Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Informatik

Konsolidierung und Erweiterung von Forschungsprototypen für ein Wissensmanagement-Tool

Themenbereiche: Software Entwicklung, Natural Language Processing, Knowledge Engineering

Studierender: Julian Bigler, Robin Bürgi

Dozent: Prof. Dr. sc. inf. Michael Kaufmann

Experte: Dipl. Inf. Ing. HTL Dominique Portmann

Wirtschaftspartner: Data Intelligence Team - HSLU

Keywords: Knowledge Management, NLP, Open-Source, Cloud, Keyphrase Extraction, Big Data

1. Aufgabenstellung

Das Data-Intelligence-Forschungsteam der Hochschule Luzern (HSLU) hat über die letzten Jahre mehrere Software Prototypen im Bereich des Knowledge-Managements entwickelt. Dabei wurde mit Verfahren wie Concept Extraction aus dem Natural Language Processing (NLP), Graph-Exploring, Peer-to-Peer (P2P) Netzwerken und Machine Learning gearbeitet. Die verschiedenen Forschungsprototypen bilden keine stabile Software und die Code Basis ist fragmentiert. Dieses Problem wird in dieser BDA angegangen.

Das Ziel ist es, ein Wissensmanagement-Tool als Open-Source Projekt unter dem Namen ***grafit*** anzubieten. Hierzu sollen die einzelnen Prototypen zu einem Projekt integriert, stabilisiert und weiterentwickelt werden. Eine Erweiterung der Funktionalität soll im Bereich User Interface, Cloud Integration oder Machine Learning getätigt werden.

2. Ergebnisse

Im Rahmen dieser BDA *Konsolidierung und Erweiterung von Forschungsprototypen für ein Wissensmanagement-Tool* wurden folgende Ergebnisse erarbeitet:

- **Stand der Prototypen:** Die aktuellen Prototypen der Data Intelligence Forschungsgruppe wurden analysiert. Zudem wurde zu diesem Punkt ein Literaturstudium zu verschiedenen NLP-Technologien und Algorithmen vorgenommen.
- **Anwendungsszenario:** Basierend auf vorherigen Arbeiten, welche den Business Case eines Knowledge Management Tools untersuchen, wurde ein Anwendungsszenario entwickelt. Dabei wurde klar, dass der Consumer Markt keinen Bedarf an Knowledge-Management-Lösungen hat. Deshalb wurde entschieden, eine Multiuser-Lösung für Teams und Firmen aufzubauen.
- **Integration/ Technologieentscheid:** Unter Beachtung des Anwendungsszenarios wurde evaluiert, welche Funktionalitäten und Konzepte der Prototypen für grafit übernommen oder neu entwickelt werden. Es wurden folgende Funktionalitäten übernommen: Keyphrase Extraction, Search, Node-Editing, Edge Labels und Tree View React Component.
- **Erweiterung:** Nebst der direkten oder konzeptuellen Übernahme mehrerer Features aus den Prototypen wurde die Software auch weiterentwickelt. Eine neue Cloud Schnittstelle, Mehrbenutzerfähigkeit und eine automatische Knowledge-Graph-Generieren wurde umgesetzt.

- **Open-Source:** Ein Open-Source-Projekt wurde auf GitHub initialisiert und der Code, die Build Scripts und eine Entwickler- so wie Benutzerdokumentation sind unter https://github.com/grafit-io/grafit/ frei verfügbar.
- **Deployment:** Das Projekt wurde unter https://grafit.io veröffentlicht.

3. Lösungskonzept

Bei dieser Arbeit handelt es sich primär um ein klassisches Software-Engineering-Projekt. Es wurde beschlossen, das Vorgehens- und Projektmodell Software Development agile (SoDa) einzusetzen. In sechs Sprints wurde eine Client-Server Webapplikation entwickelt. Diese vereint Funktionalitäten und Konzepte der Prototypen in einer Consumer Software.

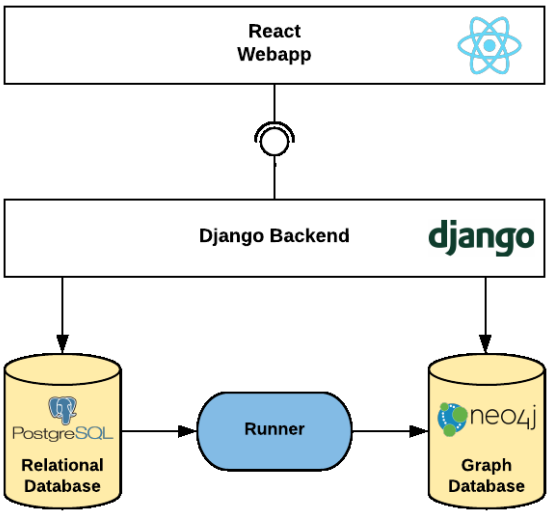

Abbildung 2: Systemübersicht grafit

Wie in der Abbildung 1 ersichtlich, wurde das Projekt mit einem Django Backend und einem React Frontend implementiert. Als Persistenzlösung wurden PostgreSQL und Neo4j eingesetzt. Neo4j ist eine Graphdatenbank und wurde zum Persistieren des Wissensnetzwerkes verwendet.

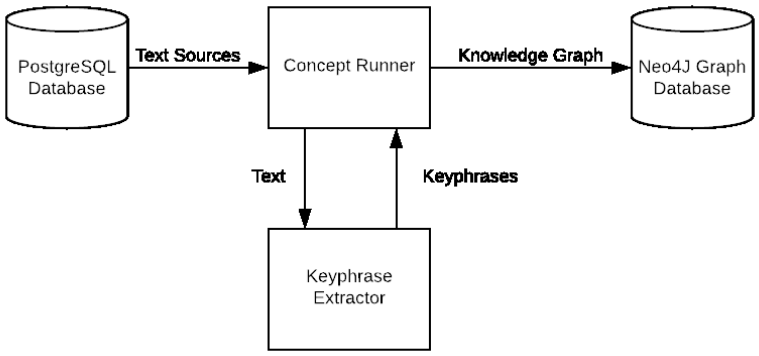

Abbildung 1: Aufbau Knowledge Graph Generierung

Wie in der Abbildung 2 ersichtlich werden die Texte, die der Benutzer in der Webapplikation erfasst, in der PostgreSQL Datenbank gespeichert und anschliessend durch die beiden Python Module **Concept Runner** und **Keyphrase Extractor** weiterverarbeitet. Ziel dieses Prozesses ist es, Verbindungen im Wissensnetzwerk automatisiert zu erkennen und den Benutzer so gezielt bei der Generierung von neuem Wissen zu unterstützen.

Daten können nicht nur als Artikel direkt in der Applikation erfasst werden, sondern auch via Cloud-Integration ins Wissensnetzwerk eingebunden werden. Um eine möglichst breite Integration zu ermöglichen, wurde eine HTTP-Crawler-basierte Lösung entwickelt.

4. Spezielle Herausforderungen

Eine Herausforderung bestand darin, dass das Software Projekt unter der MIT Lizenz auf GitHub als Open-Source Projekt veröffentlicht ist. Es musste überprüft werden, ob alle verwendeten Dependencies (Bibliotheken, Frameworks usw.) mit der MIT Lizenz vereinbar sind. Dieses Problem des Lizenzmanagements wurde durch den Einsatz von FOSSA (https://fossa.com/) gelöst. Dieses Tool wurde in die Build Pipeline integriert und prüft, ob alle vorhandenen Dependencies mit der entsprechenden Open-Source Lizenz übereinstimmen.

Eine weiter Herausforderung, welche der Aufbau eines Open-Source Projektes mit sich brachte, war die Qualitätssicherung auch für spätere Weiterentwicklungen zu garantieren. Hierzu wurde automatisiertes Testing und das Messen der Testabdeckung in den Build Prozess integriert.

5. Ausblick

Im Rahmen dieser Arbeit wurde ein stabiles Open-Source-Projekt aufgebaut. Ein Fokus war dabei, eine hohe Erweiterbarkeit des Codes zu ermöglichen. So wurden gezielt Entwurfsmuster eingesetzt und auf etablierte Technologien gesetzt. Einer Erweiterung dieses Projektes steht somit nichts im Weg. Es könnte eine andere Keyphrase Extraction implementiert werden, welche zum Beispiel mit Deep Learning-Technologien implementiert wird. Eine andere Möglichkeit wäre die Entwicklung eines alternativen Frontends. Hierbei könnte eine native Mobile Applikation oder eine Browser Extension ein innovatives Benutzererlebnis ermöglichen. Eine Möglichkeit ist eine Kommerzialisierung von grafit mit einem SaaS-Modell.