

**Fakultät Wirtschaft Studiengang Wirtschaftsinformatik**

Evaluierung einer neuen Release Historie für Energieversorger

**2. Projektarbeit**

Im Rahmen der Prüfung zum Bachelor of Science (B. Sc.)

**Verfasserin/Verfasser: Kemal Celik**

**Kurs: WWI20B4**

**Partnerunternehmen: cortility gmbh Wissenschaftlicher Betreuer: Frau Julia Dziumbla Abgabedatum: 04.09.2022**

# Selbstständigkeitserklärung

Ich versichere hiermit, dass ich die vorliegende Arbeit mit dem Thema: „Evaluierung einer neuen Release Historie für Energieversorger“ selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

(Unterschrift)

**Inhaltsübersicht**

[Selbstständigkeitserklärung II](#_bookmark0)

[Inhaltsübersicht III](#_bookmark1)

[Inhaltsverzeichnis IV](#_bookmark2)

[Abkürzungsverzeichnis VI](#_bookmark3)

[Abbildungsverzeichnis VII](#_bookmark4)

[Tabellenverzeichnis VII](#_bookmark5)

1. [Einleitung 1](#_bookmark6)

**Inhaltsverzeichnis**

1. [Einleitung 1](#_bookmark6)
   1. [Problemstellung und Motivation 1](#_bookmark7)
   2. [Zielsetzung 2](#_bookmark8)
2. [Theoretische Grundlagen 3](#_bookmark9)
   1. [Definitionen release management 3](#_bookmark10)

**Abkürzungsverzeichnis**

**Abbildungsverzeichnis**

**Tabellenverzeichnis**

1. **Einleitung**

Release Management spielt eine wichtige Rolle in dem Erfolgskonzept einer Firma. Es widerspiegelt die Produktqualität und beeinflusst die Zusammenarbeit mit dem Kunden sowie dessen Zufriedenheit. Hierdurch werden Kunden über bestehende und neue Funktionen eines Produkts aufgeklärt. Automatisierungen dieses Systems sorgen für konstante Zufriedenheit bei Kunden und durch folgende Minderungen des Aufwands, auch bei Mitarbeitern.

* 1. **Problemstellung und Motivation**

Cyber Sicherheit, Bugs oder Geschwindigkeit, all diese Aspekte benötigen neue Software oder neue Updates, um auf neustem Stand im Internet mitzuhalten. Ein Softwareanbieter hat dadurch stets die Herausforderung, seinen Kunden über die Funktionsweisen und Änderungen seiner Produkte aufzuklären. Hierfür ist das Release Management einer Firma ausschlaggebend. Die cortility gmbh nutzt hierfür eine von SAP zur Verfügung gestellte Software. Innerhalb dieser Software werden während dem Entwickeln alle für den Release notwendigen Informationen dokumentiert. Um es in PDF Form an den Kunden überbringen zu können, muss jeder Mitarbeiter diese Dokumentation ein zweites Mal in eine Excel-Tabelle übertragen, welche letztendlich in eine PDF konvertiert wird. Diese Excel-Tabelle dient Zeit gleich als Release Historie, über welche jede Softwareänderung hinterlegt und dokumentiert bleibt. Die cortility gmbh möchte diesen Ablauf komprimieren und optimieren, um vor allem an Zeit und Aufwand zu sparen.

* 1. **Zielsetzung**

Ziel dieser Arbeit ist eine optimierte Lösung durch mögliche Automatisierung der Release Historie zu finden oder zu entwickeln. Dies soll Mitarbeiter entlasten und für eine höhere Endqualität bei den Kunden sorgen. Als wissenschaftliche Frage und Ergebnis dieser Arbeit sollen innerhalb einer Soll-Ist Analyse verschiedene Lösung Ansätze auf ihre jeweiligen Vor- und Nachteile evaluiert werden. Außerdem wird diskutiert, welche Bedingungen für eine Verbesserung der aktuellen Release Historie sinnvoll erscheinen.

1. **Theoretische Grundlagen**

Neue Software oder Softwareänderungen werden immer mit Informationen zur Aufklärung über ihre Funktionen veröffentlicht. Die Qualität dieser Veröffentlichung ist das Resultat eines ausgereiften Release Managements. Die Release Historie dient als Indikator für den Erfolg des Release Managements. Eine Release Historie erfasst jegliche Releases und bietet daher die Möglichkeit Folgen und Veränderungen durch Releases nachzuvollziehen.

* 1. **Definitionen Release Management**

Release Management bezieht sich auf Prozesse rund um die Veröffentlichung von Software und Softwareänderungen. Hierzu gehören Planung, Design, Entwicklung, Testen und Controlling. Diese Prozesse sind essenziell für die Softwareentwicklung und verlaufen allerdings nicht immer in derselben Geschwindigkeit. Daher wird ein Management System benötigt, durch welches alles aufeinander abgestimmt wird. Um hierbei den Erfolg zu garantieren, müssen beteiligte Entwickler gerade bei unerfahrenen Kunden, so präzise wie möglich ihre Arbeit beschreiben. Das heißt, es muss nachvollziehbar sein, zu welcher Software Version die gelieferten Infos gehören und es muss garantiert sein, dass alle zur Nutzung relevanten Änderungen vollständig und verständlich dokumentiert werden. Sollten dennoch Probleme oder Fragen bei einer Zielperson aufkommen, muss für diesen erkennbar sein, wer für diese Änderungen verantwortlich war und wie dieser zu kontaktieren ist. Somit ist das Hauptziel von Release Management, Änderungen, Updates, Patches oder Roll-Outs in geeigneter Form an den Endnutzer zu bringen. Hierbei ist es besser, flexibel zu sein was die Form der Releases angeht. Damit sich der Kunde selbst entscheiden kann, ob er Beispielsweise eine PDF, eine ZIP-Datei oder direkt über eine Webseite auf seine Informationen zugreifen möchte.

* + 1. **Release Notes**

Release Notes müssen alle relevanten Informationen zum Nutzen einer neuen Software beinhalten. Bei Neuerungen, Verbesserungen oder Fehlerbehebungen dienen Release Notes als Kommunikation zwischen Entwickler und Nutzern der Software. Wichtige Inhalte fangen bereits bei der Dokumentation einzelner Versionen an. Nutzern sollte klar sein, zu welcher Software Version und zu welchem Datum eine Änderung gehört. Falls Probleme aufkommen, muss der Endnutzer ebenfalls über einen Verantwortlichen in Kenntnis sein, welcher in dem Release gekennzeichnet sein sollte. Um diese Probleme vorzubeugen, sollte auf komplexe Fachsprache verzichtet werden, die Beschreibung kurzgehalten werden, die Info auf die jeweilige Zielperson angepasst werden und Quellen geboten werden, welche bei eigen Bedarf eine tiefgründige Recherche ermöglichen. Durch die Pflege einer Historie aller Releases kann jederzeit nachvollzogen werden, wer wann was geändert hat. Dies hilft Entwicklern bei der Koordinierung der Dateien und bietet ihnen die Möglichkeit, alte Stände von neuen zu differenzieren und erleichtert eventuelle Wiederherstellungen.

* 1. **SAP Solution Manager**

SAP ist ein Unternehmen, welches im Jahr 1972 gegründet wurde und als Softwarehaus Lösungen rund um vor allem Enterprise resource planning Software bietet. Diese umfasst hauptsächlichen Programme in den Bereichen Finanzwesen, Personalwesen, Beschaffung, Produktion, Materialwirtschaft, Vertrieb und Marketing. Eine dieser Softwarelösungen ist der SAP Solution Manager. Dieser fungiert als eine Unterstützung der Systemumwelt, durch eine Sammlung von Werkzeugen, Inhalten und Services, welche dabei helfen, Änderungen bei zentralen Monitorings, Projektinformationen, Ereignissen oder Problemen zu verwalten. Des Weiteren dient der Solution Manager als Tunnel für den Kundensupport, da dieser zum Beispiel Daten direkt mit der Kundeninfrastruktur austauschen kann und diese Werkzeuge meistens bei den Kunden innerhalb der SAP-Infrastruktur integriert sind.

* 1. **abapGit**

ABAP ist die Abkürzung für „Advanced Business Application Programming“ und kommt als proprietäre Programmiersprache aus dem SAP-Umfeld. Diese wurde primär für die Massendatenverarbeitung in kommerziellen Anwendungen entwickelt und glänzt vor allem mit zusätzlichen Funktionen gegenüber anderer Programmier-sprachen. Beispielsweise beinhaltet ABAP für die dynamische Speicherung und Bearbeitung von tabellarischen Massendaten im Arbeitsspeicher interne Tabellen. Außerdem sind in die Sprache zum einen Schnittstellen zu anderen Programmierumgebungen, Datenbankzugriffe über SQL, Performance-Optimierung von Datenbankzugriffen über die SAP-Pufferung oder Schnittstellen zu XML integriert. Innerhalb der ABAP Entwicklung ist es ebenfalls von Vorteil, dass SAP das Prinzip der Abwärtskompatibilität betreibt. Dies bedeutet, dass nach Einführung neuer Anweisungen alte Sprachelemente bei Bedarf weiterhin verwendet werden können, was Nutzern einen großen Sprachumfang zur Verfügung stellt. Des Weiteren sorgt die Abwärtskompatibilität für den Erhalt der Funktionsfähigkeit bisheriger Entwicklungen oder Kundenanpassungen. AbapGit ist ein in ABAP vollständig entwickelter open-source git client für ABAP. Git selbst ist ein Open-Source-Projekt, entwickelt 2005, um mittlerweile als weitverbreitetstes Versionskontrollsystem zu dienen. Git dient somit als Werkzeug, unterschiedliche Softwareprojekte und Versionen zu verwalten oder zu überwachen. Benutzern ist es hierbei möglich, ihre Projekte zeitgleich auf lokalen Datenträgern zu speichern und eine Kopie auf die zentralen Ablagen von Git zu synchronisieren. Somit ist es zum einen möglich, ohne eine Server Verbindung an Projekten zu arbeiten und zum anderen neue Stände jeder Zeit auf Git zu verwalten. Die Git Historie sorgt dafür, dass bei jedem Upload(„commit“), der vorherige Stand erhalten bleibt und jeder Zeit genutzt werden kann. Commits beinhalten alle Veränderung nach dem Letzten und bis zum nächsten Commit und werden durch Zweige, welche auf den nächsten Commit zeigen, miteinander verbunden. Dies hat den Vorteil, dass der Verlauf einer Entwicklung besser nachvollzogen werden kann oder bei möglichen Fehlern, jederzeit über eine ältere Fehlerfreie Version, korrigiert werden kann. Zusätzlich ist es Nutzern möglich, die hochgeladenen Dateien zusätzlich mit Informationen in Form sogenannter „read me“ zu versehen. Außerdem ist das parallele Uploaden und Downloaden oder auch „push“ und „pull“ genannt, ebenfalls möglich. Durch die SAP-Erweiterung Abap Git ist es nun möglich, eine Git-Versionskontrolle mit ABAP-Applikationsservern zu verbinden, wodurch ABAP-Entwickler ihre Projekte darüber in die zentralen Ablagen von Git ablegen können. Ein grundlegendes Ziel der Entwicklung von ABAP Git war es, kostenlos und das Projekt zu sein, welches am einfachsten zu installieren ist. Dabei wurde vor allem Wert daraufgelegt, ohne Weiteres durch einfaches copy and paste in das eigene ABAP-Entwicklungssystem, sofort einsatzbereit zu sein. Durch das Verknüpfen mit GitHub kann anschließend zu jeder Zeit die neuste abapGit Version geladen werden

* 1. **JIRA**

Der Name Jira stammt von dem japanischen Godzilla „Gojira“ und soll eine Bezugnahme sein, zu einem Bugtracker namens „Bugzilla“. Jira selbst ist ein Web und Java EE basierte Software, welche für Teams aller Art durch Funktionen wie anpassbare Instrumententafeln, eigene Suchfilter, Statistiken, RSS oder E-Mails unterstützend wirkt. Vor allem in Bereichen Fehlerverwaltung, Problemverwaltung, Testmanagement oder operativem Projektmanagement. Durch die Flexibilität Jiras, kann diese an verschiedenste Teams individuell angepasst werden. Beispielsweise bietet Jira Tools wie das Jira-Scrum-Board, welches laufende Aufgaben visuell darstellt.

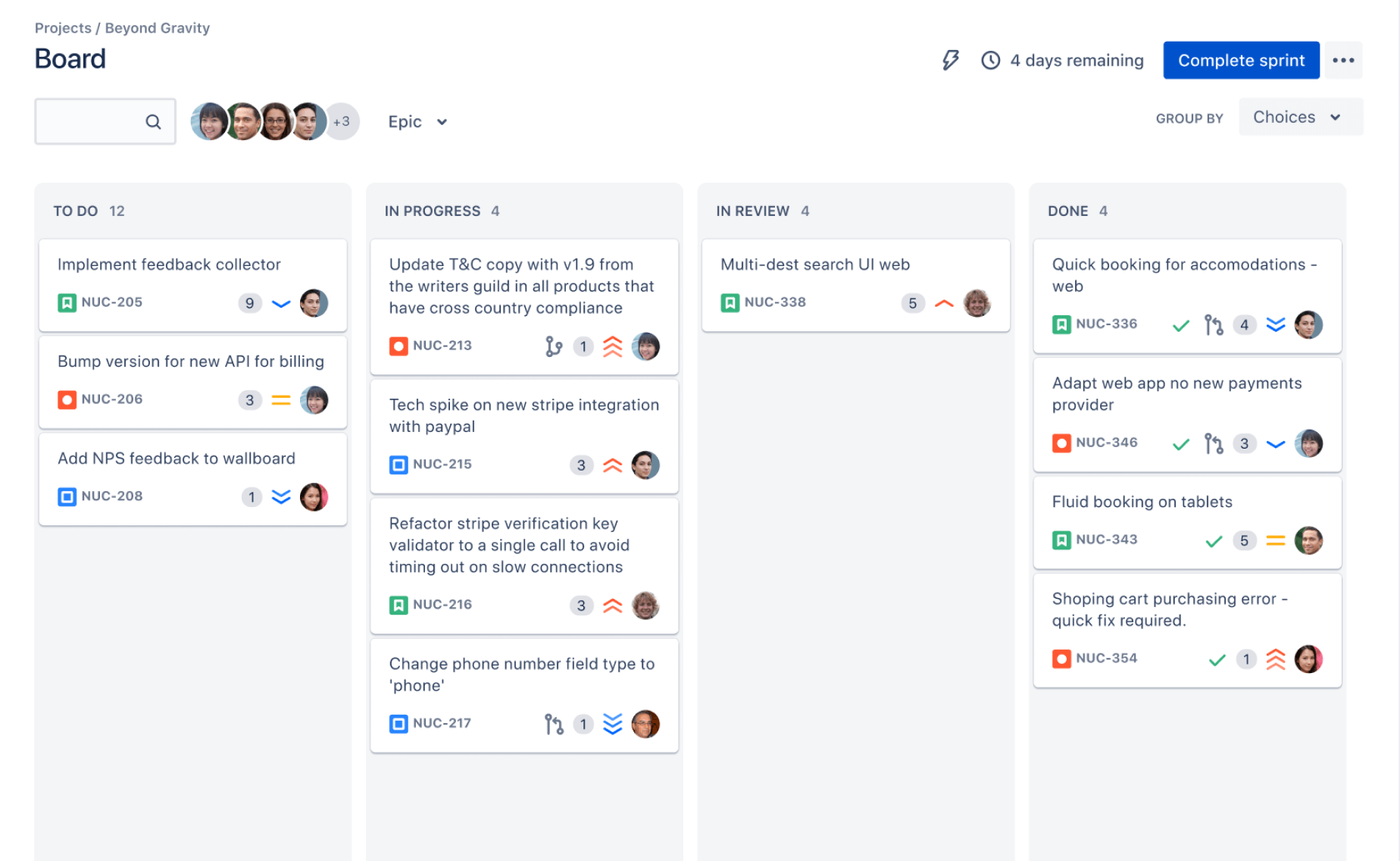
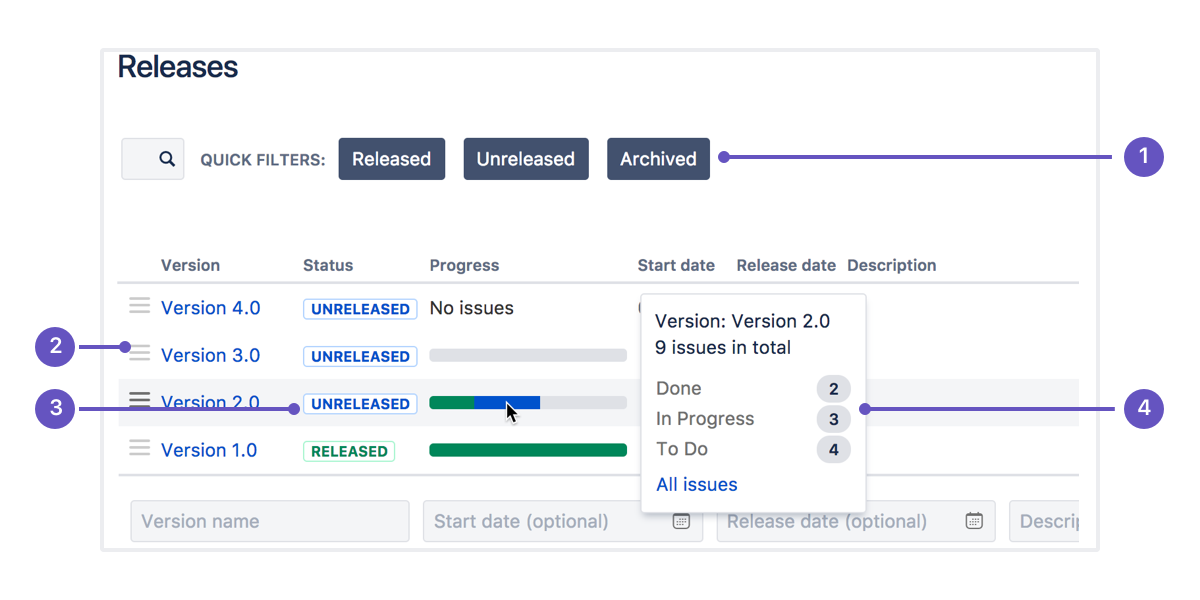


Abb. 1 Interaktionsablauf Benutzer und Chatbot [SDP18]

In Abb. 1 ist zu entnehmen, dass ein Scrum-Board eines Projekts „Board“ exakt vier angezeigten plus drei weiteren Nutzern zugeordnet ist. Innerhalb dieses Scrums ist es nun möglich, Kategorien zu erzeugen, in welchen die Aufgaben aufgeteilt werden. Den jeweiligen Aufgaben können nun dem Projekt hinzugefügte Nutzer, zugeteilt werden. In Abb. 1 ist beispielsweise die erste Aufgabe „Implement feedback collector“ der ersten Kategorie „TO DO“ zugeordnet. Jira bietet hier ebenfalls die Option einzelne Kategorien oder Aufgaben mit dem gedrückt halten, der linken Maustaste an eine beliebige Stelle zu verschieben und neu somit anzuordnen. Diese Funktionen unterstützen Teams durch einen transparenten Überblick auf die Arbeit und vereinfacht die Planung sowie die Kommunikation innerhalb des Teams.

## Release Hub

Der Release Hub ist eine wichtige Bonusfunktion in Jira. Dieser bietet die Möglichkeit, Releases innerhalb eines Projekts zu verwalten. Hierzu zählen Funktionen wie die Schnellsuche, um in der Historie vergangene Versionen zu filtrieren, eine Versionsliste, welche die Versionen per Drag-and-drop anordnet oder Status und Fortschritt der Releases. Verschiedene Versionen sind per Knopf druck neu erzeugbar, veränderbar- oder verschiebbar. Des Weiteren ist jede Version mit einem Start sowie Release Datum versehen und besitzt ein eigenes Feld für eine Versionsbeschreibung.

****

**Abb.2 Jira**

# 2.5 Excel

Excel ist ein Tabellenkalkulationsprogramm und ein Produkt der Microsoft Office Reihe „Big Four“. Nutzern ist es möglich, innerhalb von Tabellen, Zahlen oder Texte zu formatieren und zu organisieren. Hierbei unterstützen verschiedene Funktionen wie Berechnungen oder Auswertungen von Daten. Des Weiteren können Ergebnisse automatisch sortiert und in grafischen Formen als Diagramm, Pivot-Tabelle oder Chart dargestellt werden. Excel kommt vor allem zum Einsatz in den Bereichen Unternehmensanalyse, Personalwesen, Produktionswirtschaft und Projektmanagement. Das Arbeiten in Excel funktioniert über Zellen, in welche Zahlen, Texte oder auch ganze Rechenwege eingefügt werden können. Vorprogrammieren erlaubt es beispielsweise durch Einfügen eines „=“ Rechenwege automatisch berechnen zu lassen.

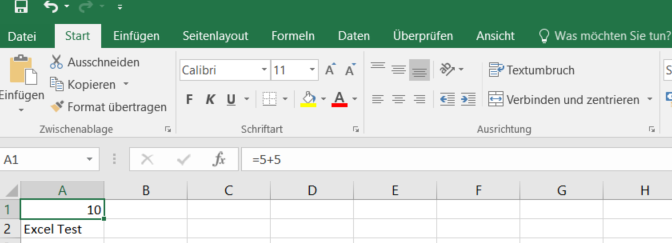


Abb3.

Außerdem lassen sich Zellen gruppieren und nach gewünschten Ergebnissen auswerten. Zum Beispiel wäre es möglich, aus einer Gruppe von Zahlen den Mittelwert, den Durchschnitt oder nur die größte Zahl zu ermitteln. Als Einstieg bietet Excel bereits verschiedene Tabellenformatvorlagen, wodurch Zellen bereits in mit ausgewählten Funktionen erzeugt und in gewünschte Position formatiert werden.

* 1. **Ist-Stand innerhalb der cortility gmbh**

Der aktuelle Prozess der Release Notes beginnt mit dem Öffnen einer sogenannten „dringenden Änderung“. Hierfür müssen die Mitarbeiter sich zunächst in dem SAP Solution Manager anmelden, welcher diese Funktion bietet. Diese Dringende Änderung unterteilt sich in zehn zwischen Schritte  


Abb4.  
In dem ersten Schritt wird diese angelegt. Hierfür müssen zunächst Allgemeine Daten hinterlegt werden, wie zum Beispiel zugehörige Mitarbeiter, ein Anlege beziehungsweise Fälligkeit Datum, oder eine Kurzbeschreibung der anzulegenden Änderung. Anschließend kann der Vorgang in den Status Entwicklung gesetzt werden, woraufhin über ein automatisches Popup-Fenster, ein dazugehöriger Transportauftrag angelegt werden muss. Dieser besitzt eine kurze Beschreibung, sowie ein SAP System und Mandant. Nach dieser Konfigurierung werden die Test Schritte durchgeführt, bevor die dringende Änderung in das Produktivsystem importiert werden darf. Ab diesem Schritt wird der Auftrag zum Ende überführt und abgeschlossen. Damit Kunden nun über Erneuerungen oder Änderungen aufgeklärt werden können, muss eine Release Historie erzeugt werden. Derzeit wird dies in der cortility gmbh über eine Tabelle in Excel gepflegt. Hierhin werden die meisten Daten aus einer Dringenden Änderung noch einmal separat in die Excel Datei kopiert und neu angeordnet. Die Datei selbst ist ein verschiedene Ebenen unterteilt. Die erste Ebene der Excel zeigt Ausliefereinheiten, welche Pakete und ihre Subpakete aufteilt. Zu diesen wird ein Datum, ein Transportauftrag und eine Bemerkung zugeordnet. Als Orientierung hilft eine Legende, welche die Zellen durch farbige Markierungen beschreibt. Diese unterteilen sich in die Farben Gelb für Fragen, Grün für neue Formate, Blau für Sicherungen, Rosa für alle Tabellen oder Objekte, welche mittransportiert werden, Lila für Unterpakete, welche mit aufgenommen werden, Rot für Fehler und Weiß für keine Veränderungen. Alle weiteren Ebenen gliedern jeweils zusammengehörige Auslieferungseinheiten. Diese werden jeweils in eine einheitliche Tabelle gegliedert, welche durch sechs Spalten innerhalb der Kopfzeile beschrieben werden. In der ersten Spalte steht „Datum Änderung“, zu diesen Zeitpunkten fand eine Änderung im Quellcode statt. Die folgende Spalte ist mit „Beschreibung der Änderung“ auf das jeweilige Datum bezogen. Es existiert keine einheitliche Bestimmung des Inhalts. Anhand der derzeitigen Verwendung sind folgende Fälle zu entnehmen: Kurzbeschreibungen des Sachbearbeiters, eine Fehlerbeschreibung vor der Korrektur oder Kommentare für bestimmte Mitarbeiter. Die dritte Spalte „Objekt“ liefert mit dem Objektnamen, die Stelle der Änderung und kann entweder eine Methode, ein Report oder eine Quellcodestelle sein. Die vierte Spalte beinhaltet den „Kundenwunsch“, dieser unterteilt sich in Auftragsgeber bei Erweiterungen eines Produkts oder in Meldende bei Problem- und Fehlermeldungen. Als Kennzeichnung beinhaltet die fünfte Spalte „MA-Kürzel“, einen zuständigen Mitarbeiter Kürzel. Die letzte Spalte „Transportauftrag“ umfasst die Auftragsnummer eines Transportauftrags, einer Dringenden Änderung oder einer Meldung aus dem Solmanager.

## 3.1.1 Umfrage zum Ist-Stand

Innerhalb dieser Arbeit wurde eine Umfrage mit 14 Fragen rund um die aktuelle Release Historie der cortiltiy gmbh entworfen. Diese wurde speziell an die Vorgegebene Excel Release Historie in der Firma angepasst. Der Fragebogen wurde in einer Rundmail an alle cortility Mitarbeiter verschickt, damit nur interne Daten festgestellt werden. Die durchschnittliche Teilnahmedauer wurde auf dreit bis fünf Minuten angesetzt und Betrug Letzt endlich bei 22 Teilnahmen 02:13 Minuten. Die Fragen unterteilten sich in die Kategorien Benutzer Methodik, Usability, Qualität, Datensicherheit und Persönliche Informationen. Die Umfrage wurde auf einen Monat terminiert, wonach die Analyse der Daten beginnen konnte. Zu Beginn bekam die Zielgruppe eine kurze Einleitung, welche als Aufklärung über die Umfrage diente. Anschließend wurde zuerst danach gefragt, welche Methode für eine Release Historie benutzt wird. Teilnehmer erhielten innerhalb des Multiple-Choice-Verfahrens vier Antwort Möglichkeiten. Mit dieser Frage sollten Abweichungen der Vorgesehenen Methode einer Release Historie in Excel, festgestellt werden. In Abb5. Ist zu entnehmen, dass von insgesamt 18 Teilnehmer Antworten 12 Excel, 2 abapGit, 6 Jira und 3 andere Methoden ergaben. Das bedeutet, dass ein Teil der Mitarbeiter mehr als eine Methode benutzten. Diese Verteilung zeigt, dass es an einer einheitlicheren Nutzung einer Release Historie mangelt. Die anschließenden Fragen waren nur noch für Excel Nutzer relevant. Für nicht Nutzer galt die „weder/noch“ Option bis zu den Persönlichen Fragen. Die erste Frage bezog sich auf die Regelmäßigkeit, in welcher Excel Nutzer die Historie Pflegen. 33,33 % gaben an die Historie eher selten bis nicht zu pflegen, wodurch Info Lücken in der Release Historie entstehen können.

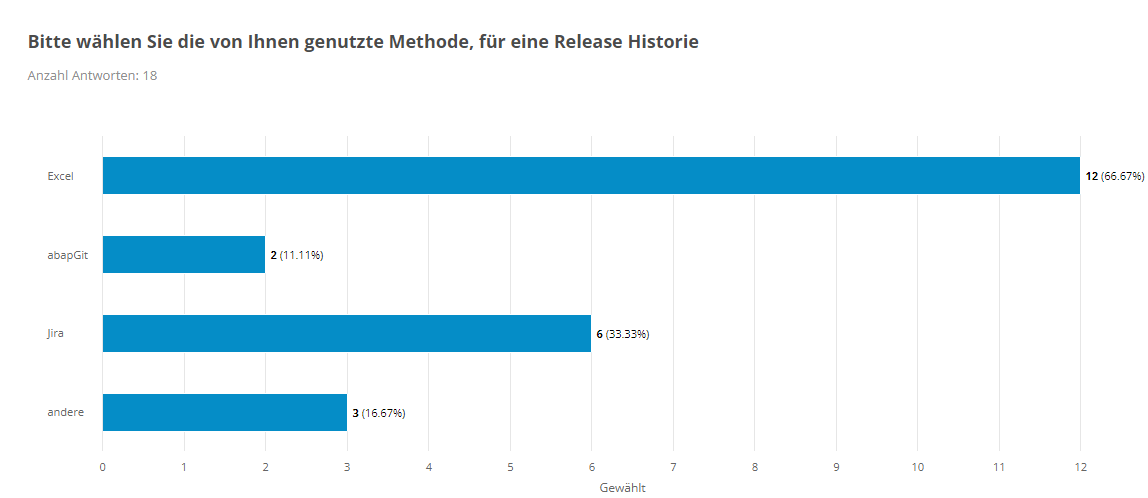


Abb5.

Die darauffolgende Seite beinhaltet vier Fragen über die Kategorie Usability. Diese sollte die Zufriedenheit des Pflegeaufwands, sowie die Benutzerfreundlichkeit hervorbringen. Als Antwort Möglichkeit hielt sich diese Umfrage für Zustimmung Fragen einheitlich. Jede Frage erhielt die fünf Antwort Optionen „zutreffend“, „eher zutreffend“, „weder/noch“, „eher unzutreffend“ und „unzutreffend“. In Abb6. Ist zu entnehmen, dass der Durschnitt der Usability Antworten in beide Richtungen schwankt. Dabei finden mehr Befragte die Excel als leicht zu pflegen und diese Pflege auch als Sinnvoll, dafür stimmten im Schnitt mehr für „eher unzutreffend“ bei der Benutzerfreundlichkeit insgesamt. Es ist hervorzuheben, dass nicht eine Frage innerhalb dieser Kategorie einmal „zutreffend“ oder „unzutreffend“ erhalten hat. Dies bedeutet zwar, dass die Usability befriedigend bis gut ist, allerdings Defizite aufweist, welche in dem Soll Stand zu berücksichtigen sind.

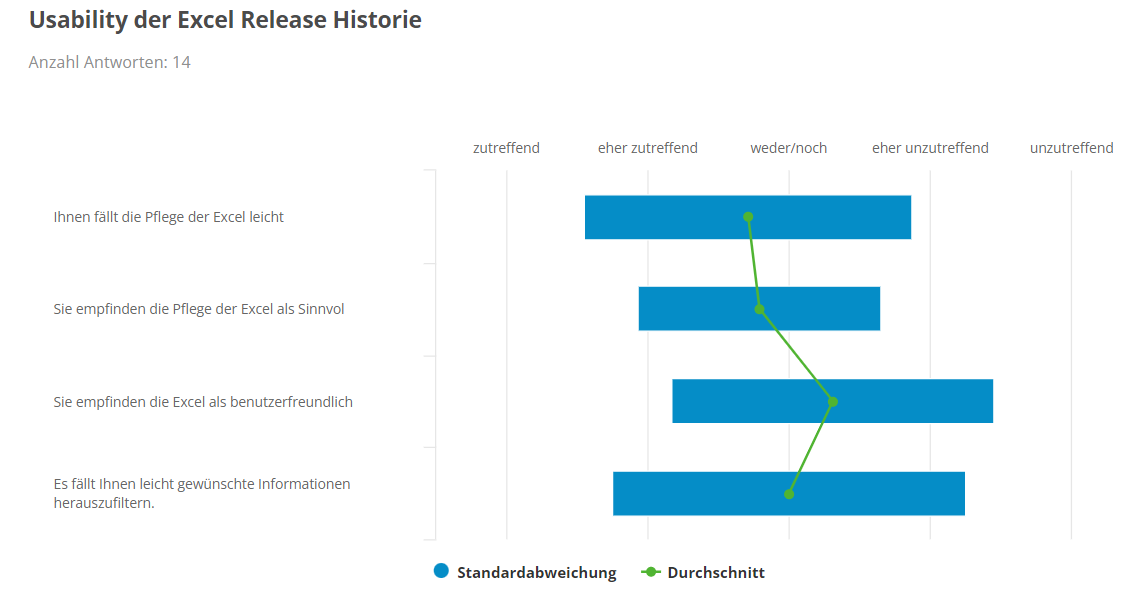


Abb.6

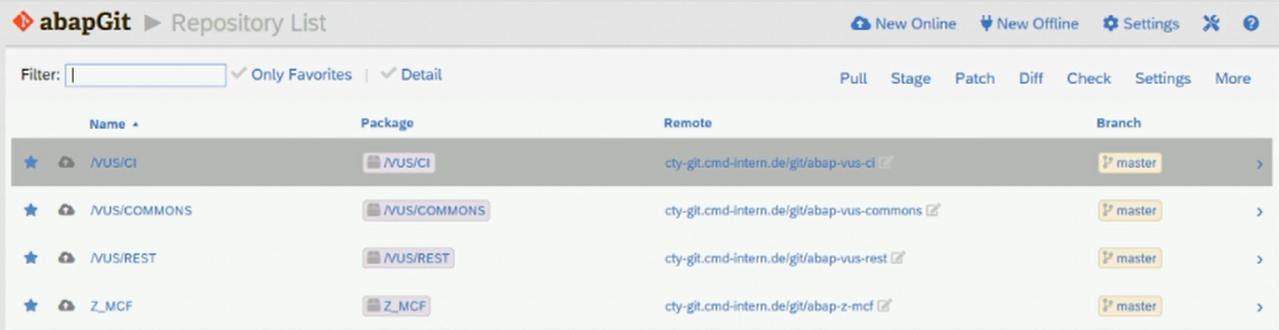
Die nächste Kategorie umfasst qualitative Fragen über die Release Historie. Zusätzlich zu eigenen Beurteilung der Befragten, wurde auch auf die Zufriedenheit der jeweiligen Kunden Wert gelegt. Der Durschnitt der Antworten innerhalb der Kategorie Qualität befand sich in allen drei Fragen im negativen Bereich. Die zweite Frage „Sie finden die Excel anschaulich und übersichtlich“ wurde am schlechtesten bewertet und als einzige Frage mit „unzutreffend“ beantwortet. Dies zeigt, dass weder Mitarbeiter noch Kunden zufrieden sind mit der Qualität der Daten innerhalb der Excel und diese sogar als eher weniger Sinnvoll erachten. Die letzte Kategorie Datensicherheit bezieht sich zum einen auf die Sicherheit vor Fremdzugriffen und die Sicherheit vor ungewolltem Datenverlust. In Abb. 7 ist zu erkennen, dass sich der Durschnitt eindeutig gegen die Sicherheit der Excel Datei stimmt. Am unzutreffendsten ist die Schutz vor Fremdzugriffen. Zusätzlich hat es den Antworten nach den Schein, dass es keine einheitliche Datenstruktur innerhalb der cortility gmbh gibt. Dies kann ebenfalls zu Datenredundanzen führen und zusätzlich die Übersichtlichkeit und Benutzerfreundlichkeit beeinträchtigen, da Benutzer sich erst mit den jeweiligen Datenstrukturen zurechtfinden müssen. Die Alterspanne der Teilnehmenden betrug zwischen 23 und 51 Jahren. In Verbindung mit dem Alter wurden die Teilnehmer nach ihrer Erfahrung mit dem Thema Release Management in Jahren befragt. Diese verteilte sich von 0 Jahren Erfahrung bis hin zu 25 Jahren Erfahrungen in dem Relevanten Themen Gebiet. Aus den Umfragen Ergebnissen war ebenfalls zu entnehmen, dass Teilnehmer mit Erfahrung zwischen 10 und 25 Jahren eher Jira oder abapGit, statt Excel als Methode für eine Release Historie angegeben haben. Lediglich Teilnehmer mit weniger Erfahrung nutzten nur ausschließlich Excel. Dies lässt vermuten, dass Jira und abapGit erfahrenen Nutzer mehr Zufriedenheit, Sicherheit und Komfort bietet, da diese Punkte sogar bei Excel Nutzer schlecht abschnitten.

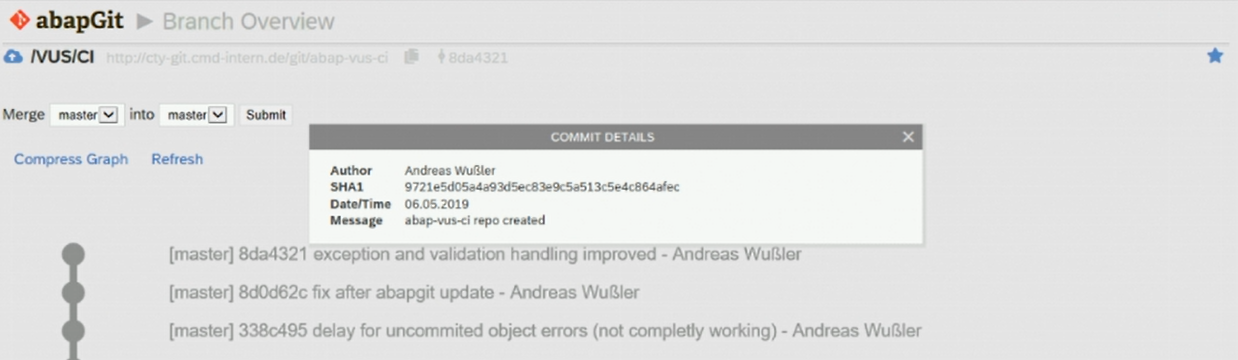
4 Lösungsansätze

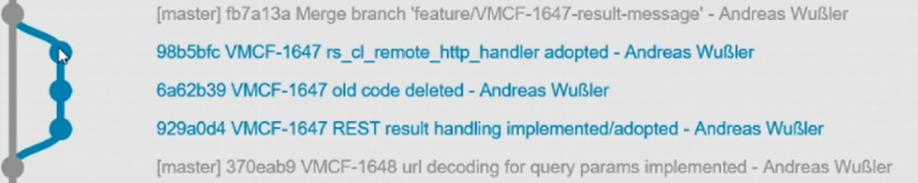
Zusammengefasst lässt das Ergebnis der Umfrage feststellen, dass die Excel Release Historie Mängel aufweist, welche als Orientierungsbasis für Anforderungen an Verbesserungsmöglichkeiten dienen. Die Verbesserungen werden aufgrund der verteilten Benutzung von Excel, Jira und abapGit unter den Mitarbeitern der cortility gmbh, auf drei Lösungsansätze begrenzt. Die Lösungen werden unter Einfluss der genutzten Kategorien aus der Umfrage ausgewertet und miteinander evaluiert.

# 4.1 abapGit

AbapGit ist ein Open Source Projekt, welches zunächst über einen GitHub Repository installiert werden muss. Der Link <https://github.com/ababGit/abapGit> führt zu dem Repository, über welches man nach einer Einführung auf die Webseite docs.abapgit.org weitergeleitet wird. Diese Webseite beinhaltet einen Downloadlink für abapGit und eine umfassende Dokumentation, welche zu Beginn eine Kurzvorstellung des Projekts aufzeigt. (abapGit1) Die Installation ist vollständig dokumentiert und gelingt in wenigen Minuten. Anschließend haben Benutzer die Möglichkeit, auf Zahlreiche Projekte zuzugreifen oder eigene zu erstellen. Um ein Projekt zu erstellen, müssen Nutzer ein Repository über Github erzeugen. Hierbei empfiehlt es sich in der Regel, wenn diese Datei mindestens eine README sowie eine Startdatei hat. Anschließend bietet Github eine Funktion dieses Repository in abapGit zu klonen. Als nächsten Schritt besteht die Möglichkeit eine Kommunikation zwischen SAP-Systemen oder zwischen Eclipse und den Projekten aufzubauen. Eine aufgebaute Verbindung zu Github sorgt außerdem dafür, dass ständig die neuste Version von abapGit zur Verfügung steht oder je nach Wunsch auch automatisch geladen wird. Die aufgebaute Verbindung zu den SAP Systemen sorgt nun dafür, dass sobald Mitarbeiter Änderungen „comitten“, diese sofort in dem für das Projekt angelegte Repository automatisch hinterlegt werden. Die einzelnen Repository´s werden in der abapGit „Repository List“ hinterlegt. In Abb. Ist eine Beispielansicht der cortility gmbh. Das Overlay von abapGit bietet dem Nutzer an, die Repositorys Normal oder als Favoriten zu hinterlegen.



Innerhalb eines Repositorys werden die Commits automatisch dokumentiert. AbapGit bietet zusätzlich die Option, einzelne Commits wie in Abb. Zu sehen, mit einer eindeutigen ID zu versehen und zusätzlich mit einem Autor, einer SHA1, einem Datum sowie einer kurzen Beschreibung zu pflegen. Praktisch hierbei ist, dass diese Funktionen als Notiz innerhalb der Commits gespeichert werden und nur bei Bedarf durch Öffnen der Notiz zum Vorschein kommen. In Abb. Verweist der Mauszeiger auf sogenannte Commit-Zweige. Die Zweige können sich auch aufteilen und müssen aber wieder zusammengeführt werden auch „merge“ genannt. (git2)  




# 4.2 Jira

Jira entstand als Testsoftware von der australischen Firma Atlassian. Diese bietet ihr Produkt kostenlos zum Download auf ihrer Website atlassian.com. Über dieses können Nutzer nun einen Account erstellen und damit beginnen, Projekte zu erzeugen, Team Mitglieder in diese Projekte einzuladen und ihre Projekte aus verschiedenen Jira Vorlagen aufzubauen. Je nach Benutzungsbedarf bietet Jira zwei Upgrade Pakete an, welche vor allem die Benutzer Zahl verbessern. Statt den 10 Benutzern je Team, können schon ab der monatlichen 75% Standard Version bis zu 35.000 Benutzer pro Projekt zugeteilt werden. Die Hauptfunktionen bleiben zunächst dieselben.

**Scrum Boards** Über diese Funktion können geplante und laufende Aufgaben nahezu auf einen Blick überschaut und gesteuert werden.

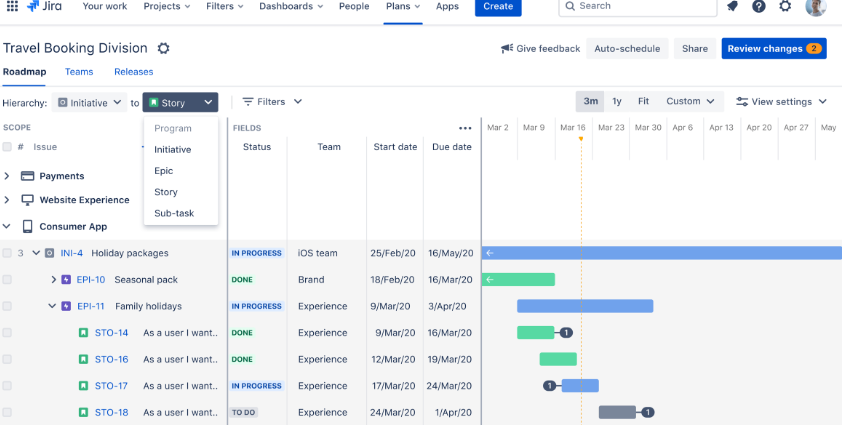
**Backlog** Das Backlog bietet die Funktionen einer To-Do Liste. Aufgaben können in dieser Liste erzeugt, abgeschlossen und überschaut werden. Nach Abschluss einer Aufgabe wird diese automatisch und zeitlich geordnet im Backlog hinterlegt und ist jederzeit nachvollziehbar.

**Agile Berichtsfunktion** Jira bietet mit über Dutzenden Grafiken Nutzern die Möglichkeit, eine Echtzeit Berichterstattung zu erzeugen, welche rückwirkend auswertbar bleibt.

**Apps und Integrationen** Jira wurde so konfiguriert, sodass es möglich ist, Projekte mit etlichen Tools unter anderem auch bekannte wie zum Beispiel Confluence, Git, Zoom oder Microsoft. Hierfür bietet Atlassian einen eigenen online Marktplatz und schafft eine Übersicht zwischen kostenlosen und kostenpflichtigen Erweiterungen, um die Jira Software zu ergänzen.

**Automation** Die Automation durch Trigger ist in Jira eine beliebte Funktion, welche sich allerdings innerhalb der Nutzer Pakete unterscheidet. Bei kostenlosen Kunden begrenzt sich die Nutzung auf 100 Automationen. Für Standard und premium Nutzer steigen die Zahlen auf 500 und 1000 Automationen pro Nutzer.

**Roadmaps** Die Jira Roadmaping Software bietet ein Interface, um Übersichtspläne über geplante Releases zu erstellen. Das Interface bietet eine anschauliche Vorlage über den Status eines Releases beteiligte Teams, Beginn und Dauer der Releases. Darüber hinaus bietet die Funktion auf die Releases sensibilisierte Filter, Zeit Einstellungen und Feedbackfunktionen. Die in Abb zu entnehmenden Releases werden hier nach Datum sortiert und verbleiben so als Release Historie. Der Nachteil hierbei ist, es fehlen Funktionen, diese Releases ausreichend zu beschreiben und zu sortieren. Des Weiteren unterscheiden sich die Funktionen für Standard und premium Kunden darin, dass bei Standardkunden lediglich ein Team je Roadmap zugewiesen werden kann. Somit sind Unternehmen mit mehr als 10 Beteiligten nur durch den Erwerb des premium Pakets befähigt, für alle diese Funktion freizuschalten. Außerdem werden weiter Funktionen wie automatisierte Vergleiche verschiedener Roadmaps von verschiedenen Teams ermöglicht oder beispielsweise Abhängigkeiten zwischen Teams ermittelt.



**Projektarchivierung** Damit abgeschlossene Projekte in Jira nicht gelöscht werden, können premium Kunden die Funktion der Projektarchivierung freischalten. Diese Funktion, welche für Standardkunden ausfällt, bietet einen Bereich, in welchem alle Projekte nach Abschluss automatisch hinterlegt werden und nach Belieben anzuordnen sind. Eigene Filter erlauben es alte Projekte leicht bei Bedarf wiederzufinden.

Zusammengefasst bietet Jira vor allem für premium Kunden umfassende Funktionen für die Projektplanung und Projektdurchführung. Jira ermöglicht es trotz einer beliebig großen Anzahl von Teams eine Übersicht über alle Projekte zu behalten und diese zu managen. Im Rahmen einer Release Historie bietet Jira zudem Automationen, welche Projekte und darin stattfindenden Abläufe und Releases hinterlegen, sortieren und durch Filter wieder auffindbar machen. Allerdings mangelt es Jira an Möglichkeiten, diese Daten für zum Beispiel Kunden der Unternehmen ausführlich zu beschreiben und zu erklären. Des Weiteren bietet Jira alleine keine Möglichkeit, Dateien wie Quellcodes oder SAP Pakete anzuhängen oder darzustellen. Da diese Funktionen essenziell innerhalb einer Release Historie sein können, wurden für Jira Verknüpfungen und add Ons evaluiert, welche Defizite für eine Release Historie ergänzen können.

## 4.2.1 Jira – SAP Solution Manager

Mit Jira ist es möglich, dritt Anbieter den SAP Solution Manager mit Jira zu verknüpfen. Bis lang haben bereits einige Produkte hierfür auf dem Markt etabliert. Unter anderem ChaRM, ConnectALL, Cross-Connector und einige auf Github geladene Projekte. Diese helfen dabei den Abläufen des SAP Solution Managers direkt mit Jira Projekten zu synchronisieren, um somit statt des Managers das Jira Interface nutzten zu können. Ziel ist es, verarbeitete Informationen des Solution Managers in eine übersichtliche und flexible Form innerhalb Jira zu überführen. Durch diese AddOns können über Jira Probleme gemeldet, Transportaufträge bearbeitet oder Kunden Feedback ausgetauscht werden, welche automatisch an den Solution Manager weiterleitet werden. Mitarbeiter können weiterhin im Solution Manager arbeiten, über welchen die Informationen automatisch in Jira wiedergegeben werden. Dies hat den Vorteil, dass alle Teilnehmer innerhalb des Jira Projekts bei Änderungen benachrichtigt werden. Diese Änderungen werden zusätzlich in dem angelegten Projekt abgespeichert und sind jederzeit für berechtigte einsehbar. Die verschiedenen Produkte unterscheiden sich teilweise in Preisen, einzelnen Funktionen und dazugehörigen Ansprechpartnern für Supportfälle. Da ein konkreter Vergleich den Rahmen dieser Arbeit sprengen würde, wurde lediglich eine Zusammenfassung der wichtigsten Funktionen erstellt, um eins dieser Produkte in Zukunft nach einer Evaluierung in Betracht ziehen zu können.

## 4.2.2 Jira – Git

Ähnlich wie bei Jira und dem Sap Solution Manager lassen sich auch die Systeme Git und Jira miteinander integrieren. Somit kann jedes Git Repository mit einem beliebigen Jira Projekt verknüpft und genutzt werden. Atlassian besitzt einen eigenen online Markt, auf welchem 75 Angebote aktiv sind für eine Git Integration mit Jira. Die Git Integration der Firma „GitKraken“ ist mit 315 Bewertungen und 9000 Installationen mehr als das Nachfolger Programm der Firma Move Work Forward, die meist genutzte Integration. Geworben wird mit einer einfach und kostenlosen Installation, worauf eine monatliche Gebühr von 10 US Dollar folgt. Innerhalb des Angebots werden folgende Funktionen gelistet, welche sich positiv für die Erzeugung einer Release Historie auswirken:

**Kunden Support** Kunden wird innerhalb der USA und der EU ein Kundensupport geboten.

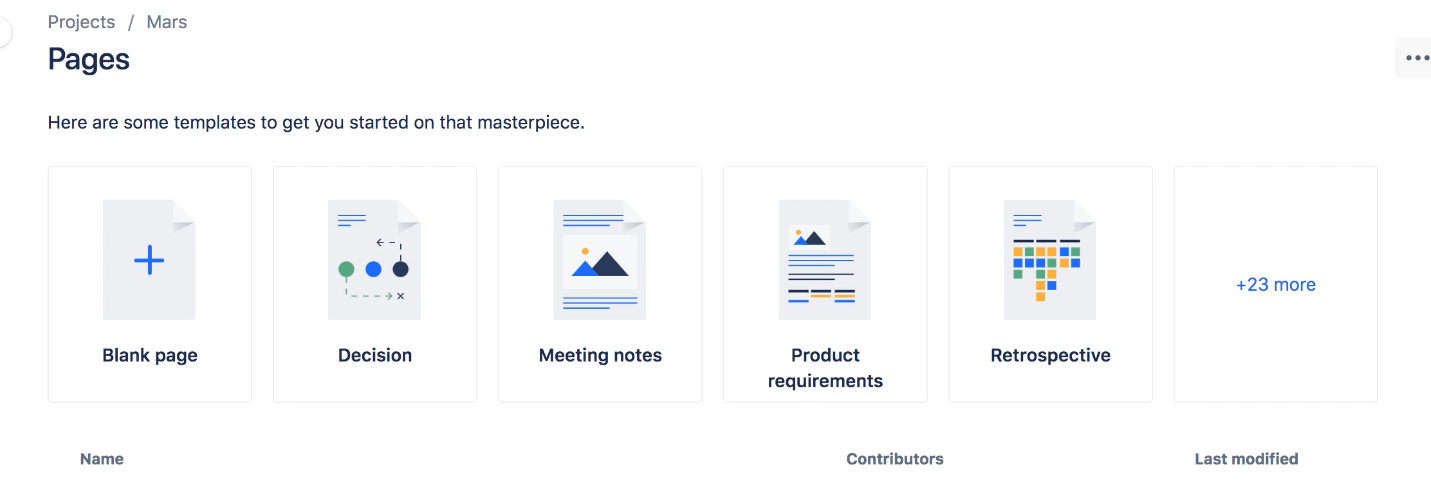
**Webhook Indexing** Git arbeitet via Webhook Indexing, sprich es braucht keine zwei Seite Kommunikation zwischen den Programmen, da Git bei eintreten von Ereignissen automatisch alle Beteiligten informiert. Diese Informationen werden automatisch in Jira als Historie gespeichert und können jederzeit nachgesehen werden.

**Automation triggers supported** Jira bietet vorgefertigte Trigger, welche unter vorbestimmten Bedingungen automatisch Funktionen auslösen. Hierbei ist es möglich bestimmte Abläufe automatisch in eine Release Historie zu überführen. Über die eigene Support Webseite von atlassian ist es möglich, alle für Jira vorgefertigten Trigger und ihre Funktionen nachzulesen.

**Smart Commits** Mit der Smart Commits Funktion wird es Repostiroy Nutzer möglich, Commits über Jira mit Informationen wie Beschreibungen, Datierungen, automatische Erkennung der Bearbeitungszeit oder einen im Jira Interface vorhandenen Status zu ergänzen. Beim Erzeugen einer Release Historie sind somit bereits alle Commits mit den notwendigsten Informationen versehen und müssen nur noch gelistet werden.

## 4.2.2 Jira – Confluence

In Jira gibt es eine eingebaute Funktion, um Jira Projekte mit Confluence-Bereichen zu verbinden. Confluence hat ebenfalls Bereiche, um Verknüpfungen hinzuzufügen. Dort kann dann die URL des Jira Projekts eingefügt werden, dadurch kann anschließend beliebig zwischen Confluence und Jira gewechselt werden. Jira bietet nun die Möglichkeit, in Jira selbst Seiten in Confluence zu erzeugen und diese anschließend dort zu bearbeiten. Hierdurch wird zeitaufwendiger Kontextwechsel vermieden. In Abb werden 28 Seiten Vorlagen durch Jira dargestellt. Innerhalb diesen Bereiches enthält Jira zusätzlich eine automatische Historie aller Änderungen für den Confluence Bereich. Nutzer, welche das Arbeiten in Confluence präferieren, können umgekehrt durch die Verknüpfung Jira Projekte innerhalb von Confluence erzeugen und verwalten. Der Vorteil in Confluence liegt darin, dass alle Daten in der Confluence Cloud hinterlegt werden und somit keine eigenen Speicherstände verbraucht werden müssen. Da diese schnell unübersichtlich werden können, bietet Jira einen umfangreichen Suchfilter. Mit diesem ist es möglich, sowohl in Jira als auch in Confluence nach Daten zu durchsuchen. Dies ermöglicht es beispielsweise in der Release Historie alte Informationen sofort wieder zu finden. Dieser Filter ist sensibilisiert, um einzelne Wörter, Projekte, bestimmte Zeitspannen oder Projektteilnehmer zu filtrieren.



Confluence mit Jira zu integrieren, hat also vor allem den Vorteil, dass alle Projekte an derselben Stelle dokumentiert bleiben. Durch die einheitliche Struktur und die Status Funktion durch Jira, können Kunden als Team Mitglied ständig Zugriff zu ihren Informationen erlangen und sich über die Filter Funktion schnell zurechtfinden. Einer Atlassian Kundenumfrage nach gaben über 60% an, dass die Integration sowohl die Kommunikation von Entwicklern und nicht Entwicklern steigerte als auch die Geschwindigkeit von Projektabschlüssen.

# 5 Entwicklung Prototyp

Als Grundlage für die Entwicklung des Release Historien Prototypen dienen die kostenlos zu Verfügung stehenden Funktionen durch Jira, abapGit und Confluence. Die Entwicklung erfolgt innerhalb einer Verknüpfung dieser Softwareumgebungen. Die genannten Softwareumgebungen werden bereits in der Cortility anderweitig verwendet und werden daher als Technologie verwendet, um einen Prototypen zu entwickeln.

# 5.1 Modellierung

Für Jira, Confluence und Git müssen zunächst Benutzerkonten angelegt werden. Jira und Confluence können gemeinsam über ein Benutzerkonto auf der Webseite von Atlassian erworben werden. Ein Git Konto wird über die Github Seite erzeugt. Um die Sicherheit zu erhöhen bieten beide Webseiten außerdem 2 Faktor Authentifizierungen.

In unbestimmter Reihenfolge ist zu Beginn ein Jira Projekt, ein Git Repository und eine Confluence Seite zu erstellen. Für das Jira Projekt reicht es über den „Erstellen“ Knopf, ein Projekt mit einem Namen und einer kurzen Beschreibung zu erzeugen. In Confluence ist genauso vorzugehen. In Github muss über „New“ ein neues Repository erzeugt werden, welches bis auf einen Namen zunächst leer bleibt. Anschließend können die Software miteinander verknüpft werden. Um Jira mit Confluence zu verknüpfen, bieten Jira und Confluence Bereiche an, um die jeweilige URL des Jira Projekts und der Confluence Seite hineinzukopieren. Daraufhin wird alles innerhalb der eingefügten URL automatisch und zeitgleich mit einer dortigen Änderung angezeigt und aktualisiert.

# 5 Fazit & Ausblick

Ziel der Arbeit war es, Lösungsansätze vorzustellen, welche die aktuelle Release Historie der cortility gmbh optimieren oder ablösen können. Hierfür wurde eine Umfrage erstellt, mit welche die Zufriedenheit der Mitarbeiter sowie Defizite der Release Historie festgestellt werden sollte. Nutzer mussten zunächst angeben, welche Methode sie für die Historie verwenden. Auf Basis dieser Antworten wurden genannte Methoden in einer Produktevaluation analysiert und spezifisch Funktionen für eine Release Historie herausgearbeitet.

Als Ergebnis lässt sich zusammenfassen, dass Teilnehmer der Umfrage die Methoden Excel, Jira und abapGit verwenden, um eine Release Historie zu pflegen. AbapGit erwies sich als am vorteilhaftesten, um Entwicklungsschritte zu dokumentieren und diese Dokumentation automatisch in Form von Commit Zweigen sortiert angezeigt zu bekommen. Im Vergleich zu Excel haben Entwickler eine direkte Verknüpfung zwischen Quellcode und relevanten Informationen wie Beschreibung des Codes, Datierung und einer Mitarbeiter Kennzeichnung. Allerdings ist die derzeitige Release Historie für den Kunden übersichtlicher und besitzt mehr Informationen, als abapGit dokumentieren lässt.

Die andere angegebene Methodik Jira bietet eine Oberfläche, um vor allem das Anforderungsmanagement einer Firma zu unterstützen. Teams können innerhalb eines Projekts auf Funktionen zurückgreifen, mit welchen sie Release relevante Informationen einfach, anschaulich und leicht durch Filter wieder auffindbar erzeugen können. Auch wenn Jira einige Informationen, welche für die Release Historie Relevant sind automatisch dokumentieren kann, fokussiert sich dieses Programm mehr um das Managen der Releases selbst, als um das Erzeugen der Release Historie. Hierbei mangelt es an Bereichen in denen Releases ausreichen beschrieben werden können und die Oberfläche selbst, bietet wenig Funktionen die Informationen zu bearbeiten. Aufgrund einiger Defizite in Jira ergaben sich Möglichkeiten Verknüpfeng zu ergänzenden Softwares zu integrieren.

Hierbei wurde zum einen Git mit Jira integriert, um somit das Entwicklungsmanagement zu ergänzen und zum Weiteren wurde eine Jira und Confluence Verknüpfung analysiert. Innerhalb Confluence können die erzeugten Releases automatisch auf der mit dem Projekt verknüpften Seite übertragen werden und durch die Funktionsweise von Confluence ausreichend beschrieben und letztendlich als PDF Datei exportiert werden, um an den Kunden zu gelangen.

Als gleichwertige Lösungsansatz wurde die Verknüpfung zwischen dem SAP Solution Manager und Jira evaluiert. Diese Verknüpfung wies dieselben Defizite wie Jira und abapGit auf. Als Schlussfolgerung der Lösungsevaluationen, um alle Vorteile gegenüber der aktuellen Release Methode in der cortlity gmbh auszuschöpfen empfiehlt sich eine Kombination aller Lösungen. Jira, SAP Solution Manager,abapGit und Confluence zu einer automatisch interagierenden Landschaft kombiniert werden, welche die Abläufe der Entwicklung aus abapGit und die Abläufe des Anforderungsmanagements aus Jira automatisch in eine angelegte Release Historie in Confluence dokumentiert. Weiterhin bietet Confluence nun die Möglichkeit mit ausführlichen Beschreibungen zu ergänzen.

**Literaturverzeichnis**

**Anhangsverzeichnis**

Anhang 1:

**Anhang**

**Anhang 1:**

**Anhang 2: Beigabenverzeichnis**

1. Verzeichnis “Projektarbeit 2“ .............................................................................

2. Verzeichnis “Quelltexte“ ....................................................................................

3. Verzeichnis “Elektronische Quellen“ .................................................................