Projektangebot Link Discovery

Projektverantwortlicher: Sascha Hahne

9. April 2015

1 Allgemeines

Das Projekt umfasst das Entwickeln eines GUI Systems auf der Basis von Java um eine Benutzeroberfläche für das Programm Limes zu bieten. Wichtig dabei ist eine leichte Bedinbarkeit welche keine Vorkenntnisse über Limes selbst voraussetzt. Zur Bewahrung der Übersichtlichkeit ist zu jedem gegebenen Zeitpunkt wärend der Benutzung nur eine limitierte Auswahl an Aktionen möglich. Je nachdem welchen Wunsch der Nutzer an das System stellt wird er durch ein Menü geführt das alle Funktionaliäten für den gewählten Pfad bereit stellt. Ein eingeschlagener Pfad führt stringent zu einer gezielten Berechnung für eine bestimme Aufgabe. Selbstverständlich sind alle Entscheidungen widerrufbar.

2 Produktübersicht

Limes benötigt zur Berechnung Linkspecs auf XML Basis. Diese Linkspecs beschreiben und definieren welche INformationen Limes miteinander verknüfpen soll und welche Daten von dem Nutzer gewünscht werden. Das GUI System ermöglicht zwei Hauptpfade, um LIMES zu starten und mit den gewünschten Linkspecs zu versorgen. Zum einen der direkte Weg indem ein XML Dokument, welches aus fertig definierten LInkspecs besteht, eingelesen wird. Dafür ist es notwendig dass die Linkspecs syntaktisch korrekt sind. Nach dem Einlesen kann LImes direkt ausgeführt werden. Der zweite mögliche Weg ist das Erstellen der Linkspecs durch einen von der Oberfläche gestellten Assistenten. Dieser besteht aus mehreren separaten Oberflächen für Einstellungen der Target und Source sowie der Operanden und Metriken welche diese verbinden. Dabei wird gesichert das keine falschen Verknüpfungen verlangt werden und die Baumstruktur der Linkspecs strikt eingehalten wird.

3 Grundsätzliche Struktur und Entwurfsprinzipien

Das GUI System basiert auf dem 'Model-View-Controll' Prinzip mit der Besonderheit das mehr View- und Controllklassen benutzt werden als Modelklassen. Der Sinn dahinter ist dass einige View gezielt zur reinen zusätzlichen Auswahl von Parametern genutzt werden und diese dann an ein Model weiter gegeben werden. Für die Übergabe jedoch ein extra Model einzufügen ist überflüssig. Die Hauptaufgabe eines Models ist eine Berechnung in Hinblick auf Präsentation von Graphen.

4 Struktur- Entwurfsprinzipien einzelner Pakete

4.1 Semanticweb

Dieser Begriff beinhaltet eine Erweiterung des bestehenden Begriffes des klassischen Webs, um zusätzliche Einträge und Ergänzungen, die die bestehenden und neu hinzukommenden Daten eindeutiger, verständlicher und auswertbarer für Mensch und vor allem Maschine gestalten. Ziel ist es Daten nicht nur zu erfassen oder zu finden, sondern durch multiple Verknüpfungen einzelner Datenbausteine eine inhaltserfassende, verstehende Basis für Maschinen zu erzeugen. Um dies zu erreichen ist es notwendig die vorhandenen Informationen auszuwerten, neu zu strukturieren und für maschinelle Vorgänge interpretierbar zu machen. In diesem Projekt erfolgt die Strukturierung durch RDF und FXML / XML Dateien. In diesen sind bereits typische Tripel der notwendigen Neustrukturierung enthalten beziehungsweise werden darin umgesetzt.

4.2 LIMES, ConfigReader und die Metrik

Die wichtigste Klasse zur Steuerung von LIMES ist der ConfigReader. Dieser speichert nicht nur die für die Verwendung von LIMES notwendigen Parameter, sondert ist auch in der Lage, eine LinkSpec-Datei (sprich XML) einzulesen und zu verarbeiten. Auf die wichtigsten Dinge soll hier kurz eingegangen werden.

Der Contructor des ConfigReaders akzeptiert keine Parameter. Sollte eine LinkSpec-Datei vorliegen, so muss diese über eine Funktion eingelesen werden.

```
public boolean validateAndRead(String filePath)
```

Hierbei bezeichnet filePath den (vollständigen) Pfad zur Datei. Durch den booleschen Rückgabewert lässt sich überprüfen, ob die eingelesene Konfiguration gültig ist.

Informationen über die Endpoints sind als eigener Datentyp gespeichert. Sollen diese nachträglich geändert werden, so geschieht dies über die Funktion

```
public void processKBDescription(String kb, NodeList children)
```

Der String kb hat hierbei den Wert SSOURCEöder "TARGET", entsprechend der Endpointkonfiguration, die man ändern möchte. Die Liste children enthält sämtliche Parameter des Endpoints, das heißt ÏD", ËNDPOINT", "PROPERTY", usw.

Alle anderen Parameter sind als Strings gespeichert und können direkt manipuliert werden. Hierbei ist lediglich auf die Formatierung zu achten, insbesondere bei der Metrik. Dafür empfiehlt es sich, sich an den Beispiel-LinkSpecs zu orientieren.

4.3 SPARQL

Hierbei handelt es sich um Daten, die durch Ergänzungen vernetzt und erweitert wurden, um eine Interpretation durch Maschinen zu ermöglichen. Die Bildung von Tripeln ermöglicht es, diese durch die Abfragesprache SPARQL zu erfassen und weiterzuverarbeiten. Speziell für dieses Projekt bedeutet das, dass LIMES, Daten aus diesen Quellen verwenden und verwerten kann. Dies ermöglicht einen effizienteren Vergleich und höhere Trefferchanen. Für unsere GUI beinhaltet dies, die einfache und schnelle Verlinkung.

4.4 JavaFX

In der konkreten Implementierung der Software wird wie vorher erwähnt mit dem 'Model-View-Controller' Prinzip gearbeitet. Aus Gründen der Arbeitserleichterung wird in diesem Projekt zusätzlich der SceneBulider als Tool verwendet, welcher es ermöglicht schnell und einfach Oberflächen zu erstellen und zu ändern. Dies und der Aufbau von JavaFX haben auch zu Folge, dass als Views nicht eigene Klassen implementiert werden sondern FXML-Dateien vom Controller eingelesen und manipuliert werden um die Oberfläche zu gestalten. Dies bietet viel Flexibilität bei der Einarbeitung und Arbeitsteilung.

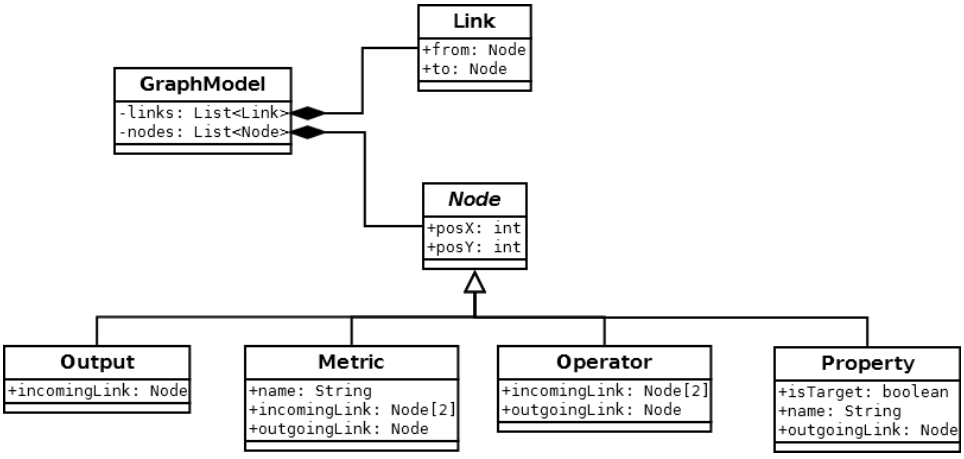

Abbildung 1: Modellierung des Datenmodells für die Graphen

5 Datenmodell

Intern verwaltet die GUI zwei Bereiche Einmal die Modellierung der Graphen mit den Metriken, Operatoren, Endpoints und deren Verbindungen und zum anderen die Resultate

5.1 Resultate

die Resultat werden aus einer externen .nt Datei gelesen. In dieser Datei sind die einzelnen Matches paar- und zeilenweise gespeichert. Das Format ist hierbei:
Sourcenode Targetnode Value
Source- und Targetnode sind URL auf die jeweiligen Datensätze in den Endpoints Der Value gibt den Wert der Auswertung des Graphen für das jeweilige Paar an.

5.2 Modellierung der Graphen

Hauptbestandteil der GUI ist die Modellierung der LIMES-Anfrage Graphen. Dabei werden die Nodes mit Position in der GUI, Typ und Namen gespeichert. Die Typen von Nodes sind hierbei: Output, Metrik, Operator und Endpoint. Die Nodes werden durch Edges verbunden, welche es im Nachhinein möglich machen aus der graphischen Modellierung den Anfrage String rekursiv auszuwerten.

6 Testkonzept

Während der Implementierungsphase werden einige Mitglieder des Teams sich ausschließlich auf die Qualitätssicherung und das Testen des Codes konzentrieren. Die Tests sind werden folgendermaßen durchgeführt

6.1 Usertests

Vor jedem Release werden Mitglieder die nicht mit der Implementierung betraut sind, simple Click-Tests durchführen um sicher zu stellen, dass die GUI stabil und zuverlässig ihren Dienst tut.

Desweiteren werden sich diese Mitglieder mit der Ausarbeitung eines Fragebogens beschäftigen, um die fertige Software am Ende durch Fachfremde Menschen evaluieren zu lassen. Während diesem Prozesses stehen diese Personen im ständigen Austausch mit dem Implementierungsteam um gegebenenfalls noch Design Änderungen für die GUI anzubringen.

6.2 Tests mit JUnit

Um die funktionale Sicherheit der Software sicherzustellen werden automatisierte Tests mit JUnit verfasst. Hierbei haben wir konkret vor das Framework TestFX zu benutzen, da es konrekt für die Tests von GUIs entwickelt wurde.

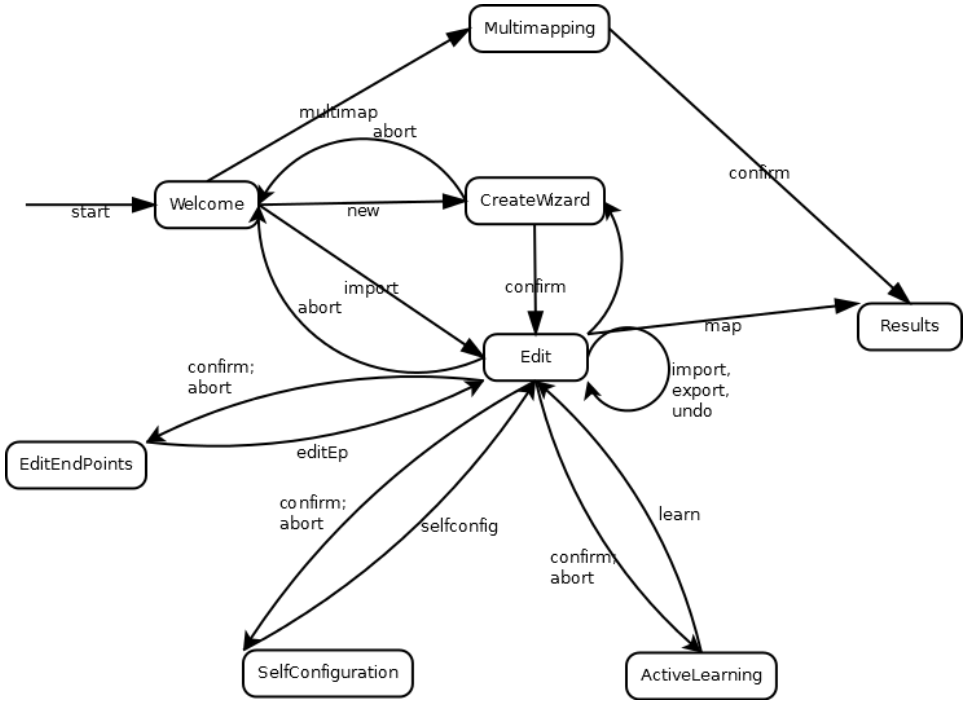

Abbildung 2: State-Diagramm der geplanten GUI

7 Appendix

Benchmarks: Im Rahmen des Vorprojektes wurden verschiedene Benchmarks angefertigt um die genaue Funktionsweise von Limes genauer zu verstehen. Diese wurden in einer extra ZIP-Datei bei der Abgabe mit versendet.

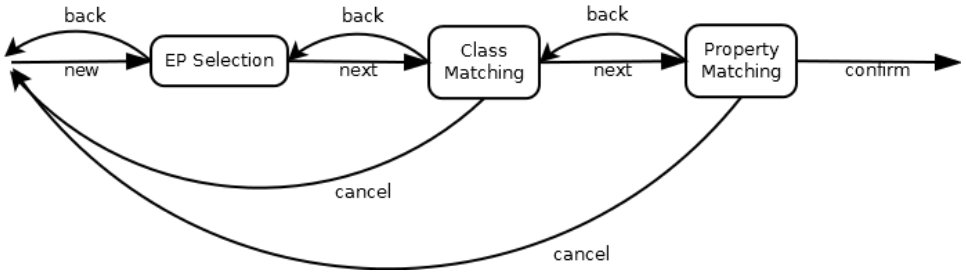

Abbildung 3: State-Diagramm des Assistenten für den Import und die Manipulation der Endpoints