Software Requirements Specification

for

GWT Mapmaker

Version 1.0 approved

Prepared by Jason Ferguson

<date created>

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 2

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 3

3.4 Communications Interfaces 3

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 4

5. Other Nonfunctional Requirements 4

5.1 Performance Requirements 4

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 5

Appendix A: Glossary 5

Appendix B: Analysis Models 5

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Initial | 22 Apr 11 |  | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

This document describes the GWT Mapmaker application, version 1.0. The purpose of the application is to allow a user to dynamically generate a map with borders and features, and export it in standard GIS formats.

## Document Conventions

## Intended Audience and Reading Suggestions

This document is intended for software developers seeking to understand the GWT Mapmaker application. It is also intended for review by persons reviewing my skills as a software developer.

This document is intended to follow the IEEE standard 830 for Software Requirements Specifications. Most headings are in accordance with the that standard, except when broken out for further detail.

## Product Scope

The primary purpose of the GWT Mapmaker application is to general KML files for end-users wishing either to display a Google Map on their own website or to create a file for use with the Google Earth application.

One potential use case for this application would involve a user looking for a list of all churches in a given ZIP code. The user would enter the ZIP code of their interest, then select churches for their feature type. The application would generate the KML file with the requested information for their use.

## References

# Overall Description

## Product Perspective

This application is intended to replace an earlier application written in the PHP language. It is intended as a demonstration of usage of the Google Web Toolkit (GWT) toolset and the Geotools library.

The earlier application was written in PHP version 5 and used hand-written Javascript to generate XML files which then required end-users to use another piece of Javascript code to read and display data on a Google Map. The XML files generated did not conform to any commonly used format such as KML. It was also written in a “wizard” format, where the user would walk through a series of pages to specify what features would be displayed.

This application is intended to take advantage of the more dynamic features offered by GWT. Users will be able to select map features and see a map dynamically generated in real time (or nearly real time). All features will be displayed on a single page, versus the wizard format of the previous application.

This version of the application will be limited to feature sets available from the United States Geological Service (USGS). Border data will be limited to line data available from the United States Census TIGER line data. Future versions may conceivably allow users to provide customized data for their own maps.

## Product Functions

The application must perform the following functions:

* Allow the user to select the features which will be displayed on the map.
* Display a map which dynamically updates based on user selection
* Allow the user to select the external formats they wish to export the map data in (i.e. KML or ESRI Shapefile)

## User Classes and Characteristics

This version of the application is intended to be “open”. Any user will be able to use this application without authentication.

Users should be familiar with basic usage of a web browser and how to use controls such as a Tree control. They must be able to follow certain directions to display the generated KML on their web page as an overlay of a Google Map.

## Operating Environment

The application is intended to take advantage of the GAE hosting platform. Therefore, it must be written to take advantage of certain limitations of that platform. These limitations include (but are not limited to) the Java class whitelist and inability to use certain common libraries (such as the Hibernate Object Relational Mapping tool).

## Design and Implementation Constraints

### Constraints Related to Technologies, Tools, and Databases

#### Constraints Related to Technologies

#### Constraints Related to Security

#### Constraints Related to Design Conventions or Programming Standards

Since one of the purposes of this application is to demonstrate the usage of Google Web Toolkit (GWT) and the Geotools library, certain decisions are forced upon this application at this time. This includes the choice of Java as the development language and the choice of GWT as a display technology.

These predetermined choices then cascade to other decisions, such as those related to software libraries, as well as the user’s Integrated Development Environment (IDE).

## User Documentation

User documentation for the application will be limited to popup windows in the application itself. Users will click on a link to activate a modal popup window with help on selecting options, and making the application work with their own website.

## Assumptions and Dependencies

2.7.1 Assumptions

The following assumptions are being made:

* A schema can be designed to link the USGS data with the Census TIGER data. Both types of data specify a Federal Information Processing System 55 (FIPS 55) code that should be able to be used, but there are slight differences in the Census TIGER data.

# External Interface Requirements

## User Interfaces

3.1.1 Primary User Interface

The primary user interface will consist of a single web page with three “panels”:

* Stack Layout Panel – The stack layout panel will occupy the majority of the left side of the user interface. The stack layout panel will contain four “sub” panels:
  + Features Panel – The features panel will contain the user interface components necessary to determine which borders and features will be displayed/exported.
  + Export Panel – The export panel will contain the user interface components necessary to allow the user to export the features in the format of their choice.
  + Admin Panel – The admin panel will allow users to import the borders and features
  + Help Panel – The help panel will show the option to display the help for the application
* Map panel – the map display panel will be displayed to the right of the tree control panel, but above the KML text panel. It will contain a Google Map that dynamically updates whenever the user updates options in the tree control panel.
* Disclaimer panel – the disclaimer panel will display application copyright information.

3.1.2 Help Dialog User Interface

The Help Dialog user interface will be a modal dialog box that will appear as an overlay of the Primary user interface described above. The user will activate it either via button or hyperlink.

3.1.3 Import Data User Interface

The import data user interface will be a popup window to allow the user to import Shapefile and feature data into the application datastore.

When importing shapefile data, if no data is currently in the datastore, the user will only receive the option to import the state borders shapefile. Once state borders are imported, the user will be able to import state-based data files. Once county borders are imported, the user will be able to import county-based shapefiles.

## Hardware Interfaces

Data to be uploaded to the application datastore will be in ZIP archive format, as provided by the USGS and Census TIGER datasets. The application will be responsible for unarchiving the data, converting it to proper format, then storing it.

## Software Interfaces

### Software Libraries to be Used

The following software libraries have currently been identified for use. Note: transitory dependencies (software libraries that these identified libraries require) are not listed.

* Google Web Toolkit
* Spring
* Apache Commons Libraries:
  + Commons-FileUpload
  + Commons-IO
  + Commons-Lang
* Apache OpenJPA Library
* JUnit (unit testing only)

### Specifications of Data to be Imported

Data to be imported will be in either Census Department TIGER/Line format (for boundaries) or USGS Comma-Separated Value (CSV) format. Data will be imported by the application directly from the source, preventing possible malicious or accidental modification of data.

### Specifications of Data to be Exported The user can specify the data to be exported in the following formats:

- KML

- ESRI Shapefile

### Specification of Datastore/Repository

Application data will be stored in a Relational Database System.

## Communications Interfaces

Access to the application will be by web browser, using the HTTP format. The application will communicate via HTTP with the USGS and TIGER websites to import data into the application datastore.

# System Features

## Select Borders and Features for Map

4.1.1 Description and Priority

4.1.1.1 Description: allow the user to select border types and features via standard HTML form controls

4.1.1.2 Priority: HIGH

## Import Shapefile and Features

4.1.1 Description and Priority

4.1.1.1 Description: Allow the user to import ESRI shapefile for borders and USGS features for displaying on screen and exporting in GIS formats

4.1.1.2 Priority: HIGH

## Export GIS Files

4.1.1 Description and Priority

4.1.1.1 Description: allow users to export map data in the GIS data format (i.e. KML or ESRI Shapefile) of their choice

4.1.1.2 PRIORITY: MEDIUM

4.1.2 Stimulus/Response Sequences

Action: User selects border type via tree control

Response: System remembers preference of border type

Action: User selects location to display

Response: System displays map of location with border. System generates KML file and displays in the KML panel.

Action: User selects features to display on map

Response: System displays map of location, with features, with border. System generates KML file and displays it in the KML panel.

4.1.3 Functional Requirements

REQ-1:

REQ-2:

## Import Data

# Other Nonfunctional Requirements

## Performance Requirements

The Map panel of the application interface is intended to update whenever the user updates an option in the Tree user panel. If this takes too much time, it may be necessary to implement a timer that waits 2-3 seconds before the update (in case the user is making multiple selections at once).

## Safety Requirements

This application is not intended for any use commercial use. A footer may be necessary to that effect.

## Security Requirements

The administrative access page is intended to be hidden from use by most end-users. Since authentication is not required for using this application, it is necessary that the knowledge of the URL for said page be limited.

## Software Quality Attributes

As a demonstration project, it will be important that the source code address, to varying degrees, the following areas:

* Adaptability
* Availability – availability is dependent upon the hosting environment.
* Correctness – unit testing will be used as much as possible to verify the correctness of the application
* Flexibility
* Maintainability – source code will be well-documented, to include Javadoc class and method descriptions and commenting code that is not easily-understood.
* Reliability
* Robustness
* Testability
* Usability

## Business Rules

<List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.>

# Other Requirements

<Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.>

Appendix A: Glossary

Commonly Used Acronyms

GWT – Google Web Toolkit

JPA – Java Persistence Architecture

KML – Keyhole Markup Language

ORM – Object-Relational Mapping

Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

Appendix C: To Be Determined List

<Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.>