Fakultät II: Department für Informatik
Projektgruppe 2011/2012

Abschlussbericht

Version 1.0

Betreuer:
Prof. Dr. Andreas Winter
M.Sc. Jan Jelschen

Mitglieder:

Tore Bierwirth
Christoph Gerken
Marion Gottschalk
Sieglinde Hahn
Maxim Klimenko
Björn Wolff
Christian Wübbeling

Oldenburg, den 30. September 2012

Inhaltsverzeichnis

1. Einleitung

Autor: Marion Gottschalk, Sieglinde Hahn

Dieses Dokument stellt den Abschlussbericht der Projektgruppe (PG) *Clonebusters* dar und soll alle wichtigen Ereignisse, Dokumente und Ergebnisse der PG zusammenfassen, um Außenstehenden einen Einblick über den einjährigen Verlauf der PG zu geben und dabei alle erreichten Ziele aufzuzeigen.

Das Ziel der PG ist die Erstellung eines komponentenbasierten Frameworks zur Erkennung von textuellen Plagiaten in verschiedenen Dokumenttypen wie z.B. PDF und HTML. Die Anwendung zur Plagiatserkennung wird PlagTag genannt. Hierbei wird die Anwenderoberfläche als Webanwendung realisiert und die rechenintensiven Operationen werden auf dem hochschuleigenen HPC-Cluster verteilt ausgeführt. Da ein komponentenbasiertes Framework erstellt wird, ist es möglich die einzelnen Komponenten später auszutauschen bzw. zu erweitern und während der Implementierung eine parallele Entwicklung zu ermöglichen.

1.1. Prozess der Plagiatsüberprüfung

Bevor die Ereignisse, Dokumente und Ergebnisse der PG vorgestellt werden, wird der Prozess der Plagiatsüberprüfung dargestellt, um dem Leser beim Verständnis dieses Dokuments zu unterstützen. Der Prozess der Plagiatsüberprüfung gliedert sich wie folgt in diese fünf Schritte:

1. Bereitstellen mindestens eines Plagiatskandidaten durch den Anwender.
2. Bereitstellen von möglichen Referenzdokumenten durch den Anwender, was optional durchzuführen ist.
3. Automatisierte Internetrecherche mit ermittelten Schlagwörtern aus den Plagiatskandidaten, was ebenfalls optional ist.
4. Plagiatsüberprüfung mit den bereitgestellten Dokumenten und den gefundenen Internetquellen durchführen.
5. Visualisierung des Ergebnisses der Plagiatsüberprüfung und mögliche Widerlegung eines Plagiatsverdachts durch den Anwender.

Ein wichtiger Punkt, der bei diesem Prozess beachtet werden muss, ist, dass die Plagiatskandidaten und die Referenzdokumente in der gleichen Sprache verfasst sind, um eine erfolgreiche Plagiatsüberprüfung durchführen zu können. Weitere Informationen bzgl. des Funktionsumfangs von PlagTag befinden sich im Pflichtenheft der PG und der detaillierte Ablauf der Plagiatsüberprüfung ist im Entwurf der PG dargestellt. Wo sich diese beiden Dokumente befinden und was sie außerdem beinhalten wird im Abschnitt 2.3 beschrieben.

1.2. Überblick

In diesem Dokument wurde bereits der Projektinhalt im ersten Abschnitt 1 vorgestellt. Im Folgenden wird der Projektverlauf in Abschnitt 2 beschrieben werden. Zudem wird eine Übersicht über alle relevanten Dokumente in Abschnitt 2.3 vorgestellt werden. Dies beinhaltet auch die Verweise darauf, wo sich diese Dokumente befinden. Abschließend wird die Zielerreichung der PG in Abschnitt 3.1 betrachtet.

2. Projektbeschreibung

Autor: Sieglinde Hahn

Im Folgenden ist der Verlauf des Projektes dokumentiert, welches zur Software PlagTag beigetragen hat. Zu Beginn wird der *Projektverlauf* 2.1 über den gesamten Zeitraum der PG beschrieben. Daraufhin wird auf durchgeführte *Interviews* 2.2 eingegangen, um auf die Anforderungserhebung einzugehen. Die *Projektdokumentation* ist ein wichtiger Bestandteil bei der Erstellung von PlagTag gewesen und ist daher in Abschnitt 2.3 festgehalten. Bestandteil der PG ist die Durchführung einer Seminarphase, um Erlerntes tiefer Verankern zu können, die gehaltenen *Seminarvorträge* 2.4 sind daher ebenfalls festgehalten. Neben der *Seminarphase* wurden nach Bedarf interne Schulungen gehalten, welche in Abschnitt 2.5 dokumentiert sind.

2.1. Projektverlauf

Das Projekt startete mit der Einarbeitung in das Thema Plagiate, bereits vorhandener Tools zur Plagiatserkennung und dem aktuellen Bezug in der Öffentlichkeit bzgl. von Plagiatsverdachtsfällen. Die PG hat sich zu Beginn damit befasst ein Exposé (siehe Abschnitt 2.3) mit der Motivation, Vision und Zeitplanung des Projektes zu erstellen. Im Anschluss daran wurde begonnen das Pflichtenheft zu erstellen. Im Rahmen dieser Erstellung wurden u.a. Interviews geführt (siehe Abschnitt 2.2). Aufbauend auf den Anforderungen aus dem Pflichtenheft wurde der Entwurf unser Anwendung PlagTag erstellt und angepasst. Die Seminarvorträge (siehe Abschnitt 2.4) wurden parallel zur Projektentwicklung durchgeführt.

Das Vorgehen innerhalb der PG fand während der Entwicklung agil und iterativ statt, wie es in Abbildung 1 dargestellt ist. Zu Beginn der PG richtete sich das Vorgehen nach dem Wasserfallmodell [Bal09], da zuerst ein initiales Pflichtenheft und ein erster Entwurf der Anwendung erstellt wurde. Erst nach der Erstellung dieser zwei Dokumente begann das agile und iterative Vorgehen, wobei das Pflichtenheft und der Entwurf rückwirkend angepasst wurden. Es wurden insgesamt drei Zyklen des Entwurfs durchlaufen, wobei die Phasen *Analyse und Design*, *Implementierung*, *Test und Reflexion* und *Fertigstellung* immer wieder aufgetreten sind. Die Phasen *Analyse und Design*, *Implementierung* und *Test und Reflexion* fanden in jedem Zyklus parallel statt.

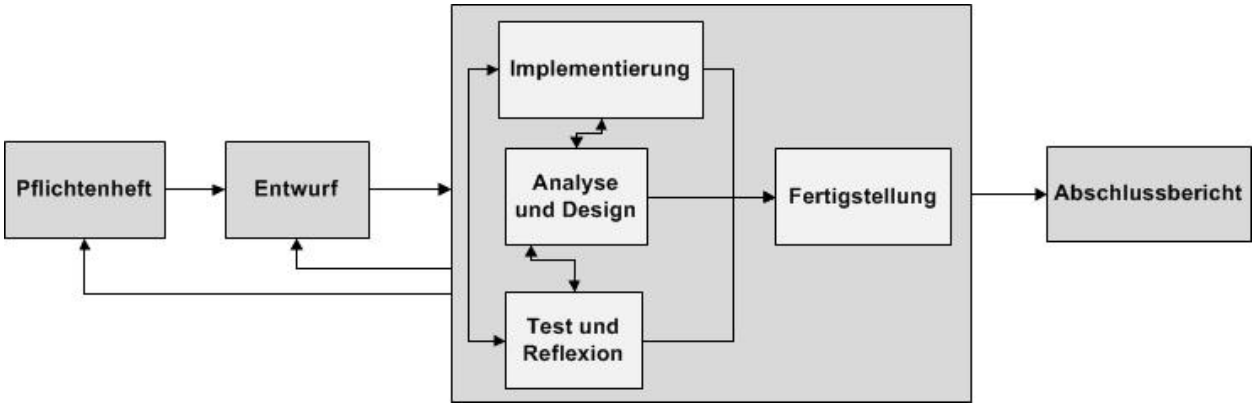

Abbildung 1: Vorgehensmodell

- **Analyse und Design:** In dieser Phase wurden die Anforderungen bestimmt, welche in diesem Zyklus realisiert werden sollten. Dabei wurden die Anforderungen mit der höchsten Priorität, also den Pflichtanforderungen, aus dem Pflichtenheft begonnen.
- **Implementierung:** In dieser Phase werden die einzelnen Komponenten, welche parallel in der Phase *Analyse und Design* bestimmt wurden, realisiert und mit den bestehenden Kom-

ponenten verbunden. Dabei findet ein ständiger Austausch mit der Phase *Test und Reflexion* und *Analyse und Design* statt.

- **Test und Reflexion:** Die Implementierung der einzelnen Komponenten wird in dieser Phase getestet, um die Korrektheit der Komponente zu gewährleisten. Außerdem wird überprüft, ob alle Anforderungen, welche in der Phase *Analyse und Design* bestimmt wurden, entsprechend der Beschreibung im Pflichtenheft realisiert wurden. Gefundene Probleme werden innerhalb der Phase *Implementierung* behoben.
- **Fertigstellung:** In dieser Phase wird die erstellte Funktionalität dem Kunden vorgestellt und übergeben.

Die Arbeitsweise in der PG sollte in vielen Teilbereichen so sein, dass wenig bürokratischer Aufwand und Regeln benötigt werden. Dabei wird auf folgende Gesichtspunkte Wert gelegt:

- effiziente Kommunikation durch ein Ticketsystem und Organisation durch eine Projektleitung,
- Dokumentation so viel wie nötig, so wenig wie möglich (KISS - keep it simple, stupid),
- enge Zusammenarbeit mit dem Auftraggeber (z.B. durch Anwesenheit bei der Mehrzahl der Besprechungen),
- relativ kurze Zeiten zwischen Meilensteinen (meist ca. 6 Wochen) und kundenbezogene Entwicklung, gemäß Anforderungen der Kunden.

Nach jedem Zyklus fand eine Zwischenpräsentation (siehe Phase *Fertigstellung*), um dem Kunden das Zwischenergebnis zu präsentieren und um mögliche Änderungswünsche anzunehmen. Zusätzlich zu den Zwischenpräsentationen hat die PG einen Stand am *Tag der Informatik* vertreten. Des Weiteren gab es für Mike Godfrey, einem Software-Engineering Professor aus Waterloo, welcher sich mit dem Thema *Code Clones* befasst hat, eine Präsentation. Eine ausführliche Dokumentation dieser Ereignisse ist im Fortschrittsbericht zu finden.

Um dem Kunden eine möglichst hohe Qualität liefern zu können, welche dessen Vorstellungen entspricht, wird dafür gesorgt, dass Änderungsanträge zeitnah innerhalb der PG und mit den Kunden auf ihre Realisierbarkeit besprochen werden.

2.2. Interviews

Zu Beginn der PG wurden mehrere Interviews mit unterschiedlichen Interviewpartnern durchgeführt. Die Interviews wurden mit dem Ziel geführt die Anforderungen des Projekts definieren zu können. Die Interviewpartner waren Andreas Winter und Jan Jelschen in der Funktion der Consulter und Betreuer. Jan Jelschen wurde darüber hinaus in seiner Funktion als Entwickler interviewt. Weitere Personen, welche interviewt wurden, waren Reinhard Leidl und Thomas Geuken. Reinhard Leidl ist der Clusterbeauftragte der Universität. In diesem Interview wurde erläutert, wie der HPC-Cluster genutzt werden kann. Thomas Geuken ist der Datenschutzbeauftragte der Universität, welcher uns ausführlich über die Speicherung von Daten informiert hat.

Nachfolgend eine detaillierte Auflistung der geführten Interviews:

- **25.11.2011** - Interviewpartner: Jan Jelschen, Interviewleitung: Christoph Gerken und Protokollführer: Björn Wolff
- **28.11.2011** - Interviewpartner: Andreas Winter, Interviewleitung: Marion Gottschalk und Protokoll: Sieglinde Hahn

- **28.11.2011** - Interviewpartner: Thomas Geuken, Interviewleitung: Christian Wübbeling und Protokoll: Tore Bierwirth
- **12.01.2012** - Interviewpartner: Reinhard Leidl, Interviewleitung: Christoph Gerken und Protokoll: Sieglinde Hahn
- **20.01.2012** - Interviewpartner: Andreas Winter, Interviewleitung und Protokoll: Christian Wübbeling

Zur Vorbereitung der Interviews wurde ein Leitfaden für die Interviewdurchführung gestellt, welcher im Wiki der PG dokumentiert ist. Die geführten Interviews wurden textuell aufbereitet und in das Pflichtenheft übernommen. Aus den geführten Interviews konnten wir zusätzliche Anforderungen an unser Projekt ableiten, umso eine detailliertere Planung und Zielsetzung des Projekts vorzunehmen.

Ergebnisse der Interviews haben zu Änderungen in der Anforderungsdefinition geführt, da z.B. permanentes Speichern von Studienarbeiten gesetzlich nicht möglich ist. Diese Ergebnisse und Einsichten sind ebenfalls dokumentiert und können dem Pflichtenheft Version 6.0 entnommen werden.

2.3. Projektdokumentation

Im Rahmen der PG wurde eine umfangreiche Dokumentation angefertigt um u.a. den Prozess der Entstehung festzuhalten, eine Weiterentwicklung von PlagTag zu ermöglichen und so ein qualitativ hochwertiges Produkt zu schaffen. Dabei werden die Formen *Projektbegleitende Dokumentation*, *Anwenderdokumentation* und *Abschlussdokumentation* unterschieden und hier aufgeführt:

Projektbegleitende Dokumentation: Vor und während der Implementierung von PlagTag wurden verschiedene Dokumente gepflegt, um die Anforderungen von PlagTag festzuhalten und auf den neusten Stand zu halten. Außerdem werden neben der Implementierung entdeckte Fehler festgehalten, um später zu prüfen, dass diese auch behoben wurden.

- **Exposé (Version 2.0)** - Das *Exposé* beinhaltet den inhaltlichen Einstieg in die Thematik der Plagiate. Daher ist dieses Dokument das zuerst erstellte Dokument der PG. In diesem Dokument werden Motivation, Vision und Zeitplanung der PG näher erläutert.
- **Pflichtenheft (Version 6.0)** - Das *Pflichtenheft* beinhaltet u.a. die Vision und das gesetzte Produktziel der PG. Die verschiedenen Plagiatstypen werden beschrieben und abgegrenzt. Konkurrenzanwendungen werden betrachtet und evaluiert, um daraus zusätzliche Anforderungen ableiten zu können. Das Vorgehen in der PG (siehe Abschnitt 2.1) sowie die gesetzten Meilensteine sind ebenfalls in diesem Dokument festgehalten. Vorgestellte Alternativen, des Deployments in der Praxis, wurden bewertet und auf die PG übertragen. Im Falle von möglichen Risiken, welche bei einem Projekt auftreten können, existiert eine Risikoanalyse. Diese Analyse beschreibt mögliche Risikoquellen und Maßnahmen zur Behebung dieser. Darüber hinaus existiert eine allgemeine Beschreibung von PlagTag. Diese Beschreibung geht auf die Produkteinbettung in die Umgebung von PlagTag ein. Mithilfe eines Analysemodels wird der eigentliche Prozess der Plagiatsüberprüfung visualisiert. Allgemeine Anwendungsfälle und konkretisierte Aktivitätsdiagramme beschreiben detailliert den Umfang von PlagTag. Die Maßnahmen zur Unterstützung der Benutzerfreundlichkeit sind ebenfalls dokumentiert. Abschließend sind im Pflichtenheft alle erfassten Anforderungen aufgelistet. Im Anhang des Pflichtenhefts befinden sich ausführliche Interviewprotokolle der geführten Interviews 2.2.
- **Entwurf (Version 5.0)** - Das *Entwurfsdokument* enthält die Architektur von PlagTag und alle getroffenen Entwurfsentscheidungen. Dokumentiert ist hier das Entwurfsvorgehen von PlagTag. Die Auswahl eines Frameworks basierend auf der Wahl einer geeigneten Software-Architektur ist in diesem Dokument ebenfalls dokumentiert. Die Grobarchitektur bildet einen

weiteren Bestandteil im Zusammenspiel mit der Systemzerlegung sowie des Funktionsumfangs von PlagTag. Zu Beginn der Implementierung wurden Machbarkeitsstudien realisiert, welche im Hinblick auf das Zusammenspiel der Komponenten dokumentiert wurden. Erst daraufhin ist die erste Feinarchitektur der Anwendung erstellt worden, welche in jedem Zyklus erweitert wurde. Abschließend ist die Ausführungs-Sicht, also die Anforderungen an die Hardware- und Software-Umgebung von PlagTag, im Entwurf festgehalten.

- **Testdokument (Version 1.0)** - Das *Testdokument* dient der Qualitätssicherung von PlagTag. In diesem Dokument enthalten sind die Grundlagen für das Testen von Software, sowie auch durchgeführte und dokumentierte Tests. Regelmäßig durchgeführte Abnahmetests inklusive der erstellten Testfälle stellen den Kernbestandteil des Testdokumentes dar.

Anwenderdokumentation: Damit der Kunde nach der Übergabe die Anwendung PlagTag schnell und einfach in Betrieb nehmen kann, werden verschiedene Dokumente zur Verwendung und Wartung von PlagTag erstellt.

- **Benutzerhandbuch (Version 1.0)** - Das *Benutzerhandbuch* dient dazu den Einstieg in PlagTag zu erleichtern. Dokumentiert sind hier die ersten Schritte bei der Verwendung von PlagTag (Version 1.0).
- **Installationshandbuch (Version 1.0)** - Das *Installationshandbuch* enthält eine Beschreibung wie PlagTag zu installieren ist. Hier wird speziell auf die Systemvoraussetzungen eingegangen, um PlagTag optimal Nutzen zu können.
- **Schulungshandbuch (Version 1.0)** - Das *Schulungshandbuch* dient als Grundlage zur Einarbeitung neuer Anwender in PlagTag. Um den bestmöglichen Einstieg in die Schulung zu ermöglichen, wurde eine Fallstudie für diese Zwecke erstellt.
- **Wartungshandbuch (Version 1.0)** - Das *Wartungshandbuch* enthält alle Informationen zur Wartung und Weiterentwicklung von PlagTag. Daher sind in diesem Dokument alle Komponenten detailreich beschrieben, um die Austauschfähigkeit der einzelner Komponenten nutzen zu können. Außerdem werden in diesem Dokument mögliche Schwierigkeiten mit dem Framework Tuscany und anderen APIs dokumentiert.

Abschlussdokumentation: Dieser Teil der Dokumentation wurde abschließend zur PG erstellt, um zu überprüfen, ob die Ziele der PG erreicht wurden. Außerdem wird der Verlauf der PG rückblickend betrachtet.

- **Qualitätsbericht (Version 1.0)** - Der *Qualitätsbericht* enthält einen Überblick über die Qualitätserreichung von PlagTag. In diesem Dokument wird auf die erzielten Qualitätsanforderungen eingegangen, welche im konkreten Zusammenhang mit der Zielerreichung stehen.
- **Fortschrittsbericht (Version 1.0)** - Der *Fortschrittsbericht* enthält einen Überblick über die Entwicklung von PlagTag. Festgehalten sind in diesem Dokument einzelne Meilensteine sowie die Zielerreichung der einzelnen Meilensteine. Zusätzlich ist die Priorisierung der Anforderungen hier dokumentiert.
- **Abschlussbericht (Version 1.0)** - Der *Abschlussbericht* enthält eine Zusammenfassung des gesamten Projekts, um einen Überblick über den Projektverlauf und die Zielerreichung zu geben. Hierbei handelt es sich um das vorliegende Dokument.

Alle Versionen der Dokumente befinden sich im Subversion-Repository im Abgabeordner. Dokumente in der aktuellsten Versionen sind zusätzlich auf der PG-Webseite zu finden[1]. Neben diesen

[1] Internetpräsenz der PG: http://www.se.uni-oldenburg.de/preprocess.php?seite=45624.html&include0=generated-content/clonebustersAbschluss.html&clearAll

Dokumenten wurde ein Wiki gepflegt[2]. In diesem Wiki sind Projekthandbuch, Liste mit wechselnden Aufgaben, Ferienplanungen, Sitzungsprotokolle, Interviewprotokolle und gesammelte Informationen der PG-Mitglieder zu finden, welche in den oben genannten Dokumenten verarbeitet wurden. Speziell zu erwähnen ist das Projekthandbuch, in diesem sind die Spielregeln der PG festgehalten so wie Zuständigkeiten innerhalb der PG. Das Wiki befindet sich in dem Tracking system *Trac*, welches außerdem ein Ticketsystems bereitstellt, mit dem die Ressourcen der PG-Mitglieder bestmöglich verteilen wurden.

2.4. Seminarvorträge

Die Seminarphase der PG fand nicht in Form einer Blockveranstaltung statt, wie es eigentlich üblich ist. Es wurde sich dazu entschieden, dass die Seminarvorträge parallel zur Projektverwirklichung zeitnah bei Bedarf gehalten werden, um so die wichtigsten Themen für die PG identifizieren zu können, was zu Beginn der PG noch nicht vollständig möglich war. Im Folgenden werden die vorgestellten Seminarvorträge kurz im Hinblick auf den Nutzen für den weiteren Verlauf der PG beschrieben.

- 16.01.2012 **Visualisierung** - Der erste Seminarvortrag von Marion Gottschalk bezog sich auf die Visualisierung gängiger Plagiatserkennungstools. Anhand dieser Präsentation konnten wir für den weiteren Verlauf des Projekts Visualisierungsmöglichkeiten in unsere Planung aufnehmen.
- 30.01.2012 **Softwarearchitektur** - Vor Beginn der Entwurfsphase wurde eine Präsentation zum Thema Softwarearchitektur von Tore Bierwirth und Christian Wübbeling gehalten, speziell behandelt wurden hier OSGi und SCA. Durch diese Präsentation wurde in der PG die Entscheidung getroffen SCA zu verwenden, um die Anforderung der Komponentenbasiertheit umsetzen zu können.
- 29.02.2012 **Algorithmen zur Plagiatserkennung** - Die Algorithmen-Präsentation hat einen Überblick über bereits vorhandene Algorithmen der Plagiatserkennung verschafft. Durch diese Vorstellung haben Maxim Klimenko und Björn Wolff dazu beigetragen Algorithmen für die Entwicklung von PlagTag identifiziert zu können.
- 30.07.2012 **Cluster** - Der Seminarvortrag zum HPC-Cluster von Christoph Gerken diente dazu einen Überblick über die Anbindungsmöglichkeiten zu geben. Des Weiteren konnte mittels der Vorstellung erkannt werden, dass die Komponente Plagiatserkennung angebunden werden sollte.
- 09.08.2012 **Deployment** - Der letzte Seminarvortrag stellt das Thema Deployment dar. Sieglinde Hahn stellte Beispiele aus der Praxis bzgl. Projektdokumentationen vor, welche an die PG angepasst wurden, umso einen Einstieg für die zu erbringenden Abschlussdokumention zu geben.

Alle Seminarvorträge wurden im Anschluss dokumentiert und im Pflichtenheft oder Entwurf eingebettet. Des Weiteren sind alle Präsentationen im Subversion-Repository zu finden.

2.5. Schulungen

Im Rahmen des Erlernens und der Aneignung von neuen Wissensbereichen, wurden in der PG neben den Seminarvorträgen interne Schulungen durchgeführt. Zu diesen Schulungen zählen:

- 06.10.2011 **Tools zur Plagiatsüberprüfung** - Um einen Überblick über die derzeit vorhandenen Plagiatserkennungs-Tools zu erhalten, hat Marion Gottschalk zu Beginn der PG eine kurze Übersicht erstellt.

[2]Wiki der PG: `http://sehome.informatik.uni-oldenburg.de:8080/`

- 06.10.2011 **Architekturen** - Direkt zu Beginn der PG wurden mögliche Architekturen von Christoph Gerken vorgestellt, welche sich zur Erstellung einer Plagiatserkennungssoftware eignen.
- 13.10.2011 **Plagiatstypen** - Um das Thema Plagiate besser verstehen und abgrenzen zu können, wurde zu diesem Thema eine interne Schulung von Maxim Klimenko gegeben.
- 16.04.2012 **Apache Tuscany** - Christian Wübbeling stellte der Gruppe die Verwendung von Apache Tuscany vor.
- 10.05.2012 **JUnit** - Die Qualitätssicherung ist ein wichtiger Bestandteil, daher wurde in diesem Rahmen eine JUnit-Schulung von Marion Gottschalk und Sieglinde Hahn gegeben. Diese Schulung sollte dabei dienen, eine mögliche Testvariante näher zu erläutern.
- 14.06.2012 **Code Review** - Eine weitere Schulung im Rahmen der Qualitätssicherung wurde von Marion Gottschalk zu dem Thema Code Review gegeben.
- 13.09.2012 **Apache Tomcat** - Christian Wübbeling führte im Rahmen einer Schulung in das Thema Apache Tomcat ein.

3. Ziel der Projektgruppe

Autor: Sieglinde Hahn, Marion Gottschalk

In diesem letzten Kapitel sollen die Erfolge der PG aufgegriffen und die Erreichung der Ziele für die Anwendung PlagTag in Abschnitt 3.1 aufgezeigt werden. Dabei werden einzelne Funktionalitäten von PlagTag hervorgehoben, welche PlagTag im Vergleich zu anderen Anwendungen auszeichnet. Außerdem werden in Abschnitt 3.2 die Möglichkeiten beschrieben, wie PlagTag in einer nächsten Version erweitert und damit zusätzlich verbessert werden könnte.

3.1. Zielerreichung

PlagTag ist, wie bereits erwähnt, ein komponentenbasiertes Framework zur Plagiatserkennung, welches als Webanwendung in das Uni-Layout eingebettet ist. Der Anwender hat die Möglichkeit über die Benutzeroberfläche verschiedene Algorithmen zur Plagiatsüberprüfung sowie verschiedene Vor- und Nachbereitungen wie u.a. Stoppwort-Entfernung und Wortersetzung der Dokumente auszuwählen, um seinen Plagiatsüberprüfungsprozess zu optimieren.

Neben den zu überprüfenden Plagiatskandidaten und den selbst hochgeladenen Referenzdokumenten kann der Anwender einer automatischen Internetrecherche zustimmen um die daraus resultierenden Ergebnisse als weitere Referenzdaten mit in die Plagiatsüberprüfung einzubeziehen. Eine Besonderheit hierbei ist, dass PlagTag die Sprache des Plagiatskandidaten automatisch erkennt und dies bei der Internetrecherche berücksichtigt, sodass nur Internetquellen gefunden werden, welche die gleiche Sprache wie der Plagiatskandidat besitzt. Die gefundenen Internetquellen werden nicht direkt mit in die Plagiatsüberprüfungen einbezogen, sondern zuerst dem Anwender aufgelistet, sodass dieser die gefundenen Internetquellen zuvor sichten und selbst entscheiden kann, ob die gefunden Internetquellen relevant für die Plagiatskandidaten sind. Nachdem alle relevanten Referenzdaten ausgewählt und hochgeladen sind, kann der Anwender zusätzliche Informationen zu den Dokumenten angeben. Dabei handelt es sich, um die Autoren und die Sprache der Dokumente. Falls die Sprache nicht angegeben wird, wird diese ebenfalls von PlagTag automatisch identifizieren.

Nachdem die Plagiatsüberprüfung von PlagTag durchgeführt wurde, kann der Anwender gefundene Plagiate verifizieren, d.h. der Anwender entscheidet, ob es sich bei den gefundenen Plagiaten wirklich um ein Plagiat handelt. Im Anschluss an die Verifizierung ist es möglich ein Ergebnisdokument zu exportieren, siehe Abbildung 2. Ein Ergebnisdokument enthält neben den gefundenen Plagiaten eine grafische Darstellung in Form eines Kuchendiagramms und eines Strichcodes. Diese beiden Visualisierungsformen geben das Verhältnis von plagiierten Textstellen zu nicht plagiierten Textstellen in einem Dokument an, sowie die Position der plagiierten Textstellen. Das Ergebnisdokument ist so aufgebaut, dass zu Beginn des Dokuments eine Erklärung zum Ergebnisdokument gegeben wird. Durch diese Erklärung wird dem Anwender deutlich gemacht, welche Inhalte das Ergebnisdokument enthält und welche Algorithmen verwendet wurden. Im Ergebnisdokument werden neben der grafischen Darstellung die gefundenen Plagiate mit dem entsprechenden Referenzdaten gegenübergestellt, siehe Abbildung 3.

Als letztes kann der Anwender die Plagiatsüberprüfung abschließen. Dies bedeutet, dass die Plagiatskandidaten und alle Referenzdaten aus der Datenhaltung entfernt werden und nur die gefundenen Plagiate mit deren Textstelle erhalten bleiben, wie es rechtlich gefordert ist. Wird die Plagiatsüberprüfung nicht selbst vom Anwender abgeschlossen, so ist PlagTag in der Lage dies zu erkennen und das Abschließen der Plagiatsüberprüfung nach 30 Tagen automatisiert vorzunehmen.

Somit wurde das Ziel erreicht PlagTag als eine qualitativ hochwertige Anwendung zu entwickeln, welches neben Anforderungen mit A-Prioritäten zusätzliche Anforderungen erfüllt. Außerdem konnte

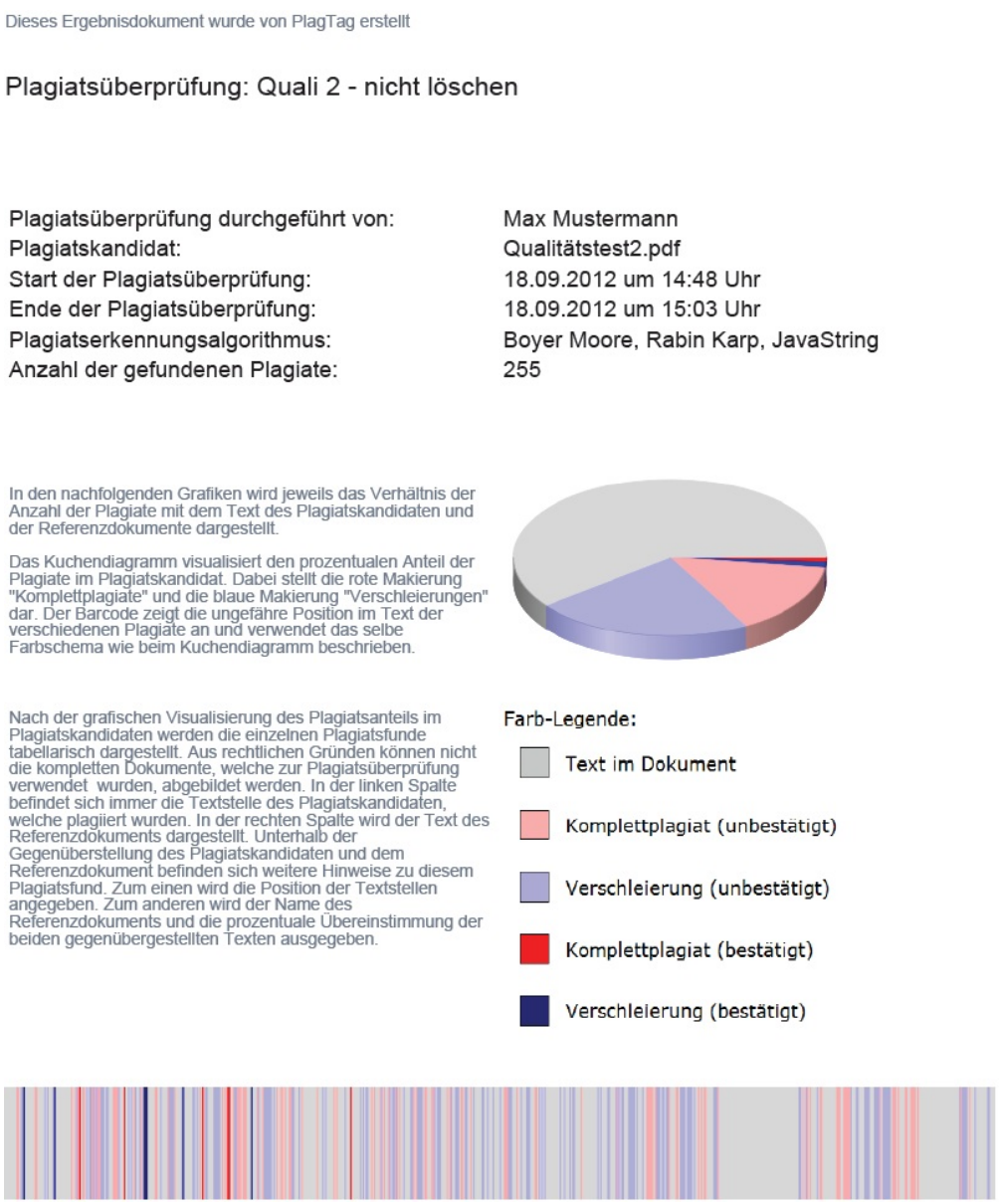

Abbildung 2: Ergebnisdokument - Visualisierung

erreicht werden, dass die Anwendung durch eine einfache Konfigurierbarkeit in jeder Komponente veränderbar und erweiterbar ist. Diese Konfigurierbarkeit ist im Wartungshandbuch (siehe Abschnitt 2.3) detailliert dokumentiert. Zusätzlich ist die Anwendung PlagTag überdurchschnittlich gut dokumentiert und kann somit von jedem Anwender bedient und ebenfalls durch andere Entwickler ergänzt bzw. gewartet werden. Die Weiterentwicklung von PlagTag kann auf Grundlage des Wartungshandbuchs vorgenommen werden. Hierfür stellt ebenfalls die Austauschbarkeit der Komponenten einen großen Vorteil dar, da die Kommunikationen zwischen den Komponenten mittels eines SCA-Framworks erstellt wurde. Dem eigentlichen Anwender wird durch ein Installations-, Benutzer- und Schulungshandbuch die Möglichkeit geboten sich intensiv mit PlagTag vertraut zu machen.

Neben den eigentlichen Anforderungen wurden zusätzlich zeitgemäße Sicherheitsmaßnahmen für SQL Injektion und Cross Site Scripting eingesetzt. Die Bedienung des Anwender-Interface ist, durch kurze und prägnante Informationstexte auf den jeweiligen Seiten, intuitiv gehalten. Die Bedienbarkeit wird zusätzlich durch ein detailgetreues Benutzerhandbuch gewährleistet. Bei PlagTag handelt

Plagiatskandidat	Referenzdokument
verfassungsmäßigeOrganisation europäischer Macht folglich ausrichten sollte. *Position: 937 - 1012*	Organisation europäischer Macht folglich ausrichten soll. *Referenzdokument: 2002_wp_eu_verfassung.pdf; Position: 86067 - 86123; Übereinstimmung: 100% (Komplettplagiat)*
vorschlagen, ohne zuvor über den politischen Sinn, den man Europa zu verleihen *Position: 1076 - 1153*	vorschlagen, ohne zuvor über den politischen Sinn, den man Europa zu verleihen *Referenzdokument: 2002_wp_eu_verfassung.pdf; Position: 86177 - 86254; Übereinstimmung: 100% (Komplettplagiat)*
Man kann keine Verfassung schaffen, ohne zu wissen, was verfasst werden soll".258 *Position: 1264 - 1344*	„Man kann keine Verfassung schaffen, ohne zu wissen, was verfasst werden soll"119 *Referenzdokument: 2002_wp_eu_verfassung.pdf; Position: 86716 - 86796; Übereinstimmung: 92% (Verschleierung)*

Abbildung 3: Ergebnisdokument - Plagiatsgegenüberstellung

es sich nicht allein um ein Hochschulprojekt, sondern um ein Anwendung, welches direkt in den Endbenutzereinsatz gehen kann, denn die Beachtung der rechtlichen Anforderungen macht PlagTag zu einer zukunftssicheren Anwendung. Die Integration des HPC-Cluster hilft dabei größere Dokumente, wie z.B. die Dissertation von Karl-Theodor zu Guttenberg, zu berechnen und erzielt durch die verteilte Berechnung deutlich schnellere Ergebnisse als geprüfte Konkurrenzprodukte. Das Konkurrenzprodukt *Ephorus*, dokumentiert im Pflichtenheft, welches bis vor kurzem an der Universität Oldenburg zum Einsatz gekommen ist, hat ca. eine Woche benötigt, um mitzuteilen, dass keine Plagiate erkannt wurden. Im Vergleich dazu kann PlagTag eine Plagiatsüberprüfung der Dissertation von zu Guttenberg innerhalb von zwei Stunden durchführen, welches detaillierter in den Leistungsanforderungen im Qualitätsbericht festgehalten ist.

Zusammenfassend sind hier die besonderen Features von PlagTag aufgeführt, welche sich von anderen Plagiatserkennungstools abheben. Diese Features sind:

- HPC-Cluster-Anbindung zur schnelleren Durchführung der Plagiatsüberprüfungen,
- die Verwendung mehrerer Algorithmen zur Plagiatserkennung, um ein genaueres Ergebnis zu erzielen,
- eine automatische Spracherkennung, um die Internetsuche zu verbessern,
- Einbeziehung von Internetquellen neben selbst ausgewählten Referenzdokumente,
- exportierbares Ergebnisdokument, welches neben zwei Visualisierungsformen eine Gegenüberstellung von gefundenem Plagiaten und seiner Referenzdaten bietet,
- die Austauschbarkeit von Komponenten für die Erweiterung der Software,
- detaillierte Dokumentation von PlagTag und
- Einhaltung der Datenschutzbestimmungen und des Urheberrechts.

3.2. Ausblick

Innerhalb der einjährigen Projektdauer wurden noch weitere Ideen und Anforderungen festgehalten, welche die Funktionen von PlagTag erweitern und verbessern könnten. Diese Anforderungen sind genau wie die realisierten Anforderungen im Pflichtenheft festgehalten, um eine erste Erweiterbarkeit von PlagTag zu erleichtern.

Neben den bereits realisierten Algorithmen besteht natürlich die Möglichkeit weitere Algorithmen in den Plagiatsüberprüfungsprozess zu integrieren, wie dies genau möglich ist, wird im Wartungshandbuch beschrieben. Weitere Optimierungsmöglichkeiten bietet der Einsatz vom Erkennen von verschiedenen Zitierweisen bei der Plagiatsüberprüfung. Aktuell ist die manuelle Verifizierung von gefundenen Plagiaten möglich, eine Erweiterung durch das automatische Erkennen von korrekt zitierten Textstellen, würde die Anzahl der False Positives von PlagTag verringern und damit den Verifizierungsprozess für den Anwender erleichtern. Dieses Verfahren könnte in Form der Erweiterung einer Komponente in den Prozessablauf integriert werden. Außerdem wäre die Verwendung des Literaturverzeichnis für die Referenzsuche bei der Internetrecherche von Vorteil, um so das Ergebnis der Internetrecherche zu verfeinern.

A. Abbildungsverzeichnis

B. Literaturverzeichnis

[Bal09] Balzert, Helmut ; Rüdinger, Andreas (Hrsg.) ; Kohl, Kerstin (Hrsg.) ; Fraude, Dagmar (Hrsg.): *Lehrbuch der Softwaretechnik: Basiskonzepte und Requirements Engineering*. Spektrum, 2009

[BBK01] Back, A. ; Becker, J. ; König, W. ; Stürken, M. (Hrsg.): *Lexikon der Wirtschaftsinformatik*. Springer, 2001

[CSFP] Collins-Sussman, B. ; Fitzpatrick, B. W. ; Pilato, M.: *Version Control with Subversion*. http://svnbook.red-bean.com/. – Letzter Zugriff am 08.08.2012

[LCF+11] Laws, Simon ; Combellack, Mark ; Feng, Raymond ; Mahbod, Haleh ; Nash, Simon: *Tuscany SCA in Action*. Manning Publications Co., 2011

[Lin05] Link, Johannes: *Softwaretests mit JUnit - Techniken der testgetriebenen Entwicklung*. Dpunkt Verlag, 2005

[OAS] OASIS: *OASIS Service Component Architecture / Assembly*. http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=sca-assembly. – Zugriff am 23.04.2012

[Teca] TechTerms: *API*. http://www.techterms.com/definition/api. – Letzter Zugriff am 08.08.2012

[Tecb] TechTerms: *PDF*. http://www.techterms.com/definition/pdf. – Letzter Zugriff am 08.08.2012

[w3s] w3schools: *Introduction to SQL*. http://www.w3schools.com/sql/sql_intro.asp. – Letzter Zugriff am 08.08.2012

[Wes] Westphal, Frank: *Unit Tests mit JUnit*. http://www.frankwestphal.de/UnitTestingmitJUnit.html. – Zugriff am 05.04.2012

[WHKL08] Wütherich, Gerd ; Hartmann, Nils ; Kolb, Bernd ; Lübken, Matthias: *Die OSGi Service Platform-Eine Einführung mit Eclipse Equinox*. dpunkt Verlag, 2008

C. Glossar

API API steht für die Abkürzung *Application Programming Interface*. Dies ist ein Programmteil, welcher von einem Softwaresystem für ein anders System zur Verfügung gestellt wird [Teca].

Code Review Das Code Review beschreibt ein Vorgehen, indem Programmabschnitte manuell von einem anderen Programmierer überprüft werden [Lin05].

Datenhaltung Die Datenhaltung dient für die Anbindung an eine Datenbank, wobei noch nicht entschieden wurde, welche Datenbank verwendet werden soll.

Ergebnisdokument Das Ergebnisdokument speichert als Prüfbericht die Informationen aus der Plagiatsüberprüfung eines Plagiatskandidaten. Zu den Informationen gehören: Anzahl der Plagiate, wo wurden die Plagiate gefunden und welche Referenzdaten bzw. Internetquellen wurden plagiiert.

Erkennungs-Algorithmen Die Erkennungs-Algorithmen sind Algorithmen zur Erkennung von Plagiate. Diese sind derzeit noch nicht bestimmt.

False Positive Ein False Positive ist in dieser Anwendung ein erkanntes Plagiat, welches aber in Wirklichkeit keins ist, da es korrekt zitiert wurde.

HPC-Cluster Ein High Performance Computing Cluster (HPC Cluster oder vereinfacht *Cluster*) stellt ein Synonym für einen Höchstleistungsrechner dar, der verteilt komplexe Aufgaben bewältigt.

HTML Hyper Text Markup Language ist eine Web-Auszeichnungssprache, ist Grundlage des World Wide Web und wird von einem Webbrowser dargestellt. Aktuell befindet sich HTML Version 5 in der Standardisierung.

Internetquelle Die Internetquellen werden durch eine automatische Internetrecherche durch PlagTag gefunden. Dabei soll PlagTag Webseiten finden, die durch den Ersteller des Plagiatskandidat plagiiert wurden.

Internetrecherche Die Internetrecherche ist eine Komponente der Anwendung PlagTag. Diese dient dem Finden von Referenzdaten im Internet.

JUnit JUnit ist ein Framework für Unit-Tests in Java. Es ist für die automatisierte Unit-Tests einzelner Units geeignet [Wes].

Komponente Eine Komponente (hier: Software-Komponente) ist ein Software-Element, das konform zu einem Komponentenmodell (z.B. aus der UML) ist und über Schnittstellen mit anderen Komponenten verknüpft und ausgeführt werden kann. Komponentenbasierte Software wird z.B. mittels Web-Services, CORBA, Enterprise Java Beans (EJBs) oder COM entwickelt.

OSGi OSGi Alliance ist eine hardwareunabhängige dynamische Softwareplattform, die es erleichtert, Anwendungen und ihre Dienste per Komponentenmodell (*Bundle*/*Service*) zu modularisieren und zu verwalten (*Service Registry*). Die OSGi-Plattform setzt eine Java Virtual Machine (JVM) voraus und bietet darauf aufbauend das OSGi-Programmiergerüst [WHKL08].

PDF Das Portable Document Format (PDF) ist ein plattformunabhängiges Dateiformat für Dokumente, das vom Unternehmen Adobe Systems entwickelt und 1993 veröffentlicht wurde. Ziel ist die originalgetreue Wiedergabe auf allen Plattformen [Tecb].

Plagiatserkennung Die Plagiatserkennung meint das Ausführen der Erkennungs-Algorithmen auf einen Computer, um Plagiate zu identifizieren.

PlagTag PlagTag ist der Name der Plagiatserkennungssoftware.

Plagiat Ein Plagiat liegt dann vor, wenn jemand vorgibt, ein Werk selbst erstellt zu haben, obwohl die Inhalte ganz oder teilweise aus fremden Quellen stammen, die falsch oder unvollständig zitiert wurden. Je nachdem wie die Zitiertechnik falsch verwendet wurde, lassen sich die Plagiate zusätzlich genauer kategorisieren Je nachdem wie die Zitiertechnik falsch verwendet wurde, lassen sich die Plagiate zusätzlich genauer kategorisieren. Es existieren vielfältige Formen geistiger Leistung, z.B. Texte, Bilder oder Videos.

Plagiatskandidat Ein Plagiatskandidat ist ein Dokument, welches vom Anwender in das System eingefügt und dort auf Plagiate überprüft wird.

Plagiatsüberprüfung Die Plagiatsüberprüfung beschreibt das Vorgehen in PlagTag, welches den gesamten Ablauf der Anwendung ab dem Anlegen einer Überprüfung vom Anwender bis zur Ausgabe des Ergebnisses der Überprüfung meint.

Referenzdaten Die Referenzdaten sind der Sammelbegriff für Referenzdokumente und Internetquellen.

Referenzdokument Die Referenzdokumente sind die Dokumente, welche durch den Anwender zusätzlich neben dem Plagiatskandidat in das System eingefügt werden, um genau diese Dokumente mit dem Plagiatskandidat zu vergleichen.

SCA Die Service Component Architecture (SCA) ist eine Sammlung an Spezifikationen, welche ein Modell einer Serviceorientierten Architektur (SOA) beschreiben. SCA basiert auf offenen Standards wie Web Services. SCA-Komponenten sind unabhängig von einer konkreten Technologie. [OAS].

SQL Die Abkürzung SQL steht für *Structured Query Language*. Es ist eine Sprache für Datenbankabfragen [w3s].

Stoppwort Stoppwörter sind gewöhnliche oft vorkommende Wörter in einem Text z.B. *Artikel ('der', 'eine')*, *Konjunktionen ('und', 'oder')* usw.

Strichcode Mithilfe eines Strichcodes werden die Menge und die Position der Plagiate im Text deutlich. Dabei werden verschiedene Farben verwendet, wobei jede Farbe für eine andere plagiierte Textpassage steht, somit wird auch die Länge der Plagiat anhand des Strichcodes deutlich.

Subversion-Repository Subversion ist eine Software zur Versionsverwaltung von Dokumenten und Programmen [CSFP]. Repository beschreibt das Verzeichnis, in dem Dokumente und Programme mit zusätzlichen Metadaten gespeichert werden, um eine Versionverwaltung zu ermöglichen [BBK01].

System Wird der System-Begriff ohne explizite Einschränkungen verwendet, so bezieht sich dieser immer auf das im Projekt zu entwickelnde Gesamtprodukt. Dieses teilt sich auf in eine zentrale Berechnungs-Komponente und ein angebundenes Webfrontend.

Tuscany Tuscany stellt ein kostenloses Framework zur Nutzung von SCA dar [LCF+11].

Wiki Eine Plattform, welche die Verwaltung von Dokumenten über das Internet ermöglicht.