Google Web Toolkit (GWT),
Systemy Informacji Geograficznej (GIS),
Internetowe Systemy Informacji Geograficznej(WEBGIS)

Małgorzata MALICKA[1]
Kamil ŻYŁA [2]

GOOGLE WEB TOOLKIT JAKO PRZYKŁAD NARZĘDZIA DO TWORZENIA WIELOFUNKCYJNYCH APLIKACJI GIS WYKORZYSTUJĄCYCH TECHNOLOGIE WEBOWE

Systemy Informacji Geograficznej (GIS), to specjalistyczne systemy służące gromadzeniu, przetwarzaniu oraz prezentacji danych geograficznych. Wraz z rozwojem Internetu wyodrębnił się WEBGIS, czyli systemy GIS wykorzystujące technologie internetowe.

Problemem twórców systemów WEBGIS jest prezentacja danych użytkownikowi. Czysty HTML jest niewystarczający, a aplikacje tworzone z wykorzystaniem popularnych technologii takich jak, JavaServer Faces są trudne w implementacji oraz nieefektywne w działaniu.

W niniejszym artykule sprawdzono możliwość uproszczenia procesu tworzenia interfejsu użytkownika w systemach WEBGIS poprzez wykorzystanie GWT (Google Web Toolkit) oraz przedstawiono wady i zalety tego rozwiązania.

AIDING THE CREATION OF MULTIFUNCTIONAL GIS WEB APPLICATIONS BY USAGE OF GOOGLE WEB TOOLKIT

Geographic Information Systems (GIS) are specialized systems that capture, store, analyze and present geographical data. Along with the Internet evolution, WEBGIS has been created. WEBGIS are the Geographic Information Systems that use web technologies.

One of the WEBGIS developers' problems is a data presentation. Pure HTML is not enough, on the other hand applications that are created with popular web technologies like JavaServerFaces are difficult in implementation and ineffective in runtime.

This paper checks if process of WEBGIS user interface creation can be simplified by the usage of GWT (Google Web Toolkit). Advantages and disadvantages of this solution are also described.

[1] GisPartner Sp. Z.O.O. ; ul. Kilińskiego 30; 50-264 Wrocław; e-mail g.malicka@gmail.com
[2] Politechnika Lubelska, Wydział Elektrotechniki i Informatyki; 20-618 Lublin; ul. Nadbystrzycka 36b
tel: +48 81 525-20-46, fax: +48 81538-43-49, e-mail: kamilz@cs.pollub.pl

1. WSTĘP

GIS, czyli Systemy Informacji Geograficznej (ang. Geographic Information Systems), to specjalistyczne systemy służące gromadzeniu, przetwarzaniu oraz prezentacji danych geograficznych. Są stosowane w wielu gałęziach nowoczesnej gospodarki m.in. w przemyśle, naukach przyrodniczych, urbanistyce, logistyce, rolnictwie. Z potencjału GIS korzysta również wojsko, administracja publiczna, służby zajmujące się ochroną zdrowia i zarządzaniem w sytuacjach kryzysowych.

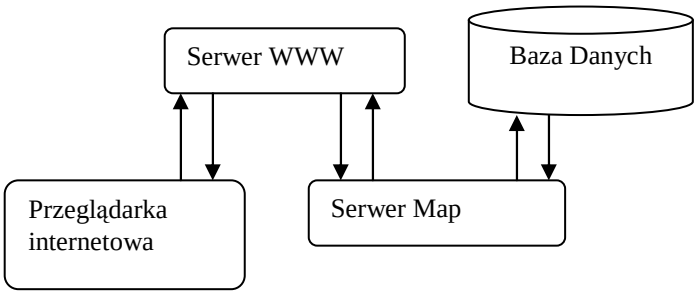
Systemy Informacji Geograficznej są tworzone po to, aby usprawniać pracę i wykonywanie codziennych obowiązków swoich użytkowników m.in. poprzez:

- wyznaczenie trasy,
- zgłoszenie dziury w drodze,
- odnalezienie najbliższego szpitala,
- wyszukanie właściciela działki,
- sprawdzenie, jakie zwierzęta występują na terenie nadleśnictwa,
- zlokalizowanie miejsca postoju samochodu firmowego,
- wykonanie analizy demograficznej,
- wsparcie dla specjalistycznych prac geodezyjnych,
- dodanie punktu adresowego,
- wydruk planu itd.

Dlatego też ważne jest zapewnienie swobodnego dostępu do systemów GIS szerokiemu gronu odbiorców, w tym specjalistom z dziedziny geodezji i kartografii, pracownikom urzędów i firm oraz zwykłym obywatelom. Stało się to możliwe dzięki rozwojowi Internetu, który przyczynił się do powstawia WEBGIS, czyli grupy Systemów Informacji Geograficznej wykorzystujących technologie internetowe [1].

2. ARCHITEKTURA SYSTEMU WEBGIS

Internetowe Systemy Informacji Geograficznej (WEBGIS) działają w architekturze klient-serwer i składają się z trzech podstawowych elementów: serwera map, bazy danych oraz interfejsu użytkownika (rys. 1).



Rys.1. Uproszczona architektura systemu WEBGIS [1]

Serwer map jest niezależną maszyną, której zadaniem jest generowanie map o zadanych parametrach na podstawie posiadanych lub też pobieranych informacji przestrzennych. Jego podstawowymi funkcjami są: wizualizacja, nawigacja, analizy przestrzenne.

Baza danych służy do przechowywania danych przestrzennych, które stanowią najważniejszy i najbardziej wartościowy element Systemów Informacji Geograficznej. Mogą pochodzić z wielu źródeł takich, jak GPS, mapy, zdjęcia lotnicze, obrazy satelitarne i bezpośrednie pomiary w terenie [2]. Przykładami danych przestrzennych mogą być: adresy klientów, granice administracyjne, sieci dystrybucyjne, lokalizacje pracowników i samochodów w terenie, ogniska chorób, rejony zagrożone powodzią itd.

Interfejs użytkownika służy do prezentowania danych oraz interakcji z użytkownikiem. To za jego pomocą użytkownik decyduje, jaką mapę wyświetlić i w jakim przybliżeniu, to dzięki niemu może wyszukiwać i przeglądać informacje oraz wykonywać analizy przestrzenne.

3. INTERFEJS UŻYTKOWNIKA W SYSTEMIE WEBGIS

Początkowo interfejs użytkownika systemu WEBGIS był tworzony z wykorzystaniem czystego HTML. Ówczesne systemy WEBGIS pełniły rolę prostych przeglądarek map, czego przykładem może być aplikacja Map Viewer opracowana przez Xerox Corporation, która generowała i wyświetlała statyczne mapy w formacie GIF [6].

Wraz z rozwojem Internetu pojawiały się nowe technologie i frameworki do tworzenia aplikacji webowych. Jednym z nich był JavaServer Faces (JSF). Elementy funkcjonalne JSF takie jak: implementacja wzorca MVC (model-widok-kontroler), stanowy, komponentowy interfejs użytkownika, wsparcie dla internacjonalizacji, mechanizmy walidacji danych po stronie serwera, sprawiły, że zyskał popularność wśród twórców części klienckiej aplikacji WEBGIS.

Wadą i zaletą JSF jest interfejs użytkownika obsługiwany przez serwer i wysyłany klientowi jako wyrenderowany HTML. Dzięki temu klient nie musi instalować u siebie dodatkowego oprogramowania, ale każde zapytanie użytkownika implikuje wygenerowanie i przesłanie całej nowej strony HTML, przez co aplikacje są mniej wydajne [7].

Przykładami wykorzystania JSF są:

- http://ikar2.pgi.gov.pl/geoportal
- http://nbii-nin.ciesin.columbia.edu/beacon/mapviewer.jsf? .

Obecnie do tworzenia interfejsu użytkownika w systemach WEBGIS, ze względu na swoją efektywność, najczęściej wykorzystywane są języki skryptowe działające po stronie klienta. Wśród nich dużą popularnością cieszy się technologia AJAX, opierająca się na JavaScript i XML, która umożliwia asynchroniczną komunikację użytkownika z serwerem, dzięki czemu w momencie wykonania akcji, może zostać przeładowany tylko wybrany obszar strony www.

Firma Google była jedną z pierwszych firm, które wykorzystały technologię AJAX w swoich aplikacji internetowych np. Google Maps. Jednakże tworzenie aplikacji przy użyciu czystego AJAX'a nie jest łatwe, w dużej mierze ze względu na problemy z kompatybilnością przeglądarek internetowych i testowaniem kodu w trakcie pisania.

W związku z tym Google postanowiło stworzyć narzędzie ułatwiające tworzenie aplikacji w tej technologii. Tak powstało GWT (Google Web Toolkit) – darmowe narzędzie do tworzenia aplikacji wykorzystujących AJAX bez konieczności znajomości tej

technologii. GWT pozwala pisać aplikacje w Javie dostarczając zestaw bibliotek do tworzenia i testowania komponentów interfejsu użytkownika i logiki biznesowej [3].

4. GOOGLE WEB TOOLKIT

4.1. Komponenty interfejsu użytkownika

Do tworzenia interfejsu użytkownika w GWT służy zbiór kontrolek bardzo podobnych do komponentów biblioteki Swing. Biblioteki interfejsu użytkownika zawierają wstępnie zdefiniowane komponenty, które podczas kompilacji tłumaczone są na ich odpowiedniki w JavaScript i HTML dla konkretnej już przeglądarki. Cały ciężar zapewnienia kompatybilności interfejsu użytkownika z przeglądarką spada na GWT.

GWT oferuje m.in.: panele, przyciski, pola tekstowe, listy rozwijane, listy hierarchiczne, tabele, których właściwości mogą być dowolnie modyfikowane poprzez style CSS i drzewo DOM. W wersji 2.0 GWT został wprowadzony UiBinder pozwalający na odseparowanie logiki biznesowej od części prezentacji poprzez tworzenie dwóch plików dla każdego komponentu interfejsu użytkownika. Elementy widoku znajdują się w pliku XML, który zawiera znaczniki HTML i komponenty GWT tzw. Widgety. Natomiast implementacja logiki biznesowej pozostaje w osobnym pliku .java. Powiązania pomiędzy elementami widoku a logiką biznesową w klasie .java następują poprzez odpowiednie adnotacje @UiField [5]. Z kolei za pomocą adnotacji *@UiHandler* można podpiąć obsługę zdarzeń (listing 1 i 2).

Listing 1. plik Test.ui.xml

```
<!DOCTYPE ui:UiBinder SYSTEM
"http://dl.google.com/gwt/DTD/xhtml.ent">
<ui:UiBinder xmlns:ui="urn:ui:com.google.gwt.uibinder"
   xmlns:g="urn:import:com.google.gwt.user.client.ui">
   <ui:style>
   </ui:style>
   <g:HTMLPanel>
      <g:Button ui:field="searchButton"
         text="Click me" width="140px"
         styleName="search"></g:Button>
   </g:HTMLPanel>
</ui:UiBinder>
```

Listing 2. plik Test.java

```
package pl.test;
public class Test extends Composite {

   private static TestUiBinder uiBinder =
GWT.create(TestUiBinder.class);

   interface TestUiBinder extends UiBinder<Widget, Test> {
   }
```

```
   @UiField
   Button searchButton;

   @UiHandler("searchButton")
   void searchButtonOnClick(ClickEvent e) {
      Window.alert("Clicked");
   }
   public Test() {
      initWidget(uiBinder.createAndBindUi(this));
   }
}
```

4.2. Modularność aplikacji

Zaletą aplikacji tworzonych przy użyciu GWT jest możliwość dzielenia ich na moduły. Każdy moduł jest osobnym niezależnym bytem, któremu jest przypisany plik XML z konfiguracją GWT odpowiadającą jednostkom funkcjonalnym aplikacji. Plik ten ma rozszerzenie gwt.xml (np.: nazwa.gwt.xml) i zawiera ustawienia niezbędne do poprawnego uruchomienia aplikacji GWT, w tym klasy które podlegają translacji do JavaScript (listing 3).

Listing 3. Przykładowy plik konfiguracyjny gwt.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<module rename-to='GIS'>
   <inherits name='com.google.gwt.user.User' />
   <inherits name="com.google.gwt.json.JSON" />
   <inherits name="com.google.gwt.uibinder.UiBinder" />
   <set-property name="locale" value="pl"/>
   <entry-point class='pl.gis.GIS' />
</module>
```

Większość swojej funkcjonalności GWT udostępnia poprzez moduły, m.in. mechanizmy do pracy z danymi w postaci JSON i XML, które są częstym sposobem udostępniania danych przez serwery map.

Modułowość aplikacji GWT, w połączeniu z możliwością dziedziczenia modułów, jest bardzo korzystna z punktu widzenia twórców systemów WEBGIS. Pozwala na wielokrotne wykorzystanie tego samego kodu w różnych aplikacjach, np. mechanizmów wyświetlania mapy czy dostępu do bazy danych. Pozwala również na łatwe rozbudowywanie istniejących aplikacji o nową funkcjonalność [4].

4.3. Kompatybilność aplikacji z przeglądarkami www

Niezależnie czy system WEBGIS jest zwykłą przeglądarką map, czy zaawansowanym systemem wspierającym specjalistyczne prace, ważne jest, aby był stabilny i jednakowo działający w dowolnej przeglądarce internetowej, z poziomu której będzie uruchamiany. Najważniejszym elementem przeglądarki internetowej jest jej silnik, który odpowiada za

przetwarzanie zawartości stron internetowych (kod HTML, XHTML, skrypty, grafiki) oraz ich elementów formatujących (arkusze CSS i XSL), a także wyświetlenie rezultatu. Wśród typów silników przeglądarek internetowych można wyróżnić:

- Trident -używany przez Internet Explorer,
- Gecko - używany m.in. przez Firefox,
- Presto - używany przez Opera od wersji 7,
- WebKid - używany m.in. przez Safari i Google Chrome.

Wśród wymienionych silników wszystkie, oprócz Trident, implementują nowoczesne standardy W3C, ale i wśród nich widoczna jest różnica w obsłudze wybranych API. Z powodu niespójności w implementacji standardów, utworzenie aplikacji z wykorzystaniem czystego JavaScriptu (czy też innych języków skryptowych) działającej w identyczny sposób pod każdą z przeglądarek wymaga niewspółmiernego nakładu pracy.

Wychodząc na przeciw tym problemom twórcy GWT udostępnili programistom spójne API wyższego poziomu, które deleguje odpowiednią logikę w zależności od sposobu implementacji standardów przez silnik przeglądarki. Następstwem tego jest generowanie w czasie kompilacji osobnych wersji skryptu dla każdego typu silnika. Dzięki temu działanie aplikacji GWT jest jednakowe w większości przeglądarek internetowych, korzystających z wymienionych silników [4].

Tworzenie wielu wersji skryptu wynikowego jest czasochłonne i nie zawsze potrzebne (np. podczas testów), dlatego GWT. pozwala ograniczyć liczbę skryptów wynikowych poprzez zdefiniowanie w pliku gwt.xml właściwości informującej dla jakich przeglądarek ma być dostępna aplikacja. Ograniczenie kompilacji do przeglądarek Internet Explorer 8 i FireFox (od wersji 2) przedstawiono na listingu 4.

Listing 4. Ograniczenie ilości kompilacji aplikacji GWT

```
<set-property name="user.agent" value="ie8"/>
<set-property name="user.agent" value="gecko1_8"/>
```

4.4. Internacjonalizacja aplikacji

Mapy wyświetlane w aplikacjach webowych swym zasięgiem mogą obejmować obszar powiatu, kraju (http://ikar2.pgi.gov.pl/mvs_viewer/mapviewermvs.jsf), a nawet całej kuli ziemskiej (http://maps.google.com/). Niezależnie od zasięgu przestrzennego aplikacji, udostępnienie jej przez Internet sprawia, że użytkownik końcowy może być mieszkańcem dowolnego miejsca na Ziemi. Dlatego też ważnym elementem aplikacji WEBGIS jest jej internacjonalizacja pod kątem przyszłych odbiorców.

GWT udostępnia kompleksowy mechanizm internacjonalizacji. Elementem, który umożliwia lokalizację tekstów w aplikacji jest moduł zawarty w pakiecie com.google.gwt.i18n, oferujący narzędzia takie, jak : Constants, Messages, Dictionary.

Dictionary, to dynamiczny sposób internacjonalizacji, pozwalający aplikacji na korzystanie z komunikatów definiowanych w czasie jej działania. Spowalnia to działanie aplikacji, ale nie wymaga ponownej kompilacji kodu, gdy nastąpią zmiany w treści komunikatów lub rozszerzona zostanie lista dostępnych języków.

Do statycznej internacjonalizacji służą Constants i Messages. Jest to najbardziej efektywny sposób tworzenia wersji językowych dla komunikatów w aplikacji. W czasie

kompilacji tworzone są osobne wersje skryptu dla poszczególnych wersji językowych, a do każdej wstawiane są odpowiednie treści zdefiniowane w plikach .properties [3].

Dostępne wersje językowe definiuje się w pliku gwt.xml, co przedstawiono na listingu 5.

Listnig 5. Wersje językowe dostępne w aplikacji GWT

```
<set-property name="locale" value="pl"/>
<set-property name="locale" value="en"/>
```

Istnieją dwa sposoby wyświetlania tekstów statycznych w danym języku:

- ustawienie właściwości <meta name="gwt:property" content="locale=pl"> bezpośrednio w pliku HTML,
- dodanie parametru do URL http://www.gis.com/GIS.html?locale=pl.

Mechanizmy, które zostały opisane można używać tylko w części klienckiej aplikacji, czyli tej, która jest kompilowana do postaci skryptu JavaScript. Stanowi to pewną wadę tego rozwiązania, ponieważ wymusza zastosowanie w części serwerowej aplikacji oddzielnych mechanizmów do internacjonalizacji, jakimi są dostarczone przez Java ResourceBundle.

4.5. JavaScript Native Interface (JSNI)

Kod źródłowy aplikacji GWT jest pisany w języku Java, jednak programiści mają do dyspozycji mechanizm pozwalający na integrowanie kodu aplikacji z istniejącymi rozwiązaniami napisanymi w języku JavaScript. Służy do tego JSNI, który pozwala na wykonanie kodu JavaScript z poziomu języka Java. JSNI można wykorzystać do:

- implementacji metod Java w JavaScript,
- wywołania kodu Java z poziomu JavaScript i odwrotnie,
- odczytywania i zapisywania atrybutów klasy Java przy pomocy JavaScript.

Metody JSNI deklarowane są jako *native*. Kod natywnej metody musi znajdować się pomiędzy znacznikami */*-{ ... }-*/*. Zapis stanowi dyrektywę dla kompilatora, aby traktował znajdujący się pomiędzy znacznikami tekst jako poprawny kod JavaScript i wstrzyknął go do wygenerowanych plików GWT. W metodzie JSNI dostęp do obiektu okna lub obiektu dokumentu, musi być poprzedzony odpowiednio *$wnd* i *$doc*. Listing 6 przedstawia metodę JSNI wywołującą metodę JavaScript wyświetlającą alert. Metody JSNI wywoływane są jak zwykłe metody Java, a z ich poziomu można wywoływać metody Java oraz modyfikować atrybuty klasy (listing 7).

Listing 6. Natywna metoda

```
public static native void alert(String msg) /*-{
      $wnd.alert(msg);
}-*/;
```

Listing 7. Korzystanie z JSNI

```
package pl.test.client;

public class JSNITest {

   String text;

   public void print(String text,int x) {
      System.out.println(text+" "+x);
   }

   public void callJSNI() {
      printJSNI("text");
   }

   public native void printJSNI(String s)/*-{
     this.@pl.test.client.JSNITest::print (Ljava/lang/String;I)(s,2);
   }-*/;

   public native void modifyFieldJSNI(String newText)/*-{
      this.@pl.test.client.JSNITest::text = newText;
   }-*/;
}
```

Korzystanie z metod JSNI pozwala znacząco rozszerzyć możliwości aplikacji, w szczególności poprzez możliwość wykorzystania istniejących rozwiązań napisanych w języku JavaScript. Jednakże pociąga to za sobą pewne niedogodności m.in. fragmenty aplikacji napisane w JavaScript nie zawsze będą działać jednakowo we wszystkich przeglądarkach i nie będą mogły być optymalizowane przez kompilator.

4.6. Integracja aplikacji GWT z istniejącymi rozwiązaniami do tworzenia części klienckiej aplikacji WEBGIS

Jak wspomniano wcześniej, popularnym sposobem tworzenia interfejsu użytkownika w systemach WEBGIS stały się języki skryptowe, a w szczególności JavaScript. Wiodący dostawcy rozwiązań GIS wychodząc na przeciw oczekiwaniom twórców WEBGIS, zaczęli tworzyć w JavaScript zestawy API pozwalające na tworzenie aplikacji mapowych działających w przeglądarkach internetowych. Tak powstały m.in.:

- ArcGIS API for JavaScript (http://help.arcgis.com/en/webapi/javascript/arcgis/), biblioteka, której producentem jest ESRI, umożliwia budowę aplikacji wykorzystujących usługi oferowane przez ArcGIS Server,
- OpenLayers (http://openlayers.org/), biblioteka utworzona przez MetaCarta, obecnie rozwijana przez Open Source Geospatial Foundation, umożliwia budowę aplikacji mapowych korzystających z różnych źródeł danych oraz dostęp do OGC web services (WMS, WFS etc.),
- Maps JavaScript API, biblioteka do tworzenia aplikacji mapowych, producentem której jest Google.

Wymienione zestawy API mogą być z powodzeniem stosowane w aplikacjach GWT dzięki mechanizmom JSNI.

Biblioteki do tworzenia aplikacji mapowych mają wiele wyspecjalizowanych klas, z których każda odpowiada za inny fragment logiki oferowanej przez interfejs systemu WEBGIS np. pokazanie mapy, dodanie do niej warstw czy wyszukanie informacji. Wiele z tych klas jest między sobą zależnych np. obiekt mapy korzysta z warstw, a różne rodzaje warstw dziedziczą z głównego obiektu warstwy.

Najprostszym sposobem na użycie tych klas w aplikacji, przy jednoczesnym zachowaniu struktury zależności pomiędzy nimi, jest ich opakowanie w klasy Java. Klasa opakowująca może zostać utworzona w dwojaki sposób: poprzez dziedziczenie po klasie *JavaScriptObject* lub przez instancyjne pole typu *JavaScriptObject(listing 8)* [4].

Listing 8. Przykład opakowania klasy Map pochodzącej z biblioteki JavaScript OpenLayers

```
public class Map {

  private JavaScriptObject map;
  private MapWidget parent;

  public Map(Element mapDomElement, MapWidget parent) {
   this.map = _newInstance(this, mapDomElement);
   this.parent = parent;
  }

  private static native JavaScriptObject _newInstance(Map m,
        Element mapDomElement) /*-{
    var map = new $wnd.OpenLayers.Map(mapDomElement);
    return map;
  }-*/;
}
```

Programiści aplikacji WEBGIS nie muszą tego robić samodzielnie, ponieważ producenci JavaScript API, dostrzegając rosnące zainteresowanie technologią GWT przy tworzeniu systemów WEBGIS, korzystając z mechanizmu JSNI utworzyli dla swojego oprogramowania biblioteki opakowujące, pozwalające na ich bezpośrednie wykorzystanie w aplikacjach GWT. Twórcy aplikacji mają do dyspozycji m.in.:

- gwt-esri opakowujące bibliotekę ArcGIS API for JavaScript,
- GWT-OpenLayers opakowujące bibliotekę OpenLayers,
- gwt-google-apis opakowujące biblioteki Maps JavaScript API.

Aby móc korzystać z wymienionych bibliotek, programiści muszą jedynie zaimportować wybraną przez siebie bibliotekę do aplikacji oraz wczytać odpowiedni JavaScript na stronie HTML.

5. WNIOSKI

GWT jest praktycznym narzędziem pozwalającym na pisanie zawansowanych aplikacji opartych na technologii AJAX. Może być z powodzeniem stosowane do tworzenia części klienckiej w systemie WEBGIS ze względu na:

- wsparcie dla JSON i XML, które są formatami danych najczęściej zwracanymi przez serwery map,
- możliwość integracji z gotowymi rozwiązaniami JavaScript do wizualizacji map i przetwarzania danych przestrzennych,
- mechanizm internacjonalizacji,
- jednakowe działanie aplikacji w większości przeglądarek internetowych,
- bogaty zestaw kontrolek pozwalających na prezentowanie danych oraz interakcje z użytkownikiem,
- możliwość modyfikacji wyglądu aplikacji za pomocą styli CSS,
- możliwość testowania aplikacji w trakcie jej tworzenia,
- tworzenie aplikacji z wykorzystaniem języka Java i podstaw HTML bez konieczności znajomości JavaScript i technologii AJAX.

6. BIBLIOGRAFIA

[1] Pinde Fu, Juilin Sun: *Web GIS: Principles and Applications*, ESRI Press 2010
[2] Scott Davis: *GIS for Web Developers: Adding 'Where' to Your Web Applications*, Pragmatic Bookshelf 2007
[3] Robert Cooper, Charles Collins: *GWT w praktyce*, Manning Publications 2008
[4] Robert Hanson, Adam Tacy: *GWT in Action: Easy Ajax with the Google Web Toolkit*, Manning Publications 2007
[5] http://code.google.com/intl/pl/webtoolkit/doc/latest/DevGuideUi.html
[6] Rzeszewski Michał, Jasiewicz Jarosław, *WebGIS - od map w internecie do geoprzetwarzania*, GIS- platforma integracyjna geografii 2007
[7] David Geary, Cay Horstmann, *Core JavaServer Faces*, Prentice Hall 2008