Gruppe: SWP12-2 Verfasser: Oliver Schurig
Datum: 17.04.2012

Projektvertrag

der Gruppe 12-2

0. Inhaltsverzeichnis

Projektvertrag

Gruppe: SWP12-2 Verfasser: Oliver Schurig
Datum: 17.04.2012

1. Verwendete Kürzel

In diesem Vertrag werden einige Namenskürzel verwendet. Zum schnelleren Navigieren wurde folgendes Verzeichnis erstellt:

1.1. LinkSpec

LinkSpec ist eine Abkürzung für Link Specification.
Nach dem LIMES-Standard sind Link Specifications definiert als XLM-Konfigurationsdateien, deren Aufbau einer gewissen, in LIMES festgelegten, Struktur entspricht, sodass über die darin enthaltenen Informationen RDF-Tripel-Paare verlinkt werden können.

1.2. Matches

Bei Matches oder Matching-Ausgaben handelt es sich um die Ausgabe eines „Matchings“. So bezeichnet man das Verwenden einer LinkSpec um die RDF-Tripel zweier Wissensbasen abzugleichen und Paare solcher Tripel festzulegen. Das entsprechende Match ist die Ausgabe dieses Vorganges, in der die Paare in einer Liste dargestellt werden. Zum besseren Verständnis dieser Tripel, gelangt durch anklicken eines jeden von ihnen über einen Link auf die Quell-Seite dieses Tripels und kann dort nachlesen, worum es sich dabei genau handelt.

1.3. Play!

„Play!“, oder einfach nur „Play“, bezieht sich auf das zur Programmierung verwendete Framework namens Play!-Framework. Weitere Informationen hierzu gibt es auf der Homepage des Play!-Frameworks: http://www.playframework.org/.

1.4. VCS

VCS steht als Abkürzung für das englische Wort Version Control System. Darunter versteht man eine Plattform, mittels derer der Quellcode für örtlich verteilte Programmierer zugänglich gemacht werden kann. Das Versionskontrollsystem bietet die Möglichkeit, den Quellcode in seiner aktuellsten Version herunterzuladen, ihn in Teilen zu bearbeiten und die aktualisierten Teile erneut in das System hochzuladen. Vorteile bieten sich neben der Möglichkeit, räumlich verteilt an einem Projekt arbeiten zu können, vor allem dadurch, dass auch gleichzeitig und dynamisch an einem Projekt gearbeitet werden kann und Änderungen im System verfolgt werden, indem vermerkt wird, wann Änderungen von wem und an welchen Teilen des Quellcodes vorgenommen wurden.

1.5. JDBC

JDBC ist ein Kürzel für Java Database Connectivity. Es stellt eine Schnittstelle der Java-Plattform dar und bietet einheitliche Schnittstellen zu verschiedenen Datenbanken verschiedener Hersteller an. Dabei ist JDBC jedoch speziell auf relationale Datenbanken ausgerichtet.

Projektvertrag

Gruppe: SWP12-2 Verfasser: Oliver Schurig
Datum: 17.04.2012

1.6. User und Gäste

In Anlehnung an das User-Management wird es zwei Nutzergruppen geben. Ist der Nutzer eingeloggt, so ist die Rede vom User. Ist er nicht eingeloggt und nimmt die eingeschränkte Funktionalität in Anspruch, so sprechen wir vom Gast.

1.7. CLOB

CLOB steht für Character Large Object. Dies ist ein gängiges Speicherformat für lange Zeichenketten, die in relationalen Datenbanken abgelegt werden.

2. Allgemeines zu Aufgabe, Arbeitsablauf und Zielsetzung

Die Zielsetzung des zu bearbeitenden Projekts ist die Erstellung einer Web-GUI für die bereits bestehende Applikation LIMES. Sie soll dazu dienen, das Arbeiten mit LIMES zum einen einfacher und übersichtlicher zu gestalten, es andererseits aber auch facettenreicher zu gestalten, beispielsweise durch die Einführung eines User-Managements, sodass die von LIMES angebotenen Funktionen den verschiedenen Nutzern oder Nutzergruppen verwehrt oder gutgeschrieben werden. Die konkret benannten Ziele dieses Prozesses folgen allerdings in den nächsten Unterpunkten.
Der zeitliche Rahmen des Projektes ist ein nicht ganz 6-monatiger Arbeitsprozess, beginnend am 24. November 2011, Deadline ist der 14. Mai 2012. Die letztendliche Abnahme des Projektes findet am 21. Mai 2012 statt.
Die Lokalität der Projektarbeit jedes Arbeitsteilnehmers ist relativ frei wählbar. Durch das Vorhandensein eines VCS sind die Projektteilnehmer weitestgehend unabhängig. Bearbeitungszeiten sind von jedem insofern frei wählbar, dass die gruppenintern gesetzten Deadlines für Teilaufgaben eingehalten werden. So kann jedes Projektmitglied für sich entscheiden, wann und wo es seine Arbeit verrichten will. Die Ausnahme zu diesem Vorgehen bilden lediglich die vereinzelten Meilensteintreffen mit dem Kunden, um den Arbeitsfortschritt aufzuzeigen, zu denen jeder in Leipzig anwesend sein soll. Zu den Kundentreffen im frühen Zeitraum der Projektzeit muss nicht jeder Projektteilnehmer anwesend sein, dennoch wurde es in der Gruppe entschieden, dass ein jeder auch zu diesen Treffen kommen sollte, soweit es möglich ist. So ist sichergestellt, dass ein jeder den Tatbestand des Projektes kennt, versteht, und somit die Fehleranfälligkeit des Projektes reduziert wird.
Durch die Komplexität der Aufgabe bietet es sich an, zur Programmierung ein Framework hinzuzuziehen, um in kürzerer Zeit eine bessere Qualität der Web-GUI erzielen zu können. Nach diversen Recherchen zu den zahlreichen Frameworks und Vergleichen zu ihrer Güte und Verwendbarkeit, fiel die Wahl auf Play!. Play! zeichnet sich dadurch aus, dass es relativ überschaubar und einfach zu handhaben ist. Es unterstützt die Einbindung von HTML-Pages in Java-Quellcode, den Betrieb eines Servers sowie das Aufstellen von Datenbanken. Die Datenbank wird vor allem für das User-Management sowie die Verwaltung der Sichtbarkeit von LinkSpecs benötigt und als Derby-Datenbank realisiert werden. Die Wahl fällt wegen ihrer Plattformunabhängigkeit auf Derby und kann einfach mittels JDBC durch Java-Quellcode angesprochen werden.

3. Grafische Oberfläche und Tab-Management

Allem voran geht die Entwicklung einer Grafischen Oberfläche – sie ist ein zentraler Aspekt des gesamten Projekts. Hier werden Ein- sowie Ausgaben getätigt und in angenehmer, nutzerfreundlicher Weise dargestellt. Im Zuge dessen ist eine zu implementierende Vorgabe des Projekts, ein Tab-System. Innerhalb des Browserfensters läuft das Geschehen ab, dem Nutzer werden die Funktionen präsentiert und hier kann er Arbeiten. Innerhalb des Screens wird ein gesonderter Teil wiederum als Screen eingeteilt sein. Werden neue LinkSpecs geöffnet, so werden diese LinkSpecs in diesem kleineren Screen geöffnet und bekommen einen Karteireiter. Sind mehrere solcher Tabs geöffnet, so muss ein Tab gewählt sein, der Aktiv ist. Zum Aktiven Tab wird eine zweite Karteireiterleiste unter der ersten dargestellt, in der weitere Tabs dargestellt sind, in denen verschiedene Eigenschaften dieser LinkSpec eingesehen werden können. Beispielsweise kann man sich die XML-Datei des aktiven LinkSpecs anschauen, man kann den LinkSpec weiter bearbeiten, in einem Bearbeitungs-Tab oder in einem weiteren Tab die Matching-Ergebnisse einsehen, die mittels dieses LinkSpecs generiert wurden. So entsteht ein 2-Ebenen-Tab-System. Jedem Karteireiter wird ein kleiner Schließen-Button hinzugefügt, so dass zu jeder Zeit dieser Tab aus der Liste der Karteireiter entfernt werden kann.

4. User-Management

Das User-Management ist ein wichtiger Bestandteil der Web-GUI. Konkret heißt das, dass es zwei Arten Nutzer gibt: Gäste und User. Beim ersten öffnen der Web-GUI kann man bereits agieren, man befindet sich im Gastmodus. In diesem Modus sind die Möglichkeiten der Interaktion beschränkt. Zur Auswahl steht das Erstellen von LinkSpecs, sowie das Laden von als öffentlich gekennzeichneten, in der Datenbank gespeicherten LinkSpecs. Diese LinkSpecs kann der Gast nutzen, um ein Matching zu generieren. Dieses Matching kann der Gast dann einsehen, und bei Bedarf auch auf seine lokale Maschine als Datei herunterladen.
Zu jeder Zeit steht in der oberen rechten Ecke des Browserfensters eine Fläche, in der man sich über Eingabe von E-Mail und Passwort einloggen kann. Ist man noch nicht registriert, so kann man über den „Registrieren“-Button an selbiger Stelle ein Konto erstellen, sodass die Daten in der Datenbank aufgenommen werden und man anschließend eingeloggt ist.
Der nun eingeloggte User, hat immernoch alle Möglichkeiten, die sich zuvor dem Gast geboten haben. Hinzu kommen aber die Möglichkeiten, LinkSpecs in die Datenbank hochzuladen und sie, entweder als privat oder als öffentlich gespeichert, dort abzulegen. Außerdem hat der User Zugriff auf den Learning-Mechanismus, der im Folgenden noch erläutert wird.
Ist man eingeloggt, so wird die Schaltfläche zum Einloggen durch einen Button zum ausloggen ersetzt.
Unterstützt wird der Prozess des User-Managements von Play!. Mittels des Tools Deadbolt, das in Play! enthalten ist, lässt sich ein Rollensystem entwickeln. Über Java-Klassen kann hiermit festgelegt werden, welche Nutzer welche Rollen bekommen, und welche Funktionalitäten sie damit für sich in Anspruch nehmen dürfen.

5. Sichtbarkeit und Speichern von LinkSpecs für unterschiedliche User-Gruppen

Auch die Sichtbarkeit von LinkSpecs ist ein wichtiger Punkt in der Entwicklung des Web-GUI. Entstehen durch Aktionen der User neue Linkspecs, die in der Datenbank abgelegt werden sollen, so soll der User, der diese speichert wählen können, ob diese LinkSpec nur für den User selbst besteht, also privat abgelegt wird, so dass nur er oder sie selbst Zugriff darauf hat, oder ob sie so gespeichert werden, dass sie der Öffentlichkeit zugänglich sind. Nur User können LinkSpecs in der Datenbank speichern, allerdings können sowohl User, als auch Gäste LinkSpecs laden. Gäste haben nur Zugriff auf öffentliche LinkSpecs. User hingegen haben Zugriff auf öffentliche LinkSpecs, sowie solche LinkSpecs, die sie zuvor als privat gespeichert haben.
Diese Aufteilung wird auch mittels der Datenbank gelöst. In der relationalen Datenbank werden die LinkSpecs in einer Relation abgelegt. Die Relation wird ein bool'sches Attribut der Art „private?“enthalten, das angibt, ob der LinkSpec privat abgespeichert wurde oder nicht. Über ein weiteres Attribut wird dann die Zuordnung der privaten LinkSpecs zum „Besitzer“ abgehandelt. Bei öffentlichen LinkSpecs bleibt dieses Attribut ein NULL-Wert. Der Upload einer Datei an sich kann durch die Java-Bibliothek „FileUpload“ von Apache realisiert werden. Diese Bibliothek stellt Möglichkeiten bereit um eine „RFC 1867“ Anfrage zu bearbeiten.
Weiterhin ist zu sagen, dass die LinkSpecs zur Speicherung in ein CLOB-Format konvertiert werden. Die Standardgröße eines solchen CLOB-Feldes beträgt 2GB, sollte keine spezielle Größe angegeben sein. Die CLOB-Konvertierung wird von JDBC unterstützt. Auf diese Weise wird die Datenbank zwar umfangreich, bleibt aber dennoch durch die nötigen Constraints sicher.

6. Bearbeiten von LinkSpecs

Es besteht die Möglichkeit, einen LinkSpec zu kreieren. So wird dem Nutzer ein Default-LinkSpec zur Verfügung gestellt, dessen Eigenschaften der Nutzer über eine grafische Oberfläche einfacher modellieren kann, bis der LinkSpec den vom Nutzer gewünschten Zustand erreicht hat. Da LinkSpecs in einem solchen Zustand gespeichert werden können, ist es auch wünschenswert, einen gespeicherten LinkSpec zu gegebener Zeit zu laden und weiter zu modellieren. Hierzu ist eine Konvertierung vom CLOB-Format zurück in das XML-Format nötig. Dazu ist eine Java-Klasse nötig, die entsprechende Transformationen durchführt. Auch hier bietet JDBC entsprechende Hilfe an.

7. Ausgabe gefundener Matches als Datei

Die Ausgabe von Limes sind die gefundenen Matches, welche in der Datei accepted.nt gespeichert werden. Diese Datei wird durch den LinkSpec fest vorgegeben. Im Kopfteil der accepted.nt-Datei befinden sich die genutzten Präfixe und im Hauptteil die gefundenen Matches. Dabei wird der durch Limes vorgegebene strukturelle Aufbau der Matches nicht verändert. Somit werden sie in folgender Form gesichert:

<URI-Link_Source> Relation <URI-Link_Target>.

Dieses Matching wird als Untertab des LinkSpecs geöffnet, zu dem es gehört. Es besteht außerdem die Möglichkeit, diese Datei über einen Download-Button herunterzuladen und lokal abzuspeichern. Eine Speicherung in der Datenbank ist im Projekt nicht vorgesehen, wäre jedoch mit dem Stand der Dinge gegen Ende des Projektes ohne großen Aufwand realisierbar.

8. Learning von LinkSpecs

Das Lernen von LinkSpecs erfolgt über einen eigenen Tab in der WebGUI. Diese Funktionalität steht nur Usern zur Verfügung. Am Anfang hat der Nutzer die Möglichkeit sich für eine Learning-Methode zu entscheiden (Active oder Batch Learning). Unabhängig von der genutzten Methode wird der Benutzer anschließend aufgefordert den Ausgangs-LinkSpec anzugeben. Diese wird dann über einen Pfad der dem User zur Verfügung stehenden LinkSpecs ausgewählt. Wie bei der graphischen Oberfläche wird auch hier viel Wert auf die Erweiterbarkeit und Modularisierung gelegt. Dabei wird versucht Funktionen so allgemein wie möglich zu halten um, unter anderem, die Hinzunahme weiterer Lernmethoden zu ermöglichen.

8.1. Active Learning

Beim Active Learning kann der Nutzer die Anzahl der Iterationen über ein Textfeld angeben, wobei verschiedene Voreinstellungen möglich sind. Des weiteren kann er noch die Anzahl der Matches angeben, die nach jedem Durchlauf zu überprüfen sind. Nach der Übergabe des LinkSpecs erscheint eine erste Liste von Matches zwischen Instanzen aus Source und Target. Der Nutzer kann nun über Checkboxen bei jedem potentiellen Match auswählen, ob es sich dabei um ein Match handelt oder nicht. Dieser Vorgang wird der Anzahl der Iterationen entsprechend oft wiederholt. Als Ergebnis wird dem Nutzer der angepasste LinkSpec als XML-Datei mit veränderten Treshold-Werten übergeben, welche er auf dem Server, wie gewohnt als privat oder auch öffentlich, speichern kann.

8.2. Batch Learning

Ähnlich dem Active Learning kann der Nutzer hier die Anzahl der Matches angeben, die er zu überprüfen wünscht. Jedoch wird eine Mindestanzahl an zu überprüfenden Matches verlangt. Ein optimaler Wert für diesen Prozess wird sich während der Implementierungsarbeit zeigen. Da das Batch Learning nicht auf Iterationen beruht und dieser Prozess viel automatisierter abläuft, muss hierbei nichts weiter beachtet werden. Auch hier erhält der Nutzer nach Abschluss des Lernens den optimierten LinkSpec, den er wiederum wie gehabt in der Datenbank speichern kann.