Referenzsuche und Referenzgrapherstellung

Anwendungen von Semantic MediaWiki

Seminararbeit
von

Marius Walk

Am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB)

24. Januar 2014

Inhaltsverzeichnis

1 Einleitung

Die vorliegende Arbeit ist im Rahmen des Seminars „Anwendungen von Semantic MediaWiki“ am Institut für Angewandte Informatik und Formale Beschreibungsverfahren (AIFB) des Karlsruher Instituts für Technologie (KIT) entstanden. Aufgabe des Seminars war es, eine im Voraus definierte Problemstellung durch den Entwurf und die Implementierung einer Erweiterung (Extension) für „Semantic Media Wiki“ zu bearbeiten. Diese Ausarbeitung geht dabei im Folgenden näher auf diese Erweiterung ein und beschreibt dabei die funktionalen Anforderungen an diese, das Konzept, sowie die letztendliche Umsetzung.

1.1 Original Aufgabenstellung

Um einen Überblick über einen Forschungsbereich zu erhalten, ist die Sichtung von Fachliteratur unerlässlich. Interessant ist hier insbesondere, welche Publikationen andere Publikationen zitieren, d.h. welchen "Impact" Publikationen haben und wie sie möglicherweise in unterschiedlichen Communities verankert sind. In einem SMW sind daher besonders relevante Publikationen als Wiki-Seiten (mittels einer Vorlage) gespeichert. Die Aufgabe besteht darin, die Referenzen aus dem damit verlinkten Publikations-PDF zu extrahieren und auf diese Weise entweder auf schon im Wiki vorhandene andere Publikations-Wiki-Seiten zu verlinken oder entsprechende Publikations-Wiki-Seiten anzulegen, falls diese noch nicht existieren. Darauf aufbauend kann ein Referenzgraph erstellt werden, der die Referenzstruktur widerspiegelt [1].

1.2 Spezifikation der Erweiterung

Aus der genannten Aufgabenstellung lassen sich genauere Spezifikationen der gesuchten Erweiterung ableiten, auf die im Folgenden genauer eingegangen wird.

1.2.1 Zielbestimmung

Die Erweiterung soll den Nutzer in die Lage versetzen, wissenschaftliche Dokumente auf angegebene Referenzen zu durchsuchen. Die sich daraus ergebenden Beziehungen der verschiedenen Publikationen untereinander sollen dann in einem Wiki abgebildet und gespeichert werden können. Daraus lassen sich die in folgendem Abschnitt ermittelten funktionalen Anforderungen ableiten.

1.2.2 Funktionale Anforderungen

Die Erweiterung muss folgende Nutzeraufgaben unterstützen:

- /F10/: Einlesen von PDF-Dokumenten.
- /F20/: Extraktion einzelner Referenzen aus einer PDF.
- /F30/: Verbinden einer Publikation mit deren Referenzen.
- /F40/: Anlegen neuer Publikationen aus gefundenen Referenzen.

Folgende Anforderungen an die Erweiterung sind optional:

- /F50/: Visualisierung der sich aus Referenzen ergebenden Beziehungen zwischen Publikationen.

1.2.3 Einsatzumgebung

Die Lösung der angegebenen Problemstellung soll in einem durch „Semantic Media Wiki“ betriebenen Wiki realisiert werden. Ein Wiki ist ein System für Webseiten, deren Inhalte von Benutzern nicht nur gelesen, sondern auch online direkt im Webbrowser geändert werden können. Diese Eigenschaft wird durch ein vereinfachtes Content-Management-System bereitgestellt, in diesem Fall durch „Semantic Media Wiki“ [2].

Dabei handelt es sich um ein vollständiges Framework, mit dessen Hilfe das Wiki als leistungsfähiges und flexibles System zum Wissensmanagement bereitgestellt wird [3]. „Semantic Media Wiki“ basiert dabei auf der open-source Software „Mediawiki“ und ist in der Programmiersprache PHP implementiert. Die Realisierung, der im Rahmen dieses Seminars zu erstellenden Erweiterung für „Semantic Media Wiki“, erfolgt daher ebenfalls in PHP.

2 Konzept

Bevor in Abschnitt drei auf die konkrete Implementierung eingegangen wird, wird hier zunächst das grundsätzliche Konzept zur Realisierung beschrieben. Dazu wird zuerst der gewünschte Ablauf aus Sicht des Nutzers beschrieben um möglicherweise problematische Stellen leichter zu erkennen. Für die erkannten Probleme wird dann jeweils ein möglicher Lösungsweg aufgezeigt.

2.1 Ablauf der Suche nach Referenzen

Eine Publikation wird in einer Wiki-Seite dargestellt. Diese Publikation kann der Nutzer nun durch Klick auf „Referenzen suchen“ auf mögliche Referenzen durchsuchen (siehe Mock-Up in Abbildung 2.1). Dazu gibt der Nutzer im nächsten Schritt den Pfad zur PDF der momentan angezeigten Publikation an. Sobald ein PDF-Dokument hinzugefügt wurde, wird automatisch das Literaturverzeichnis der Publikation durchsucht und entsprechende Verweise auf andere Publikationen extrahiert.

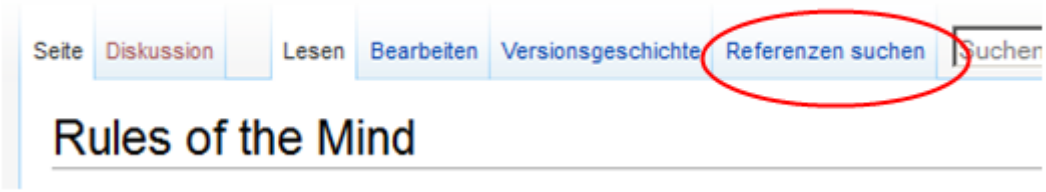

Abbildung 2.1: Mockup eines zusätzlichen Menü-Tabs

Der Nutzer kann nun die extrahierten Referenzen einsehen und entweder bestätigen oder ablehnen. Die bestätigten Referenzen werden dann mit der aufgerufenen Publikation verlinkt. Existiert von der genannten Referenz noch keine Wiki-Seite, wird diese angelegt und mit Informationen aus dem durchsuchten Literaturverzeichnis angereichert.

In folgendem Aktivitätsdiagramm wird dieser Ablauf nochmals verdeutlicht:

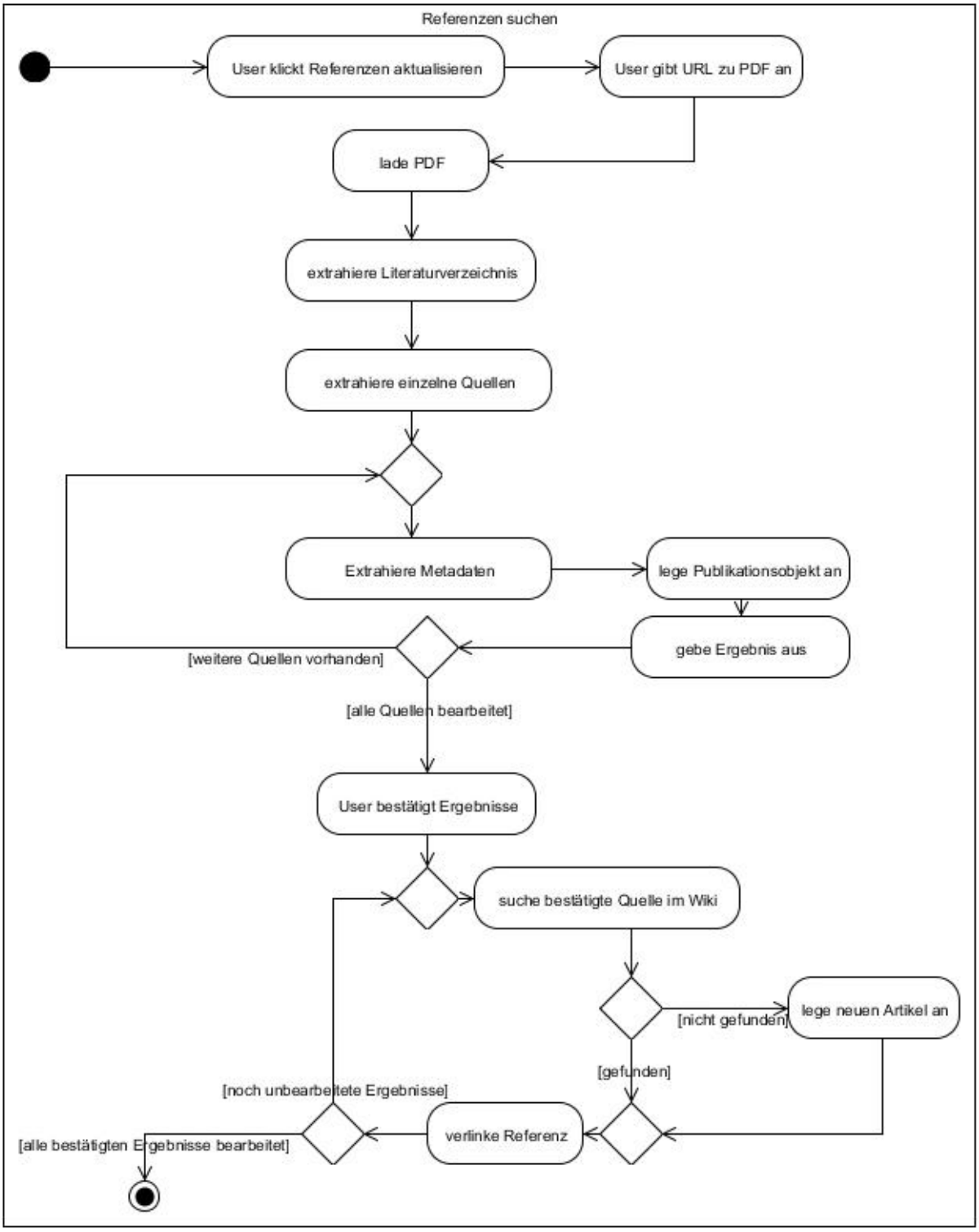

Abbildung 2.2: Aktivitätsdiagramm Referenzen suchen

2.2 Auftretende Problematik

Ausgehend vom dargestellten Aktivitätsdiagramm werden die benötigten Arbeitsschritte im Folgenden jeweils analysiert. Dabei lassen sich die einzelnen Schritte grob in drei Bereiche einteilen: Auslesen von Text aus einer PDF, Extrahieren von Metadaten (Titel der Publikation, Autoren, Journal, Jahr der Veröffentlichung, etc.) aus den einzelnen im Text

vorhandenen Referenzangaben und Auswertung der gewonnenen Metadaten, beziehungsweise Speicherung dieser. Hierbei ist bereits ersichtlich, dass die einzelnen Bereiche aufeinander aufbauend sind. Liefert die Bearbeitung des ersten Bereichs keinen Text als Ergebnis, kann mit der Extraktion der Metadaten nicht begonnen werden. Schlägt die Extraktion der Metadaten fehl, muss und kann im dritten Bereich auch nichts gespeichert werden. Der erste, das Auslesen einer PDF, und der zweite Bereich, die Extraktion der Metadaten, sind somit kritische Bereiche mit deren Ergebnis die erfolgreiche Ausführung der Erweiterung steht und fällt. Daher wird in den nächsten Abschnitten näher auf die aufkommende Problematik in diesen Bereichen eingegangen.

2.2.1 Auslesen einer PDF

Das Portable Document Format (PDF) ist ein Format zur Plattform unabhängigen Darstellung und dem Austausch verschiedener Dokumente. Ein PDF Dokumente besteht dabei aus verschiedenen Objekten, die in hierarchischer Form (Baumstruktur) angeordnet sind. Die Objekte können verschiedene Inhalte, Attribute oder externe Quellen enthalten. Um den Text aus einer PDF zu extrahieren, gilt es also alle Objekte zu finden, die Text enthalten und deren Inhalten dann zusammenzufügen [4].

Dies ist ein komplizierter Prozess bei dem unterschiedliche Formatierungen von PDF-Dokumenten berücksichtigt werden müssen. Um dafür eine Lösung zu finden, mit der es gelingt, möglichst viele verschieden formatierte PDF-Dokumente auszulesen, wurden im Vorfeld unterschiedliche bereits bestehende PHP Bibliotheken zur Textextraktion getestet. Das beste Ergebnis erzielte dabei pdfparser [5], weshalb diese Bibliothek letztendlich zum Auslesen von PDFs eingesetzt wird.

2.2.2 Metadatenextraktion

Um die Metadaten aus dem Literaturverzeichnis einer Publikation zu extrahieren, müssen zunächst die einzelnen Referenzangaben analysiert werden. Referenzangabe bezeichnet dabei einen Aufzählungspunkt aus dem Literaturabschnitt am Ende eines Dokuments.
Jede einzelne Referenzangabe besteht wiederum aus einzelnen Feldern, die Informationen enthalten (zum Beispiel Autor, Titel, Jahr, Journal). Durch verschiedene Signalwörter oder Zeichen, wie beispielsweise Punkte, sind diese Felder voneinander getrennt. Die Vielfalt verschiedener Zitierstandards, gepaart mit unbeabsichtigten Fehlern auf Seiten der Autoren, erschwert den Prozess des automatisierten Auslesens dieser Felder erheblich.

Zitierweise nach APA-Richtliniene:

Alon, Uri. “How to choose a good scientific problem.” *Molecular cell* 35.6 (2009) : 726-728. Print.

Zitierweise nach MLA:

Alon, U. (2009). How to choose a good scientific problem. *Molecular cell*, *35*(6), 726-728.

Abbildung 2.3: Unterschiedliche Zitierweisen

Das Problem der Zuordnung der Felder zu bestimmten Informationen ist unter dem Namen „Sequence Labeling Problem“ bekannt und fordert zur Lösung verschiede heuristische

Maßnahmen [6]. Da die Implementierung einer solchen Lösung den Rahmen dieser Seminararbeit sprengen würde, muss im Folgenden auf bereits existierende Softwarekomponenten zurückgegriffen werden, die sich diesem Problem annehmen.

Dafür wurden zum Einen ParsCit [7] und zum Anderen FreeCite [8] im Vorfeld getestet. Beides sind open-source Anwendungen, die Referenzangaben parsen und in geordneten Daten zurückgeben können. ParsCit lieferte bei Tests mit diversen PDFs die genaueren Ergebnisse.

Während ParsCit in Python implementiert ist, basiert FreeCite auf Ruby on Rails. Da Mediawiki jedoch in PHP programmiert ist, kann eine direkte Einbindung in die Erweiterung nicht erfolgen. In beiden Fällen muss somit auf eine vorhandene Web-Schnittstelle zurückgegriffen werden. Die von ParsCit bereitgestellte SOAP-Schnittstelle hat bei den Tests einen sehr unzuverlässigen Eindruck hinterlassen und war oftmals nicht zu erreichen. Die REST-Schnittstelle von FreeCite hingegen hatte keine Ausfälle zu vermerken.

Daher fiel die Entscheidung trotz des schlechteren Testergebnisses beim Parsen von Referenzangaben auf die Implementierung der FreeCite-Schnittstelle.

3 Umsetzung

In diesem Kapital wird auf die konkrete Realisierung der Lösung eingegangen. Dies ist die im Rahmen des Seminars entstandene und vollständig implementierte Erweiterung Reference Helper für Semantic Media Wiki.

3.1 Aufbau

Ausgehend von der Dateistruktur der Erweiterung lassen sich Aufbau und daraus resultierend die Funktionsweise von Reference Helper beschreiben. Die Erweiterung besteht aus folgenden Dateien:

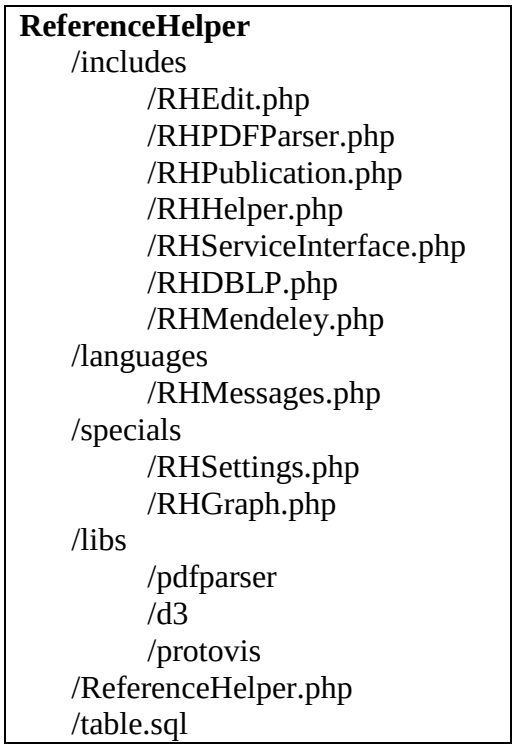

Abbildung 3.1: Dateistruktur Reference Helper

Die Erweiterung ist somit in die vier Ordner, /includes, /languages, /specials und /libs aufgeteilt. Zusätzlich liegen im Hauptverzeichnis die Dateien ReferenceHelper.php und table.sql, welche zur Installation der Erweiterung benötigt werden. Diese erfolgt durch Einbinden von ReferenceHelper.php in MediaWikis LocalSettings.php. Darüber hinaus wird ein Update des Datenbankschemas von MediaWiki benötigt (über den Befehl: php update.php). Dabei wird dann mit Hilfe der SQL-Anweisungen in table.sql eine neue Tabelle in der Datenbank angelegt. In ReferenceHelper.php werden weiterhin alle für die Erweiterung benötigten Dateien geladen und verwendete Hooks registriert.

Im Ordner /libs liegen alle externen Bibliotheken auf die die Erweiterung zurückgreifen muss. Dies sind zum einen pdfparser [5] zum Auslesen diverser PDF-Dokumente und die Javascript Bibliotheken d3js [9] sowie protovis [10] zur Generierung und Darstellung von Graphen.

Das Verzeichnis /languages beinhaltet die Datei RHMessages.php, welche alle benötigten Textbaustein-Variablen von Reference Helper enthält. Dadurch wird ein sehr wichtiges Prinzip der MediaWiki Software, die leichte Anpassung an verschiedene Sprachen,

gewährleistet. Je nach eingestellter Sprachversion des Wikis, können die in dieser Datei definierten Variablen mit der jeweiligen Sprache belegt werden.

Die von der Erweiterung bereitgestellten Spezialseiten werden durch die Dateien RHSettings.php und RHGraph.php im Verzeichnis /specials erstellt. Dabei enthält RHSettings.php Logik und Aufbau des Einstellungsmenüs (siehe Abschnitt 3.2.1) und RHGraph.php die Implementierung zur Visualisierung der Referenzbeziehungen (siehe Abschnitt 3.2.5).

Im Ordner /includes befinden sich die Dateien, mit deren Hilfe der eigentliche Prozess des Referenzen Suchens und Speicherns abgewickelt wird (siehe Abbildung 2.2). Dazu werden folgende PHP-Klassen verwendet:

RHEdit.php:	RHEdit steuert den gesamten Prozess des Referenzen Suchens und Speicherns. Dabei stellt die Klasse das entsprechende User Interface bereit und verarbeitet die Eingaben des Nutzers.
RHPDFParser.php:	RHPDFParser stellt Werkzeuge zum Auslesen des Textes und Finden von Referenzen in diversen PDFs bereit (siehe Abschnitt 3.2.2).
RHPublication.php:	Diese Klasse repräsentiert Publikationen. Publikationsobjekte enthalten alle bereitstehenden Informationen (Titel, Autor, etc.) einer Publikation. RHPublication stellt darüber hinaus Werkzeuge zur Bearbeitung von Publikationen, wie beispielsweise einer Methode zur Erstellung eines Wiki-Artikels aus einem Publikations-Objekt, bereit.
RHHelper.php:	RHHelper stellt verschiedene statische Methoden mit Operationen bereit, die im Programmablauf mehrmals benötigt werden. Dadurch werden Codewiederholungen vermieden.
RHServiceInterface.php:	Diese Klasse stellt ein Interface dar und definiert Methoden zur Anbindung verschiedener Services zur Informationsgewinnung (siehe Abschnitt 3.2.3).
RHDBLP & RHMendeley:	RHDBLP & RHMendeley implementieren jeweils das Interface RHServiceInterface und bieten die Möglichkeit die Datenbestände von DBLP [11] und Mendely.com [12] nach Publikationen zu durchsuchen (siehe Abschnitt 3.2.3).

3.2 Funktionsweise ausgewählter Programmteile

Im Folgenden sollen abschnittsweise die wichtigsten Arbeitsschritte der Erweiterung erklärt werden. Zuerst wird auf die generelle Funktion der Komponente eingegangen, danach die konkrete Implementierung und die dazugehörigen Abläufe erläutert und gegebenenfalls mit Codeauszügen ergänzt.

3.2.1 Einstellungsmenü

Dieser Abschnitt behandelt das über die Spezialseite RH Settings bereitgestellte Einstellungsmenü. Eine der Funktionen von Reference Helper ist es, für in Referenzen gefundene Publikationen, die bisher noch nicht im Wiki hinterlegt waren, neue Wikiseiten anzulegen. Damit für den Nutzer der Erweiterung die Möglichkeit besteht, die Struktur und die Zugehörigkeit zu einer Wikikategorie mitzubestimmen, ist ein solches Einstellungsmenü unabdingbar.

Es gibt es in MediaWiki die Möglichkeit für das Erstellen von neuen Seiten eine Vorlage zu verwenden. Vorlagen sind Wikiseiten, welche in anderen Seiten auf folgende Weise per Wikitext eingebunden werden können: {{Name}}. Dieser Befehl bindet dann den Inhalt der Vorlage "[[Vorlage:Name]]" an dieser Stelle ein [13]. Damit Vorlagen auch dynamisch verschiedene Inhalte darstellen können werden Parameter verwendet. Diese sind Platzhalter die mit beliebigen Werten gefüllt werden können, wie folgendes Beispiel verdeutlicht.

Vorlage (Wikitext):	*Artikel (Wikitext):*	*Anzeige des Artikels:*
Diese Publikation hat den Titel {{{Titel}}}.	{{Vorlage|Titel=Seminararbeit}}	Diese Publikation hat den Titel Seminararbeit.

Mit Hilfe von Vorlagen können Artikel des gleichen Typs (hier Publikation) somit einheitlich mit derselben Struktur dargestellt werden.

Das Einstellungsmenü unterteilt sich somit in drei verschiedene Bereiche. Im ersten Bereich wird die Kategorie festgelegt, in der Reference Helper aktiviert ist. Im zweiten Bereich wird die zu verwendende Vorlage selektiert und im dritten Bereich werden dann die Parameter der gewählten Vorlage den von Reference Helper bereitgestellten Informationen zugeordnet (siehe Abbildung 3.2).

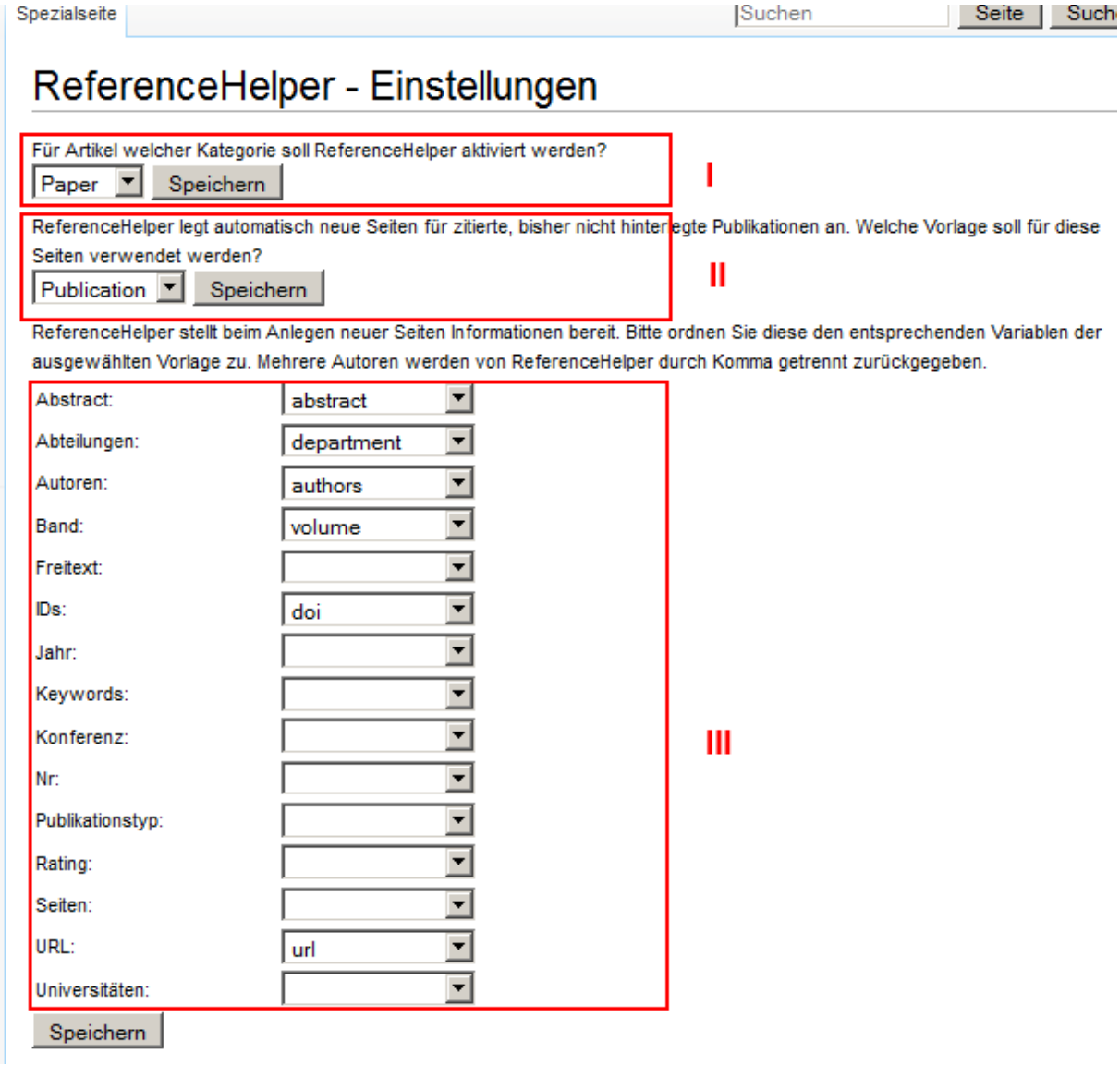

Abbildung 3.2: Einstellungsmenü

Die Bereitstellung der Formulare des Menüs, sowie die Auswertung und Speicherung der Eingaben übernimmt dabei die Klasse RHSettings. Für jeden Menübereich stellt diese Klasse eine Methode zum Aufbau des jeweiligen Formulars (beispielsweise *getTemplateSection*) und zur Speicherung der eingegebenen Daten (hier *saveTemplate*) zur Verfügung. Für die ersten beiden Bereiche werden in den jeweiligen Methoden zuerst alle im Wiki vorhanden Kategorien oder Vorlagen per SQL-Anfrage geladen und dann in einem Dropdown Menü zur Auswahl dargestellt.

Der dritte Bereich erfordert zunächst eine andere Vorgehensweise. Die Methode *getVariableSection* bestimmt zuerst die momentan gewählte Vorlage und extrahiert dann aus dieser alle verwendeten Parameter. Zu jedem dieser gefundenen Parameter kann dann per Dropdown Menü ein Wert ausgewählt werden, mit dem der Parameter durch Reference Helper gefüllt werden soll (siehe Abbildung 3.2). Alle durch Reference Helper bereitgestellten Werte sind in der Klasse RHPublication definiert (Attribute).

Die Speicherung aller Einstellungen erfolgt in der Datenbanktabelle reference_helper, die wie folgt strukturiert ist:

#	Name	Typ
1	field_type	varchar(15)
2	standard	varchar(25)
3	user	varchar(25)

Abbildung 3.3: Datenbanktabelle reference_helper

Die Spalte *field_type* bezeichnet dabei aus welchem der drei Einstellungsbereiche der gespeicherte Wert stammt (Vorlage, Kategorie oder Parameter). In der Spalte *user* werden dann alle Angaben des Nutzers gespeichert. Ist als *field_type* der Bereich Parameter angegeben ist zusätzlich in der Spalte *standard* noch der durch Reference Helper bereitzustellende Wert angegeben.

3.2.2 Extrahieren einzelner Publikationen aus Literaturverzeichnis

Hier wird die zentrale Funktion der Erweiterung Reference Helper behandelt. Aus einem vom Nutzer angegebenen PDF-Dokument werden alle zitierten Publikationen extrahiert. Diese Aufgabe übernimmt die Klasse RHPDFParser, deren Konstruktor zunächst der Pfad zu einem PDF-Dokument als Parameter übergeben werden muss.

Im Konstruktor werden dann bereits die einzelnen Einträge des Literaturverzeichnisses voneinander getrennt und jeweils als String in einem Array gespeichert (siehe Abbildung 3.4). Dieses Array wird daraufhin im citation Attribut hinterlegt. Als Ergebnis gibt der Konstruktor dann ein RHPDFParser Objekt zurück.

REFERENCES

Aleven, V., & Koedinger, K. R. (2002). An effective meta-cognitive strategy: Learning by doing and explaining with a computer-based Cognitive Tutor. *Cognitive Science*, 26(2), 147-179. hier trennen

Aleven, V., & Koedinger, K. R. (2000). Limitations of Student Control: Do Students Know when they need help? In G. Gauthier, C. Frasson & K. VanLehn (Eds.) *Proceedings of the 5th International Conference on Intelligent Tutoring Systems, ITS 2000* (pp. 292-303). Berlin: Springer Verlag. hier trennen

Aleven, V., McLaren, B. M., & Koedinger, K. R. (2006). Towards Computer-Based Tutoring of Help-Seeking Skills. In S. Karabenick & R. Newman (Eds.) *Help Seeking in Academic Settings: Goals, Groups, and Contexts* (pp. 259-296). Mahwah, NJ: Erlbaum.

Abbildung 3.4: Ausschnitt aus Literaturverzeichnis

Zur Bewältigung dieser Aufgabe stehen dem Konstruktor diverse Hilfsmethoden zur Verfügung. Zuallererst wird mit Hilfe der eingebundenen pdfparser-Bibliothek der reine Text des PDF-Dokuments extrahiert. Dieser steht im Anschluss als String zur Verfügung und wird der Hilfsmethode *getReferenceSection* übergeben.

Diese Methode durchsucht den Text nach dem Literaturverzeichnis, beziehungsweise dem Abschnitt in dem alle Referenzen angegeben sind. Nach Analyse verschiedener Publikationen und deren strukturellem Aufbau ergab sich dafür folgende Vorgehensweise. Der Text wird von hinten nach verschiedenen Schlüsselwörtern durchsucht (hier: „References“,

„Bibliography“, „Literatur“, „Literaturverzeichnis“). Wird einer dieser Begriffe gefunden, markiert die Fundstelle den Beginn des Literaturverzeichnisses, der Text wird an dieser Stelle abgeschnitten und als String zurückgegeben.

Dieser String wird dann an die Methode *getCitationStyle* weitergereicht. Hier wird untersucht durch welches charakteristische Merkmal einzelne Referenzangaben voneinander zu trennen sind (siehe Abbildung 3.4). Hierzu wurden im Vorhinein der Implementierung mehrere Publikationen analysiert. Dabei traten vermehrt folgende drei Zeichenfolgen zur Trennung einzelner Referenzangaben auf:

1.) Jeder Literaturverweis ist durch eine in eckigen Klammern vorangestellte Zahl durchnummeriert. Das Literaturverzeichnis lässt sich danach mit folgendem regulären Ausdruck durchsuchen: „@\[[0-9]{1,3}\]@“.
2.) Jeder Literaturverweis wird mit einem Punkt abgeschlossen, auf den dann ein erzwungener Absatz folgt. Der reguläre Ausdruck zur Suche hierfür lautet „@\.\n@“.
3.) Jeder Literaturverweis wird mit einem Punkt abgeschlossen, auf den ein Leerzeichen und dann ein erzwungener Absatz folgen. Hier lautet der reguläre Ausdruck „@\. \n@“.

Die Methode durchsucht nun das Literaturverzeichnis nach den angegebenen regulären Ausdrücken und gibt den am häufigsten gefundenen Ausdruck als String.

```
    /**
     * Gives back a regex of the most used citation style.
     * @return $regex
     */
    private function getCitationStyle($references) {
          $styles = array('@\[[0-9]{1,3}\]@' => 0,
                '@\.\n@' => 0,
                '@\. \n@' => 0);
          foreach ($styles as $regex => $quantity) {
                $styles[$regex] = preg_match_all($regex, $references);
          }
          arsort($styles);
          $keys = array_keys($styles);
          return array_shift($keys);
    }
```

Abbildung 3.5: implementierte Methode getCitationStyle

Mit Hilfe des zurückgegebenen regulären Ausdrucks kann nun per PHP Methode *preg_split* das Literaturverzeichnis in die einzelnen Literaturverweise aufgeteilt werden. Diese werden dann als Array gespeichert.

Nachdem der Konstruktor ein RHPDFParser-Objekt erfolgreich erstellt hat, kann auf dieses die Methode *getCitatedPublications* angewendet werden. Diese schickt die gefundenen Literaturverweise an die Schnittstelle von FreeCite (siehe Abschnitt 2.2.2), verwertet die XML-Rückgabe dieser und gibt dann in Form eines Arrays für jeden Literaturverweis ein RHPublication-Objekt zurück.

3.2.3 Anreichern der Informationen durch Abfrage verschiedener Services

In einem Literaturverweis stehen oftmals nur wenige Metadaten, wie beispielsweise Titel, Autoren und Erscheinungsjahr. Um weitere Informationen über die zitierte Publikation zu erhalten (Journal, Abstract, Universität, URL, etc.) können öffentliche Datenbanken mit einer Fülle an wissenschaftlichen Artikeln abgefragt werden. Mit Hilfe der zusätzlichen Informationen dieser kann Reference Helper dann aussagekräftigere Wikiseiten zur jeweiligen Publikation anlegen. Auch für den Fall, dass der Verfasser bei der Erstellung des Literaturverzeichnisses kleinere Fehler macht, hilft die zusätzliche Abfrage diverser Services zur eventuellen Korrektur.

Zur Einbindung eines öffentlichen Services stellt Reference Helper das Interface RHServiceInterface zur Verfügung. Klassen, die die Kommunikation mit einem Service steuern, müssen dieses Interface und somit auch die Methode *getSearchResults* implementieren. Diese Methode bekommt als Parameter ein RHPublication-Objekt übergeben und kann dann die verbundene öffentliche Datenbank nach beispielsweise dessen Titel oder Autor durchsuchen. Der Rückgabewert der Methode ist wiederrum ein Array aus RHPublication-Objekten, welches die Suchergebnisse wiederspiegelt.

Bereits implementiert sind mit den Klassen RHMendeley und RHDBLP Suchanfragen auf Mendeley.com und der Datenbank DBLP, die viele wissenschaftliche Publikationen mit Schwerpunkt Informatik bereitstellt. Vorteil von Mendeley.com ist die Möglichkeit einer Volltextsuche. Hier kann also ein kompletter Literaturverweis als Suchbegriff dienen und somit Fehler dessen Verfassers korrigiert werden. Auf DBLP wird dagegen nur nach dem entsprechenden Titel gesucht und alle vorhandenen Metadaten der Treffer zurückgegeben.

Die Steuerung des Suchprozesses und des Auswertens der gefundenen Resultate übernimmt dabei die Methode *getInfoFromServices* der Klasse RHPublication. Als Parameter muss dieser Methode ein Array an Serviceobjekten (beispielsweise ein RHMendeley-Objekt) übergeben werden. Jeder übergebende Service wird dann anschließend durchsucht. Durch die Methode *compare* wird daraufhin verglichen ob unter den Suchtreffern eine Übereinstimmung mit dem gesuchten RHPublication-Objekt existiert. Ist dies der Fall, werden die durch den Service bereitgestellten Metadaten übernommen.

Die Methode *getInfoFromServices* wird im momentanen Programmablauf von der Klasse RHEdit aufgerufen. Hier ist auch durch die Methode *getServices* definiert welche Services in Reference Helper verwendet werden dürfen

Der komplette Ablauf und das Zusammenspiel verschiedener Objekte und Methoden sind in unten stehendem Sequenzdiagramm nochmals verdeutlicht.

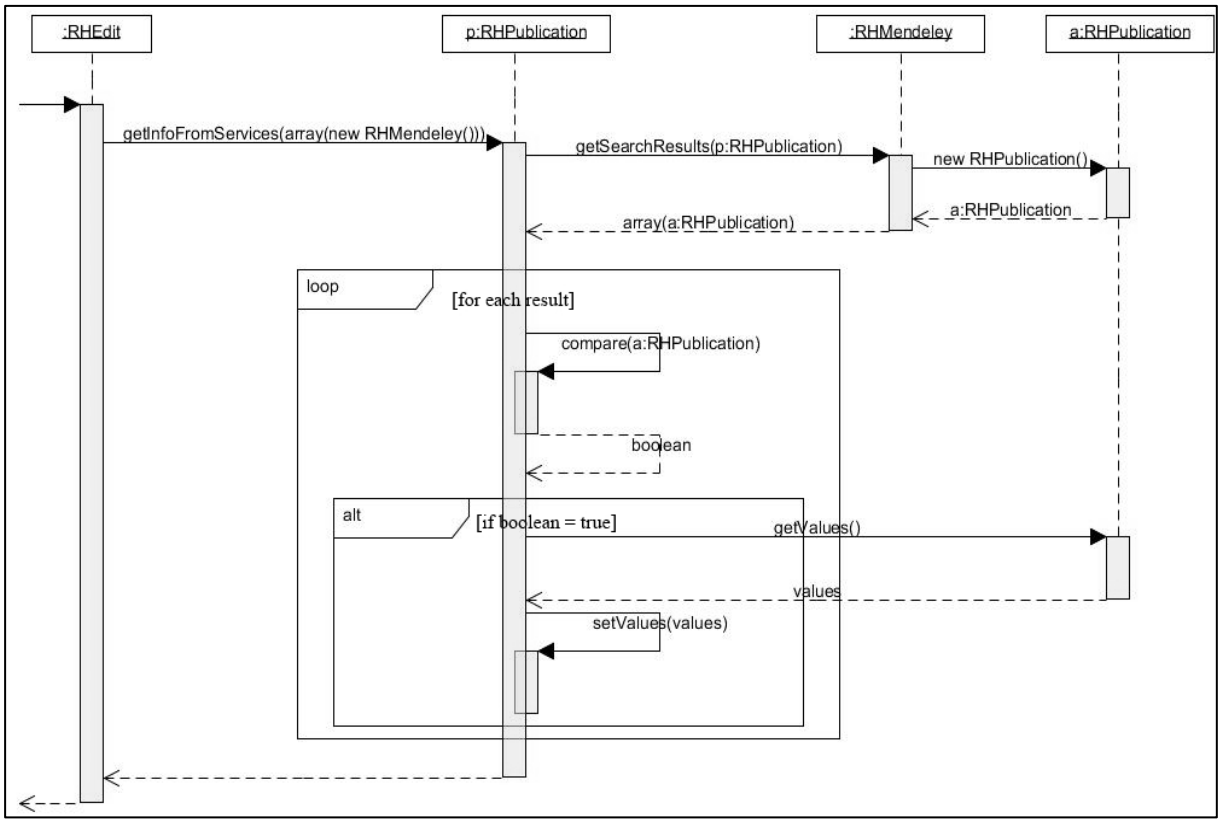

Abbildung 3.6: Sequenzdiagramm Einbindung von Services

3.2.4 Verarbeiten der Ergebnisse

Das Verarbeiten der Ergebnisse unterteilt sich in zwei Hauptaufgaben. Zum Einen wird überprüft, ob bereits eine Wiki-Seite über die gefundenen Referenzen besteht. Ist dies nicht der Fall legt Reference Helper dafür jeweils eine neue Wiki-Seite an. Die zu einer gegebenen Publikation gefundenen Referenzen, müssen zum Anderen auf der Wiki-Seite, der referenzierenden Publikation, hinterlegt werden. Damit keine Fehlerhaften Informationen im Wiki gespeichert werden, gibt Reference Helper dem Nutzer zusätzlich die Möglichkeit, die gefundenen Referenzen vor dem Speichern einzusehen und wie gewünscht zu bestätigen oder abzulehnen.

Dieser gesamte Prozess wird von der Klasse RHEdit innerhalb der Methode *saveResults* gesteuert. Für jede vom Nutzer bestätigte Referenz wird dort zunächst ein RHPublication-Objekt angelegt. Danach wird überprüft ob für die durch diese Objekte dargestellten Publikationen bereits eine Wiki-Seite existiert. Hierzu macht sich Reference Helper die von der in Media Wiki bereits integrierte Suche und deren Funktionen zu Nutze.

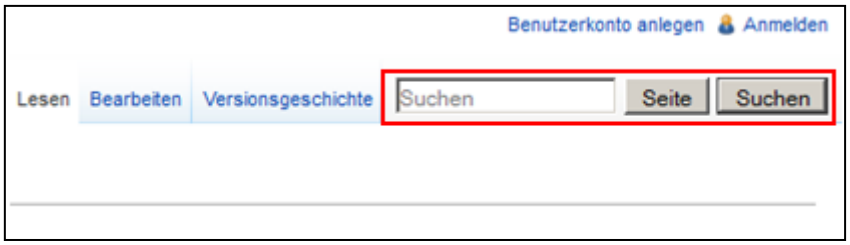

Abbildung 3.7: Suchfunktion in Media Wiki

Als Ergebnis werden bei dieser Suche für beliebige Suchbegriffe auch Übereinstimmungen mit Seitentiteln zurückgegeben. Geschickt ist, dass auch mögliche Abweichung durch unterschiedliche Groß- oder Kleinschreibung oder Zeichensetzung berücksichtigt werden. Weicht der Titel einer gefundenen Referenz also nur minimal vom Titel einer bereits bestehenden Wiki-Seite ab, wird diese Seite als Suchergebnis zurückgegeben.

Reference Helper bindet somit den von Media Wiki verwendeten Search Engine (hier: SearchMySQL) ein und sucht mit Hilfe dessen Methode *getTitle* nach übereinstimmenden Seitentiteln. Wird kein Treffer gefunden, erstellt Reference Helper daraufhin eine neue Wiki-Seite.

Dies erfolgt über die von der Klasse RHPublication bereitgestellte Methode *saveAsWikiPage.* Unter Verwendung der im Einstellungsmenü der Erweiterung angegebenen Vorlage (siehe Abschnitt 3.2.1) wird dort der benötigte Wiki-Text generiert und anschließend gespeichert.

Um die Verarbeitung der Ergebnisse fertigzustellen werden abschließend alle bestätigten Referenzen auf der Wiki-Seite der referenzierenden Publikation hinterlegt. Dafür wird folgender Wiki-Text generiert und an das Ende der Seite angehängt:

{{#set: Has Reference = Titel Referenz A| … | Has Reference = Titel Referenz Z}}

#set ist dabei eine von Semantic Media Wiki bereitgestellte Parserfunktion, die es ermöglicht stille Annotationen von Attribut-Attributwertpaaren vorzunehmen [14]. Diese werden dann zwar nicht auf der Wiki-Seite angezeigt sind aber hinterlegt und abrufbar. Das Attribut ist hierbei „Has Reference“ und der Wert der jeweilige Titel der referenzierten Publikation.

3.2.5 Visualisierung der Referenzstruktur

Durch die mit Reference Helper gesammelten und hinterlegten Daten stehen die in einem Wiki gespeicherten Publikationen in einer Beziehung zueinander. Durch eine Visualisierung dieser lässt sich der Einfluss, den eine Publikation beispielsweise in einem bestimmten Themengebiet hat leichter ermitteln. Durch das in den Wiki-Seiten hinterlegte Attribut „Has Reference“ (siehe Abschnitt 3.2.4) kann dann für jede Publikation die Häufigkeit der Zitierungen bestimmt werden.

Die Auswertung dieser Daten und die Visualisierung dieser übernimmt dabei die Klasse RHGraph, die von der Media Wiki Klasse SpecialPage erbt und somit eine Spezialseite bereitstellt. Hier kann der Nutzer zunächst über folgendes Formular die Menge der zu visualisierenden Publikationen eingrenzen:

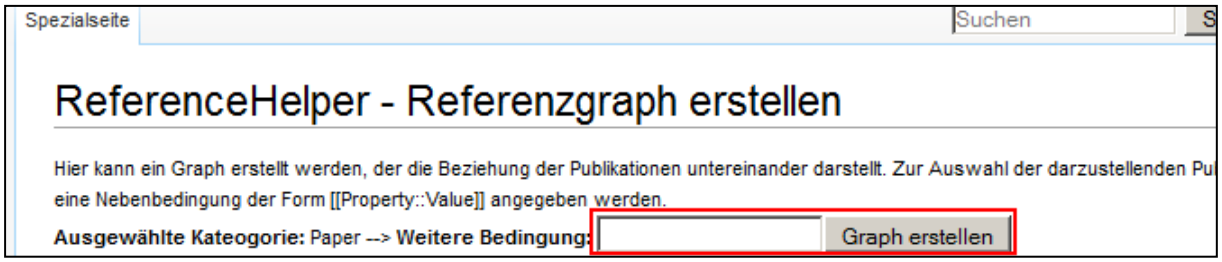

Abbildung 3.8: Formular der Spezialseite RHGraph

Durch Eingabe der Bedingung [[Jahr::2002]] wird die Menge beispielsweise auf alle im Jahr 2002 erschienen Publikationen begrenzt. Bleibt das Formularfeld leer, erfolgt keine Eingrenzung und es werden alle Artikel der im Einstellungsmenü definierten Kategorie ausgelesen. Die Abfrage der zur Visualisierung benötigten Daten erfolgt dabei über die von Semantic Media Wiki bereitgestellte Sprache für Abfragen [15]. In diesem Fall lautet die Abfrage wie folgt:

{{#ask: [[Category:Kategorie]] Nebenbedingung
| ?Has Reference
}}

Kategorie und Nebenbedingung sind hier Platzhalter, die von Reference Helper gefüllt werden. Die Methode *getResult* der Klasse RHGraph führt die Abfrage dann mit Hilfe der in Semantic Media Wiki enthaltenen Klasse SMWQueryProcessor aus und gibt das Ergebnis in Form eines mehrdimensionalen Arrays zurück.

Key		**Value**
[Titel 1]	=>	array(Titel Referenz 1, Titel Referenz 2, …)
[Titel 2]	=>	array(Titel Referenz 1, Titel Referenz 2, …)
…	=>	…

Abbildung 3.9: Aufbau des Ergebnisarrays

Dieses Array wird dann im weiteren Verlauf grafisch dargestellt. Dazu stellt Reference Helper zwei Diagrammformen bereit. Zum Einen einen gerichteten Graphen, der durch die Javascript Bibliothek d3js [9] bereitgestellt wird und zum anderen ein Arc Diagramm, realisiert durch die Javascript Bibliothek protovis [10].

Die Bibliotheken werden mit Hilfe des Hooks BeforePageDisplay von der in der Klasse RHGraph implementierten Methode *onBeforePageDisplay* eingebunden. Die Methoden *getArcDiagram* und *getDirectedGraph* generieren dann den jeweils benötigten Javascript Code, der auf der Spezialseite ausgegeben wird.

4 Bewertung

In Abschnitt 1.2 wurden die Ziele und im Speziellen die funktionalen Anforderungen an die zu implementierende Erweiterung definiert. Im Folgenden soll anhand dieser Anforderungen

das Ergebnis der Umsetzung bewertet werden. Darüber hinaus wird auf Maßnahmen eingegangen, die zur endgültigen Erfüllung aller Anforderungen noch erforderlich sind oder zur Verbesserung des bisher erreichten Ergebnisses beitragen.

4.1 Erfüllung der funktionalen Anforderungen

/F10/: Einlesen von PDF-Dokumenten.

Die von Reference Helper bereitgestellte Klasse RHEdit bietet dem Nutzer die Möglichkeit die URL zu einem PDF-Dokument anzugeben. Mit Hilfe der PHP Bibliothek pdfparser [5] wird dann der Text dieses Dokuments als String extrahiert.

Bisher bietet das Formular zur Angabe eines PDF-Dokuments nur die Möglichkeit, der Angabe einer URL. Der Server auf dem Media Wiki verwendet wird muss auf diese URL zugreifen können. Lokal gespeicherte Dokumente sind somit außen vor und können von Reference Helper nicht geparst werden.

Die in Abschnitt 2.2.1 beschriebene Problematik führt dazu, dass eine vollständige Abdeckung aller PDF-Dokumente nicht garantiert werden kann. Tests verschieden formatierter Dokumente ergaben in seltenen Fällen eine unvollständige Extraktion des Textinhalts in einen String.

Aus diesem Grund kann die Anforderung F10 nur als bedingt erfüllt angesehen werden.

/F20/: Extraktion einzelner Referenzen aus einer PDF.

Dank der Einbindung der Schnittstelle von FreeCite (siehe Abschnitt 2.2.2) ist eine Aufschlüsselung der einzelnen Metadaten eines Literaturverweises möglich. Aufgrund der vielen verschiedenen Zitierweisen können jedoch Ungenauigkeiten und Fehler auftreten. Um die Fehlerquote zu reduzieren bietet Reference Helper zusätzlich die Möglichkeit, öffentliche Datenbanken nach Informationen zu Publikationen zu durchsuchen und eventuelle Fehler so zu verbessern (siehe Abschnitt 3.2.3). Bevor die von Reference Helper erfassten Daten im Wiki gespeichert werden, können diese dann nochmal vom Nutzer überprüft und gegebenenfalls korrigiert werden. Eine vollständige Übernahme der Referenzen eines PDF-Dokuments mit Hilfe von Reference Helper ist somit gewährleistet. Die Anforderung F20 kann daher als erfüllt bewertet werden.

/F30/: Verbinden einer Publikation mit deren Referenzen.

Wie in Abschnitt 3.2.4 beschrieben sorgt die Methode *saveResults* der Klasse RHEdit für die stille Annotation des Attributs „Has Reference“ in einer Publikation. Dieses steht für die Verbindung der Publikation mit seinen Referenzen. Durch Abfragen des Attributs lassen sich somit Verbindungen der Publikationen untereinander nachvollziehen. Die Anforderung kann somit als erfüllt betrachtet werden.

/F40/: Anlegen neuer Publikationen aus gefundenen Referenzen.

Die Methode *saveAsWikiPage* der Klasse RHPublication legt für ein beliebiges RHPublication-Objekt eine Wiki-Seite an. Dies ermöglicht auch das Speichern einer gefundenen Referenz als solche. Darüber verhindert Reference Helper wie in Abschnitt 3.2.4 beschrieben das Anlegen von Duplikaten, durch die Überprüfung des Titels.

Bevor Reference Helper jedoch automatisch Seiten anlegen kann, muss der Nutzer eine Vorlage erstellen und diese im Einstellungsmenü angeben. Die Anforderung F40 wird daher zusammenfassend als nahezu erfüllt betrachtet.

/F50/: Visualisierung der sich aus Referenzen ergebenden Beziehungen zwischen Publikationen.

Auf der Spezialseite RHGraph werden zwei verschiedene Diagramme zur Verdeutlichung der vorherrschenden Beziehungen zwischen Publikationen angezeigt (siehe Abschnitt 3.2.5). Zusätzlich kann der Nutzer die Menge der anzuzeigenden Publikationen durch benutzerdefinierte Bedingungen einschränken. Anforderung F50 wird somit als erfüllt bewertet.

4.2 Mögliche Weiterentwicklungen

4.2.1 Autorennamen

In einer möglichen Weiterentwicklung könnten Richtlinien für Autorennamen berücksichtigt werden. Die vorliegende Version setzt keine besondere Schreibweise des Namens eines Autors voraus. Der Name besteht lediglich aus einem String und es wird keine Unterscheidung zwischen Vor- und Nachnamen gemacht. Der Name Max Mustermann beispielsweise, kann durch verschiedene Angaben in Zitaten einmal als „M. Mustermann“ und in einem anderen Fall als „Max Mustermann“ als Autor hinterlegt werden.

Würde hier eine einheitliche Linie durchgesetzt und realisiert wären die Ergebnisse semantischer Abfragen, die den Autor als Attribut beinhalten vollständiger.

4.2.2 Einlesen von PDF-Dokumenten

Wie in Abschnitt 4.1 beschrieben ist das Einlesen von PDF-Dokumenten nur eingeschränkt möglich. Um beispielsweise auch lokal gespeicherte PDFs nach Referenzen zu durchsuchen, könnte ein zusätzliches Formular zum Upload von Daten eingerichtet werden.

Darüber hinaus kann das Eingabeformular um einen Textbereich erweitert werden, in den Text aus beliebigen Dokumenten manuell eingefügt werden kann. Schlägt beispielsweise die Extraktion von Text aus einem PDF-Dokument fehl, würde so die Analyse des Textes durch Reference Helper dennoch möglich sein.

4.2.3 Erweiterung der verfügbaren Schnittstellen

Bisher sind wie in Abschnitt 3.2.3 aufgeführt Schnittstellen für die die öffentlichen Services von Mendely.com [12] und DBLP [11] implementiert. DBLP umfasst dabei vor allem Publikationen aus dem Bereich der Informatik. Um auch auf Daten anderer Schwerpunkte zugreifen zu können, könnten weitere Schnittstellen realisiert werden.

Literaturverzeichnis

[1] „Institut AIFB,“ [Online]. Available: http://www.aifb.kit.edu/web/Pr%C3%BCfung/Seminare/WS2013/SMW. [Zugriff am 03 Dezember 2013].

[2] „Wikipedia,“ [Online]. Available: http://de.wikipedia.org/wiki/Wiki. [Zugriff am 05 Dezember 2013].

[3] „semantic-mediawiki.org,“ [Online]. Available: http://semantic-mediawiki.org/. [Zugriff am 27 November 2013].

[4] „WebCheatSheet.Com,“ [Online]. Available: http://webcheatsheet.com/php/reading_clean_text_from_pdf.php. [Zugriff am 03 Dezember 2013].

[5] „PDF Parser,“ [Online]. Available: http://www.pdfparser.org/. [Zugriff am 21 November 2013].

[6] I. Councill, C. Lee Giles und M.-Y. Kan, „ParsCit: An open-source CRF Reference String Parsing Package.,“ *Proceedings of LREC*, pp. 661-667, 2008.

[7] „ParsCit,“ [Online]. Available: http://aye.comp.nus.edu.sg/parsCit/. [Zugriff am 03 Dezember 2013].

[8] „FreeCite,“ [Online]. Available: http://freecite.library.brown.edu/. [Zugriff am 03 Dezember 2013].

[9] „d3js,“ [Online]. Available: http://d3js.org/. [Zugriff am 30 November 2013].

[10] „Protovis,“ [Online]. Available: http://mbostock.github.io/protovis/. [Zugriff am 27 November 2013].

[11] „DBLP,“ [Online]. Available: http://dblp.uni-trier.de/. [Zugriff am 01 Dezember 2013].

[12] „Mendeley,“ [Online]. Available: http://www.mendeley.com/. [Zugriff am 14 November 2013].

[13] „MediaWiki,“ [Online]. Available: http://www.mediawiki.org/wiki/Help:Templates/de. [Zugriff am 01 Dezember 2013].

[14] „semantic-mediawiki.org,“ [Online]. Available: http://semantic-mediawiki.org/wiki/Help:Parsererweiterung_set. [Zugriff am 29 November 2013].

[15] „semantic-mediawiki.org,“ [Online]. Available: http://semantic-mediawiki.org/wiki/Inline_query. [Zugriff am 29 November 2013].

Abbildungsverzeichnis

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt, alle benutzten Hilfsmittel vollständig und genau angegeben und alles kenntlich gemacht zu haben, was aus Arbeiten anderer unverändert oder mit Abänderungen entnommen wurde.

(Ort, Datum) (Unterschrift)