Entwurfsbeschreibung

Gruppe dbp-17

Gliederung

1. Allgemeines
2. Produktbeschreibung
3. Grundsätzliche Struktur- und Entwurfsprinzipien
4. Struktur- und Entwurfsprinzipien einzelner Pakete
5. Datenmodell
6. Glossar

1. Allgemeines

Im Rahmen des Softwaretechnikpraktikums 2017 an der Universität Leipzig soll ein Programm entwickelt werden, welches zu in dbpedia vorliegenden Fakten die Provenienz ermittelt. Mit Provenienz sind hier Änderungen an Triplen, sowie deren Zeitpunkt, Autor und Revision gemeint. In dieser Entwurfsbeschreibung ist der Stand der Anwendung beschrieben wie er im Rahmen des finalen Release veröffentlicht wurde. Das Produkt liegt in diesem Release in vorlaeufig fertiger Form vor.

2. Produktübersicht

Das Produkt wird als Kommandozeilenanwendung ausgeliefert. Je nach den übergebenen Parametern wird nach Abschluss des Programmablaufs ein entsprechender Output im .tsv Format gespeichert. Dieser wird unter dem Punkt 5.1 genau spezifiziert.

3. Grundsätzliche Struktur- und Entwurfsprinzipien

3.1 Programmiersprache

Die gesamte Anwendung wird in Java 1.8 geschrieben.

3.2 Parallelisierung

Die Anwendung läuft teilweise parallel, wie genau wird im folgenden beschrieben. Im Prinzip sollen die Artikel, nachdem die für den jeweiligen Artikel relevanten Revisionsnummern extrahiert wurden, parallel verarbeitet werden. Um das umzusetzen ist der die Funktion goProvenancerGo() der Verteiler der Aufgaben an die Threads.

Die Methode teilt eine Datei auf die ProvenanceManager auf, die in beliebiger Anzahl in eigenen Threads laufen. Jeder Thread arbeitet dann die Artikel aus der Datei ab, bei denen Artikelnr. % Threadanzahl = Threadnr. gilt. Dafür lässt er sich die entsprechende Liste von Revisionsobjekten Artikel für Artikel erzeugen und arbeitet diese ab. Die Verteilung der Arbeit erfolgt durch das Executors Framework von Java.

3.3 Maven

Der Code der Anwendung wird als Apache Maven Projekt geschrieben.

3.4 Frameworks

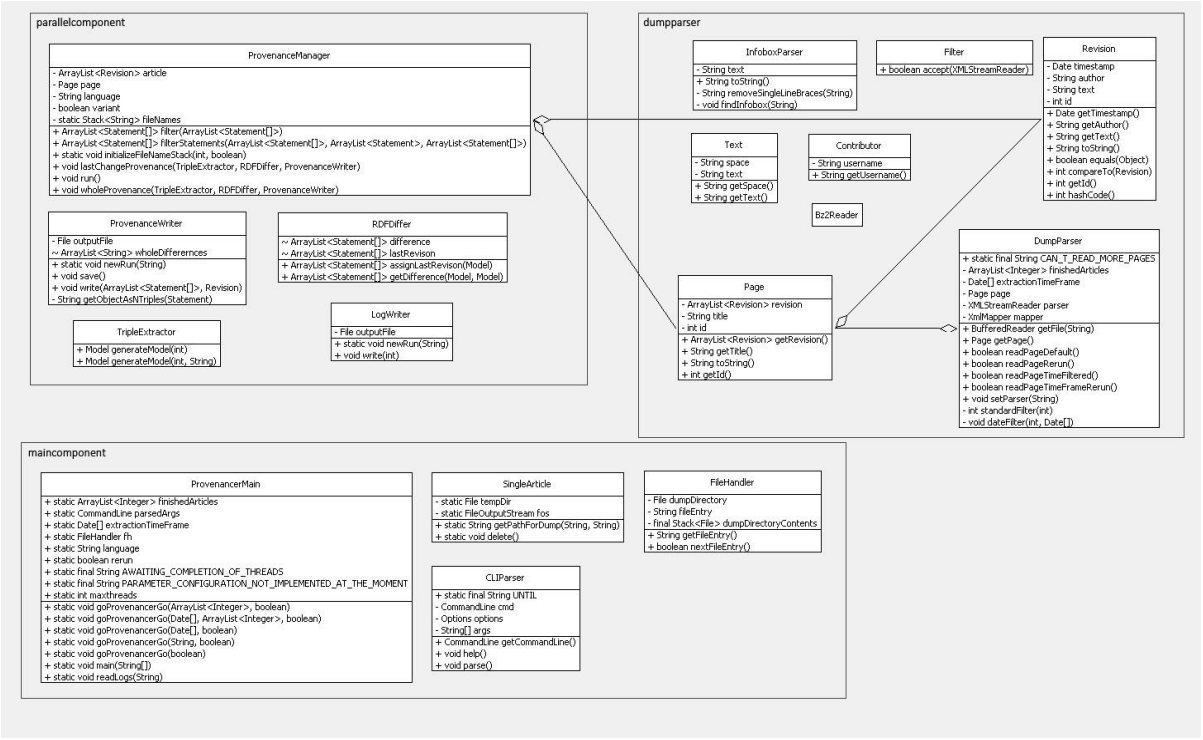
Fuer den Umgang mit dem RDF-Datenmodell wird das Framework Apache Jena verwendet. Um mit komprimierten Dateien umzugehen wird im Bz2Reader.java das Framework Apache Compress genutzt. Zum einlesen der unkomprimierten oder dekomprimierten XML-Dateien wird Jackson/FasterXML verwendet. Um die Konsoleneingaben einzulesen wird Apache Commons CLI verwendet.

3.5 Programmstruktur

Grundsätzlich besteht das Programm aus den folgenden Paketen

1. DumpParser
    - Das DumpParser-Paket enthält alle Klassen die zum verarbeiten der Dumps im XML-Format nötig sind.
2. ParallelComponent
    - Dieses Paket enthält alle Klassen die später in einzelnen Threads zusammen arbeiten.
3. MainComponent
    - Dieses Paket enthält alles was vor dem eigentlichen Algorithmus stattfinden soll.

Das Programm inklusive der Klassen wird von dem folgenden UML-Diagramm beschrieben.



Eine genaue Beschreibung wie diese Klassen miteinander verzahnt sind ist in Punkt 4 enthalten.

3.6 Parameteruebergabe

Zur Steuerung des Programms existieren die folgenden Parameter die beim Programmaufruf angegeben werden koennen.

- -a <arg> Article to provenance
- -from <arg> Earliest Timestamp(Date in yyyy-MM-dd) to extract
- -help Show help
- -lang <arg> Dump Language as "de" or "en"
- -lastchange Only last change to an existingtriple will be saved
- -path <arg> Path to the dump containing directory
- -rerun Rerun program after a crash using the log
- -threads <arg> Number of threads to run
- -until <arg> Last Timestamp (Date in yyyy-MM-dd) to extract

4. Struktur- und Entwurfsprinzipien einzelner Pakete

4.1 DumpParser

Der DumpParser wird vom ProvenanceManager per Konstruktor erzeugt; dabei gibt es verschiedene Varianten für die Use Cases, diesen aufzurufen:

- Mit dem Default-Konstruktor liefert der DumpParser die gesamte Provenienz bzw. ein Page-Objekt das alle Revisionen mit Änderungen der Infobox beinhaltet.
- Optional kann eine Extraktions-Zeitraum angegeben werden; dann werden nur Revisionen aus dem angegebenen Zeitraum zurückgegeben.
- Wird das Programm nach einem Absturz mit vorhandenen Log-Dateien gestartet, kann auch eine Liste der IDs von bereits fertiggestellten Artikeln / Pages als ein weiterer Parameter übergeben werden.

Durch einen Aufruf von “setParser()” kann ein Pfad zu einer XML-Dump-Datei gesetzt werden. Bei Aufruf der entsprechenden “read…()” Methoden wird diese Artikel für Artikel mit ihren Revisionen eingelesen und je nach angegebenen Parametern gefiltert; im Erfolgsfall wird true zurückgegeben. Mit “getPage()” kann anschließend auf diese Daten zugegriffen werden.

4.1.2 Revision

Die Revisionsklasse ermöglicht es alle Daten zu einer bestimmten Revision in einem Objekt zu speichern. Die Verwendung von Konstruktor, Gettern und Settern erfolgt dabei ganz normal.

4.2 ParallelComponent

Diese Komponente enthält mehrere Klassen die wie folgt verzahnt sind:

4.2.1 ProvenanceManager

Der ProvenanceManager steuert den Ablauf des zweiten Programmteils in einzelnen Threads. Er wird mit einem Pageobjekt, einer Sprache und einem boolean variant initialisiert. Ist variant true, wird die komplette Provenienz ermittelt, ist variant false nur die letzte Aenderung eines jeden aktuell vorhandenen Triples. Fuer jeden Artikel wird ein neuer Manager in einem freien Thread gestartet. Anschließend übergibt er chronologisch immer ein chronologisch benachbartes Paar (oder nacheinander das neuste Model mit allen anderen) von durch den TripleExtractor erzeugten Models von RevisionsNummern an den RDFDiffer, und übergibt anschließend die Rückgabe des RDFDiffers mit der neueren

benutzten Revision an die write() Funktion des ProvenanceWriters, um das Ergebnis zu speichern. So geht er sequentiell durch alle Revisionsnummern bis der Artikel verarbeitet und die Ausgabe fertig produziert ist.

4.2.2 TripleExtraktor

Der TripleExtractor ermittelt für eine Revisionsummer mit Hilfe des DBPedia Online Extractors1[1] (im Folgenden als DOE abgekürzt) das entsprechende Model in Apache Jena. Dafür öffnet er die Seite des DOE mit den entsprechenden Parametern, verarbeitet den Seiteninhalt und speist ihn in ein Model ein. Die Klasse ist statisch implementiert. Übergibt man der Methode generateModel() eine Revisionsnummer, wird per Default das Model für die RevisionsNummer aus der englischen Wikipedia zurückgegeben. Will man das Model für eine Seite aus einer anderssprachigen Wikipediaversion, wird zusätzlich zur Revisionsnummer noch das entsprechende Sprachtag übergeben.

4.2.3 RDFDiffer

Der RDFDiffer ist statisch implementiert. Übergibt man der Methode getDifference() zwei Models, gibt die Methode eine ArrayList<Statemtent[]> zurück. In dieser sind ArrayList sind Paare von Statements enthalten. Das Statement mit dem Index 0 taucht neu in dem neueren Model auf, und das Statement mit dem Index 1 wurde zum anderen verändert.

4.2.4 ProvenanceWriter

Der ProvenanceWriter wird mit einem Dateipfad, in den geschrieben werden soll, und einem Boolean, der angibt ob das vorhandene File entleert werden soll, initialisiert. Mit der write() Funktion wird bei Uebergabe eines Statement Arrays und einem Revisionsobjekt die entsprechende Outputzeile generiert und zunaechst zwischengespeichert. Wird save() aufgerufen, werden die gespeicherten Outputzeilen in eine Datei geschrieben. Dadurch wird weitgehend verhindert dass bei einem Programmabsturz Artikel nur teilweise abgearbeitet sind. Die newRun() Methode loescht den Inhalt des Files am uebergebenen Pfad, und schreibt in die oberste Zeile der Files das Datum sowie den aktuellen Commithash.

[1] nachfolgend als DOE abgekürzt

4.3 MainComponent

Dieses Paket verarbeitet die Konsoleneingabe und startet anschließend den restlichen Programmablauf.

4.3.1 FileHandler

Die Klasse FileHandler dient dazu einen Iterator fuer die zu verarbeitenden Dumps zu erstellen. Dem Konstruktor wird der Pfad uebergeben, in dem die Dumps liegen. Dieser wirft jedes DumpFile dann auf einen Stack.

Mit der Methode nextFileEntry() wird der Klassenvariable fileEntry der Pfad zur obersten Datei im Stack zugewiesen, die danach mit getFileEntry() immer wieder abgerufen werden kann. Ist der Stack leer, gibt nextFileEntry() statt einem true ein false zurueck.

4.3.2 SingleArticle

Die Klasse ermöglicht das temporäre herunterladen des Dumps eines Artikels. Wird der Methode getPathForDump() ein Artikelname sowie ein Sprachtag der Form “en”, “de” etc. uebergeben, laedt sie den entsprechenden Dump herunter und gibt den Pfad zur Datei zurueck. Mit der Funktion delete() wird der Ordner, in den die Dumps temporaer heruntergeladen wurden, geloescht.

4.3.3 CLIParser

Die CLIParser Klasse analysiert mit Hilfe von Apache Commons CLI die uebergebenen Parameter. Im Konstruktor werden entsprechend 3.6 die moeglichen Parameter definiert und die eingegebenen Argumente gespeichert. Mit parse() werden dann die Argumente interpretiert und in einer CommandLine als Klassenvariable gespeichert. Diese CommandLine kann dann mit getCommandLine() abgerufen und analysiert werden.

4.3.4 ProvenancerMain

Bei Aufruf der Mainmethode in der Klasse ProvenancerMain werden zunaechst mit Hilfe des CLIParsers die ihr uebergebenen Argumente extrahiert. Anschliessend werden die Argumente interpretiert, und abhaengig davon wird die entsprechende goProvenancerGo() Methode aufgerufen, die den weiteren Ablauf steuert.

In den goProvenancerGo() Methoden wird in einer aeusseren Schleife mit Hilfe des FileHandlers durch die Dateien im uebergebenen Inputpfad iteriert. In einer Inneren Schleife wird dann fuer jede Datei Artikel fuer Artikel per DumpParser extrahiert und an einen

ProvenanceManager in einem eigenen Thread gegeben. Die Methode ist ueberladen um entsprechend den uebergebenen Parametern zu arbeiten.

5. Datenmodell

Für den Umgang mit dem Datentypen RDF wird Apache Jena verwendet.

5.1 Output

Der Output wird im .tsv Format ausgegeben.
Dieser hat dabei folgende Form:

1. **[Timestamp] [commithash]\n**
2. **[neues Triple]\t[altes Triple]\t[Revisionsnr.]\t[Autor]\t[Timestamp in W3C Standard]\n**

...

n. **[neues Triple]\t ...**

\t steht dabei fuer ein Tab, \n fuer einen Zeilenumbruch

In der ersten Zeile steht hier der Zeitpunkt der Erstellung und der Commithash der aktuellen Version, anschliessend folgen die Tripleaenderungen.

6. Glossar

DBpedia is a crowd-sourced community effort to extract structured information from Wikipedia and make this information available on the Web. DBpedia allows you to ask sophisticated queries against Wikipedia, and to link the different data sets on the Web to Wikipedia data.[2]

Das **Resource Description Framework** (RDF, engl. sinngemäß „System zur Beschreibung von Ressourcen“) bezeichnet eine technische Herangehensweise im Internet zur Formulierung logischer Aussagen über beliebige Dinge (Ressourcen).[3]

Parallelverarbeitung beschreibt die simultane Bearbeitung mehrerer Befehlsteile, Befehle oder Programmteile durch eine Zentraleinheit.

Als **Provenienz** werden in der Informatik Informationen über Entitäten, Aktivitäten und Personen, die an der Erzeugung oder Veränderung von Daten beteiligt sind bezeichnet.

Das **DBpedia Extraction Framework** ist ein flexibles und erweiterbares Framework um von einzelnen Wikipediaseiten strukturierte Informationen zu extrahieren und in die DBpedia-Ontologie zu überführen.

Apache Jena ist ein Framework zum Umgang mit dem RDF Datenmodel

[2] Quelle: http://wiki.dbpedia.org/about
[3] Quelle: https://de.wikipedia.org/wiki/Resource_Description_Framework