Friedrich-Alexander-Universität Erlangen-Nürnberg
Technische Fakultät, Department Informatik

VANESSA SCHINDLER
BACHELOR THESIS

ZUSAMMENARBEIT IN WIKIS
AM BEISPIEL VON ENTWURFSMUSTERN

Eingereicht am 1. April 2015

Betreuer: Prof. Dr. Dirk Riehle, M.B.A.; Dipl.-Inf. Hannes Dohrn
Professur für Open-Source-Software
Department Informatik, Technische Fakultät
Friedrich-Alexander-Universität Erlangen-Nürnberg

Versicherung

Ich versichere, dass ich die Arbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat und von dieser als Teil einer Prüfungsleistung angenommen wurde. Alle Ausführungen, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

Erlangen, 1. April 2015

License

Erlangen, 1. April 2015

Abstract

Collaboration has changed because of the Internet. Large projects are more often performed on public online platforms. Even whole encyclopedias or large scale software projects are developed by multiple remote actors in a collaborative way. Wikis are frequently used as an online reference plattform of informations, and there are two types: public and private enterprise Wikis. Currently GitHub (a collaborative software development platform) is widely used. It provides various ways of collaborative work - based on Git (a source code control system). In this thesis I examine different forms of collaboration based on wikis and GitHub, in order to evaluate a combination of both paradigms. I will examine feasibility of identified combinations against a theoretical model, and I am using different user group definitions and suitable Usecase diagrams to visualize the requirements. That way I can give an accurate impression of the theoretical model.

Zusammenfassung

Zusammenarbeit wurde durch das Internet stark verändert und so werden große Projekte, immer häufiger, auf öffentlichen Online Plattformen durchgeführt. Ganze Enzyklopädien werden erzeugt oder Softwareprojekte durchgeführt, durch viele, räumlich getrennte, Akteure, die kollaborieren. Wikis werden dabei häufig als Nachschlagewerke verwendet, wobei es vor allem zwei Arten gibt: öffentliche, über das Internet erreichbare, und private Enterprise Wikis. In der Softwareentwicklung ist derzeit GitHub, eine Softwareentwicklungsplattform, für gemeinschaftliches entwickeln weit verbreitet und bietet verschiedene Arbeitsweisen aufbauend auf Git (ein Versionsverwaltungssystem) an. In dieser Arbeit untersuche ich die Zusammenarbeitsformen, die in Wikis und GitHub vorherrschen, um darauf aufbauend ein Modell zu entwickeln. Ob eine Kombination beider Arbeitsweisen sinnvoll ist, wird ebenfalls durch Diskussion des theoretischen Modells erfolgen. Die Anforderungen der Benutzergruppen visualisiere ich dabei mit Use-Case Diagrammen, die einen genauen Eindruck vom System erzeugen sollen.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einführung

1.1 Original thesis goals

Ursprünglich war geplant das entwickelte theoretische Modell, auch in einer Implementierung, umzusetzen. Das theoretische Modell soll aufbauend, auf den Erkenntnissen der Zusammenarbeit in Wikis und GitHub, entwickelt werden. Mit Testfällen sollte die Implementierung anschließend überprüft werden.

1.2 Changes to thesis goals

Statt der Implementierung des Systems wird nun ein theoretisches Modell, aus den Erkenntnissen der Zusammenarbeit, entwickelt und mit Anforderungsfällen visualisiert. Die Anforderungen werden dabei mit Use-Case Diagrammen dargestellt.

2 Forschung

2.1 Einführung

Im folgenden Teil zeige ich auf wie die Zusammenarbeit, in öffentlichen und privaten Wikis, aufgebaut ist und wie die Organisationsprozesse die Qualität der Artikel beeinflusst. Anschließend werde ich auf die Zusammenarbeit in GitHub eingehen. Dieses Wissen dient als Grundlage, um anschließend ein Modell für eine eigene Plattform zu entwickeln, welche Wikis auf einer Plattform hosten soll. Für diese Plattform habe ich bereits das Buch, "Entwurfsmuster: Elemente wiederverwendbarer objektorientierer Software" von Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, in Wikitext übertragen, so dass bereits Softwaremuster auf der zukünftigen Plattform zu finden sein werden. Dieses Buch zeigt wiederverwendbare Muster, für häufige Programmieraufgaben, auf. Ab Kapitel 2.5, werde ich auch die Benutzeranforderungen, mittels Use-Case Diagrammen, herausarbeiten. Die Diskussion und Ausblick der Plattform bilden den Abschluss des Kapitels.

2.2 Verwandte Arbeiten

In dem Buch, "Erfolgsfaktoren von Social Media: Wie "funktionieren" Wikis?" von Mayer (2013), wird unter anderem untersucht wie die Kollaboration in Wikis durchgeführt und diese beeinflusst wird. Auch in dem Buch, "Wiki: Kooperation im Web" von Ebersbach et al. (2008), behandeln die Autoren Zusammenarbeit in Wikis, beziehen sich aber hauptsächlich auf konkrete Beispiele, wie beispielsweise das MediaWiki. In dem Paper, "Studying Cooperation and Conflict between Authors with history flow Visualizations" von Viégas et al. (2004), wird die Zusammenarbeit mit einer neuen Methode, beispielhaft an Wikipedia, analysiert, der "history flow visualization", die Zusammenarbeitsmuster aufdecken konnte. In dem Artikel, "Innovating Collaborative Content Creation: The Role of Altruism and Wiki Technology" von Wagner und Prasarnphanich (2007), werden die Gründe und Motivation behandelt, die Menschen dazu veranlassen an einem Projekt, wie einem Wiki, mitzuarbeiten. Diese Literatur analysieren die Zusammenarbeit in Wikis aus unterschiedlichen Perspektiven und fließen damit auch in die Exploration der bestehenden Zusammenarbeit, ab Kapitel 2.4.1, hinein.

Zur Zusammenarbeit auf GitHub gibt es eine Fülle von Artikeln, die den GitHub Workflow, analysieren und deren Erfolg untersuchen. So wurden im Artikel, "The Promises and Perils of Mining GitHub" von Kalliamvakou, Singer et al. (2014)

untersucht, wie Nutzer GitHub verwenden und welche Aktivitäten in Repositories durchgeführt werden. “Open Source-Style Collaborative Development Practices in Commercial Projects Using GitHub“ Kalliamvakou et al. (2015) und “The Code-Centric Collaboration Perspective: Evidence from GitHub“ Kalliamvakou, Damian et al. (2014), behandeln beide die Zusammenarbeit in Entwicklerteams und wie diese durch GitHub unterstützt werden. Das erste Paper konzentriert sich dabei auf kommerzielle Projekte, die bei GitHub kostenpflichtig sind, aber dennoch zumeist einen Workflow, wie Open-Source-Projekte, einsetzen. Der Artikel, “Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository“ von Dabbish et al. (2012), untersucht die sozialen Komponenten von GitHub, die einen starken Einfluss auf die Kollaboration auf der Plattform haben. Durch Verständnis der Zusammenarbeit in GitHub, wird auch die Grundlage für ein eigenes Modell gelegt. Dieses Modell wird sich ähnlich wie GitHub verhalten.

Es gibt demnach eine große Anzahl an Literatur, die sich mit Kollaboration im Web, im speziellen mit Wikis und GitHub, beschäftigen, jedoch im Zusammenhang wurden diese beiden Plattform bisher nicht, oder kaum, bearbeitet.

2.3 Fragestellung

In dieser Bachelorarbeit soll die Frage beantwortet werden, wie die Zusammenarbeit, vor allem in Wikis und GitHub, im Internet durchgeführt wird. Diese Frage soll durch eine Literaturrecherche beantwortet werden. Außerdem soll herausgefunden werden, ob darauf aufbauend ein theoretisches Modell entwickelt werden kann, welches beide Formen vereint.

2.4 Exploration der bestehenden Zusammenarbeit

In diesem Kapitel betrachte ich die Zusammenarbeit in Wikis und GitHub, wobei allgemeine Zusammenarbeitsmodelle mit Git, in Kapitel 3.3, aufgezeigt werden. In Kapitel 3.1, der Ausarbeitung, stelle ich allgemein die Kollaboration im Internet dar und vergleiche Wikis mit cloudbasierten Kollaborationsmöglichkeiten. Auch auf die Unterschiede zwischen Kooperation und Kollaboration gehe ich, in Kapitel 3.2, genauer ein.

2.4.1 Zusammenarbeit in Wikis

Ward Cunningham, entwickelte 1995 das erste Wiki, welches auch heute unter `http://c2.com/cgi/wiki` erreichbar ist. Das ursprüngliche Ziel war es, die gemeinschaftliche Softwareentwicklung durch editierbare Seiten zu vereinfachen, die zusätzlich die komplette Historie speichern. Dieser Anwendungsbereich hat sich heute erweitert. Ebersbach et al. (2008, S. 14), definiert Wikis auf folgende Weise:

> "Ein Wiki ist eine webbasierte Software, die es allen Betrachtern einer Seite erlaubt, den Inhalt zu ändern, indem sie diese Seite online im Browser editieren."

Themenbezogene Wikis, wie auch das erste Wiki, sind über das Internet für jeden erreichbar. Es gibt Fan-Wikis, die sich zum Beispiel einer Serie widmen, das erste Wiki, mit Schwerpunkt Softwareentwicklung und Wikipedia eine freie Enzyklopädie. Fan-Wikis und andere Wikis, die sich hauptsächlich einem Thema widmen, gehen tiefer auf ihre Thematik ein, als es bei Wikipedia möglich wäre, da Wikipedia eine große Bandbreite an Themen bietet. (Ebersbach et al., 2008) Wiki Farmen, wie beispielsweise Wikia, bieten ein einfaches einrichten eines eigenen Wikis. Die Plattform ermöglicht einen Austausch über eine Fülle von Themen, die vor allem im Bereich, Entertainment, Videospiele und Lifestyle, angesiedelt sind (Mayer, 2013).

Wikis besitzen eine Reihe von Charakteristika, die sie von anderen Webseiten, unterscheiden. Für einen möglichst reibungslosen Ablauf der Zusammenarbeit, werden alle Versionen einer Seite automatisch gespeichert, um im Falle von Vandalismus, oder falscher Angaben, schnell wieder auf eine korrekte Vorversion zurückgreifen zu können. Nach Moskaliuk (2008), sind Wikis auch durch deren dynamisch entstehendes Netwerk gekennzeichnet, welches durch Verlinkung anderer Beiträge ensteht. Die Struktur der Links auf Wikiseiten ist dynamisch, weil es keine feste Reihenfolge der Links gibt und ebenfalls neue Links gesetzt werden zu noch nicht existenten Wikiseiten, die das Netzwerk erweitern. Oftmals wird das Bearbeiten auch durch einen WYSIWYG-Editor erleichtert, sodass auch ohne Kenntnis der spezifischen Wiki-Syntax ein Text bearbeitet werden kann und damit auch eine geringere Hürde darstellt, um mit dem Bearbeiten eines Wikis zu beginnen. Die Wiki Syntax ist sehr einfach gehalten, so dass ohne große Einarbeitungszeit und meist ohne notwendige Installationen, mit Änderungen begonnen werden kann. Für Links gibt es beispielsweise ebenfalls eine vereinfachte Syntax, ohne Angabe einer URL, die entweder, durch setzen des Wikiseiten Titels in Klammern, oder durch CamelCase, dargestellt werden. (Moskaliuk, 2008)

Im Folgenden soll nun die Zusammenarbeit der verschiedenen Wiki Typen betrachtet werden, die an unterschiedliche Anforderungen und Ziele angepasst sind. Es gibt öffentliche Wikis, die von überall erreichbar sind, Firmenwikis, die aus

dem Firmennetz erreichbar sind, und persönliche Wikis, die aber keine, oder kaum, Zusammenarbeit aufweisen. (Mayer, 2013)

Öffentliche Wikis

Es gibt größere Wikis, abseits von Wikipedia, die auf einer kleineren aktiven Gruppe aufgebaut sind. Zum Beispiel Wikis auf der Wikia Plattform besitzen, zumeist, nur eine geringe aktive Nutzerbeteiligung pro Wiki, und dennoch häufig eine große Anzahl an Artikeln. Fällt diese Basis weg, die eigene Organisationsstrukturen entwickelt haben, bricht das ganze Wiki zusammen. Bei Wikipedia fällt es hingegen kaum auf, wenn selbst ein größerer Teil der aktiven Nutzer wegbricht, da Wikipedia eine so große aktive Nutzergemeinschaft besitzt, die den Nutzerschwund ausgleichen können. (Mayer, 2013) Trotz der Unterschiede gibt es einige Merkmale der Zusammenarbeit, die sowohl in Wikipedia, als auch in kleineren Wikis, vorkommen.

Wiki-Effekt Bei den meisten Wikis steht ein kollaboratives Endprodukt im Mittelpunkt, welches einem stetigen dynamischen Prozess unterworfen ist. Öffentliche Wikis, besitzen auch keine, a priori, festgelegten Rollen. (Moskaliuk, 2008) Jeder Benutzer, hat anfangs die gleichen Möglichkeiten, ohne zentrale Steurung, am Wiki zu arbeiten. Durch den sogenannten "Wiki-Effekt", beginnt die Gemeinschaft sich selbst zu organisieren und Normen, sowie Richtlinien für die Zusammenarbeit festzulegen. Ebersbach et al. (2008, S.27) definiert den Wiki-Effekt folgendermaßen:

> " Unter Wiki-Effekt verstehen wir in erster Linie die Selbstorganisationsprozesse, die sich bei den bekannten und erfolgreichen Wiki-Projekten beobachten lassen. Hier verblüfft, dass Menschen eigenständig recherchieren, organisieren, schreiben und publizieren, um der Allgemeinheit eine kostenlose Dienstleistung zur Verfügung zu stellen. So haben sich große Communities gebildet, die weitgehend auf zentrale Steuerungsmodelle verzichten."

Diese flachen Hierarchien in öffentlichen Wikis, verstärken demnach die Selbstorganisationsprozesse der Nutzer.(Ebersbach et al., 2008)

Koordinationsmöglichkeiten der Zusammenarbeit Die Koordination der Zusammenarbeit in Wikis kann unterschiedlich erfolgen. Wachsen die Mitglieder der sozialen Plattform eines Wikis immer stärker zusammen, so erwächst daraus eine Community, die, wenn sie sich auf ein Thema verständigt hat und Regeln aufstellt, zu einer Organisation erwächst. Es bilden sich Redaktionen innerhalb der

Wikis, die sich gemeinsam auf Schreib- und Darstellungsformen geeinigt haben (Mayer, 2013, S. 56 ff).

Beispielsweise die Koordinierungsformen in Wikipedia: Nutzer können eine Seiten alleine editieren, oder sie schließen sich mit Anderen als Team in Projekten zusammen. Projekte zeigen einen inhaltlich zusammengehörigen Themenkomplex, mit seinen dazugehörigen Seiten auf, zum Beispiel zum Thema Astronomie. Auch Redaktionen richten sich an einen Themenkomplex, sind aber weit gestreuter, wie beispielsweiße die Redaktion Physik. Man kann sich auch auf Portalen engagieren, die einen Einstieg in Themenkomplexe bieten.[1] Bei Wikia trifft man, in Form der neuesten Wikis und Wikis nach Thema geordnet, ebenfalls auf eine Art Portal. (Mayer, 2013, S. 129)

Qualitätskontrolle Die Qualitätskontrolle in Wikis ist sehr effektiv, so wird der meiste Anteil an Spam innerhalb weniger Minuten beseitigt, auch Dank der Versionsgeschichte die ein schnelles Wiederherstellen ermöglicht. (Viégas et al., 2004) In größeren Wikis ist die Masse an Benutzern, die Artikel kontrollieren und ausbessern, ausschlaggebend für eine gute Qualität der Artikel. In Wikipedia beispielsweise, haben sich eine Fülle von Regeln und Richtlinien etabliert, die ein hohes Nivea der Artikel begünstigen sollen. Es gibt Formatierungsrichtlinien, oder es werden exzellente Artikel mit hoher Qualität ausgezeichnet. Mit dem Mentorenprogramm, werden neue Benutzer auch schnell an die Richtlinien herangeführt, zum Beispiel der Einhaltung eines neutralen Standpunkts. (Mayer, 2013)

Entstehende Rollen Wikis bieten meistens die Möglichkeit, als anonymer Nutzer ohne Anmeldung, Änderungen vorzunehmen. Nutzer können aber auch, mit einem registrierten Benutzernamen, Änderungen durchführen, unter dem alle Bearbeitungen gespeichert werden. Bei anonymen Änderungen wird die IP-Adresse des Nutzers, anstelle des Nutzernamens, in der Historie gespeichert. (Viégas et al., 2004) Außerdem gibt es in größeren Wikis, wie Wikipedia oder Wikia, auch automatisch ablaufende Skripte, genannt Bots, die unter einem eigenen Namen angemeldet sind. Sie durchsuchen die Wikis und nehmen kleine Änderungen automatisch vor, wie Anpassungen von Schreibweisen. (Ebersbach et al., 2008)

Probleme in der Zusammenarbeit Es können einige Probleme in der Zusammenarbeit in Wikis gefunden werden, zum Beispiel Vandalismus und Editierkriege zwischen Nutzern. Wikipedia und auch andere öffentliche Wikis sind stetig, trotz Bots und Nutzergruppen, mit dem Problem des Vandalismus konfrontiert, bei

[1] `https://de.wikipedia.org/wiki/Wikipedia:Beteiligen`
aufgerufen am 04.03.2015

dem falsche Änderungen oder Massenlöschungen vorgenommen werden. Wie bereits aufgezeigt, wird Spam in der Wikipedia meistens innerhalb weniger Minuten korrigiert. Die schnelle Korrektur, wird durch Beobachtungslisten und der Liste der kürzlichen Änderungen, erst für die Nutzer ermöglicht. Es besteht auch die Möglichkeit Seiten für einen Zeitraum zu sperren, beim Aufdecken des Vandalen diesen zeitweise zu blockieren, oder bei fehlender Besserung dauerhaft zu blockieren. Wobei auch im Fall der Verbannung, der Nutzer einen neuen Account mit neuem Benutzernamen erstellen könnte. Es ist auch möglich, dass Benutzer, die anonym editieren, aufgrund ihrer IP-Adresse, durch die Befugnisse eines Administrators blockiert werden. Dieses Vorgehen ist kritisch zu sehen, da IP-Adressen sich im Laufe der Zeit verändern und neu zugewiesen werden können. (Reagle Jr., 2010)

Ein weiteres Problem sind Editierkriege, in denen unterschiedliche Meinungen, aufeinanderprallen. Zu diesem Zweck gibt es auf Wikipedia Vermittlungsinstanzen, die versuchen einen Konsens zu finden. (Ebersbach et al., 2008, S. 35) In diesen Editierkriegen auf Wikipedia wird zwischen den Versionen der unterschiedlichen Parteien gewechselt, dabei werden die Diskussionsseiten oftmals als Kommunikationsmittel zur Auseinandersetzung genutzt. Normalerweise sind die Diskussionsseiten genau dafür da, um Konflikte zu vermeiden und seine Änderungen zu erklären. Sollten aber andere Nutzer mit der Änderung nicht einverstanden sein, kann diese Kontroverse in einen Editierkrieg ausarten. (Viégas et al., 2004)

Auch kleiner Probleme, wie Bearbeitungskonflikte, findet man in beinahe jedem Wiki. Bearbeitungskonflikte treten dann auf, wenn, zwei oder mehr Nutzer, an einer Wikiseite gleichzeitig Änderungen vornehmen, ohne sich abgesprochen zu haben und ein Akteur schneller abspeichert als die Anderen. Damit nun die neuen Änderungen nicht einfach überschrieben werden, werden die anderen Bearbeiter, beispielsweise gewarnt ihre Änderungen manuell mit der neuen Version zu verbinden. In Wikipedia kann, vor der Bearbeitung, ein vorgefertigter Hinweis auf der Seite hinterlegt werden, der signalisieren soll, dass die Seite derzeit in Bearbeitung und zu einem späteren Zeitpunkt wieder frei ist. (Ebersbach et al., 2008)

Enterprise Wikis

Enterprise Wikis gehören zu den Wikis in geschlossenen Projektgruppen und sind damit nicht für jeden einsehbar oder editierbar. Sie sind für die Mitarbeiter einer Firma gedacht um gemeinsam Wissen, Erfahrungen und Ideen auszutauschen. Dabei kann man zwei Arten von Unternehmenswikis unterscheiden: Einmal abteilungs - oder unternehmensweite Wikis und auf Projekte zugeschnittene Wikis (Gatzweiler, o. J.). Enterprise Wikis, die in die Infrastruktur der Firma integriert werden, verfügen auch über eine größere Funktionalität als öffentlich zugängliche

Wikis wie Wikipedia. Beispielsweise kann die Sichtbarkeit der Wikiseiten, nach den Anforderungen der Firma, genau eingestellt werden. In Wikis wie Wikipedia, entwickeln sich die Zuständigkeiten flexibel, während Wikis die in Firmen eingesetzt werden, auf bereits bestehende Strukturen in Abteilungen treffen, die sich gegebenenfalls auch auf die Entwicklung des Wikis ausüben. (Ebersbach et al., 2008) Auf diese verschiedenen Zusammenarbeitsmuster gehe ich in Kapitel 3.2 genauer ein. Zur Qualitätssicherung ist in öffentlichen Wikis, wie aufgezeigt, die Masse an Nutzern verantwortlich, die sich zu einem großen Grad, selbst organisieren kann. In Unternehmen dagegen werden oftmals, für spezielle Arbeiten, klare Rollen und Zuständigkeiten, wie Korrekturlesen oder Verlinkungen, festgelegt. (Mayer, 2013)

Auch werden Teile des Enterprise Wikis nicht für jeden Mitarbeiter in der Firma gleich zugänglich gemacht, auch wegen rechtlicher Probleme, und sind nach Abschluss eines Projekts möglicherweiße nicht mehr notwendig. In dieser Form eines Wikis ist die Spamgefahr weniger ein Problem, da auch nur die Mitarbeiter zugreifen können, die im Normalfall keinen Nutzen daraus ziehen, Seiten zu zerstören. So ist in Firmenwikis eher das Problem gegeben, Mitarbeiter dazu zu bewegen am Wiki zusammenzuarbeiten. Als bekanntes Beispiel eines Enterprise Wikis ist Confluence zu sehen. (Ebersbach et al., 2008)

Zusammenfassung

Woran kann man nun die verschiedenen Wikis unterscheiden? Öffentliche Wikis haben eine große Reichweite über das ganze Internet und dadurch auch eine sehr heterogene Nutzerschaft, dabei ist das Wiki meist langfristig angelegt. Die Teilnahme geschieht freiwillig und als Inhalte, werden zumeist nur das Thema des Wikis, akzeptiert. Organisationsweite Wikis haben, in Gegensatz zu projektinternen Wikis, noch eine mittlere Reichweite, während projektinterne Wikis nur noch eine geringe Reichweite haben. In Projektteams ist dadurch auch die Nutzerschaft relativ homogen, das Projektwiki bezieht sich auf das Ziel des Projekts und hat damit nur eine kurze Lebensdauer. Organisationsweite Wikis sind normalerweise dauerhaft angelegt und werden zum Wissensmanagement intern genutzt. (Mayer, 2013, S. 67 ff.)

2.4.2 Zusammenarbeit in GitHub

GitHub ist eine Webseite auf der jeder seine Git Repositories online stellen kann und mit vielen anderen Entwicklern einfach an Projekten kollaborieren kann.[2]

[2]Git ist ein verteiltes Versionsverwaltungssystem, weiteres zu Git in Kapitel 3.3

Es wird eine zentrale webbasierte Oberfläche geboten, auf der Nutzer ihre Repositories, zumeist Softwareprojekte, hochladen und effektiv mit anderen Nutzern arbeiten können, indem Mechanismen von Git einfach umgesetzt werden. (Chacon & Straub, 2014, S.195 ff.)

Auf GitHub ist es jedem Benutzer möglich eigene Repositories auf der Plattform zu erstellen und Code darin hochzuladen. Dies geschieht mit einem "push" vom lokalen Computer, mit Git, auf die GitHub Plattform. Außerdem können die Mitarbeiter eines Projekts direkt Änderungen in das Repository hochladen. Sie begutachten außerdem Änderungen, die von Aussenstehenden beigetragen wurden und in das Repository eingepflegt werden sollen. Diese Mitarbeiter, die Lese- und Schreibrechte für das Projekt haben, wurden vom Besitzer als Mitarbeiter zum Projekt hinzugefügt. Ein Entwickler kann dabei in mehreren Projekten mitarbeiten und Projekte können mehrere Mitarbeiter beinhalten. (Thung, Bissyandé, Lo & Jiang, 2013)

Es gibt zwei Arten von Akteuren, die zusammenarbeiten. Die Mitarbeiter im Projekt die direkt Änderungen einpflegen können, und alle weiteren Nutzer, die ebenfalls mitarbeiten wollen, aber nicht Teil des Projekts sind. Die zweite Gruppe der Akteure besitzt nicht die nötigen Berechtigungen, um direkt Änderungen vorzunehmen und muss deshalb einen "fork" anstoßen, mit dem das komplette Projekt in den Namensraum dieser Person kopiert wird und wodurch diese völlig unabhängig vom Ursprungsprojekt arbeiten können. (Lima, Rossi & Musolesi, 2014)

Hat man ein Projekt mit einem "fork" kopiert, ist das Projekt nur auf der GitHub Plattform, weshalb nun ein "clone" notwendig ist, wodurch das Projekt auf den Computer kopiert wird. Ein "clone" ist immer dann notwenig, wenn Nutzer ein Repository lokal auf den Computer kopieren wollen, um daran zu arbeiten. In GitHub wird nun meistens ein eigener Zweig "topic branch" erstellt, der das neue Repository enthält, um neue Funktionen zu testen und einen besseren Überblick zu erhalten. Ein Branch, Zweig, ist eine parallele Version des Projekts, auf der frei Änderungen durchgeführt werden können, ohne Gefahr die auslieferbare Software im Hauptbranch zu stören.[3] Branching mit Git wird in Abbildung 2.1 dargestellt. An seinem Computer kann man nun Änderungen durchführen und auf GitHub im eigenen Namensraum hochladen. (Chacon & Straub, 2014, S. 203 ff.)

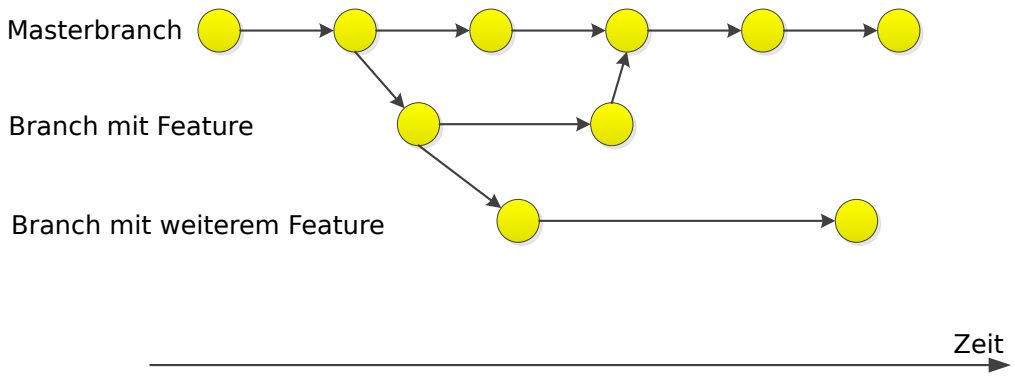

Abbildung 2.1: Git Branches

Dafür wird ein "commit" der überarbeiteten Dateien, die hochgeladen werden sollen, angelegt, welche die Dateien eindeutig identifizierbar abspeichert. Mit einer Commit Nachricht werden die gemachten Änderungen beschrieben und schließlich mit einem "push" in ein externes Repository, hier GitHub, hochgeladen und sind damit nicht mehr lokal, sondern für jeden sichtbar auf GitHub (außer es handelt sich um ein privates Repository).[3]

Ein "commit" und "push" sind immer notwendig, wenn Änderungen auf GitHub, in ein Repository, hochgeladen werden sollen. Anschließend kann man eine Anfrage, "pull request", an einen Mitarbeiter des Ursprungsrepository stellen, der die Änderungen, nach gemeinsamer Diskussion, in sein Repository mit einem "merge" vereinigen kann. (Chacon & Straub, 2014)

In Abbildung 2.3, sieht man ein hochgeladenes Repository, bei dem GitHub, mit einem Button, automatisch die Möglichkeit zum Pull Request offeriert.Die Änderungen werden dann in den Masterbranch, oder Mainbranch, des Ursprungsprojekts hochgeladen. Dies ist der voreingestelle "default" Branch, auf dem die auslieferbare Software hochgeladen wird[3]. (Bell & Beer, 2015)

Falls die Änderungen nicht zufriedenstellend sein sollten, können, vor Integration in den Masterbranch, Verbesserungen verlangt werden. (Chacon & Straub, 2014, S. 202)

In Abbildung 2.2, sieht man den beispielhaften Ablauf um mit GitHub zu arbeiten, dies ist der sogennante "fork and pull" Ablauf. Mit Git werden dabei lokale Änderungen auf die Plattform geladen, dies geschieht mit den Befehlen "commit" und "push". Auf der GitHub Plattform werden die Projekte kopiert mit dem Befehl "fork", auf den Computer übertragen mit dem Befehl "clone" und die Anfrage versendet mit einem Pull Request. Nutzer verwenden demnach

[3] `https://help.github.com/articles/github-glossary/`
aufgerufen am 04.03.2015

sowohl Git Befehle, als auch GitHub Befehle auf der Plattform.

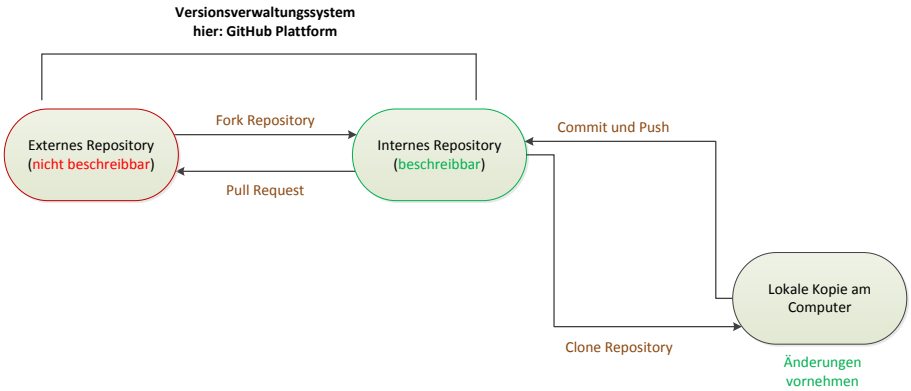

Abbildung 2.2: GitHub Workflow; Quelle: nach (RightScale Inc., 2014)

GitHub ist nicht nur eine Weboberfläche um, mit Git, Projekte hochzuladen, sondern auch um Probleme und Fragen, "issues", zu verfolgen. Eine weitere wichtige Eigenschaft sind die eingebettenden sozialen Funktionalitäten: " [...] this service has much emphasis on its social features, as summarized in its motto "GitHub: social coding"." Lima et al. (2014, S. 1) Dies hebt GitHub von anderen Codehosting Plattformen ab. Dabei bietet GitHub kostenlose, für jeden sichtbare, Repositories an oder private, kostenpflichtige, Repositories, die nicht für jeden sichtbar sind.[4]

Bestandteile von GitHub Auf einer GitHub Projektseite werden die Codedateien des Repositories, die gemachten Commits, Issues, Pull Requests und ein dazugehöriges Wiki, angezeigt, wie in Abbildung 2.3 dargestellt. Aktionen die durchgeführt werden können, können eingeordnet werden in Codebezogene Handlungen, Kommunikation und soziale Funktionalitäten. Code relevante Aktionen, wie Fork, Commit, Pull Request und deren Einsatz wurden bereits besprochen. Kommunikation mit anderen Akteuren kann bei Code relevanten Aktionen durchgeführt werden. So können Kommentare bei einem Pull Request, Commit oder Issue hinzugefügt werden, die als direktes Feedback und zum Code Review verwendet werden. Auf GitHub gibt es nicht nur Projektseiten, sondern auch Profile der Nutzer, die persönliche Informationen beinhalten können und für jeden sichtbar sind. (Dabbish et al., 2012)

[4] siehe `https://github.com/pricing`, aufgerufen am 28.03.2015

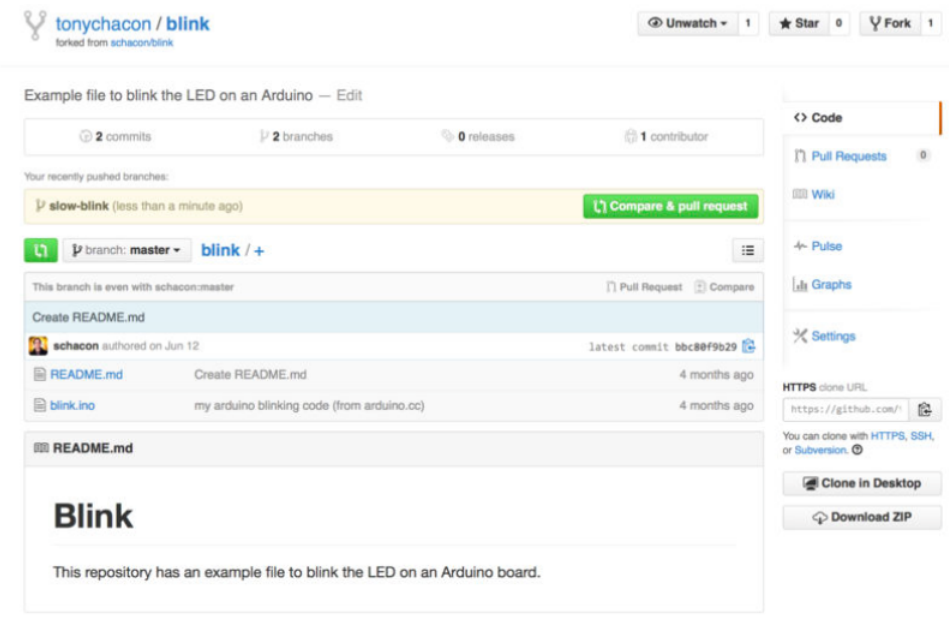

Abbildung 2.3: GitHub Plattform mit Pull Request Button; Quelle: (Chacon & Straub, 2014)

- Mit “issues“ können Fehler und neue Features der Software besprochen werden, dabei können Nutzer direkt mit @mention angesprochen werden, um Rückmeldung von bestimmten Personen zu bekommen. Außerdem können “issues“ zu Meilensteinen hinzugefügt werden, wenn eine bestimmte Deadline eingehalten werden muss. Mit # können Commits mit der Nummer des “issues“ hochgeladen werden und werden dann auch im Verlauf des “issues“ angezeigt. @mention und # können auch bei anderen Kommentaren, außerhalb von “issues“, verwendet werden. (Bell & Beer, 2015)
- Pull Requests werden in GitHub “upstream“ hochgeladen, dies ist das Originalrepository, welches kopiert wurde und von dem Änderungen geholt werden. Ein Pull Request wird oftmals als Code Review Möglichkeit genutzt, dabei kann die Begutachtung der Änderungen auf zwei Arten erfolgen (Kalliamvakou, Singer et al., 2014):
  - durch Diskussion des kompletten “pull requests“ als Ganzes
  - als Code Review von speziellen Codeteilen (Kalliamvakou, Singer et al., 2014)

Soziale Funktionalitäten in GitHub bestehen aus:

- verfolgen “follow“ anderer Nutzer von GitHub (Dabbish et al., 2012)
- beobachten “ watch“ anderer Repositories (Dabbish et al., 2012)

- unterstützen von Repositories mit Sternen, dies zeigt die Popularität eines Projekts an (Aiello & McFarland, 2015, S. 84)

Diese sozialen Funktionalitäten sind auch in Abbildung 2.3 sichtbar, die rechts oben auf der Projektseite, angezeigt werden.

Zusammenarbeitsmodelle Es gibt drei Zusammenarbeitsmodelle auf GitHub:

- "fork and pull": Diese Variante benötigt wenig Koordination, da Mitarbeiter selbstständig arbeiten und erst durch einen Pull Request Koordinationsbedarf anzeigen. (Kalliamvakou et al., 2015)
- "shared access": Hier haben die Mitarbeiter direkte Schreibrechte auf das Repository, sie können demnach ohne Pull Request, die Änderungen direkt hochladen. Diese zentrale Variante ist eher für kleiner Gruppen geeignet, wie auch beim zentralisierten Workflow von Git[5]. (Kalliamvakou et al., 2015)
- "branch and pull" Workflow: hier gibt es ein Repository, zu dem alle Mitarbeiter direkte Schreibrechte haben, aber einzelne Branches für Features verwenden werden, wobei zwischen den Zweigen, Pull Requests eingesetzt werden. Dieser Workflow wird auch auf GitHub und nach Chacon und Straub (2014) als GitHub Workflow beschrieben, wird sowohl in öffentlichen, als auch privaten Repositories, eingesetzt.

Beim "branch and pull" Workflow ist der Masterbranch immer in einem auslieferbaren Zustand und enthält keine kaputten Codeteile. Damit wird sichergestellt, dass es immer eine einsatzbereite Software gibt und dennoch jeder Entwickler an neuen Features arbeiten kann.

Sollte es zu Konflikten beim vereinigen des Pull Requests mit dem Masterbranch kommen, zeigt dies GitHub an, bis die Konflikte aufgelöst wurden. Dazu müssen die Änderungen erst lokal geholt, eingearbeitet und die Konflikte aufgelöst werden. (Chacon & Straub, 2014)

Tabelle 2.1: Vergeich der Workflows

Workflow	Vorteile	Nachteile
"fork and pull"	- Änderungen können einfach verfolgt werden - Koordinierungspunkt - Möglichkeit eines Code Review	- Mitglieder müssten alle Kopien betrachten, um den tatsächlichen Fortschritt zu sehen - Schlechterer Überblick als mit Branches
"shared access"	- Änderungen können einfach direkt hochgeladen werden	- Schlechter Überblick der Änderungen

In der Tabelle 2.1 werden noch die Vor- und Nachteile der beiden anderen Workflows dargestellt. Der "branch and pull" Workflow verbindet dabei die Vorteile der beiden anderen Workflows, nämlich schnelle Übersicht von Änderungen und

[5]siehe dazu Kapitel 3.3

einfaches Hochladen, mit kombiniertem Code Review, bei Bedarf. (Kalliamvakou et al., 2015)

Wie unterstützt GitHub die Kollaboration?

- GitHub unterstützt die Kollaboration, kommerzieller und nichtkommerzieller Projekte, in der eigenständigen Arbeit der Entwickler durch Forking und Branching. Daraus resultiert ein geringerer Koordinierungs- und Kommunikationsaufwand. (Kalliamvakou, Damian et al., 2014)
- Code Reviews werden durchgeführt, bei "pull requests", die eine Koordinierungs- und Kommunikationsbedarf kennzeichnen (Kalliamvakou, Damian et al., 2014)
- Die Kommunikation ist Codebasiert um "issues", "commits" und "pull requests" (Kalliamvakou, Damian et al., 2014)
- Selbstorganisation wird durch die öffentlichen "issues" ermöglicht, durch die eine selbstbestimmte Arbeitsaufteilung erfolgen kann. (Kalliamvakou, Damian et al., 2014)
- Entwicklung und Status von Projekten können verfolgt werden durch Benachrichtigungen, "watch" und "follow" (Kalliamvakou, Damian et al., 2014)

In Kapitel 3.4 und 3.5 gehe ich noch allgemein darauf ein, wie GitHub genutzt wird.

2.5 Bestimmung des Möglichkeitsraums für Zusammenarbeitsmodelle

Verschiedene Zusammenarbeitsmodelle lassen sich nach verschiedenen Kriterien unterscheiden. In diesem Kapitel entwickle ich ein Wiki, aufbauen auf den Erkenntnissen des Kapitels 2.4 und verdeutliche visuell die verschiedenen Dimensionen.

2.5.1 Das Musterwiki

Die Musterwiki Seite soll eine Plattform werden, auf der verschiedene Muster zu unterschiedlichen Bereichen, gesammelt werden können. Diese Plattform besteht also aus mehreren Wikis. Nicht nur die bereits umgesetzten Entwurfsmuster aus dem Buch, "Entwurfsmuster: Elemente wiederverwendbarer objektorientierer Software" von Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides, in der Softwareentwicklung, auch Entwurfsmuster im Bereich, beispielsweise Safety und Security, können dort eingestellt werden.

Muster sind bekannte und erprobte Vorlagen, um in bestimmten Situationen, häufig entstehende Probleme zu lösen. Muster klassifizieren demnach praktische Lösungen häufig auftretender Probleme.[6] Die Muster können direkt in einem Editor, mit der Wiki spezifischen Syntax, auf der Plattform eingetragen werden. Dabei soll die Musterwiki Plattform öffentlich, für jeden über das Internet, aufrufbar sein, aber Unterschiede in der Sichtbarkeit machen.

So soll es einerseits öffentliche Wikis geben und Wikis, die nur für eine bestimmte Benutzergruppe sichtbar sind, beispielsweise Firmenwikis. Die Musterwiki Plattform soll auf diese Weise aufgebaut werden, um Nutzer mit verschiedenen Anforderungen bei der Zusammenarbeit zu unterstützen. Es gibt sowohl Muster, die öffentlich bekannt sind, als auch Muster, die nur in Unternehmen eingesetzt und bekannt sind.

Da Muster in verschiedenen Bereichen häufig eingesetzt werden, sollte eine Möglichkeit gefunden werden, möglichst qualitativhochwertige Prozesse zu begünstigen und Vandalismus, so gut es geht, zu vermeiden. Es sollte also eine eingeschränkte Bearbeitungsmöglichkeit für anonyme Nutzer geben und die Möglichkeit Wikis nach eigenen Anforderungen, anzupassen. Für diese Anforderungen wird ein Modell gewählt, bei dem jedes Wiki einen Besitzer und von diesem festgelegte Mitarbeiter hat, wobei diese ihr Wiki frei bearbeiten können.

[6] siehe `http://www.e-teaching.org/didaktik/konzeption/entwurfsmuster/`, aufgerufen am 28.03.2015

Dadurch, dass jedes Wiki einen Besitzer hat und dieser entscheidet wer mitarbeiten darf und wer nicht, ist die Gefahr durch, beispielsweise Massenlöschungen von “Vandalen“, weniger gegeben. Dafür sorgt, ein integrierter Editor zum Bearbeiten des Wikis, der sich nur öffnet, wenn Besitzer oder Mitarbeiter des Wikis auf das Wiki zugreifen.

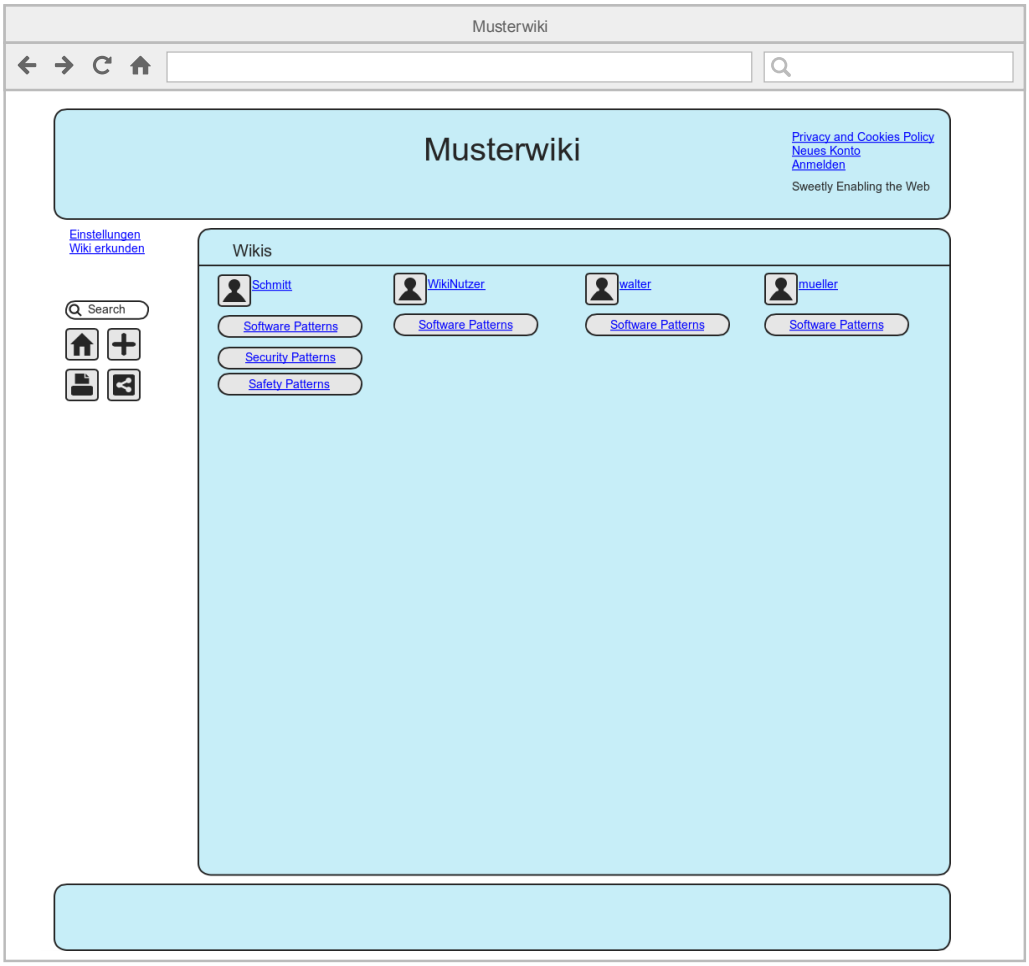

Abbildung 2.4: Mockup der Musterwiki Seite

In Abbildung 2.4, sieht man ein Mockup eines möglichen Designs der Plattform, auf der die öffentlichen Nutzer und ihre angelegten Wikis aufgezeigt werden. Durch den Link “Wiki erkunden“ könnten Wikis mit neuesten Bearbeitungen angezeigt werden, oder auch weitere Kategorien.

2.5.2 Zentralisiertes oder dezentralisiertes Konfigurationsmanagement

Als Zusammenarbeitsmodell kann ein zentralisierter Workflow, wie in Subversion, oder ein dezentralisierter Workflow, gewählt werden. Bei der dezentralen Versionsverwaltung besteht eine große Auswahl an Workflows, wie bereits bei GitHub vorgestellt. Allgemeine Zusammenarbeitsmuster von Git finden sich in Kapitel 3.3.

Vorteile dezentraler Versionsverwaltung ist, dass jeder Mitarbeiter lokal am Computer eine komplette Kopie vom Projekt hat und somit jeder unabhängig arbeiten kann. Dezentrale Versionsverwaltungssysteme ermöglichen also ein arbeiten ohne Internetverbindung am Computer, wobei die Änderungen lokal am Computer übergeben werden können und auf eine externe Plattform hochgeladen werden. Damit ist der einzelne Nutzer unabhängig vom Server und kann selbst bestimmen, welche Änderungen genau hochgeladen werden sollen.

Um Probleme in Wiki Systemen zu umgehen, wie sie Kapitel 2.4.1 vorgestellt wurden, wird für das Entwurfsmuster Wiki ein dezentrales Versionsverwaltungssystem, mit Git, gewählt. Dadurch wird es verschiedenen Nutzern ermöglicht, an der gleichen Wikiseite parallel zu arbeiten, wobei Git Bearbeitungskonflikte meist selbstständig auflösen kann.[7] Dies ist ein großer Vorteil zu den meisten anderen öffentlichen Wikis, bei denen Bearbeitungskonflikte mühsam per Hand korrigiert werden müssen. Probleme, wie Editierkriege oder Massenlöschungen, werden durch Besitzen von Wikis zusätzlich minimiert, da hier jeder an seiner eigenen Version und Vorstellung arbeiten kann.

Dadurch sind verschiedene Arbeitsweisen möglich und erzeugen dennoch größtmögliche Freiheit für den einzelnen Nutzer. Der Wikibesitzer und die Mitarbeiter können dabei ausschließlich den integrierten Editor verwenden, oder sie installieren zusätzlich Git auf ihrem Computer und können sich dadurch privat so organisieren, wie sie es für nötig halten. Aber auch für Akteure, die nicht Mitarbeiter eines Wikis sind, und dennoch teilnehmen wollen, können mit Git am Wiki mitarbeiten. Darauf gehe ich genauer bei der Zusammenarbeit ein.

Für Git macht es keinen Unterschied, ob nun Wikiseiten damit verwaltet werden, oder Codedateien. Die Versionsverwaltung auf der Plattform funktioniert ebenfalls mit Git. Die Softwareinfrastruktur kümmert sich darum, die Textänderungen entgegen zu nehmen, sie Git zu liefern und das resultierende Ergebnis dem Nutzer auf der Plattform, zu übergeben. Beispielsweise:

- Der User beendet über einen Knopf auf der Webseite „Speichern“ das Bearbeiten einer Wiki Seite

[7] siehe dazu (Bell & Beer, 2015)

- Der Server verarbeitet den Text in das passende Format und schickt die Textdatei, unter der Kontrolle von Git, in das Filesystem
- Die Webseite wird aktualisiert, wodurch der Server die neuste Version von Git holt, in HTML umwandelt, und zurück an den Browser schickt

Zu beachten ist, dass in der Arbeit zwar Begriffe von GitHub für die Plattform übernommen werden, um die Verbindung aufzuzeigen, jedoch auf der Musterwiki Plattform, auf der nicht nur Softwareentwickler arbeiten werden, eine andere Umschreibung gefunden werden sollte.

Wikis auf Basis von GitHub arbeiten das Zusammenarbeitsmodell aus, indem das Branching eine weitere wichtige Rolle spielt. So kann man verschiedene Versionen seines Wikis halten und wieder zusammenführen, auf Basis von Git. Jeder Akteur des Wikis könnte durch Verwendung mit Git, lokal auf seinen Computer, ohne Internetverbindung, arbeiten und bei Bedarf seine Änderungen online in sein Wiki hochstellen. Dass jeder an einem eigenen Wiki arbeiten kann, zeigt sich auch an der URL, die den Namen des Benutzers und dessen Projekt beinhaltet, beispielsweiße: http:\\www.musterwiki.de\u\nutzer1\p\projekt1\. Durch einen Fork könnte ein anderer Wikibenutzer projekt1 in seinen Namensraum kopieren, http:\\www.musterwiki.de\u\nutzer2\p\projekt1\, und eigenständig daran arbeiten. Wie bei GitHub könnte durch eine anschließende Anfrage von nutzer2 an nutzer1, dessen Änderungen im Original übernommen werden. Aber auch die Möglichkeit zentraler Arbeitsweise, wie sie auch in GitHub mit “shared access“, geboten wird, kann in dem Musterwiki umgesetzt werden.

2.5.3 Einzelpersonen oder Mehrpersonen-Wikis

Wie in GitHub kann auf der Musterwiki Seite jeder Nutzer sein eigenes Repository, beziehungsweise hier Wiki, erstellen, dieses liegt dann im Namensraum des Nutzers. Wollen mehrere Personen an einem Wiki arbeiten, können sie einen Workflow, wie auf GitHub zu finden, nutzen.

2.5.4 Öffentliche und private Wikis

Die Musterwiki Seite kann sowohl für Wikis genutzt werden, die öffentlich sichtbar sind oder privat, nur für einen bestimmten Nutzerkreis. Wissen, welches öffentlich geteilt werden will und kann, wird so durch die Plattform für alle sichtbar. In Unternehmen ist es aber wichtig, dass firmeninternes Wissen oftmals nicht an die Öffentlichkeit gelangen darf. So kann das Musterwiki im Intranet der Firma, firmenweit, eingesetzt werden, oder nur auf Projektebene. Die Firmen können

Muster aus einem Bereich der Musterwiki Seite übernehmen oder eigene Muster erstellen, die entweder, für die Öffentlichkeit, firmenweit oder nur für eine bestimmte Nutzerschaft, sichtbar sind.

Ein weiterer Anwendungsfall ist, dass Muster von Firmen übernommen werden und diese nach den firmeninternen Vorstellungen angepasst werden. Diese Muster können öffentlich gestellt werden, um eine Diskussion mit der breiten Öffentlichkeit anzustreben, oder nur intern, verwendet werden. Die privaten Wikis könnten die gleichen Workflows anwenden, wie sie auch die öffentlichen Wikis verwenden. Es kann also auch Kollaboration zwischen öffentlichen und privaten Wikis geben, je nach dem in welchem Ausmaß das gewünscht wird. Öffentliche Wikis können ebenfalls Muster übernehmen, anpassen, oder neue hinzufügen.

Im Use-Case Diagramm Abbilung 2.5, sieht man die generellen Anforderungen der Nutzer an das System. Öffentliche und private Nutzer, die sich in der Sichtbarkeit der Wikiartikel unterscheiden, wollen sich beide als Benutzergruppe auf der Plattform an- und abmelden können und nach Anmeldung ein eigenes Nutzerprofil anlegen. Durch die Registrierung kann der Nutzer ein eigenes Wiki erstellen, welches in dessen Nutzerraum liegt. Da die Plattform sowohl öffentlich, als auch mit eingeschränkten Sichtbarkeitseinstellungen, genutzt werden soll, können die Nutzer bei der Registrierung ein öffentliches oder privates Profil anlegen. Möglich wäre hier auch, die Musterwiki Plattform mit Serverkomponente an private Wikis anzubieten, damit diese auch bei Bedarf die kompletten Daten bei sich haben und damit keinen externen Server benötigen.

Die Anforderungen der Nutzer beim bearbeiten eines Wikis werden in Abbildung 2.6 dargestellt. Die Mitabeiter und der Besitzer arbeiten zusammen am Wiki und entscheiden über Anfragen anderer Nutzer, in Form eines Pull Requests. Durch Kopie, also “forken“, eines Wikis oder Erstellen eines eigenen Wikis, soll dieses automatisch in den Namensraum des Akteurs gestellt werden, der diese Aktionen angestoßen hat.

Das Wiki soll ein einfaches bearbeiten ermöglichen und dabei zusätzlich einen größeren Schutz vor Spam gewähren, dadurch dass Wikis besitzt werden. Viele Arbeitsweisen werden ermöglicht und damit die Kollaboration gefördert, die im Resultat auch zu einer höheren Qualität der Artikel führen kann.

Weitere Anforderungen der Benutzer betreffen die Möglichkeiten mit Git. So soll es möglich sein, das Wiki auf den Computer zu kopieren, um damit lokal, ohne Internetverbindung, zu arbeiten und mit Git Änderungen wieder auf die Plattform hochzuladen. Wie auch bei GitHub können Bearbeitungskonflikte lokal mit Git aufgelöst werden und mit den Änderungen vereinigt werden. Außerdem sollte es dem Wikibesitzer möglich sein, seine Mitarbeiter, welche direkten Schreibzugriff auf das Wiki haben, festzulegen.

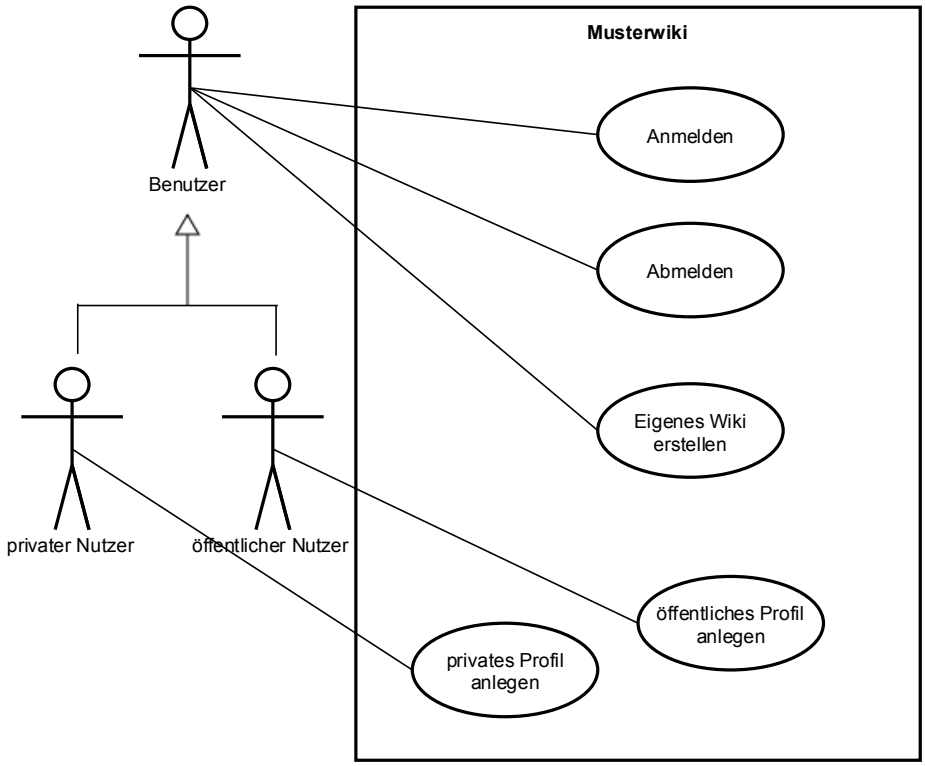

Abbildung 2.5: Allgemeine Benutzeranforderungen

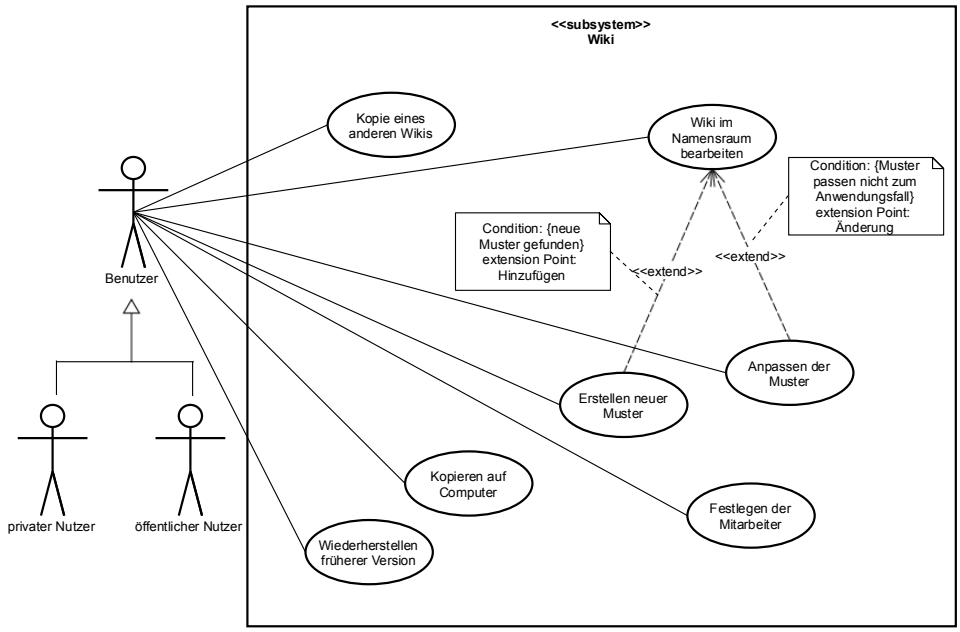

Abbildung 2.6: Benutzeranforderung beim Bearbeiten von Wikis

Private Nutzer, welche das Wiki firmenweit verwenden, haben zusätzlich zu diesen Anforderungen noch weitere, wie in Abbildung 2.7 dargestellt.

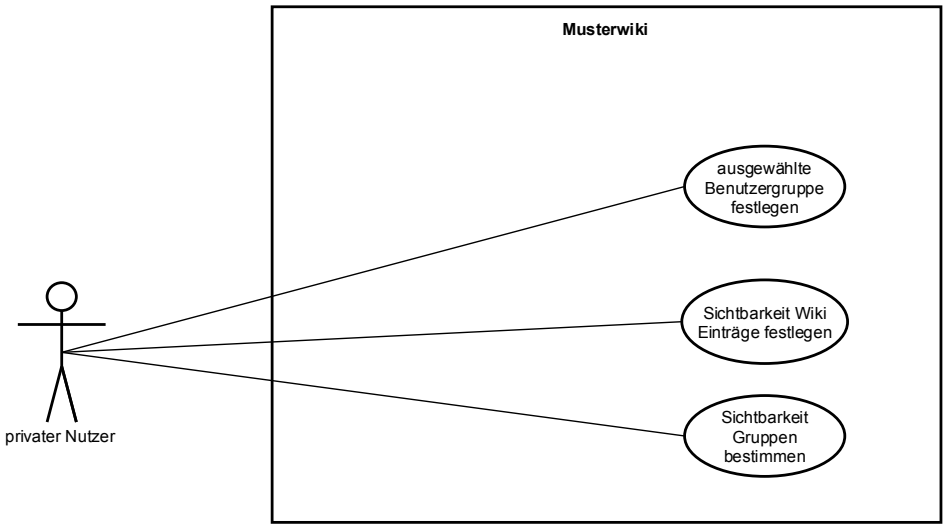

Abbildung 2.7: Benutzeranforderungen privater Nutzer

Hier ist die Festlegung der Sichtbarkeit sehr wichtig, da firmeninternes Wissen, welches für den Wettbewerbsvorteil relevant ist, nicht nach außen an die Öffentlichkeit dringen darf. So muss das Musterwiki, welches auch in Unternehmen genutzt wird, die Sichtbarkeit von Wikiartikeln einstellen lassen können. Ebenso wichtig ist eine Festlegung der Sichtbarkeit, auf eine bestimmte Gruppe, innerhalb, beispielsweise der Firma. Somit sollte es auch möglich sein, beim Konfigurieren des Wikis einzustellen, ob das betreffende Wiki unternehmensweit oder nur abteilungs-/projektweit sichtbar sein soll.

Nach der Festlegung der Benutzergruppe, sollte als Grundeinstellung festgelegt sein, Wikieinträge nur in diesem Sichtbarkeitsbereich sichtbar zu machen, um, beim Erstellen einer neuen Wikiseite nicht zufällig in einen größeren Sichtbarkeitsbereich zu veröffentlichen, als erlaubt. Wenn gewünscht, können aber eingestellte Wikiseiten, auch über das Internet für alle sichtbar veröffentlicht werden, oder bei abteilungsweiten Wikis, auch firmenweit. So besteht die Möglichkeit, Meinungen von außen einzuholen und mit verschiedenen Menschen zusammenzuarbeiten, wenn es der Anwendungsfall erlaubt. Durch das eigenständige Arbeiten am Wiki im eigenen Namensraum, werden die Mitarbeiter einer Firma möglicherweise motiviert mehr daran zu arbeiten, da sie nun kontinuierlich Änderungen, lokal am Computer, vornehmen können und nicht extra für das Aufschreiben neuer Ideen auf die Plattform wechseln müssen.

2.5.5 Anmeldung und Anonymität

Die Vorteile eines Wikis liegen darin, dass jeder Änderungen einfach vornehmen kann und die Hürden dafür gering sind, weil auch, beispielsweise die Syntax, sehr einfach gehalten wird. Ein integrierter Editor zum einfachen bearbeiten der Wikiseiten, könnte sich, beim Besitzer oder Mitarbeiter des Wikis, öffnen und Außenstehenden nur eine Leseansicht bieten. Dieser Editor könnte zum einfachen bearbeiten ein WYSIWYG-Editor ("What you see is what you get") sein, um Änderungen zu ermöglichen, auch ohne die Syntax zu kennen.

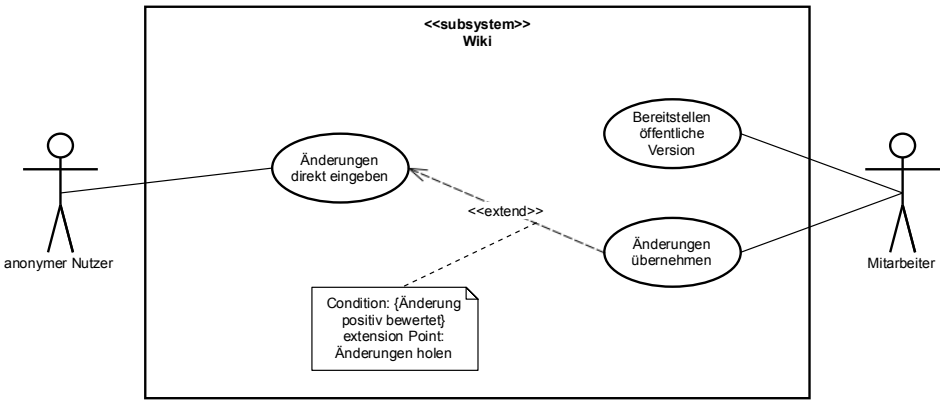

Abbildung 2.8: Anonymer Nutzer

So wird gewährleistet, dass Änderungen eines Wikis vor Spam geschützt werden, aber dennoch im eigenen Namensraum geändert werden können. Dies führt aber zu höheren Hürden um überhaupt mit der Bearbeitung einer Wikiseite anfangen zu können. Man könnte sich also überlegen, auf der gesperrten Wikiseite, für einen Außenstehenden einen Link zu der bearbeitbaren Wikiseite anzubieten, auf der jeder Schreibrechte hat. So bleibt einerseits die ursprüngliche Seite sicher vor Spam, anderseits kann jeder, auch ohne Anmeldung, mitarbeiten. Mitarbeiter des nicht öffentlich bearbeitbaren Wikis, können dann auch Änderungen aus dem öffentlich beschreibaren Wiki in ihr Wiki übernehmen. Im Use-Case Diagramm Abbildung 2.8 sieht man die beiden Benutzergruppen, anonymer Nutzer und Mitarbeiter eines Wikis. Dieser Mitarbeiter kann eine öffentlich bearbeitbare Version des Wikis auf der Musterwiki Plattform online stellen und lässt somit Änderungen von jedem zu, auch ohne Anmeldung. Dieses Wiki besitzt dann keine festgelegte Liste von Mitarbeitern, die Änderungen vornehmen dürfen.

2.5.6 Zusammenarbeit

Die Anforderungen an die Zusammenarbeit sind in Abbildung 2.9 dargestellt. Nun wird ein GitHub ähnliche Vorgehensweise gewählt. Möchte man an einem Wiki Änderungen vornehmen und ist kein Mitarbeiter mit Schreibrechten darauf, so macht man eine Kopie des Wikis und kann Änderungen in seinem Namensraum ausführen. Diesem Akteur soll es beim Ursprungswiki möglich sein, seine Änderungen mit einem "pull request", also einer Anfrage, einzureichen, da diese keine Schreibrechte auf das Wiki haben. Über eingereichte Änderungen können dann die Mitarbeiter und der Besitzer des Wikis diskutieren. Anschließend können sie Änderungen ablehnen, ins Wiki einarbeiten oder gemeinsam mit dem Beiträger überarbeiten.

Wenn die Mitglieder des Wikis entscheiden, dass eine Änderung den Ansprüchen genügt, kann diese sofort in das Wiki eingearbeitet werden. Ablehnungen können stattfinden, nachdem eine Überarbeitung als nicht zufriedenstellend angesehen wurde, oder falls die Änderungen von anfang an nicht abnehmbar scheinen. Sollten sie die Änderungen einbauen und somit das Wiki anpassen, könnte als Funktion im Wiki eingebaut werden, dass alle Kopien des Wikis, also die Beiträger, eine Benachrichtigung erhalten.

Da Wikis im eigenen Namensraum an die eigenen Anforderungen angepasst sein können, sollten die Änderungen im Ursprungswiki, nicht automatisch das kopierte Wiki ändern. Ein unabhängiges Arbeiten wäre sonst nicht vollständig gegeben. Man kann aber zusätzlich einbauen, für die Mitarbeiter einer Wikikopie, einen Hinweis anzuzeigen, dass es eine neue Version gibt und sie diese holen und einfügen können. Bei angepassten Wikis werden lokal die beiden Änderungen geholt und nach Bedarf vereinigt, oder es wird auf der Plattform direkt die neue Version geholt, die das kopierte Wiki überschreibt. Der Beiträger kann also nach Begutachtung die Änderungen holen und übernehmen.

Innerhalb eines Wikis können Rollen und Aufgaben, je nach Anforderung und Größe unterschiedlich sein. So können die Mitarbeiter direkt Änderungen im Editor einfügen, oder mit Hilfe von Git, Änderungen am lokalen Computer auf den Server hochladen. Mit Git und dem kompletten Wiki, welches auf dem Computer gespeichert werden kann, können alle Zusammenarbeitsmodelle von Git genutzt werden, die in Kapitel 3.3 dargestellt werden. Den Nutzern bleibt also selbst überlassen, wie sie sich organisieren wollen und können dies anpassen.

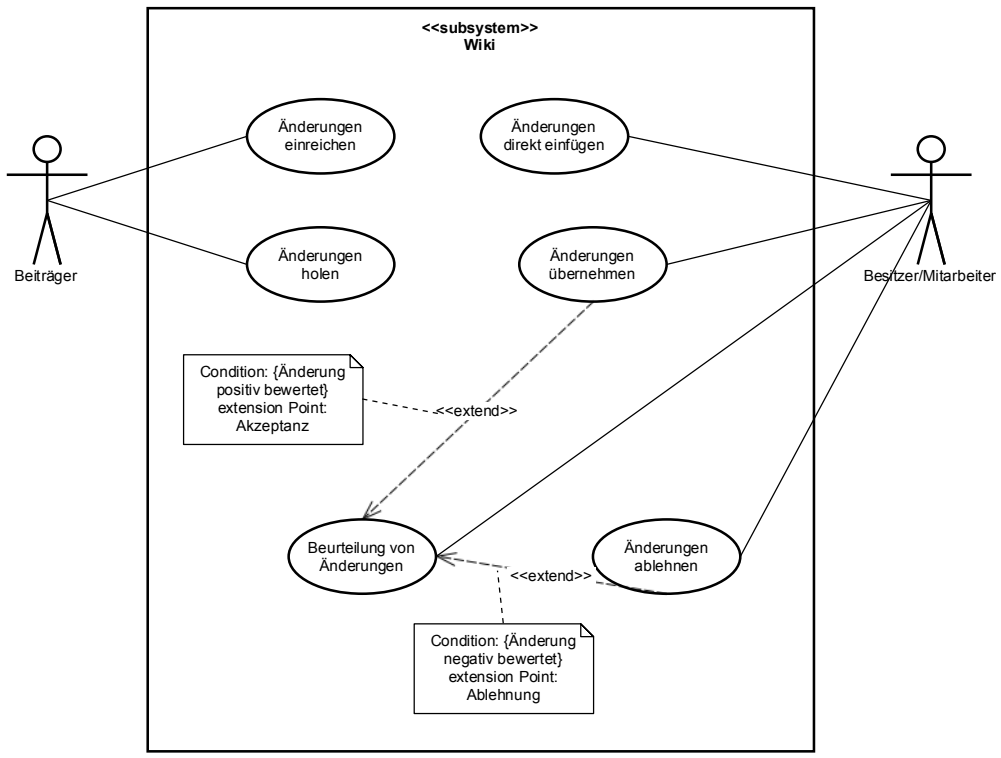

Abbildung 2.9: Anforderungen an die Zusammenarbeit

2.6 Ergebnisse

Als Ergebnis konnte ich mittels Literaturrecherche die Zusammenarbeit auf Onlineplattformen, wie Wikis und GitHub, herausarbeiten. Diese Exploration war wichtig, um ein geeignetes Modell für die Musterwiki Plattform zu finden, welche eine hohe Qualität gewährleisten soll und dabei verschiedene Zusammenarbeitsmodelle unterstützt. Da auch lokal und eigenständig am Wiki gearbeitet werden soll, scheint Git ein geeignetes Versionsverwaltungssystem. Die Wikis werden auf einer Plattform gehostet, wodurch ein GitHub ähnlicher Ablauf entstehen kann. Die möglichen Dimensionen und Workflows auf der Plattform, wurden im vorgehenden Kapitel besprochen.

2.7 Diskussion

Es stellt sich die Frage, wie sinnvoll ein Zusammenarbeitsmodell beim bearbeiten von Wikis ist, welches ein Modell ähnlich zu GitHub implementiert. Bei der Softwareentwicklung ist ein Ablauf mit Fork des Projekts und Pull Request, wenn Änderungen eingereicht werden wollen, sinnvoll, da mit dem Pull Request ein Reviewverfahren eingeleitet wird, mit dem noch Softwarebugs gefunden werden können. Außerdem können so weitere Probleme, wie Performanceschwierigkeiten, besprochen werden. Auch die Arbeitsweise mit verschiedenen Branches für Features, die den Mainbranch mit ausführbarer Software unberührt lassen, ist in der Softwareentwicklung ein gängiges und bewährtes Mittel, um qualitativ hochwärtige Software zu entwickeln. In Wiki Artikeln werden aber keine Softwarefehler sein, sollte die Syntax falsch geschrieben sein, so merkt dies der Autor nach Speicherung sofort an der angezeigten HTML Seite.

Wiki Artikel können jedoch falsche Informationen enthalten oder wenig ausgereift sein, die erst durch die Kontrolle der großen Masse gesichtet werden können, wie in Wikipedia. Auf der Musterwiki Seite werden aber Wikis zum Bereich patterns angeboten, die in ihrer Modellierung in der Zusammenarbeit mit Anderen entstehen. Im Unterschied zu Wikipedia, wird hier nicht schon bekanntes Wissen, wiedergegeben und gesammelt, sondern es entsteht neues Wissen und dient als Wissensmanagement gefundener Muster aus verschiedenen Bereichen. Es wird demnach effektiv nichts produktives entwickelt, aber der Grundstein für viele Anwendungen gelegt, die nach den Mustern ausgeführt werden können. Da viele Prozesse darauf aufbauen, sollte, um eine hohe Qualität zu gewährleisten, Abläufe gewählt werden, die ein möglichst fehlerfreies Erzeugen von Mustern verstärken.

Die Zusammenarbeitsmodelle die Änderungen, mit einem Pull Request, an das Ursprungswiki übergeben, können eine Diskussion um die bearbeiteten Wikis er-

zeugen und so optimale Abläufe in den Mustern abbilden. Je nach Größe der Gruppe in einem Wiki, können, wie bereits besprochen, verschiedene Workflows mehr Sinn machen und gewählt werden. Somit kann beim Festellen des falschen Zusammenarbeitsmodell, auch ein anderes Modell gewählt werden, um die Kollaboration so gut wie möglich zu unterstützen. Ein Modell welches an GitHub angelehnt ist, scheint in diesem Fall zur Bearbeitung der Wikis durchaus sinnvoll, wenn nicht sogar nötig, um die Kollaboration beim Entstehen neuer Muster zu unterstützen.

2.8 Ausblick

Als Ausblick kann auf der Plattform, wie auf GitHub, ebenfalls eine Reihe von sozialen Features eingebaut werden, die die Zusammenarbeit noch verstärken. Dabei können Funktionen zum Verfolgen, “follow“, von Benutzern eingestellt werden, oder besonders ausgereifte Wiki Artikel, durch die Nutzerschaft ausgezeichnet werden. Da die sozialen Komponenten auf Plattformen immer wichtiger werden, kann von einem positivem Gewinn, ähnlich zu GitHub, bei der Zusammenarbeit ausgegangen werden.

3 Ausarbeitung der Forschung

3.1 Online Kollaboration allgemein

Das Internet hat die Zusammenarbeit unter Menschen stark verändert und bietet heute eine große Bandbreite an verschiedenen Anwendungen, um auch Akteure mit räumlicher Distanz effektiv zu unterstützen. Dabei sollen diese Tools unterstützend wirken und dienen auch als Hilfestellung, um Arbeitsleistungen großer Gruppen, die an einer Aufgabe arbeiten, weiterhin effizient zu gestalten. (Hornstein, Fischer, Pertek & Koller, 03.12.2008)

Hornstein et al. (03.12.2008) sieht Online Kollaboration folgendermaßen:

> " Ziel der E-Collaboration ist es, mit Hilfe von webbasierten Informations- und Kommunikationslösungen kollaborationsintensive Prozesse zu optimieren. Der Schwerpunkt dieser Systeme liegt dabei auf denjenigen Prozessen, an denen viele Personen gemeinsam arbeiten und somit ein hohes Mass an Kommunikation erfordern."

Als asynchrones Zusammenarbeitstool wird, auch heute hauptsächlich, die E-Mail Kommunikation genutzt, da dies oftmals das erste Online Kollaborationstool ist, welches verwendet wird. [1] Als synchrones Tool werden häufig Instant Messenger eingesetzt, die eine zeitlich synchrone Kommunikation, mit Beteiligten, ermöglicht und so auch Gruppenunterhaltungen, trotz räumlicher Distanz, vereinfacht. (Fichter, 2005)

Die anfangs statische Struktur des Internets brach auf, nun wurde aus dem Internet eine sowohl lesbare als auch schreibbare Plattform, mit sozialem Schwerpunkt. Dies wirkte sich auch auf die Kollaboration aus. Beispiele sind Wikis, Blogs, soziale Netzwerke, Foren, "shared workspaces". (Mayer, 2013, S. 33 ff.)

Die bekannteste Form eines Wikis ist derzeit Wikipedia. Bei der cloudbasierten Kollaboration ist als wichtiger Vertreter, Google Drive, zu sehen.

3.1.1 Wikis und cloudbasierte Kollaboration

Da das Internet eine immer stärkere Rolle in der Zusammenarbeit einnimmt, werden ganze Dokumente zur gemeinschaftlichen Nutzung in "shared workspaces"

[1] Die Asynchronität entsteht dadurch, dass die Kommunikationspartner zu unterschiedlichen Zeiten miteinander kommunizieren, im Gegensatz zur synchronen Kommunikation, die zur selben Zeit für die Kommunikationspartner stattfindet. `siehehttp://www.uni-hildesheim.de/meum/index.php?option=com_content&task=view&id=267&Itemid=244`

ausgelagert. Beispiele hierfür sind Google Drive oder Dropbox mit einer cloudbasierten Technologie. (Notari & Hornegger, 2013, S. 23ff.)

Bei den meisten Wikis gibt es zwei wichtige Ansichten der Seite. Einmal die Ansicht einer klassischen Webseite und einmal die Bearbeitungsansicht zum Editieren des Inhalts. Beim Zusammenarbeiten an Wikis wird zwischen diesen beiden Ansichten gewechselt. Zum Bearbeiten kommt man in den Bearbeitungsmodus und beim Abspeichern wieder zur normalen Webseiten Ansicht. Diese Ansichten bieten auch für viele Wikis, beispielsweise Wikipedia, das Fundament der Versionsverwaltung. Speichert man eine neu bearbeitete Wiki-Seite, so wird von der Wiki Software eine neue Version erstellt, auf die bei Bedarf zurückgegriffen werden kann. Sollte an einer Wiki Seite mehrere Personen arbeiten, so wird die erstgespeicherte Version übernommen und alle anderen Bearbeiter werden beim Abspeichern auf einen Bearbeitungskonflikt hingewiesen, da sonst Änderungen überschrieben werden können. Eine andere Möglichkeiten, Bearbeitungskonflikte zu regeln, ist die Seite während der Bearbeitung für andere Bearbeitungen zu sperren, oder ein automatischer Vereinigungsvorgang mit aufzeigen von Konflikten. (Notari & Hornegger, 2013, S. 23ff.)

Im Gegensatz dazu, das Online Kollaborationstool Google Drive, ehemals Google Docs, welches ein hochladen von Dokumenten über einen Account, ermöglicht. Hier werden einzelne Dokumente hochgeladen und nicht wie bei Wikis über Links miteinander verknüpft. Die beiden Ansichten von Wikis gibt es bei Google Drive nicht mehr, da Dokumente nun immer in einem bearbeitungsfähigem Modus sind und von mehreren Akteuren parallel bearbeitet werden können. Hier liegt der Vorteil darin, dass es nicht zu Bearbeitungskonflikten kommt und eine neue Version nicht erst nach einem Abspeicherungsbefehl angelegt wird, sondern bereits bei Eingabe eines einzelnen Zeichens. Somit ist ein explizites Abspeichern nicht notwendig, da bereits ein Tastendruck eine automatische Speicherung generiert. Das bearbeitete Dokument wird sofort in allen anzeigenden Webbrowsern aktualisiert und lässt somit ein paralleles arbeiten zu. (Notari & Hornegger, 2013, S. 23ff.)

Da auf der Musterwiki Plattform, ähnlich zu GitHub, Wikiseiten hochgeladen werden und lokal am Computer ohne Internetverbindung bearbeitet werden können, ist die Musterwiki Plattform sowohl ein Wiki, als auch ein “shared workspace“. Die direkten Editiermöglichkeiten von Google Drive, könnten in einem Wiki ebenfalls übernommen werden, dafür müsste jedoch den Nutzern die Wiki spezifische Syntax bekannt sein.

3.2 Unterschiede in der Zusammenarbeit

Die Zusammenarbeit unter verschiedenen Menschen kann auf zwei unterschiedliche Arten erfolgen: durch Kooperation oder Kollaboration. Nach Schmalz (2007) gibt es eine Unterscheidung in der Aufteilung der Arbeit auf die Einzelpersonen. So ist eine Kooperation verschiedener Menschen an einer Aufgabe zu sehen, wenn eine klare Aufteilung des Aufgabengebiets auf eine gewisse Anzahl Personen, feststeht. Dies kann durch eine hierarchische Struktur, mit unterschiedlichen Entscheidungsebenen, durchgeführt werden oder indem jedem Beteiligten heterarchisch gleiche Rechte zugeteilt werden, die untereinander zu einer Einigung führen. So werden auch in Wiki-Systemen, beispielsweise in Projektwikis, diese Strukturen bedient, da hier das Wiki auf die bereits vorhandene Firmenstruktur trifft. (Schmalz, 2007)

Die Kollaboration unterscheidet sich darin, dass es keine klare Abgrenzung der Arbeitsteile gibt, sondern jeder an der kompletten Aufgabe mitarbeitet. Alle Mitarbeiter sind dabei gleichberechtigt und die Aufteilung der Arbeit ergibt sich dynamisch je nach Anforderung. Diese Form findet man heute, begünstigt durch die Vernetzung durch das Internet, in vielen Bereichen wieder. So findet sich diese Form auch in öffentlichen Wikis, wie der Wikipedia, bei der Akteure verschiedene Aufgaben übernehmen können, wieder. (Schmalz, 2007)

3.3 Zusammenarbeit mit Git

Git ist ein verteiltes Versionverwaltungssystem, welches die Änderungen an Dateien verwaltet. Verteilt bedeutet, dass beim arbeiten mit Git sich die komplette Projektgeschichte lokal auf dem Computer befindet, somit das komplette Repository und die Versionsgeschichte. So kann man auch ohne Internetzugriff an seinen Dateien arbeiten, was bei anderen Versionsverwaltungssystemen wie Subversion schwieriger möglich ist, da diese einen zentralen Server verwenden. Diese Versionsverwaltungssysteme wie Subversion sind demnach zentral. (Chacon & Straub, 2014, S. 27 ff)

Git ist bei der Zusammenarbeit sehr anpassungsfähig an den Anwendungsfall, da jeder Akteur zentraler Server sein kann, der Repositories bereitstellt, oder auch an anderen Repositories mitarbeitet. Git unterstützt auch ein einfaches Erstellen von eigenen Entwicklungszweigen, Branching, und vereinigen mehrerer Zweige, Merging. Im Folgenden soll nun betrachtet werden, wie Zusammenarbeitsmodelle auf Basis von Git aufgebaut werden können. (Chacon & Straub, 2014, S. 27 ff)

Zentralisierter Workflow

Der Zentralisierte Workflow, stellt einen Arbeitsablauf wie bei anderen zentralen Versionsverwaltunssystemen, beispielsweise Subversion dar. Dargestellt wird dies in Abbildung 3.1.

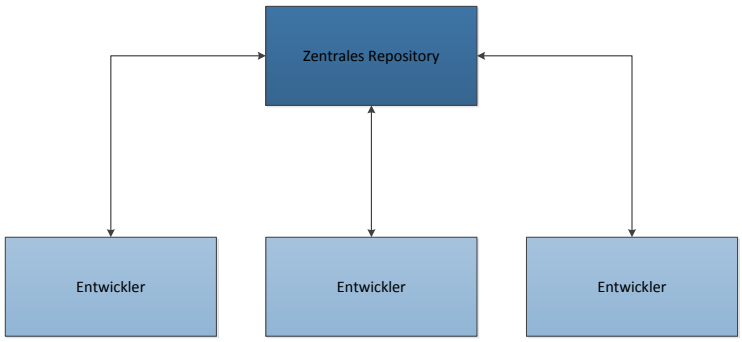

Abbildung 3.1: Zentralisierter Workflow; Quelle: nach (Chacon & Straub, 2014)

Es gibt ein zentrales Repository, welches Daten von anderen Repository Knotenpunkten entgegennimmt und damit synchronisiert wird. Auch hier ist es möglich, dass es zu Konflikten bei Änderungen kommt, wie in anderen zentralen Versionsverwaltungssystemen. Wenn, AkteurA und AkteurB, an Dateien Änderungen vornehmen und AkteurA seine Änderungen in das zentrale Repository liefert, wird AkteurB beim "pushen", hochladen, seiner Änderungen von Git gehindert, da sonst die Änderungen zuvor überschrieben werden. Git weist darauf hin, dass erst ein zusammenführen der beiden Änderungen stattfinden muss, um schließlich hochladen zu können. Um diese Zusammenarbeit zu realisieren wird ein zentrales Repository aufgesetzt, auf welches die Akteure Schreibzugriff bekommen und darauf hingewiesen werden, falls es zu Konflikten kommt. Dieses Zusammenarbeitsmodell ist in kleinen Gruppen gut einsetzbar. (Chacon & Straub, 2014, S. 151 ff.)

Integration-Manager Workflow

Ein weitverbreitetes Zusammenarbeitsmodell ist der Integration-Manager Workflow, wie in Abbildung 3.2 dargestellt. In diesem Modell, hat jeder Akteur Schreibrechte auf sein eigenes Repository und Leserechte auf externe Repositories. AkteurA möchte in seinem RepositoryA an einem Projekt von AkteurB, mit RepositoryB, arbeiten. Änderungen an externen Projekten können vorgenommen

werden, indem das externe RepositoryB, von AkteurA, nach RepositoryA kopiert wird. AkteurA kann Änderungen lokal durchführen und in das eigene, öffentliche, RepositoryA hochladen. Nach Kommunikation von AkteurA mit Projektbesitzer AkteurB, kann AkteurB den Klon als externes Repository auf seinem Computer speichern. Jetzt kann AkteurB die Änderungen mit einem seiner Zweige zusammenführen und sie öffentlich zugänglich in sein RepositoryB hochladen. Da hier das komplette Projekt kopiert und daran gearbeitet wird, kann man in seinem eigenen Rhythmus an eigenen Ideen arbeiten, die dann übernommen werden können, aber nicht müssen. (Chacon & Straub, 2014, S. 151 ff.)

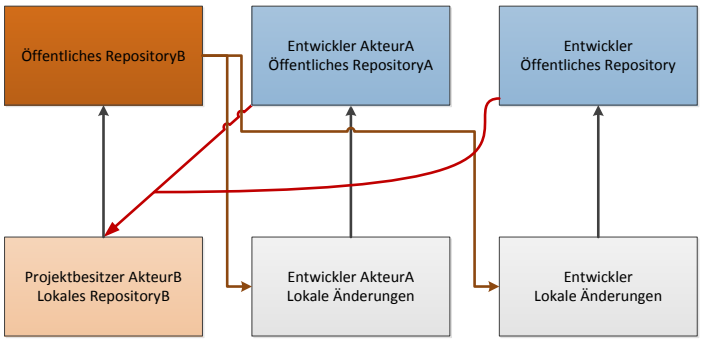

Abbildung 3.2: Integration-Manager Workflow; Quelle: nach (Chacon & Straub, 2014)

Dieses Zusammenarbeitsmodell ist auch auf GitHub zu finden, als “fork and pull“ Workflow, aus dem Kapitel 2.4.2.

Diktator und Leutnants Workflow

Der Diktator und Leutnant Workflow, Abbildung 3.3, bietet ein Zusammenarbeitsmodell, vor allem für größere Gruppen oder hierarchisch aufgebaute Strukturen, an, ist jedoch weniger verbreitet als der Integration-Manager Workflow. Hier gibt es viele normale Entwickler, die auf einem Arbeitszweig arbeiten und sich mit einem Masterzweig synchronisieren. Nun kommen die Leutnants zum Einsatz, die als Integration Manager für verschiedene Teile des Projekts verantwortlich sind, und “mergen“ die Änderungen der Entwickler in ihre Masterbranches. Für alle Leutnants ist am Ende ein wohlwollender Diktator zuständig, aus dessen Repository sich alle anderen Mitarbeiter ihre Repositories aktualisieren. Der Diktator wird die Änderungen der Leutnants mit seinem Masterbranch einpflegen und lädt die Änderungen in ein Referenzrepository. Diese Arbeitsweise wird beispielsweise

beim Entwickeln des Linux-Kernels angewandt.(Chacon & Straub, 2014, S. 151 ff.)

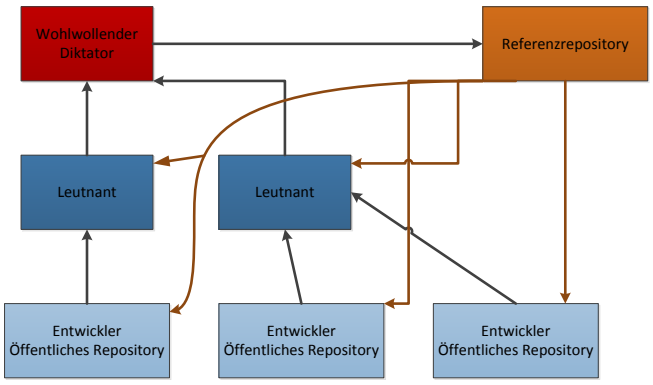

Abbildung 3.3: Diktator und Leutnants Workflow; Quelle: nach (Chacon & Straub, 2014)

3.4 Nutzung von GitHub

GitHub kann, wie bereits erwähnt, kostenfrei & öffentlich und kostenpflichtig & privat, verwendet werden. Auch größere Firmen, wie Windows, verwenden GitHub zur privaten Nutzung, um die Selbstorganisationsprozesse zu nutzen. (Kalliamvakou et al., 2015) Nach Kalliamvakou, Damian et al. (2014) verwenden Firmen, sowohl private, als auch öffentliche Repositories. Die privaten Repositories um wichtige Codeteile, die für den Firmenerfolg wichtig sind, nur intern sichtbar zu machen und öffentliche, um Entwicklern die Möglichkeit zu geben gewisse Codeteile, zu erweitern. Kalliamvakou, Singer et al. (2014) haben herausgefunden, dass viele Repositories auf GitHub nur für den persönlichen Gebrauch verwendet werden, ohne Kollaboration, auszuüben. Außerdem wird GitHub auch häufig außerhalb der Softwareentwicklung verwendet, beispielsweise als Speichermöglichkeit oder kostenlose Website. Anzumerken ist auch, dass viele Repositories inaktiv sind und nur wenige Commits aufweisen. (Kalliamvakou, Singer et al., 2014)

3.5 GitHub und andere Plattformen

Nicht nur GitHub bietet eine Codehosting Plattform auch Sourceforge, Google Code oder Bitbucket. Sourceforge ist im Gegensatz zu GitHub nicht dezentral, sondern zentral verwaltet, mit dem Versionsverwaltungssystem SVN. Nach Thung et al. (2013, S. 325) werden durch die sozialen Komponenten in GitHub, die es in Sourceforge nicht gibt, die Zusammenarbeit unter den Entwicklern stark gefördert. Open Source Software Projekte, sind unter anderem, gekennzeichnet durch Selbstorganisation, eigenständiges arbeiten und weniger Kommunikationsaufwand. Diese Arbeitsweise wird durch die dezentrale Arbeitsweise von GitHub unterstützt. BitBucket nutzt ebenfalls Git, jedoch sind bei Bitbucket private Repositories ebenfalls kostenlos[2]. GitHub stellt Mirrors vieler Projekte online, die auf anderen Plattformen gehostet werden und auch andere Versionsverwaltungssysteme verwenden. Viele Entwickler verwenden nicht nur GitHub in der Softwareentwicklung, sondern auch andere, externe Tools. (Kalliamvakou, Singer et al., 2014)

[2] `https://bitbucket.org/plans`
aufgerufen am 03.03.2015

References

Aiello, M. L. & McFarland, D. (2015). *Social Informatics - SocInfo 2014 International Workshops.* Springer. Zugriff auf http://dx.doi.org/10.1007/978-3-319-15168-7

Bell, P. & Beer, B. (2015). *Introducing GitHub.* O'Reilly.

Chacon, S. & Straub, B. (2014). *Pro Git.* Apress.

Dabbish, L., Stuart, C., Tsay, J. & Herbsleb, J. (2012). Social Coding in GitHub: Transparency and Collaboration in an Open Software Repository. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work* (S. 1277–1286). Zugriff auf http://doi.acm.org/10.1145/2145204.2145396

Ebersbach, A., Glaser, M., Heigl, R. & Warta, A. (2008). *Wiki Kooperation im Web.* Springer.

Fichter, D. (2005). *The Many Forms of E-Collaboration: Blogs, Wikis, Portals, Groupware, Discussion Boards, and Instant Messaging* (Bericht). Northern Lights Internet Solutions.

Gatzweiler, A. (o. J.). *Enterprise Wiki.* Zugriff am 29.03.2015 auf http://www.innovationsmethoden.info/files/method/Enterprise%20Wiki.pdf

Hornstein, M., Fischer, A., Pertek, M. & Koller, M. (03.12.2008). *E-Collaboration: Mehrwert durch moderne Kommunikationsmittel schaffen* (Bericht). namics ag.

Kalliamvakou, E., Damian, D., Blincoe, K., Singer, L. & German, D. (2015). Open Source-Style Collaborative Development Practices in Commercial Projects Using GitHub. In *Proceedings of the 37th International Conference on Software Engineering (ICSE '15).* Zugriff auf http://leif.me/papers/Kalliamvakou2015.pdf

Kalliamvakou, E., Damian, D., German, D. & Singer, L. (2014). *The Code-Centric Collaboration Perspective: Evidence from GitHub* (Bericht). DCS-352-IR, University of Victoria.

Kalliamvakou, E., Singer, L., Gousios, G., German, D., Blincoe, K. & Damian, D. (2014). The Promises and Perils of Mining GitHub. In *Proceedings of the 11th Working Conference on Mining Software Repositories.* ACM. Zugriff auf http://doi.acm.org/10.1145/2597073.2597074

Lima, A., Rossi, L. & Musolesi, M. (2014). Coding Together at Scale: GitHub as a Collaborative Social Network. In *Proceedings of 8th AAAI International Conference on Weblogs and Social Media (ICWSM 2014).* Zugriff auf http://arxiv.org/abs/1407.2535

Mayer, F. L. (2013). *Erfolgsfaktoren von Social Media: Wie "funktionieren"Wikis?* Lit Verlag.

Moskaliuk, J. (2008). Das Wiki-Prinzip. *Konstruktion und Kommunikation von Wissen mit Wikis. Theorie und Praxis.*, S. 17-27.

Notari, M. & Hornegger, B. D. (2013). *Der Wiki-Weg des Lernens.* hep verlag.

Reagle Jr., J. M. (2010). *Good Faith Collaboration.* The MIT Press.

RightScale Inc. (2014). *GitHub Workflow.* Zugriff am 29.03.2015 auf `http://support.rightscale.com/12-Guides/Chef_Cookbooks_Developer_Guide/04-Developer/Source_Control_Management_Systems/GitHub/Clone_vs._Fork`

Schmalz, S. (2007). Zwischen Kooperation und Kollaboration, zwischen Hierarchie und Heterarchie. Organisationsprinzipien und -strukturen von Wikis. *kommunikation@gesellschaft, Jg. 8.*. Zugriff auf `http://nbn-resolving.de/urn:nbn:de:0228-200708083`

Thung, F., Bissyandé, T., Lo, D. & Jiang, L. (2013). Network Structure of Social Coding in GitHub. In *Proceedings of the 2013 17th European Conference on Software Maintenance and Reengineering* (S. 323–326). IEEE Computer Society. Zugriff auf `http://dx.doi.org/10.1109/CSMR.2013.41`

Viégas, F., Wattenberg, M. & Dave, K. (2004). Studying Cooperation and Conflict between Authors with history flow Visualizations. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (S. 575–582). Zugriff auf `http://doi.acm.org/10.1145/985692.985765`

Wagner, C. & Prasarnphanich, P. (2007). Innovating Collaborative Content Creation: The Role of Altruism and Wiki Technology. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences.* IEEE Computer Society. Zugriff auf `http://dx.doi.org/10.1109/HICSS.2007.277`