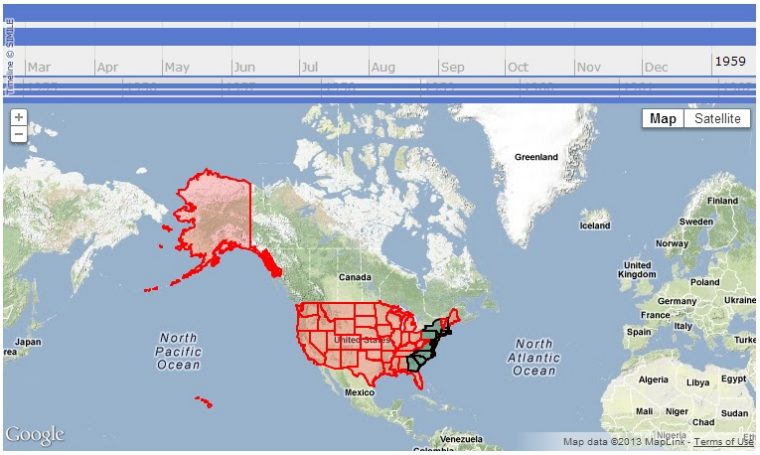
Rysunek 7.1: rok.

Kolejny rysunek 7.2 opisuje stan z 1912 roku. Dzięki możliwości stworzenia dwuch lub pasm które będą odpowiadały za zmianę daty, przesunięcie się o ponad 100 lat nie sprawia większego trudu i możliwe w krótkim czasie.



Rysunek 7.2: rok.

Ostani element tej serii ukazuje aktualny stan granic. Widać że utrzymuje się on od 1959 roku.



Rysunek 7.3: rok.

8. Podsumowanie

Celem mojej pracy magisterskiej było stworzenie lekkiej i szybkiem aplikacji pozwalającej na łatwą prezentację zmian zachodzących na danym obszarze terytorialnym. Chciałem połączyć tradycyjne podejście do map, statycznej prezentacji treści z funkcjonalnością zmiany okresu czasu do któego informacje te się odnoszą tak aby w łatwy sposób możliwa byłaby prezentacja zmian zachodzących chociażby na polu walki, granic państw czy nawet zmian pogodowych.

Znaczna część pracy jest poświęcona analizie obecnie istniejącym technologiom pozwalającym na pracę z obrazami, przeglądzie nowoczesnych technologii. Zwrócono szczególną uwagę na możliwości dostarczane przez nową wersję języka HTML, zawiera on wiele opcji które wcześniej były możliwe w wyspecjalizowanych językach(np. wprowaszenie tagów określających plik audio i wideo). Ważne było aby nie próbować rozwiązywać problemów które juz wcześniej zostały rozwiązane. Mam nadzieję że analiza zamieszczona w pracy może stanowić wartościowy przegląd i zbiór przydatnych infromacji, nie tylko dla osób zainteresowanych rozojem aplikacji związanych z infromacjami geograficznymi ale także dla każdego chcącego zapoznać się z najnowszymi trendami w tworzeniu aplikacji mobilnych(HTML5,LESS).

Głównymi problemami które należało rozwiązać były związane z obsługą dużej ilości danych, przechowywaniem i prezenetacją. W początkowej fazie napotkano równiez problem wydajności, wyświetlenie kilkudziesięciu tysięcy punktów zajmowała kilka sekund. Jest to nieakceptowalna sytuacja, użytkownik który musiałby długo czekać na aktualizację podczas najmniejszej zmiany oglądanego przedziału czasu szybko zniehęciłby się, a sama aplikacja była by bezużyteczna. Wykorzystano najnowocześniejsze rozwiązania które pozwalają na otrzymanie bardzo dobrych wyników lecz stawiają pewne wymagania, głównie do wersji przeglądarki na któej działa aplikacja. Zdecydowano się na stosunkowo nową możliwość stworzenia bazy danych po stronie klienta, w jego przegladarce. Tradycyjne rozwiązania opierają się na utrzymywaniu danych po stronie serwera, po stronie użytkownika znajdowała się małą ilość informacji, głównie służąca do identyfikacji sesji zapisane w ciasteczkach(cookies).

Stworzona aplikacja spełnia minimalne warunki aby można było ją uznać za przydatną podczas procesu nauki. Posiada intuicyjny interejs pozwalający na łatwą obsługę, dostarcza cennych informacji a szeroki zakres prezentacji danych, możliwość tworzenia płynnych przejśc kształtów, kolorów sprawia że jest interesująca i przyciąga uwagę użytkownika. Wykorzystanie jej m.in. na lekcji histori pozwoli zmienić statyczne reprezentacje ruchów wojennych w ciekawe i interaktywne, taka forma może zwiększyć przyswajalność widzy i zachęcić do czestych powtórek poza salą lekcyjną.

A. Dodatek A

A.1. Oryginalny plik kml

```
<?xml version="1.0" encoding="UTF-8"?>
<kml>
 <Document>
  <name><![CDATA[US States]]></name>
  <open>1</open>
  <Style id="Style_5">
   <LabelStyle>
    <color>9900ffff</color>
    <scale>1</scale>
   </LabelStyle>
   <LineStyle>
    <color>990000ff</color>
    <width>2</width>
   </LineStyle>
   <PolyStyle>
    <color>997f7fff</color>
    <fill>1</fill>
    <outline>1</outline>
   </PolyStyle>
  </Style>
  <Placemark id="pm251">
   <name><![CDATA[Hawaii (1959)]]></name>
   <Snippet maxLines="0">empty</Snippet>
   <description></description>
   <TimeSpan>
    <begin>1989</begin>
   </TimeSpan>
   <styleUrl>#Style_5</styleUrl>
   <MultiGeometry>
    <MultiGeometry>
     <Polygon id="g867">
      <altitudeMode>clampToGround</altitudeMode>
      <outerBoundaryIs>
       <LinearRing>
        <coordinates>
-159.335174733889,21.9483433404175
```

```
-159.327130348878,22.0446395507162
-159.295025589769,22.1248124949548
-159.343195828355,22.1970166285359
-159.391366885913,22.2291198667724
-159.576012589057,22.2131796383001
-159.712505933171,22.1490592515515
-159.800814224332,22.0366665967853
-159.736592652746,21.9644203111023
-159.640246973766,21.9483657695954
-159.576021285803,21.8841361312636
-159.439545188912,21.8680716835921
-159.335174733889,21.9483433404175
            </coordinates>
          </LinearRing>
        </outerBoundaryIs>
      </Polygon>
    </MultiGeometry>
  </MultiGeometry>
 </Placemark>
</Document>
</kml>
```

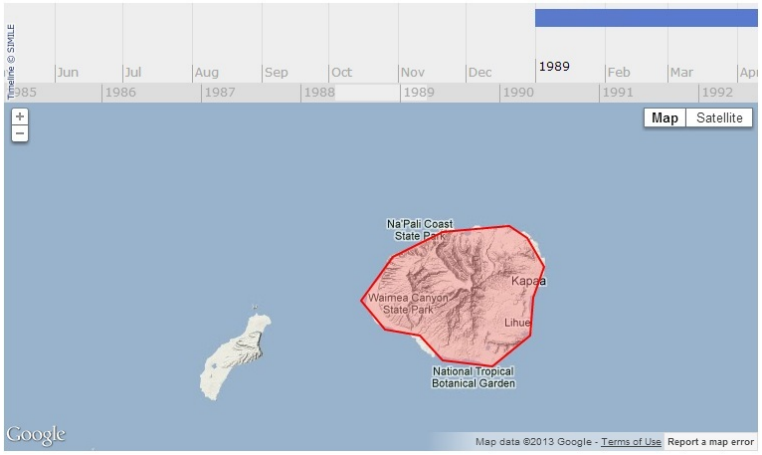
Listing A.1: caption

A.2. Zapis w pamięci sesyjnej

```
[{"start":"1989","end":"2010","polygon":[{"lat":21.9483433404175,"lon"
   :-159.335174733889},{"lat":22.0446395507162,"lon":-159.32713034887797},{"lat"
   :22.1248124949548,"lon":-159.295025589769},{"lat":22.1970166285359,"lon"
   :-159.34319582835496},{"lat":22.2291198667724,"lon":-159.391366885913},{"lat"
   :22.2131796383001,"lon":-159.57601258905697},{"lat":22.1490592515515,"lon"
   :-159.71250593317097},{"lat":22.0366665967853,"lon":-159.800814224332},{"lat"
   :21.9644203111023,"lon":-159.73659265274603},{"lat":21.9483657695954,"lon"
   :-159.640246973766},{"lat":21.8841361312636,"lon":-159.576021285803},{"lat"
   :21.8680716835921,"lon":-159.439545188912},{"lat":21.9483433404175,"lon"
   :-159.335174733889}],"title":"Hawaii (1959)","mId":"pm251","pId":"g867","color":
   "#ff0000","width":2,"opacity":0.59765625,"fillcolor":"#ff7f7f","name":"Hawaii
   (1959)","desc":""}]
```

Listing A.2: caption

A.3. Wynik końcowy



Rysunek A.1: Hawaje

Spis rysunków

Spis tablic

Bibliografia

[BE07] Thiru Thangarathinam Michael Kay Alessandro Vernet Sam Ferguson Bill Evjen, Kent Sharkey. *Professional XML*. Wrox, 2007.

[BG13] Shyam Seshadri Brad Green. *AngularJS*. OReilly Media, 2013.

[Cho09] C. Choi. *A palaeolithic map from 13,660 calBP: engraved stone blocks from the Late Magdalenian in Abauntz Cave (Navarra, Spain)*. Journal of Human Evolution, 2009.

[chr]

[flaa]

[flab]

[GS10] Cameron Turner Gabriel Svennerberg. *Beginning Google Maps API 3*. Apress, 2010.

[Har03] Elliotte Rusty Harold. *Effective XML: 50 Specific Ways to Improve Your XML*. Addison-Wesley Professional, 2003.

[Irv28] Washington Irving. *A History of the Life and Voyages of Christopher Columbus*. G. & C. Carvill, 1828.

[JB] Stanisława Porzycka Justyna Biała. Zastosowanie języka svg do wizualizacji danych geoprzestrzennych.

[Lam09] L. Lamport. *LaTeX system przygotowywania dokumentów*. Wydawnictwo Ariel, Krakow, 2009.

[Mae12] Kazuaki Maeda. Performance evaluation of object serialization libraries in xml, json and binary formats. *IEEE Press*, 2012.

[MK08] Andrzej Chybicki Marcin Kulawiak, Mariusz Łuba. Web-based gis technologies dedicated for presenting semi-dynamic geospatial data. *IEEE Press*, 2008.

[mon]

[Pil10] M. Pilgrim. *Dive into HTML5*. 2010.

[PL10] Frank Salim Peter Lubbers, Brian Albers. *Pro HTML5 Programming*. Apress, 2010.

[SM08] Weichang Du Sai Ma, Minruo Li. Service composition for gis. *IEEE Press*, 2008.

[Wan11] Guanhua Wang. Improving data transmission in web applications via the translation between xml and json. *IEEE Press*, 2011.

[wor]