Studienarbeit: Google Fusion Table Prototype

Jürg Hunziker
jhunzike@hsr.ch

Stefan Oderbolz
soderbol@hsr.ch

29. Mai 2012

Todo list

Abstract schreiben 5

Management Summary erstellen 6

Glossar funktioniert nicht korrekt 27

Impressum

Ersteller: Jürg Hunziker, Stefan Oderbolz

Änderungsverlauf

Datum	Änderungen	Bearbeiter
27.02.2012	Dokumententwurf erstellt	Stefan Oderbolz
01.03.2012	Struktur erstellt	J??rg Hunziker

Zusammenfassung

Abstract schreiben

Ein Abstract ist eine rein textuelle kurze Zusammenfassung der Arbeit. Der Abstract ist für die Recherche in grossen Dokumentensammlungen geeignet. Er umfasst nie mehr als eine Seite, typisch sogar nur etwa 200 Worte (etwa 20 Zeilen). Der Begriff 'Kurzfassung' ist zuwenig genau definiert; er soll wenn möglich vermieden werden.

Management Summary und Web-Publikation

Management Summary erstellen

Das Management Summary soll 2-5 Seiten umfassen sowie eine bis zwei Figuren enthalten. Es richtet sich an den „gebildete Laien“ auf dem Gebiet und beschreibt daher in erster Linie die (neuen und eigenen) Ergebnisse und Resultate der Arbeit. Die Sprache soll knapp, klar und stark untergliedert sein. Grundlage für das Management Summary kann der Broschüren-Eintrag sein, den die Abteilung bei Diplomarbeiten jeweils früh verlangt, um eine Broschüre zu drucken. Das Management Summary dient als Vorlage für eine allfällige Web-Publikation. Das Abstract und das Management Summary werden - zeitlich gesehen - gegen Schluss der Arbeit geschrieben und bilden zusammen mit den Schlussfolgerungen im technischen Bericht den am häufigsten gelesenen Teil der Arbeit. Diese Dokumente sollen daher am Sorgfältigsten ausgearbeitet sein. Die folgenden Stichworte sollen die typische Struktur illustrieren, wobei die genaue Ausführung jeweils auf die spezifischen Bedürfnisse und Randbedingungen eines Projekts anzupassen ist. Diese Struktur kann auch für die Präsentation der Arbeit als "Richtschnur"dienen.

1. Ausgangslage
    - Warum machen wir das Projekt?
    - Welche Ziele wurden gesteckt (Kann-Ziele, Muss-Ziele)
    - Was machen andere / welche ähnlichen Arbeiten gibt es zum Thema?
    - Vorgehen: Was wurde gemacht? In welchen Teilschritten?
    - Risiken der Arbeit?
    - Wer war involviert (Durchführung, Entscheide usw.)?
    - Was konnte von anderen verwendet werden?
2. Ergebnisse
    - Was ist das Resultat?

- Bewertung der Resultate, was ist Neuartig an der Arbeit?
- Zielerreichung bezüglich Kann-/Muss-Zielen
- Abweichungen (positiv und negativ) und kurze Begründung dafür (Externe) Kosten der Arbeit?
- Was ist der Nutzen (quantifizierbar/nicht quantifizierbar)?

3. Ausblick
   - Was hat man mit Durchführung des Projekts gelernt?
   - Verbleibende Probleme, (zukünftige) Gegenmassnahmen bez. Risiken
   - Was würde man anders machen, was ist weiter zu tun

Inhaltsverzeichnis

I. Einleitung 10

1. Einleitung 11

1.1. Problemstellung . . . 11
1.2. Aufgabenstellung . . . 11
1.3. Ziele . . . 11
1.4. Rahmenbedingung . . . 12
1.5. Vorgehen . . . 13
1.5.1. Potential von Google Fusion Tables . . . 13
1.5.2. Prototypen . . . 13
1.6. Aufbau der Arbeit . . . 14
1.7. Stand der Technik . . . 14
1.8. Bewertung (Evaluation) . . . 14
1.9. Vision/Umsetzungskonzept . . . 14
1.10. Resultate der Arbeit . . . 14
1.11. Schlussfolgerungen . . . 14
1.12. Ausblick . . . 14

II. Projektdokumentation 15

2. Einführung in Google Fusion Tables 16

2.1. SQL API . . . 16
2.1.1. Client Libraries . . . 16
2.2. Geocodierung . . . 17
2.3. Google Maps API - FusionTablesLayer . . . 17
2.3.1. Spatial-Queries . . . 18
2.3.2. Karten-Stile . . . 19
2.3.3. Heatmaps . . . 19
2.3.4. Vorteil . . . 20
2.3.5. Nachteil . . . 21
2.3.6. Einschränkungen . . . 21
2.4. Google Fusion Table Javascript Library (gftlib-js) . . . 21
2.4.1. Abhängikeiten . . . 22

2.4.2. Methoden . . . 22
2.5. Use Case 1: WorldData Explorer . . . 23
2.5.1. Einleitung . . . 23
2.5.2. Anforderungsspezifikation . . . 23
2.5.3. Analyse . . . 23
2.5.4. Design . . . 23
2.5.5. Implementation . . . 23
2.5.6. Test . . . 23
2.5.7. Resultate . . . 23
2.5.8. Weiterentwicklung . . . 23
2.5.9. Benutzerdokumentation . . . 23

III. Projektmanagement **24**

3. Projektmanagement **25**
3.1. Allgemeines . . . 25
3.2. Projektmanagement . . . 25
3.3. Projektmonitoring . . . 25

IV. Anhänge **26**

Teil I.

Einleitung

1. Einleitung

In unserer Arbeit untersuchen wir die Möglichkeiten welche die Clouddatenbank Google Fusion Tables bietet. Es sollen einige Prototypen für verschiedenste Anwendungsfälle im GIS-Bereich erstellt werden, welche das Potential der Datenbank aufzeigen.

1.1. Problemstellung

Die Aufgabenstellung stammt von der GEOINFO AG (www.geoinfo.ch), welche massgeschneiderte GIS-Softwarelösungen für ihre Kunden entwickelt.

Für solche Unternehmen wird es nach und nach schwieriger sich auf dem Markt zu beweisen, da bereits viele cloudbasierte GIS-Lösungen sehr günstig oder gar kostenlos erhältlich sind. Durch die Prototypen soll ersichtlich gemacht werden, welche Anwendungsfälle von bestehenden proprietären GIS-Sytemen bereits mit Google Fusion Tables realisierbar wären und welchen Aufwand dies darstellen würde.

1.2. Aufgabenstellung

Im Rahmen dieser Arbeit sollen das Potential aber auch Einschränkungen von Google Fusion Tables für den Einsatzbereich eines öffentlichen Web GIS evaluiert werden. Es ist aufzuzeigen, welche der typischen Anwendungsfälle, wie sie in aktuellen Web GIS Lösungen (z.B. www.geoportal.ch, www.stadtplan.stadt-zuerich.ch) implementiert sind, auf Basis von Google Fusion Tables und Google Maps realisiert werden könnten. Eine Auswahl dieser Grundfunktionen ist anhand eines Prototypen zu implementieren. Die Zielgruppe sind demnach GIS-Sachbearbeiter.

1.3. Ziele

In der Aufgabenstellung der Arbeit wurden folgende Ziele definiert:

- Evaluation von Google Fusion Talbes in Kombination mit Google Maps u.a. mit Blick auf deren Funktionalität, Anwendbarkeit, Zuverlässigkeit und Performance
- Tutorial für Übungen auf Bachelor/Master-Stufe über die Verwendung von Google Fusion Tables (inkl. Video)
- Entwurf und Dokumentation einer GIS-Architektur, welche einerseits Cloud Services (am Beispiel von Fusion Talbes) andererseits die Geodaten- und Serviceinfrastruktur einer Organisation integriert oder migriert.
- Analyse verschiedener Use Cases wobei einer davon als Prototyp einer Webapplikation (voll) implementiert werden soll.
- Implementierung und Bewertung von verschiedenen cloudbasierten GIS-Prototypen unter Verwendung der Google Fusion Tables API
  - Prototyp(en) aus Use Case-Evaluation (oben). Dabei sollen v.a. auch die Geometrietypen Linestring und Polygon berücksichtigt werden.
  - Prototyp einer Datenerfassung/Verwaltung am Beispiel eines Point-of-Interest (POI) Layers mit grossen Datenmengen
- Prototyping für zukünftige (GIS-) Kollaborationsplattformen. Es soll aufgezeigt werden, wie sich bestehende Konzepte verbessern lassen oder weiterentwickeln könnten (z.B. www.mysg.ch/locations oder http://ch.tilllate.com/de/locations).

1.4. Rahmenbedingung

- Es gelten die Rahmenbedingungen, Vorgaben und Termine der HSR
- Die Projektabwicklung orientiert sich an einer iterativen, agilen Vorgehensweise. Als Vorgabe dient dabei Scrum, wobei bedingt durch das kleine Projektteam gewisse Vereinfachungen vorgenommen werden. Meilensteine werden bezüglich Termin und Inhalt mit dem verantwortlichen Dozenten und dem Projektpartner vereinbart.
- Die Kommunikation in der Projektgruppe, in der Dokumentation und an den Präsentationen erfolgt in Deutsch.
- Eine Prototypen-Website ist in HTML/JavaScript zu implementieren und sollte auf verschiedenen Plattformen lauffähig sein.
- Realisierung eines Videos.

1.5. Vorgehen

Die Arbeit umfasst zwei Themenbereiche. Einen theoretischen Teil in dem untersucht wird, inwiefern Google Fusion Tables eine Konkurrenz für bestehende proprietäre GIS-Lösungen darstellt. Als zweiten Teil sollen verschiedene Anwendungsfälle mit Google Fusion Tables als Prototypen nachgebaut werden, wobei einer davon als mobile Webapplikation ausprogrammiert werden soll.

1.5.1. Potential von Google Fusion Tables

In einem theoretischen Teil sollen Erkenntnisse gewonnen werden, inwiefern sich bestehende GIS-Lösungen mit Google Fusion Tables ablösen lassen.

Dies soll am Beispiel eines Migrationsszenarios festgestellt werden. Die bestehenden GIS-Daten sollen möglichst einfach Exportiert und anschliessend in eine Google Fusion Tables-Datenbank importiert werden. Dafür werden die geeigneten Formate zur Übertragung der Daten evaluiert.

Da sich die Datenstrukturen von bestehenden Lösungen stark unterscheiden können, ist es nicht das Ziel eine Schritt-für-Schritt Anleitung zu erstellen. Primär geht es darum verschiedene Migrationslösungen zu entwickeln und miteinander zu vergleichen.

Dadurch soll das gegenwärtige Potential von Google Fusion Tables abgeschätzt werden. Diese Erkenntnis soll GIS-Lösungsprovidern einen Überblick verschaffen, inwiefern Google Fusion Tables bereits als einen möglichen Konkurrenten angesehen werden muss.

1.5.2. Prototypen

Der beschriebene theoretische Teil soll anschliessend durch verschiedene Prototypen belegt werden. Es soll versucht werden mehrere Standard-Anwendungsfälle im GIS-Bereich mittels Google Fusion Tables zu realisieren. Diese Prototypen sollen hauptsächlich als Webclients entwickelt werden.

Ein Anwendungsfall soll zusätzlich als vollwertige Webapplikation für Mobilgeräte ausprogrammiert werden.

1.6. Aufbau der Arbeit

1.7. Stand der Technik

Der Neuste

1.8. Bewertung (Evaluation)

1.9. Vision/Umsetzungskonzept

1.10. Resultate der Arbeit

1.11. Schlussfolgerungen

1.12. Ausblick

Teil II.

Projektdokumentation

2. Einführung in Google Fusion Tables

2.1. SQL API

Das SQL API bietet eine REST-Schnittstelle mit welcher man mit SQL-ähnlichen Befehlen Daten aus Google Fusion Tables abfragen oder verändern kann. Sie verfügt bereits über eine grosse Palette an möglichen Befehlen.

Befehl	Beschreibung
SHOW TABLES	Abfrage aller Tabellen des angemeldeten Benutzers
DESCRIBE	Bezeichnung und Datentypen aller Spalten in einer Tabelle
CREATE TABLE	Erstellen einer neuen Tabelle
CREATE VIEW	Erstellen einer View auf Grundlage einer bestehenden Tabelle
SELECT	Selektieren von Daten einer Tabelle
INSERT	Neue Zeile zu einer Tabelle hinzufügen
UPDATE	Daten in einer Tabelle verändern
DELETE	Daten aus einer Tabelle löschen
DROP TABLE	Löschen einer Tabelle

2.1.1. Client Libraries

Google bietet zum API bereits auch Client Libraries in den Sprachen PHP und Phyton an. Da unserer Applikation aber möglichst nur in Javascript implementiert werden soll erstellten wir uns eine Javascript Library zur Verwendung des SQL APIs.

Durch die Same origin policy[1], welche es uns daran hinderte AJAX-Requests direkt auf das Google API abzusetzen, mussten wir zuerst nach Lösungen für dieses Problem suchen. Wir wollten es verhindern einen PHP-Server dazwischen zu schalten, welcher uns die Abfragen abnimmt.

So fanden wir in den Google Groups ein inoffizielles JSONP API, welches es erlaubt

[1] Die Same-Origin-Policy (SOP) ist ein Sicherheitskonzept, das es JavaScript und ActionScript nur dann erlaubt, auf Objekte einer anderen Webseite zuzugreifen, wenn sie aus derselben Quelle (Origin) stammen.[1]

AJAX-Requests auch über die eigene Domäne hinweg zu senden.

2.2. Geocodierung

Ein grosser Vorteil der Google Fusion Tables ist die automatische Geocodierung von Standortdaten. Sobald eine neue Zeile zu einer Tabelle hinzugefügt wird, werden alle Zellen vom Typ *Location* einem eindeutigen Standort auf der Karte zugewiesen. Ist dies nicht möglich, da beispielsweise eine Adresse in mehreren Orten vorkommen kann, bleibt die Zelle gelb hinterlegt.

name	description	population	location
Springfield	gelbe Stadt	100000	Springfield

Diese geocodierten Standorte werden in der Tabelle hinterlegt sind aber mit den SQL API nicht selektierbar. Man müsste also für jede Zeile die man als Resultat erhält die Geocodierung manuell vornehmen, was sich negativ auf die Ladezeit der Karte auswirken würde. Es gibt verschiedene Dienste, welche eine solche Geocodierung von Standortdaten anbieten. Die meisten davon haben aber eine Begrenzung der möglichen Anfragen pro Tag.

Anbieter	Anfragen pro Tag	URL
Google Maps Geocoding API	2500	`https://developers.google.com/maps/documentation/geocoding/?hl=de`
Yahoo! PlaceFinder API	50000	`http://developer.yahoo.com/geo/placefinder/`
MapQuest Geocoding API	keine Begrenzung	`http://developer.mapquest.com/web/products/dev-services/geocoding-ws`

Wie man sieht erreicht man mit diesen Diensten beim Arbeiten mit grossen Datenmengen schnell die Grenzen.

2.3. Google Maps API - FusionTablesLayer

Google bietet von Haus aus aber bereits eine Fusion Table-Integration im Google Maps API V3 an. Damit ist es möglich Fusion Tables als eigenständige Layer direkt auf der Karte darzustellen. Die Möglichkeiten dieser Layer sind noch stark eingeschränkt aber die grundlegenden Funktionalitäten für das Arbeiten mit Geodaten sind bereits vorhanden.

So ist es möglich Abfragen mit WHERE-Conditions einzuschränken oder die Stile des Layers selbst zu bestimmen. Man kann beispielsweise Flächen mit Zeckengebieten je nach Intensität des Befalls anders einfärben.

2.3.1. Spatial-Queries

Zudem bieten die FusionTablesLayer bereits eine Reihe von speziellen ortsabhängigen Abfrage-Möglichkeiten.

Spatial Keyword	Beschreibung
ST_INTERSECTS(<location_column>, <geometry>)	Kann als Bedingung in der WHERE-Klausel des Statements verwendet werden. Liefert alle Zeilen zurück, welche sich innerhalb der definierten Geometrie *<geometry>* befinden. • Als *<location_column>* muss eine Spalte der Tabelle angegeben werden, welche den Typ *location* hat. • Als *<geometry>* kann entweder ein *CIRCLE* oder ein *RECTANGLE* verwendet werden. *Hinweis: ST_INTERSECTS und ST_DISTANCE dürfen nicht zusammen im gleichen Statement verwendet werden.*
ST_DISTANCE(<location_column>, <coordinate>)	Kann als Bedingung in der ORDER BY-Klausel des Statements verwendet werden. Liefert die Datensätze sortiert nach der Distanz zur angegebenen Koordinate *<coordinate>* zurück. • Als *<location_column>* muss eine Spalte der Tabelle angegeben werden, welche den Typ *location* hat. • Die *<coordinate>* stellt die Koordinate dar, zu welcher der Abstand gemessen werden soll. *Hinweis: ST_INTERSECTS und ST_DISTANCE dürfen nicht zusammen im gleichen Statement verwendet werden.*
CIRCLE(<coordinate>, <radius>)	Wird verwendet, um einen Kreis von der angegebenen Koordinate *coordinate* mit den Radius *radius* zu erhalten.
POLYGON(<coordinate_1>, <coordinate_2>, ...)	Wird verwendet um ein Polygon bestehend aus den angegebenen Koordinaten *coordinate_x* zu erhalten.
RECTANGLE(<coordinate_1>, <coordinate_2>)	Wird verwendet um ein Rechteck mit den Ecken *coordinate_1* (links oben) und *coordinate_2* (rechts unten) zu erhalten.

2.3.2. Karten-Stile

Die FusionTablesLayer bieten ein abfragebasiertes Styling der Ebene an. Damit ist es möglich Flächen oder Linien farblich hervorzuheben oder Custom-Icons für Markierung zu verwenden. Zu jedem Stil kann man eine Einschränkung festlegen, welche bestimmt, ob dieser für den aktuellen Datensatz angewendet wird oder nicht. Damit ist es möglich die Ebene stark zu personalisieren.

Beispiel eines Stils:

```
styles: [{
  polygonOptions: {
    fillColor: "#00FF00", // gruen
    fillOpacity: 0.3
  }
}, {
  where: "birds > 300",
  polygonOptions: {
    fillColor: "#0000FF" // blau
  }
}]
```

In diesem Beispiel werden alle Polygone der Tabelle, welche in der Spalte *birds* eine Zahl grösser als 300 eingetragen haben, *blau* eingefärbt. Die restlichen Polygone erhalten eine *grüne* Färbung mit einer Deckkraft von 30%.

Einschränkungen

Das Google Maps API hat momentan folgende Einschränkungen bezüglich den Ebenenstilen definiert.

- Auf einer Karte können maximal 5 FusionTablesLayer gleichzeitig angezeigt werden
- Stile können dabei nur für eine dieser Ebenen angewendet werden
- Zudem dürfen für diese Ebene maximal 5 Stile definiert sein

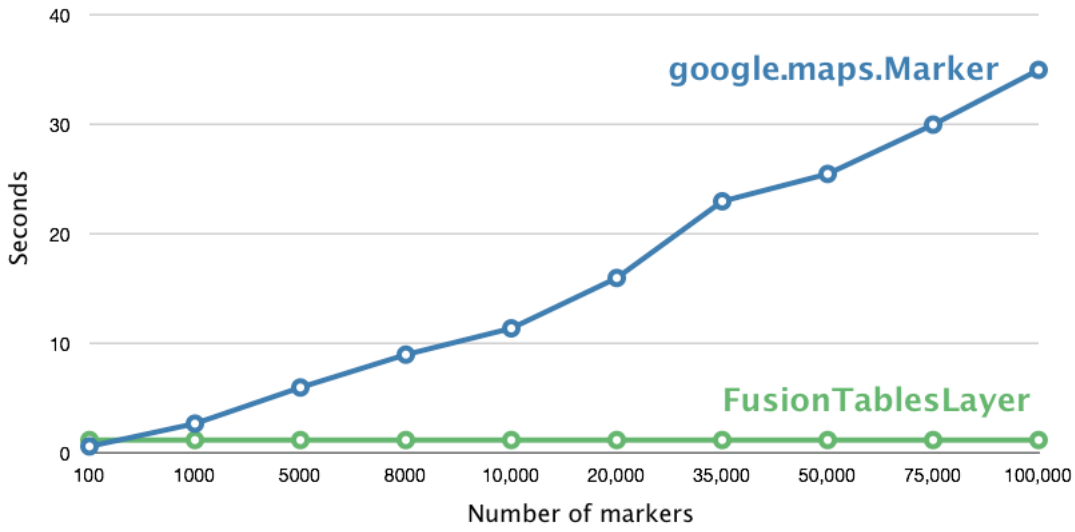
2.3.3. Heatmaps

Ein weiteres mächtiges Feature ist zudem die Möglichkeit die Daten der Tabelle direkt als Heatmap darzustellen. Dabei werden die Daten automatisch nach der Häufigkeit der

Vorkommnisse an einem Ort anders eingefärbt. Der verwendete Farbverlauf geht dabei von Grün (für wenig Daten) bis Rot (für viele Daten).

2.3.4. Vorteil

Der grösste Vorteil der Fusion Table-Ebenen findet man aber eher darin, dass die Geocodierung der Standort-Daten direkt aus der Tabelle gelesen wird und nicht manuell abgefragt werden muss. Dadurch kann die Ebene komplett auf den Servern von Google aufbereitet werden. Der Client muss die erhaltenen Daten lediglich noch darstellen. Der Vorteil davon wird durch das folgende Diagramm schnell ersichtlich.



Die Zeit für das Rendering der Karte bleibt demnach bei der Verwendung von Fusion Table-Ebenen konstant und somit unabhängig von der Anzahl Markierungen, welche gesetzt werden müssen. Der Rechenaufwand, der für das Erstellen der Javascript Marker-Objekte verwendet werden müsste, wird direkt von den Google Servern übernommen und das Resultat als Bild zum Client gesendet. Daraus resultiert die konstante Zeit, welche für die Anfrage zum Server und für das Senden der Antwort zum Client verwendet wird.

2.3.5. Nachteil

Ein grosser Nachteil dieser Fusion Table-Ebenen besteht aber darin, dass die verwendeten Fusion Tables als *öffentlich* markiert sein müssen. Sprich jeder kann die Tabellen anzeigen oder auslesen. Es ist also nicht möglich eine Tabelle mit sensiblen Daten als Fusion Table-Ebene darzustellen.

Von Google wird zur Lösung dieses Problems aber folgendes Vorgehen vorgeschlagen: Man kann für Tabellen mit sensiblen Inhalten eine View erstellen, welche lediglich die öffentlichen Spalten und Zeilen selektiert. Diese View könnte man dann als *öffentlich* markieren und in einer Fusion Table-Ebene verwenden.

Es bleibt die Frage offen, wie es möglich ist sensible Daten trotzdem in eine Ebene einzubringen.

2.3.6. Einschränkungen

Einige Einschränkungen sind aber auch bei der Verwendung der Fusion Tables-Ebenen zu beachten. Wie bereits erwähnt ist es nur möglich *öffentliche* Fusion Tables als FusionTablesLayer darzustellen.

Zusätzlich ist die Anzahl an Markierungen, die auf einer Fusion Table-Ebene angezeigt werden können, auf 100000 Stück beschränkt. Diese Zahl sollte grundsätzlich aber für die meisten Anwendungsfälle gut ausreichen.

Bei einer grösseren Anzahl von Daten sollte man sich bezüglich Übersicht zuerst die Überlegung machen, ob es Sinn macht alle Markierungen auf einmal darzustellen. Man könnte beispielsweise nur die Daten in einem gegebenen Umkreis darstellen. Auch die Performance nimmt bei einer grossen Anzahl von Markierungen schnell ab.

2.4. Google Fusion Table Javascript Library (gftlib-js)

Die Google Fusion Table Javascript Library vereinfacht die Kommunikation mit dem Google Fusion Table SQL API. Sie hilft dabei SQL-Queries zu erstellen und per AJAX an das API zu versenden.

Zur Erstellung der AJAX-Requests werden die $.get()- und $.post()-Helpermethoden der jQuery Library in der Version 1.7.1 (Minified) verwendet.

2.4.1. Abhängikeiten

Library	Version
jQuery	1.7.1-min

2.4.2. Methoden

Methode	Beschreibung	Parameter
execSql(callback, query)	Führt einen SQL-Befehl	callback (Funktion): Callback-Methode welche nach Beendigung der Methode aufgerufen wird. query (String): SQL-Query
execSelect(callback, options)	Führt einen SQL-Abfrage aus	callback (Funktion): Callback-Methode welche nach Beendigung der Methode aufgerufen wird. query (String): SQL-Query
convertToObject(gftData)	Konvertiert das Resultat einer Abfrage in sprechende Objekte	•

Use Cases

Unsere Hauptaufgabe war es nun verschiedene reale GIS-Anwendungsfälle zu finden und diese mit Google Fusion Tables umzusetzen. Daraus sollen Erkenntnisse betreffend dessen Potential gefunden werden.

3. Use Case 1: WorldData Explorer

3.1. Einführung

Im ersten Use Case geht es hauptsächlich um die Anzeige grosser Datenmengen auf der Karte. Dazu importieren wir bestehende Datenbestände in die Google Fusion Tables und visualisieren diese mittels Google Maps API auf der Karte.

3.1.1. Ziel

Es sollen länderspezifische Daten auf einer Weltkarte angezeigt werden. Diese Daten lassen sich dann mittels einer Zeitachse pro Jahr visualisieren. Daraus können beispielsweise Zusammenhänge von verschiedenen Datenkategorien gefunden werden.

Um die Daten pro Land zu visualisieren, werden zuerst die Landesgrenzen als Geometrie-Datensätze in eine separate Fusion Tabelle importiert. In eine andere Tabelle werden dann die ganzen Daten importiert unterteilt nach Land und Jahr.

ERD

3.1.2. Datenqullen

3.2. Anforderungsspezifikation

3.3. Analyse

3.4. Design

3.5. Implementation

3.6. Test

3.7. Resultate

3.8. Weiterentwicklung

3.9. Benutzerdokumentation

Teil III.

Projektmanagement

4. Projektmanagement

4.1. Allgemeines

4.2. Projektmanagement

4.3. Projektmonitoring

Teil IV.

Anhänge

Glossar funktioniert nicht korrekt

Literaturverzeichnis

[1] Wikipedia. Same-origin-policy — wikipedia, die freie enzyklopädie, 2012. [Online; Stand 18. März 2012].