

Wirtschaftsinformatik Master an der Hochschule der Medien

## **Technologische Grundlagen Cloudbasierter Anwendungen**

### **Thema 5 „User Experience“**

#### **Projektdokumentation**

Wintersemester 2016/2017

#### **Projektteam:**

Florian Blessing

Lisa Böhler

Markus Götz

Jan Habersetzer

Wendelin Herrmann

## Inhalt

1	Darstellung des Themas .....	5
1.1	Projektorganisation .....	6
1.2	Entwicklungsumgebung .....	8
2	Theoretische Grundlagen .....	9
2.1	Web-Anwendungen.....	9
2.2	User Experience.....	9
2.3	Google Hangouts .....	9
3	Projektschritt 1 .....	11
3.1	Grundlegende Ansätze .....	11
3.2	Anforderungsanalyse.....	14
3.2.1	Konstruktion Anforderungskatalog .....	15
3.2.2	Anforderungskatalog.....	15
3.3	Visualisierungskonzepte .....	16
3.3.1	Visualisierungskonzept I - Konventionell.....	16
3.3.2	Visualisierungskonzept II - MindMap .....	17
3.3.3	Visualisierungskonzept III - Pyramide.....	17
3.3.4	Visualisierungskonzept IV - Litfaßsäule .....	18
3.4	Technologie Scouting .....	18
3.4.1	Auswahl der Programmiermethodik .....	19
3.4.2	Frameworkauswahl .....	20
3.5	Fazit nach Projektschritt 1 .....	22
4	Projektschritt 2 .....	24
4.1	Änderungen in Bezug auf Projektschritt 1.....	24
4.1.1	Review der Ergebnisse des ersten Projektschrittes .....	24

4.1.2	Abschaltung der Hangouts-API.....	24
4.1.3	Gestaltung Landingpage.....	25
4.1.4	Einführung von User Stories.....	27
4.2	Fallstudie 1 - Explorer.....	35
4.2.1	Entwurf.....	35
4.2.2	Systemspezifikation.....	37
4.2.3	Implementierung.....	37
4.2.4	Test.....	39
4.2.5	Fazit der Fallstudie.....	40
4.3	Fallstudie 2 "Bubbles".....	40
4.3.1	Entwurf.....	40
4.3.2	Systemspezifikation.....	42
4.3.3	Implementierung.....	42
4.3.4	Test.....	44
4.3.5	Fazit der Fallstudie.....	44
4	Projektfazit und Ausblick.....	44
5	Anhang.....	46

## Abbildungsverzeichnis

Abbildung 1: 3-Schichten-Architektur des Gesamtsystems	6
Abbildung 2: Meilensteinplan	7
Abbildung 3: Google Hangouts	10
Abbildung 4: Ansätze zur Umsetzung	11
Abbildung 5: Hangouts-Applikation	12
Abbildung 6: Hangouts-Erweiterung	13
Abbildung 7: Hangouts "Problem-Button"	14
Abbildung 8: Visualisierungskonzept I - Konventionell	16
Abbildung 9: Visualisierungskonzept II - MindMap	17
Abbildung 10: Visualisierungskonzept III - Pyramide	17
Abbildung 11: Visualisierungskonzept IV - Litfaßsäule	18
Abbildung 12: Vor- und Nachteile der Programmiermethodiken	19
Abbildung 13: Vor- und Nachteile der Visualisierungs-Frameworks	21
Abbildung 14: Vor- und Nachteile der Code-Frameworks	22
Abbildung 15: Systemübersicht Hangouts App	26
Abbildung 16: Systemübersicht GUI Website	27
Abbildung 17: UML Use Case Diagramm	29
Abbildung 18: Erster Entwurf - Explorer	36
Abbildung 19: UML-Diagramm für den Explorer-Prototyp	39
Abbildung 20: Erster Entwurf - Bubbles	41
Abbildung 21: UML-Diagramm für den Bubble Prototyp	43

# 1 Darstellung des Themas

In global vernetzten Unternehmen stehen Mitarbeiter immer öfter vor der Herausforderung mit Kollegen an anderen Standorten oder externen Projektpartnern zusammenzuarbeiten, um den Projekterfolg zu garantieren. Absprachen und Meetings werden nicht mehr nur vor Ort gehalten, sondern finden mittlerweile auch vermehrt online mittels verschiedener digitaler Medien statt.

Im Rahmen der Vorlesung „Technologische Grundlagen cloudbasierter Internet-Anwendungen“ soll nun eine Cloudanwendung in Verbindung mit Google Hangouts und Google Drive entwickelt werden. Diese Anwendung ermöglicht es Videokonferenzen mittels Speech Tokenization zu analysieren und relevante Dokumente automatisch zu ermitteln und anzuzeigen.

Das zu entwickelnde Zielsystem wurde unterteilt in fünf einzeln zu betrachtende Challenges, welche in kleineren Projektteams bearbeitet werden. Für die Anwendung sollen bereits am Markt verfügbare Module und vorgefertigte Softwarepakete genutzt werden. Ziel ist es dabei, dass die Studierenden sich mit den Herausforderungen des Cloud-Computing und der Entwicklung einer Software in Teamarbeit auseinandersetzen. Die Studenten sollen sich dabei an einem professionellen Software-Engineering-Prozess orientieren und während der Projektarbeit einen wasserfallartigen oder agilen Ansatz der Softwareentwicklung verfolgen. Die Studierenden sollen dabei Wissen zu cloudbasierten Anwendungen eigenständig erwerben und im Bachelorstudium erworbenes fachliches Wissen vertiefen und anwenden.

Das Gesamtsystem soll innerhalb einer Drei-Schichten-Architektur aus Präsentationsschicht, Applikationsschicht und Datenbankschicht implementiert werden. Eine Übersicht über den genauen Systemaufbau kann Abbildung 1 entnommen werden. Die konkrete Aufgabenstellung dieses Projektteams ist es dabei die fünfte Challenge „User Experience“ umzusetzen. Diese sieht vor, dass der Recommender Client innerhalb der Präsentationsschicht entwickelt wird, welcher eine adäquate Visualisierung und Interaktion mit den in der Applikationsschicht erarbeiteten Recommendations ermöglicht. Das Team ist somit mit der Entwicklung des Graphical User Interface (kurz „GUI“) beauftragt. Weitere technische und funktionale Anforderungen an das zu entwickelnde System werden im Anforderungskatalog erläutert (siehe Kapitel 3.2.2).

Der Ablauf dieser Projektarbeit lässt sich in zwei Phasen aufteilen. Die erste Phase betrifft das Technologie-Scouting, in der zweiten Phase geht es um die konkrete Entwicklung von Prototypen bzw. Fallstudien. Für das Technologie-Scouting werden zunächst alle funktionalen und technischen Anforderungen an die Software ermittelt. Die Ergebnisse werden in einem Pflichtenheft, bzw. Anforderungskatalog festgehalten. Anschließend wird auf dieser Basis auf dem Markt vorhandene Technologien und Frameworks gesucht und mit den Anforderungen abgeglichen. Die Ergebnisse werden in Form einer Präsentation vorgestellt. Anschließend folgt die Entwicklung der Prototypen, die zum Ende des Semesters vorgestellt werden. Die Zusammenführung der fünf einzelnen

Challenges zu einem Gesamtsystem erfolgt im anschließenden Semester.

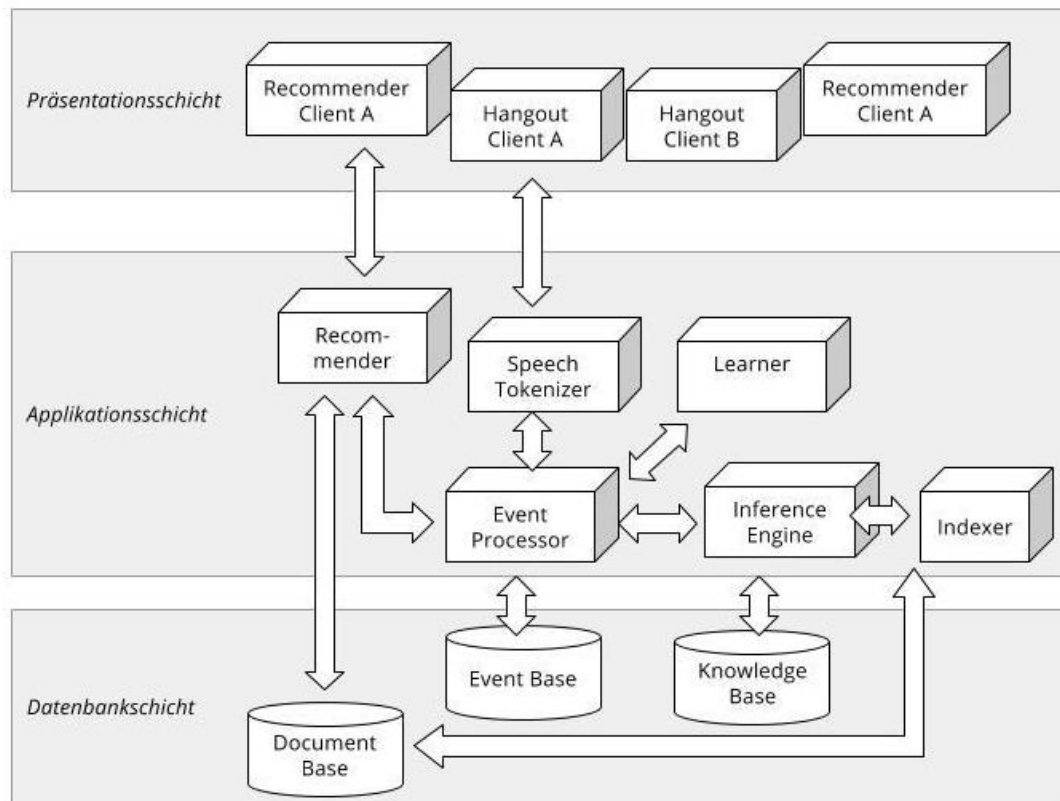


Abbildung 1: 3-Schichten-Architektur des Gesamtsystems

## 1.1 Projektorganisation

Das Projektteam setzt sich aus fünf Studenten im Masterstudiengang Wirtschaftsinformatik der Hochschule der Medien Stuttgart zusammen. Ansprechpartner und Leiter der Veranstaltung ist Prof. Dr. Christian Rathke. Während in Projektphase eins das Team aus Florian Blessing, Lisa Böhler, Andrea Breimayer, Markus Götz und Wendelin Herrmann bestand, gab es zur Projektphase zwei einen Wechsel. Andrea Breimayer verließ das Team, an ihrer Stelle kam Jan Habersetzer neu hinzu.

Ein Projektleiter wurde aufgrund der geringen Teamgröße nicht festgelegt, stattdessen wurden anstehende Aufgaben nach den individuellen und auf Vorkenntnissen basierenden Kompetenzen der Teammitglieder gemeinsam besprochen und verteilt. Einen Überblick über die angefallenen Todos sowie die Verantwortlichen und den zeitlichen Rahmen, findet man in der Excel-Tabelle "ToDo-Liste", die unter dem Ordner 1.1 Projektorganisation auf der Google Site abgelegt ist.

Da das Ziel zwar klar, das finale Produkt anfangs aber noch nicht definiert war, wurde im Team nach einem agilen Ansatz, angelehnt an die Scrum-typischen Sprints, vorgegangen. In regelmäßigen und mindestens ein- bis zweimal wöchentlich stattfindenden Meetings berichten die Teammitglieder über ihre Fortschritte und aufgetretene Probleme. Bei akuten Problemstellungen oder zusätzlich benötigtem Informationsaustausch wurde, auch vor dem nächsten festgelegten Meeting, über Online-Kanäle kommuniziert. So schärfte sich das Vorgehen und das Bild des finalen Produktes mit den neuen Kenntnissen und Informationen. Darüber hinaus nahm das Team die angebotenen Sprechstunden bei Prof. Dr. Rathke wahr, um die Ideen aus dem Team mit den Vorstellungen des Auftraggebers abzugleichen. Rahmen für dieses Vorgehen waren zwei vorgegebene Projektschritte sowie einige größere Meilensteine, die in Abbildung 2 dargestellt werden. Dabei umfassen Meilenstein 1 bis 3 den ersten Projektschritt und die Meilensteine 4 und 5 den zweiten.



Abbildung 2: Meilensteinplan

## **1.2 Entwicklungsumgebung**

Nachfolgend sollen noch kurz die grundlegenden Rahmenbedingungen des Dozenten zur Entwicklungsumgebung dargelegt werden. Vorgegeben ist die Nutzung von Java APIs – sofern diese vorhanden sind. Zur Erstellung von Source Code ist die Entwicklungsumgebung Eclipse (mit zugehörigem Papyrus-Plug-in für die UML Diagramme) zu verwenden. Der Source-Code ist den Konventionen der jeweiligen Programmiersprache entsprechend zu erstellen. Dabei ist auf eine saubere Dokumentation aller APIs und Frameworks zu achten. Zudem soll GIT als Werkzeug zur Versionsverwaltung in Form eines Plug-ins in Eclipse integriert werden. Für die Entwicklung wird Eclipse Juno in der neuesten Version verwendet.



## 2 Theoretische Grundlagen

Um ein Verständnis für das vorliegende Problem zu erhalten und ein gemeinsames Verständnis für den theoretischen Hintergrund zu schaffen, werden im nachfolgenden Kapitel einige Grundlagen erarbeitet. Hierfür wird zum einen die Charakteristik einer Web-Anwendung herausgearbeitet, der Dienst Google Hangouts vorgestellt und der Begriff “User Experience” näher definiert.

### 2.1 Web-Anwendungen

Web-Anwendungen, oder kurz Web Apps zeichnen sich vor allem durch die Erreichbarkeit einer bestimmten Funktion über eine Webseite aus. Web Anwendungen bestehen serverseitig häufig aus Scriptsprachen wie PHP oder ASP und clientseitig aus HTML, JavaScript und Adobe Flash. Ein wichtiges Merkmal von Web Apps ist, dass diese über einen Webbrowser erreicht werden können, ohne dass eine Installation einer App von Nöten ist.<sup>1</sup> Typische Beispiele für Web Apps sind etwa Onlinebanking-Systeme oder Webshops.

### 2.2 User Experience

User Experience beschreibt das Nutzer- bzw. Nutzungserleben eines Produktes. In der aktuellen ISO 9241-210 wird User Experience wie folgt beschrieben: “all aspects of the user’s experience when interacting with the product, service, environment or facility. [...] It includes all aspects of usability and desirability of a product, system or service from the user’s perspective”.<sup>2</sup>

In der Forschung gibt es eine Reihe unterschiedlicher Ansätze, User Experience zu definieren. Dabei stößt man immer wieder auf drei Charakteristiken: Ganzheitlich, Subjektiv und Positiv. Ganzheitlich meint dabei, die “Balance zwischen instrumentellen und nicht-instrumentellen Qualitäten, wie Schönheit, Neuartigkeit, Herausforderung oder Selbstausdruck”. Die Subjektivität beschreibt die wahrgenommene und nicht die tatsächliche Qualität des Produktes. Diese Wahrnehmung bestimmt zukünftig die Nutzung und Kommunikation über das Produkt. Die Charakteristik des Positiven fokussiert sich auf Freude, Spaß, Attraktivität, Herausforderungen und Schönheit.<sup>3</sup>

### 2.3 Google Hangouts

Google Hangouts ist ein Dienst des Entwicklers Google. Er kann zur klassischen Videotelefonie verwendet werden und hierbei durch eingebaute Applikationen um Funktionen erweitert werden.

---

<sup>1</sup> <http://searchsoftwarequality.techtarget.com/definition/Web-application-Web-app>

<sup>2</sup> ISO 9241-210

<sup>3</sup> Vgl. Hassenzahl, Burmester und Koller (2008) - “Der User Experience auf der Spur”

Hangouts kann sowohl im Browser, über eine eigene Applikation als auch auf Mobilgeräte gestartet werden<sup>4</sup>. In Abbildung 3 ist der typische Layout eines Hangout-Gesprächs abgebildet. Während am unteren Rand eine Übersicht aller Gesprächsteilnehmer platziert ist, wird der Großteil des Bildes automatisch mit dem Video des aktuell Sprechenden gefüllt.

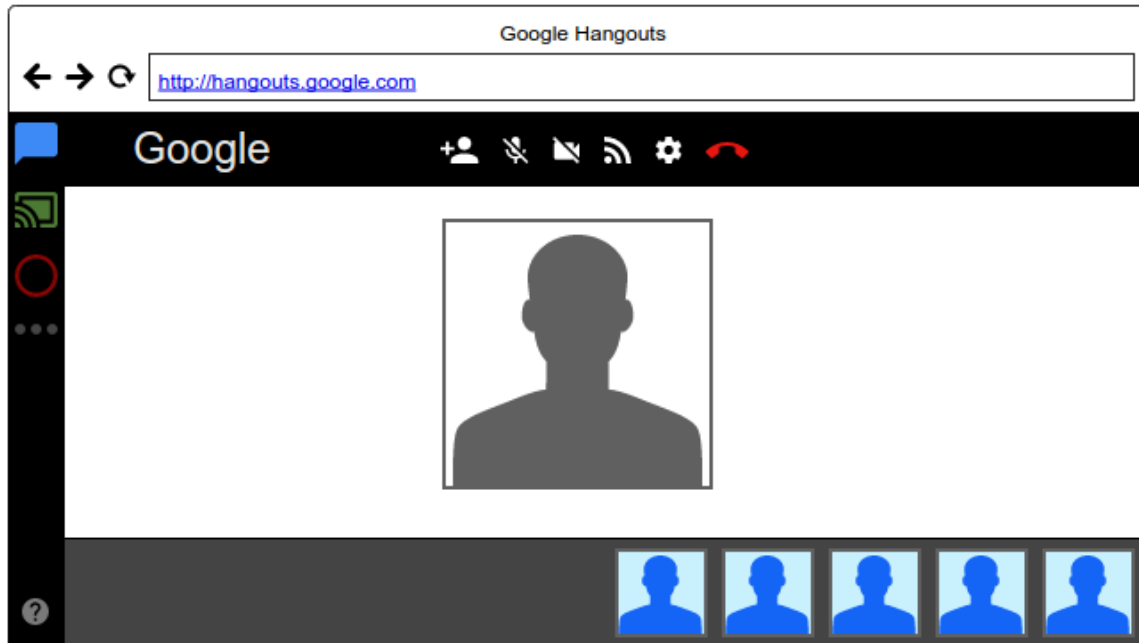


Abbildung 3: Google Hangouts

---

<sup>4</sup> [http://praxistipps.chip.de/google-hangouts-was-ist-das-verstaendlich-erklaert\\_41469](http://praxistipps.chip.de/google-hangouts-was-ist-das-verstaendlich-erklaert_41469)

### 3 Projektschritt 1

Wie bereits beschrieben dient der erste Projektschritt hauptsächlich der Einarbeitung in das Thema sowie dem Technologie-Scouting. Der erste Projektschritt endet mit der Zwischenpräsentation vor dem Dozenten sowie den anderen Teams am 21.11.2016.

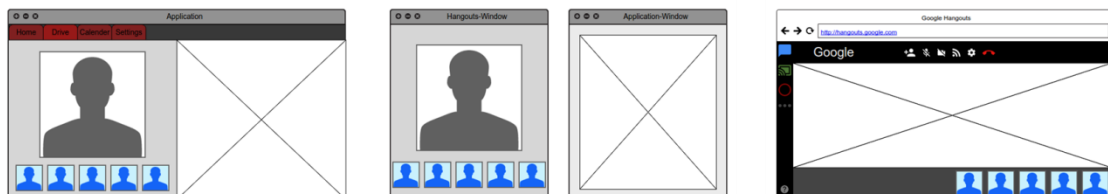
#### 3.1 Grundlegende Ansätze

Nach gründlicher Recherche ist das Projektteam zu dem Schluss gekommen, dass es keinen Anbieter auf dem Markt gibt, welcher eine zur Fragestellung der Projektarbeit passende Lösung anbietet. Daher hat sich das Projektteam in Absprache mit dem Dozenten darauf verständigt, eine Eigenentwicklung der Software vorzunehmen. Somit gilt es nicht länger Anbieter und deren Lösungen zu suchen und zu vergleichen, sondern verschiedene Herangehensweisen und Frameworks für die Eigenentwicklung zu finden und kritisch zu evaluieren.

Nach einer ersten Recherche standen dem Team drei grundsätzliche Ansätze der Umsetzung zur Auswahl:

- Hangouts in eine eigenentwickelte Anwendung/Homepage einbetten
- Aufspaltung von Videotelefonie und Web-Anwendung in zwei separate Browserfenster
- Entwicklung einer Hangouts-Applikation

Die nachfolgende Grafik stellt die drei Möglichkeiten gegenüber und enthält eine kurze Bewertung:



++			Von Google unterstützt
+	Kombination Video & DMS		Kombination Video & DMS
0		Keine Konflikte mit Google	
-		Trennung Video & DMS	
--	Von Google nicht unterstützt		

Abbildung 4: Ansätze zur Umsetzung

So hat eine Recherche und Nachfrage bei Google ergeben, dass eine Einbindung von Google Hangouts in eine eigenentwickelte App oder Software nicht möglich ist, da die benötigten

Schnittstellen von Google nicht zur Verfügung gestellt werden. Google empfiehlt hier die Entwicklung einer Hangouts Applikation. Somit wurde Ansatz 1 auf Grund der fehlenden Unterstützung Googles, die auch am seit mehr als fünf Jahren bestehenden Feature Request ausgemacht werden kann<sup>5</sup>, nicht weiterverfolgt.<sup>6 7</sup>

Eine Trennung von Hangouts und Sprachanalyse inklusive Dokumentenvorschläge ist insofern nicht ergonomisch, als dass zwei Browserfenster verwendet werden müssen, welche sich unter Umständen überlappen können und somit dem User ein einfaches Arbeiten erschweren würden. Dies wird gerade bei kleinen Bildschirmen ohne Möglichkeit der Erweiterung auf einen zweiten Bildschirm zum Problem. Da dieses Szenario im heutigen sehr mobilen Geschäftsleben des öfteren auftreten kann, hat sich das Team dagegen entschieden Ansatz 2 weiter zu verfolgen.

Bei der Evaluation des dritten Ansatzes zeigt sich, dass Google die Möglichkeit bietet eigenentwickelte Apps in Hangouts zu integrieren. Hierbei können zwei Arten von Hangouts-Applikationen unterschieden werden. So kann die Integration sowohl in Form einer klassischen App, die in der Mitte des Hangouts-Fensters dargestellt wird (siehe Abbildung 5), als auch in Form einer sogenannten Erweiterung (siehe Abbildung 6), die am rechten Seitenrand des Fensters angezeigt wird, erfolgen. Der größte Unterschied der beiden Applikationstypen ist die Größe der zur Verfügung stehenden Fläche.

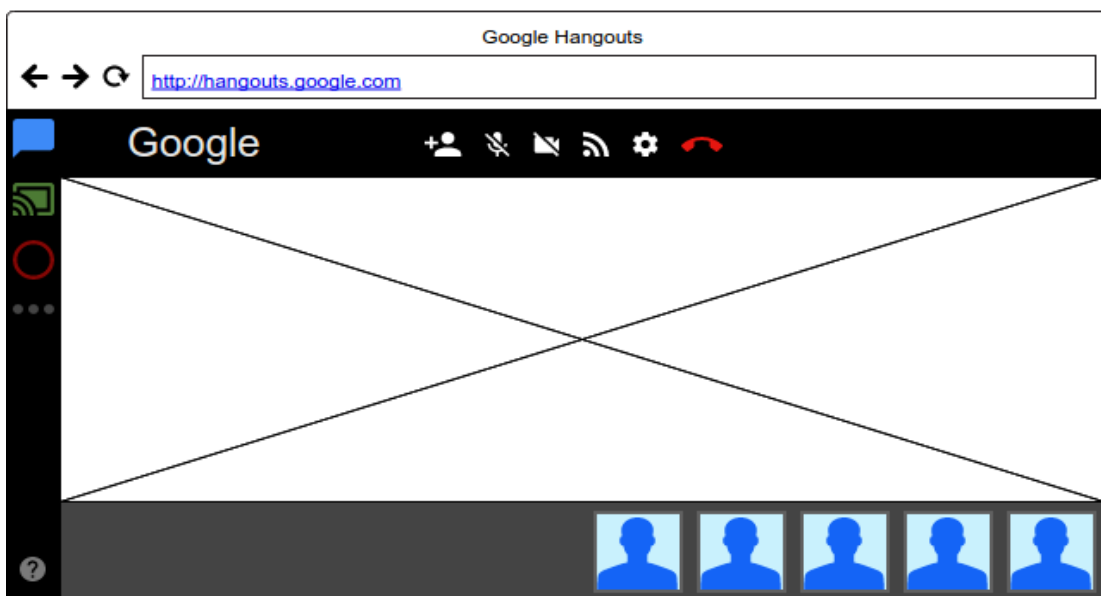


Abbildung 5: Hangouts-Applikation

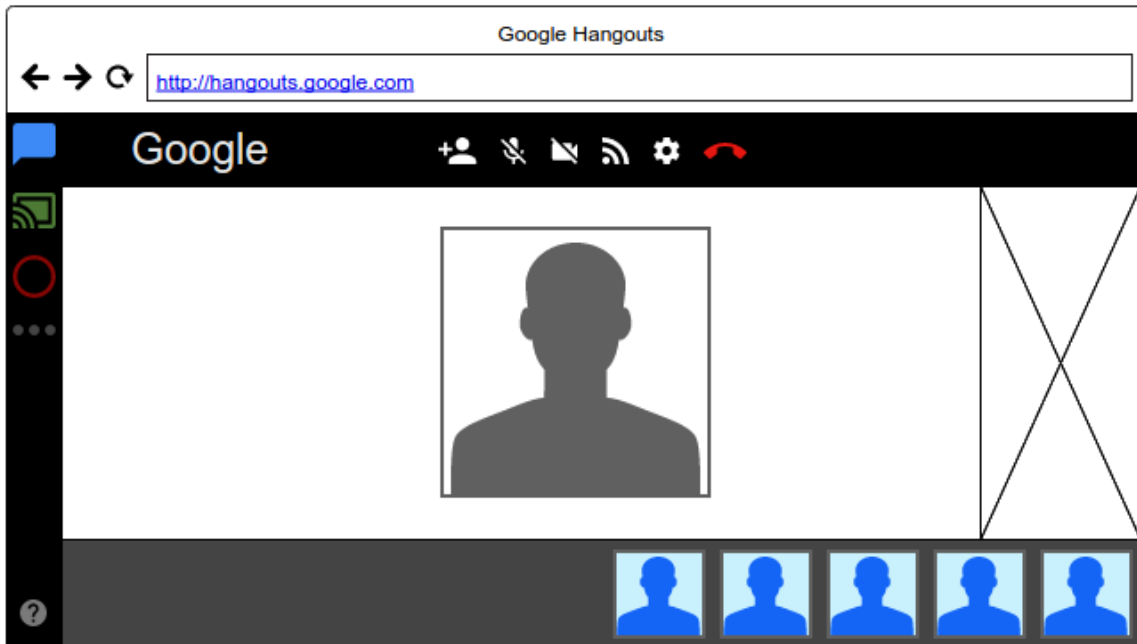
<sup>5</sup> <https://code.google.com/p/google-plus-platform/issues/detail?id=28>

<sup>6</sup> <https://groups.google.com/forum/#!topic/google-plus-developers/nZ7vC3zJOx8>

<sup>7</sup>

<http://www.allesuebergoogleplus.de/allgemein/neue-moeglichkeiten-der-hangout-nutzung-hangout-api-fuer-entwickler/>

Während die klassische App die Großansicht des Gesprächspartners ersetzt und somit den größten Teil des Fensters zur Verfügung hat, wird die Erweiterung am rechten Fensterrand eingeschoben und verfügt somit über eine stark limitierte Fläche im Vergleich zur Alternative, zeigt allerdings nach wie vor die Großansicht des sprechenden Gesprächspartners.



*Abbildung 6: Hangouts-Erweiterung*

Das Team präferiert in diesem Fall die Umsetzung als klassische Hangouts-Applikation, da für das zu betrachtende Kollaborationssystem der Darstellung der erarbeiteten Vorschläge Vorzug gegenüber der Großansicht des Gesprächspartners einzuräumen ist. Zudem sind die Gesprächspartner in beiden Darstellungsformen zu jedem Zeitpunkt am unteren Bildschirmrand sichtbar. Sollte dies nicht ausreichen, kann die Applikation temporär geschlossen werden, wodurch die Großansicht des Gesprächspartners gegeben wäre. Die Applikation kann jederzeit während einem Videotelefonat geöffnet und geschlossen werden, ohne dass dies einen Einfluss auf das Hangouts-Gespräch hat.

Durch den beschriebenen dritten Ansatz kann die Darstellung von Videotelefonat und Applikationslogik in einem Fenster unter Einhaltung der Richtlinien von Google erreicht werden. Da dies exklusiv bei diesem Ansatz der Fall ist, entscheidet sich das Team für die Umsetzung der GUI als Hangouts-Applikation in klassischer Form.

Bei der weiteren Evaluation zur Verwendung einer Hangouts-Applikation ist das Team auf ein Problem gestoßen. So wurde die in Abbildung 7 markierte Applikationsleiste zwar bei jedem Gesprächsteilnehmer angezeigt, allerdings fehlte bei mehreren Mitgliedern das dargestellte Symbol mit den drei Punkten, das zur Einbindung einer Applikation notwendig ist. Diese Komplikation konnte auch nicht durch den Neustart des Telefonats, den Wechsel des Browsers oder die Verwendung der expliziten Hangouts-Erweiterung für Google Chrome behoben werden. Die Ursache des Problems

konnte durch das Team nicht ausgemacht und isoliert werden.

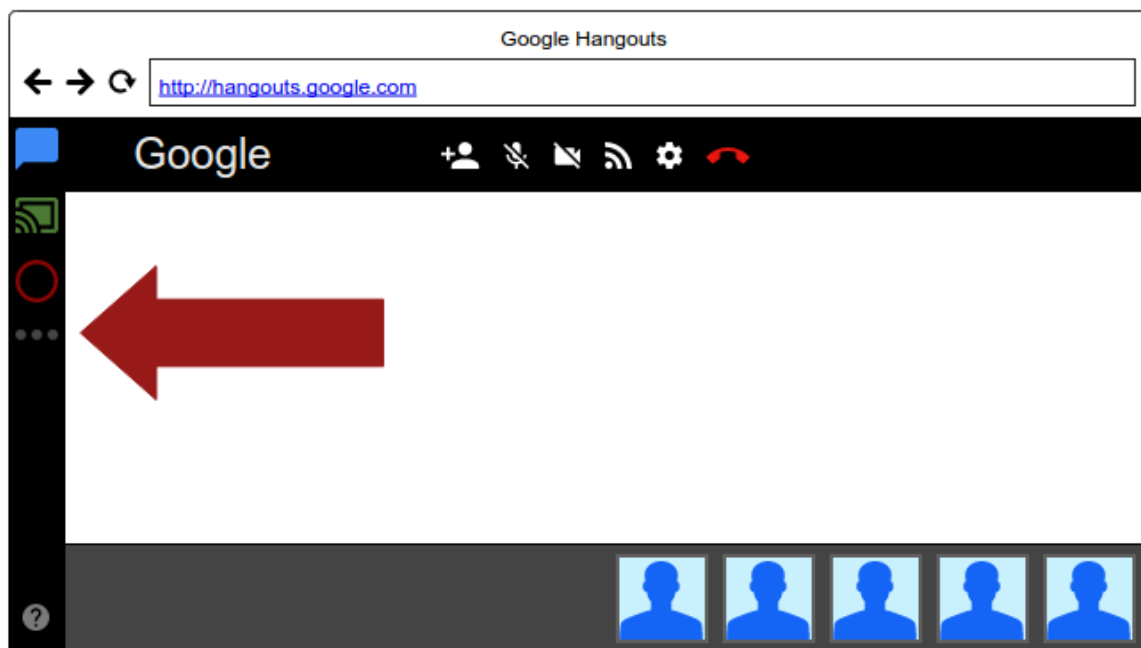


Abbildung 7: Hangouts "Problem-Button"

Es gelang allerdings durch einen einfachen Workaround die Komplikationen zu beheben und die Funktionalität der Applikation zu gewährleisten. So konnte ein Link erstellt werden, der Google Hangouts automatisch mit der gewünschten Applikation öffnet und somit neben der Behebung des Problems weitere Vorteile mit sich bringt. Durch die Einbindung dieses Links im Look & Feel von Google Hangouts in eine Landingpage, kann das Frontend des Systems beliebigen Corporate Identities angepasst werden. Zudem wird dem Nutzer die Suche und Installation der Applikation abgenommen und hierdurch das Risiko der Installation einer falschen Applikation vermieden. Insgesamt kann also mit Hilfe dieser Erweiterung des Ansatzes einer Hangout-Applikation durch eine vorgeschaltete Landingpage der gewünschte Prozess gewährleistet, optimiert und userfreundlich angeboten werden.

Es kann als Zwischenfazit festgehalten werden, dass das GUI im Rahmen einer Hangouts-Applikation umgesetzt werden soll.

### 3.2 Anforderungsanalyse

Zu Beginn der darauf aufbauenden Anforderungsanalyse definierte das Team die einzelnen Schritte und Aktionen des Tools während eines Videotelefonats. Hierfür wurden als Zwischenprodukte eine EPK sowie ein Wireframe mit integriertem Prozess erarbeitet, die als grafische Grundlagen für die Erfassung der Anforderungen dienen sollten. Beide Diagramme sind auf der Projektsite unter "3.5 Workflow User" einsehbar. Im Anschluss daran wurden mit Hilfe der Prozessvisualisierungen die grundlegenden Anforderungen an das Teilsystem erfasst und festgehalten.

### 3.2.1 Konstruktion Anforderungskatalog

Um die erfassten Anforderungen geeignet dokumentieren zu können wurde in Google Tabellen die Schablone eines Anforderungskataloges eingerichtet. Google Tabellen wurde gewählt, da es lizenzfrei erhältlich ist, die gleichzeitige Bearbeitung der Anforderungen durch mehrere Autoren ermöglicht und durch die gegebenen Filtermechaniken eine einfache Verwaltung der Anforderungen anhand der Attribute bietet. Bei der Wahl der zu verwendeten Attribute hat sich das Team dazu entschieden, sich auf das Mindeste zu beschränken, um zusätzlichen Pflegeaufwand und das Risiko von Redundanzen und veralteten Daten zu minimieren. Zudem wurden Regeln erarbeitet, die eine einheitliche Dokumentation der Anforderungen fördern und somit die Validität des Anforderungskataloges gewährleisten sollen. Außerdem wurde ein Review-Prozess eingerichtet, in dem zwei Studierende, die aus ihrem Bachelorstudium bereits Erfahrung mit der Anforderungsanalyse haben, nach dem Vier-Augen-Prinzip die neuen Anforderungen überprüfen und anschließend freigeben. Eine Erklärung der Attribute, die Liste der Regeln sowie eine grafische Darstellung des Review-Prozesses können auf der Google-Projektsite unter "4.1 Funktionale Anforderungen" im Leitfaden eingesehen werden.

### 3.2.2 Anforderungskatalog

Der gesamte Anforderungskatalog wird aus Gründen der Übersichtlichkeit separat auf der Google-Projektsite als Excel-Datei zur Verfügung gestellt. Der Katalog dient als strukturierte Sammlung sämtlicher im Zuge der Projektarbeit erfassten Anforderungen.

Nach aktuellem Stand werden folgende Anforderungen aufgrund des Projektauftrages oder sonstigen Rahmenbedingungen und Zusammenhängen als "kritisch" betrachtet:

*Tabelle 1: Kritische Anforderungen*

Eclipse	Das System muss in Eclipse entwickelt werden können
Java	Das System muss auf Java bzw. Java Script, CSS und HTML basieren
Grafische Darstellung	Das System muss eine grafische Darstellung anbieten
Dateiauswahl	Das System muss dem Benutzer erlauben konkrete Dateien auszuwählen
Dokumentenübersicht	Das System muss eine Gesamtübersicht der Dokumente anzeigen können
Datenweiterleitung	Das System muss die Nutzerdaten beim Wechsel in Drive weiterleiten
Schließen	Das System muss vom Benutzer geschlossen werden können

Google Docs	Das System muss einzelne Dateien in Google Docs öffnen können
Google Tabellen	Das System muss einzelne Dateien in Google Tabellen öffnen können
Google Kalender	Das System muss einzelne Daten in Google Kalender öffnen können

Diese kritischen Faktoren waren Grundlage für den weiteren Projektverlauf, da diese auf jeden Fall erreicht werden müssen.

### 3.3 Visualisierungskonzepte

Obwohl eine nähere Betrachtung und Evaluierung verschiedener Visualisierungskonzepte erst im zweiten Abschnitt des Semesters stattfinden soll, hat sich das Team bereits im ersten Schritt Gedanken über mögliche Visualisierungen gemacht und diese entworfen. Grund hierfür ist, dass diese essentiell für das durchzuführende Technologie-Scouting sind. Es muss bekannt sein, welche Anforderungen über die Visualisierungsformen (z.B. die Notwendigkeit von Animationen oder ähnlichem) entstehen, damit diese bei der Auswahl eines Frameworks beachtet werden können. Hierdurch soll ein guter Übergang vom ersten in den zweiten Schritt gewährleistet und eventuelle Kompatibilitätsprobleme vermieden werden.

Das Team hat im Ausblick auf die zweite Phase vier Visualisierungskonzepte erarbeitet, die als Grundlage für das Technologie-Scouting dienen sollen. Im Folgenden werden diese Konzepte benannt und kurz beschrieben. Auf eine konkrete Ausarbeitung und Wertung wird an dieser Stelle verzichtet, da diese für den weiteren Fortschritt in der aktuellen Projektphase keine Rolle spielen.

Bei der Betrachtung der folgenden Entwürfe ist grundlegend festzuhalten, dass der schwarz eingezeichnete Rahmen jeweils das Sichtfeld des Users symbolisiert. Der Nutzer sieht somit in keiner Darstellungsform die Gesamtmenge aller potentieller Vorschläge, da dies eine mit der Datenmenge skalierende Unübersichtlichkeit mit sich bringen würde. Es ist allerdings angedacht, dass in jeder Visualisierung über eine entsprechende Schaltfläche sämtliche Dateien angezeigt werden können, um dem User eine proaktive Auswahl zu ermöglichen. Dies ist beispielsweise denkbar, wenn der Nutzer auf ein im bisherigen Gespräch noch nicht genanntes Dokument ansprechen möchte, sich zuvor jedoch nochmal über dessen Inhalt versichern möchte.

#### 3.3.1 Visualisierungskonzept I - Konventionell

Die konventionelle Visualisierung ist an die Dokumentendarstellung in Google Drive angelehnt. Einzelne Dokumente werden als Kästchen dargestellt, der Dokumentenname und Typ stehen darunter. Erlangt eine Datei im Gespräch Relevanz wird diese dynamisch in die Anzeige nachgeladen. Wird eine Datei nicht mehr als relevant betrachtet, wird sie aus der Anzeige entfernt.



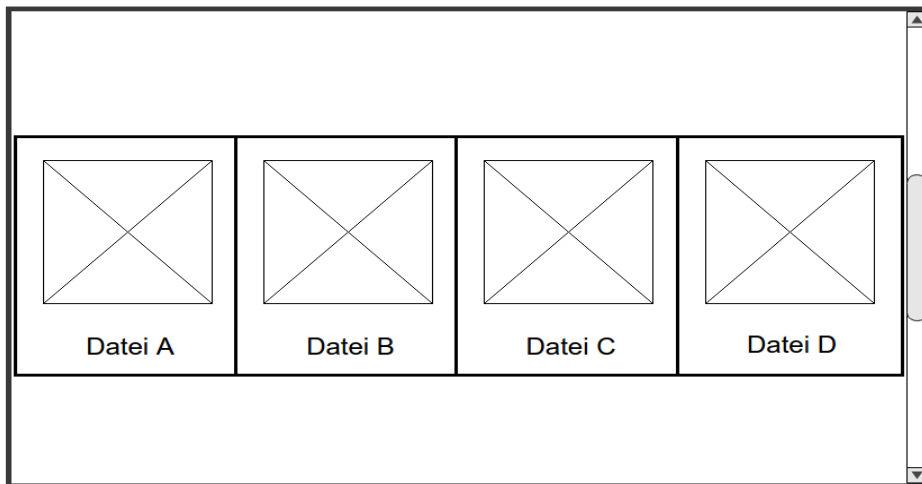


Abbildung 8: Visualisierungskonzept I - Konventionell

### 3.3.2 Visualisierungskonzept II - MindMap

Das zweite Visualisierungskonzept arbeitet wie eine MindMap mit einer Baumstruktur, wobei die Dateiordner die Knotenpunkte darstellen. So wird zunächst der Projektordner mit seinen Unterordnern dargestellt, bis das Gespräch sich einem speziellen Bereich zuwendet. Erkennt das System, dass beispielsweise über Dokumente aus einem etwaigen Ordner "Protokolle" gesprochen wird, verändert sich der Fokus und die nächste Ebene des Protokoll-Ordners werden angezeigt, wohingegen die restlichen Ordner der ursprünglichen Ebene aus dem Sichtfeld verschwinden.

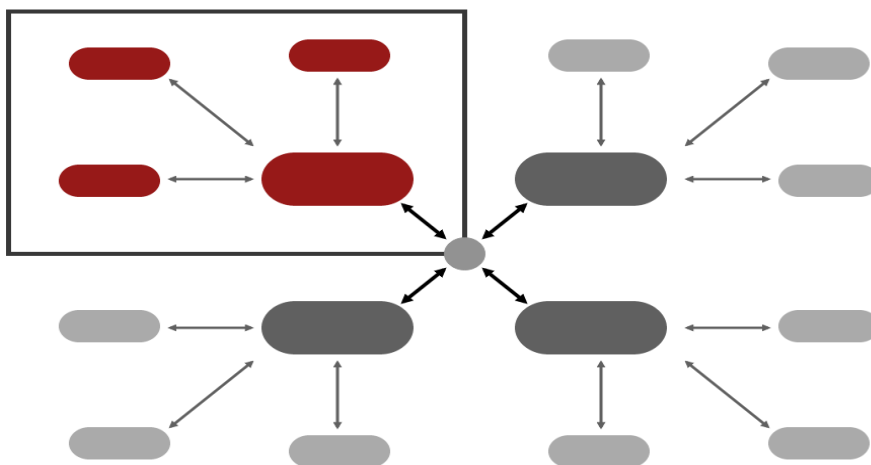


Abbildung 9: Visualisierungskonzept II - MindMap

### 3.3.3 Visualisierungskonzept III - Pyramide

Die Pyramide ist ein hierarchisches System mit visuellem Fokus auf die Dokumentenrelevanz. Je relevanter ein Dokument auf Basis des Gesprächs erscheint, desto höher rückt es an die Spitze der Pyramide. Im Gegenzug steigen Dokumente mit geringerer Relevanz wieder ab oder verschwinden

zuletzt ganz in den “Dokumentenpool”, welcher für den Nutzer nicht sichtbar ist.

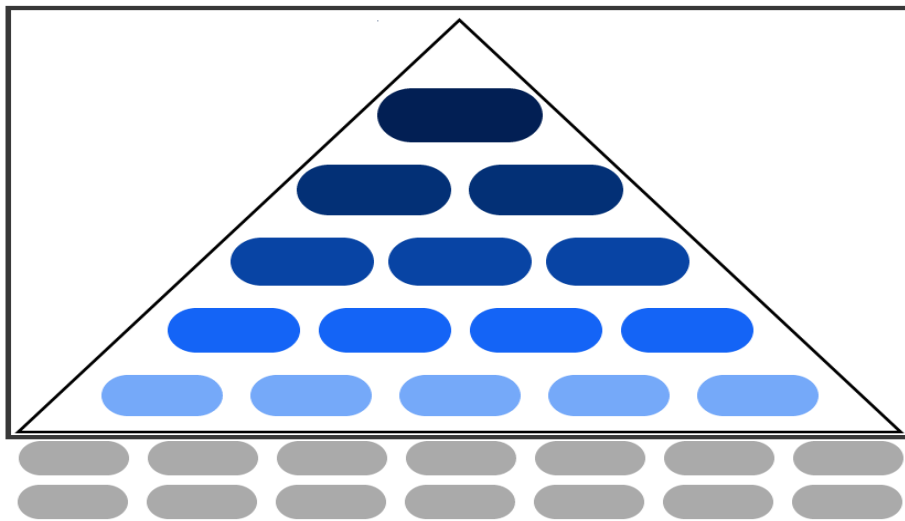


Abbildung 10: Visualisierungskonzept III - Pyramide

### 3.3.4 Visualisierungskonzept IV - Litfaßsäule

Das vierte Visualisierungskonzept ist grafisch an einen sich drehenden Zylinder angelehnt. Dabei wird ein Ordner - vergleichbar mit einem Plakat an einer Litfaßsäule - auf der Außenseite des Zylinders aufgelistet. Ändert sich der Fokus des Gesprächs, dreht sich der Zylinder zum entsprechenden Ordner bzw. Plakat, welches von nun an im Sichtfeld des Nutzers angezeigt wird.

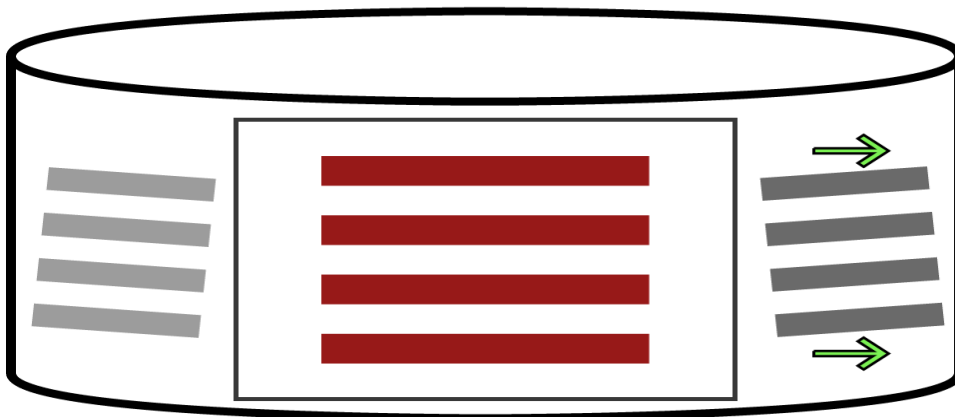


Abbildung 11: Visualisierungskonzept IV - Litfaßsäule

## 3.4 Technologie Scouting

Unter Berücksichtigung der bis hierher ermittelten Rahmenbedingungen, also der Umsetzung als Hangouts-Applikation sowie den angedachten Visualisierungsformen, begann das Team anschließend mit dem eigentlichen Technologie-Scouting.

### 3.4.1 Auswahl der Programmiermethodik

Vor dem Hintergrund, dass eine Google Hangouts Applikation auf JavaScript<sup>8</sup>, HTML und CSS basiert, wurde bei der Auswahl einer Programmiermethodik insbesondere auf die Unterstützung von JavaScript geachtet. Vom Projektteam wurden drei Möglichkeiten für die programmiertechnische Umsetzung ermittelt, die JavaScript erzeugen könnten.

1. Verwendung des Google Web Toolkits (GWT)<sup>9</sup> zur Programmierung in Java und anschließenden Kompilierung in JavaScript sowie Verwendung des enthaltenen GUI-Frameworks
2. GWT als Java zu JavaScript Compiler, mit Verwendung eines zusätzlichen Visualisierungs-Frameworks
3. Natives JavaScript unter Verwendung eines Code- sowie eines Visualisierungs-Frameworks



++	Sehr umfangreich und vielseitig	Viele Möglichkeiten	Viele Möglichkeiten Einfache Struktur
+	Erfahrung bereits vorhanden	Erfahrung bereits vorhanden	Erfahrung bereits vorhanden
0			
-	Komplex kostenpflichtig	Komplex kostenpflichtig	Funktionales JavaScript Kontakt zur Applikationslogik
--		Aufwendig in der Umsetzung	

Abbildung 12: Vor- und Nachteile der Programmiermethodiken

Bei der reinen Nutzung von GWT würde die Anwendung vollständig in Java Code entwickelt werden. Das bedeutet, dass die clientseitige Kommunikation ebenfalls in Java entwickelt wird. Zusätzlich stellt das Web Toolkit ein eigenes Framework für die Entwicklung von Grafikoberflächen zur Verfügung. Vorteil dieser Möglichkeit ist es, dass auf die Funktionen auf Serverseite einfach zugegriffen werden kann. Die GUI Visualisierung ist jedoch komplexer. Ein weiterer Nachteil ist, dass GWT nicht kostenlos genutzt werden kann und somit Kosten entstehen würden.

Im Falle der Nutzung von GWT und einem zusätzlichen Visualisierungs-Frameworks würde der Source

<sup>8</sup> JavaScript ist eine Skriptsprache, die ursprünglich für dynamisches HTML in Webbrowsern entwickelt wurde. Sie wird verwendet, um Benutzerinteraktionen auszuwerten, Inhalte zu verändern, nachzuladen oder zu generieren und so die Möglichkeiten von HTML und CSS zu erweitern.

<sup>9</sup> Bei GWT (Google Web Toolkit) handelt es sich um einen Werkzeugsatz zur Entwicklung von Webanwendungen. GWT ist ein Java-nach-JavaScript-Compiler, der ermöglicht, dass nahezu die gesamte Entwicklung von Client und Server mit Java realisiert werden kann. Das GWT beinhaltet einen XML-Parser sowie eine Schnittstelle für Remote Procedure Calls. Zudem ist ein Widget-Paket zur Gestaltung der grafischen Oberfläche (GUI) enthalten.

Code ebenfalls in Java entwickelt werden. Man kann somit die Vorteile von GWT und dem gewählten Framework verbinden. Nachteil ist, dass dadurch aber auch die Programmierung aufwendiger wird und die Nutzung von GWT erneut mit Kosten verbunden ist.

Wird die Präsentationsschicht unter Verwendung eines Visualisierungs-Frameworks ausschließlich in JavaScript entwickelt, handelt es sich um keine objektorientierte Programmierung. Dadurch sinkt die Komplexität der Entwicklung, ohne dass funktionale Einschränkungen hinzunehmen wären.

Aufgrund der geringeren Komplexität, den niedrigeren Kosten sowie der Gegebenheit, dass im Team bereits JavaScript-Erfahrung vorhanden ist, wurde die Entscheidung getroffen, dass System ohne GWT zu entwickeln. Somit wird eine Entwicklung mit nativem JavaScript unter Zuhilfenahme von zwei Frameworks stattfinden.

Hervorzuheben ist hierbei, dass sich das Team darauf verständigt hat auf zwei verschiedene Arten von Frameworks zurückzugreifen. Zum einen soll ein Framework als Unterstützung bei der Visualisierung der Daten eingesetzt werden. Das vom Team als "Visualisierungs-Framework" betitelte Framework soll die für die Visualisierungskonzepte benötigten Animationen zur Verfügung stellen, damit bei diesen Animationen auf bewährte Methodik zurückgegriffen werden kann. Zum anderen soll ein "Code-Framework" benutzt werden, um große Teile der Programmierung einer Web-Applikation auf bereits bestehende Fragmente auslagern zu können. Durch den Rückgriff auf Frameworks in beiden Bereichen erhofft sich das Team eine bessere Struktur sowie Zeit- und Ressourceneinsparung bei der Entwicklung, was eine Kosteneinsparung in einem Realszenario bedeuten würde.

### **3.4.2 Frameworkauswahl**

Bei der Sichtung der zur Verfügung stehenden Frameworks wurde eine Vorauswahl hinsichtlich der Verbreitung und Erprobtheit der Tools getroffen. Da es sich bei der zu entwickelnden Lösung um ein Businessprodukt handeln soll, hat es oberste Priorität, dass ein reibungsloser Betrieb gewährleistet werden kann. Zudem erhofft das Team sich durch die Verwendung verbreiteter Bibliotheken, dass ein Support der Lösung durch die Bekanntheit und Dokumentation der Frameworks einfacher vonstattengeht und somit personenunabhängig gewährleistet werden kann.

Zur Evaluation der einzelnen Frameworks wurde ein auf den Anforderungen aufbauender Kriterienkatalog erstellt, der auf der Projektsite unter "4.2 Auswahl Frameworks" eingesehen werden kann.

#### **3.4.2.1 Auswahl der Visualisierungs-Frameworks**

Bei der Recherche nach geeigneten Visualisierungs-Frameworks zeigte sich, dass die meisten verfügbaren Frameworks nicht auf die Darstellung von Dateien sowie komplexeren Datenstrukturen abzielen. Stattdessen sind sie vielmehr dafür ausgelegt, Daten in bestimmten Zusammenhängen zu

veranschaulichen. Gewünschte Anwendungsfelder wären hier etwa Diagramme, Karten oder Hilfsanimationen.

Bei den verbliebenen Frameworks kristallisierten sich im folgenden zwei Varianten heraus. Zum einen könnten MindMap-artige Visualisierungen, wie etwa bei den Frameworks “arbor.js” oder “cola.js”, zur Darstellung von Dateibäumen verwendet werden. Eine alternative Variante bildet die Modifizierung bestimmter Diagramm-Frameworks. Hierbei könnten etwa Googles “Bubble Chart”<sup>10</sup> dahingehend eingesetzt werden, dass die Relevanz der einzelnen Dateien über die Größe der Blasen und der Dokumententyp über deren Färbung ausgedrückt werden könnte.

Der schlussendlich anpassungsfähigste und freieste Weg eine individuelle und innovative Visualisierung zu erreichen, wäre, mittels dynamischem HTML (DHTML) und JavaScript eine eigene Visualisierungskomponente zu entwerfen. Durch den Charakter der Eigenkonstruktion könnte diese in Funktionalität, Form und Umfang genau auf das Projekt und seine Bedürfnisse angepasst werden. Diesem Vorteil steht allerdings gegenüber, dass dies die mit Abstand aufwändigste Realisierungsvariante wäre. Zudem ist durch die fehlende Erprobtheit nicht sichergestellt, dass das gewünschte Ergebnis erreicht werden kann.




	Arbor.js	Google Charts	Cytoscape.js & cola.js
++			
+	Übersichtliche Darstellung	Eigene Gestaltungsmöglichkeiten	Viele Möglichkeiten
0			
-	Wenig anpassbar	Höherer Eigenaufwand	Unübersichtlich bei großer Datenmenge
--	Security Issue in Hangouts		Security Issue in Hangouts

Abbildung 13: Vor- und Nachteile der Visualisierungs-Frameworks

Bei einer abschließenden Evaluierung wurden die am besten geeigneten MindMap- und Diagramm-Frameworks sowie die potentielle Eigenentwicklung anhand des Kriterienkatalogs getestet und auf deren Umsetzbarkeit geprüft. Eines der am vielseitigsten einsetzbaren Frameworks war Cola.js in Verbindung mit Cytoscape.js. Diese Visualisierung wird bei hohem Dateibestand jedoch schnell unübersichtlich. Arbor.js schafft es durch selbsteinklappende Unterpunkte auch bei großen Dateibäumen eine übersichtliche Darstellung zu erhalten. Beide Frameworks werden jedoch durch die Security Richtlinien von Google nicht für Hangouts unterstützt. Die dritte Möglichkeit bezieht sich darauf entweder ein Google Bubble Chart zu animieren oder eine eigene Visualisierung zu

<sup>10</sup> <https://developers.google.com/chart/interactive/docs/gallery/bubblechart>

programmieren. Dies führt zu keinerlei Security Problemen und wird von Google unterstützt, bedeutet jedoch einen deutlichen programmiertechnischen Mehraufwand.

### 3.4.2.2 Auswahl der Code-Frameworks

Die Code-Frameworks sollten neben der allgemeinen Erleichterung des Programmierens vor allem dafür sorgen, dass eine Serververbindung durch AJAX problemlos zustande kommt und dabei helfen Webseiten oder Apps in verschiedenen Browsern gleich darzustellen.



++		Wird von Google Charts benötigt	
+	Funktionalität ähnelt JQuery	Weltweit verbreitetstes JS Framework	Für End to End Anwendungen
0			
-	Rendert direkt im Client		In Verbindung mit node.js
--			benötigt auf Clientseite JQuery

Abbildung 14: Vor- und Nachteile der Code-Frameworks

Drei der beliebtesten Frameworks sind derzeit jQuery, AngularJS und Meteor. Alle zeichnen sich durch eine ausgezeichnete Usability sowie eine weite Verbreitung aus. Bei der detaillierten Betrachtung mit Blick auf die gewünschte Funktionalität stellte sich Meteor als das am wenigsten geeignete Framework heraus. Dies liegt darin begründet, dass Meteor auf dem serverseitigen Node.JS basiert und clientseitig jQuery voraussetzt. Angular.js bietet zwar die meisten und vielseitigsten Möglichkeiten, ist jedoch aus mehreren Gründen für eine animierte Hangouts App ungeeignet. Im Gegensatz zu jQuery manipuliert Angular.js nicht den HTML Dom, sondern rendert clientseitig, ohne Serverbeteiligung. Dies kann bei schwachen Client-Systemen, die schon durch eine laufende Hangouts Verbindung sowie durch eine Animation ausgelastet sind, zu Problemen führen. jQuery gilt als weltweit verbreitetes JavaScript Framework und ist explizit für Animationen geeignet. Wie bereits erwähnt, manipuliert jQuery clientseitig lediglich den vom Server empfangenen DOM des HTML Dokuments und ist somit sehr performant. Erschwerend hinzu kommt, dass jQuery vom gewählten Visualisierungs-Framework Google Charts als Grundvoraussetzung eingefordert wird. Somit wird jQuery als Code-Framework ausgewählt.

## 3.5 Fazit nach Projektschritt 1

In dieser ersten Projektphase galt es sich mit den aktuellen Technologien zur Visualisierung einer

Systemoberfläche auseinander zu setzen und durch ein Scouting diese Technologien mit den Anforderungen, die im Rahmen des Projektes an das System gestellt werden, abzugleichen. Dabei erlangte das Projektteam folgende Erkenntnisse: es ist mit Google Hangouts nicht möglich einen Hangouts Videochat in eine eigene Webanwendung zu integrieren. Aus diesem Grund wird der umgekehrte, aber von Google unterstützte, Weg gegangen und eine eigene Webapp in Hangouts integriert. Da es offensichtlich aber mehrere unterschiedliche Hangouts-Versionen gibt, welche nicht alle die Integration von Apps unterstützen, ist es notwendig die richtige Version mit Hilfe einer Landingpage zu verlinken.

Zur Visualisierung der vorgeschlagenen Dateien gibt es mehrere Ansätze. Diese reichen von einer konventionellen Ordnerstruktur über eine MindMap oder Litfaßsäule bis hin zu einem Blasendiagramm. Bei der Visualisierung kommt es auf eine ansprechende Präsentation der vorgeschlagenen Dateien und eine intuitive Bedienung der Anwendung an.

Eine in Hangouts lauffähige App besteht aus den drei Technologien HTML, CSS und JavaScript. Das Team hat sich dazu entschlossen in nativem JavaScript zu programmieren und zwei verschiedene Frameworks zur Unterstützung beim Code sowie der Visualisierung einzusetzen. Durch das JavaScript Framework jQuery ist es möglich in der Hangouts App auf geeignete Methoden für Animationen zurückzugreifen. Da es für die gewünschte Darstellung keinerlei fertige Frameworks gibt, kommt das Team zu dem Schluss, dass eine selbst geschriebene Anwendung die beste Methode ist, die gegebenen Projektziele zu erreichen. Diese selbst entworfene Anwendung soll optisch an ein Google Bubble Chart erinnern und durch eine Größenänderung der entsprechenden Blase Wichtigkeit symbolisieren. Für unterschiedliche Dateitypen ist es somit nicht nur möglich verschiedenen Farben, sondern auch unterschiedliche Symbole oder Formen zu verwenden.

## 4 Projektschritt 2

Der zweite Projektschritt, der direkt an den Ersten anschließt, beschäftigt sich nun mit der Verwendung der Ergebnisse aus dem ersten Schritt in Form von der Entwicklung lauffähiger Prototypen. Konkret sollen verschiedene Visualisierungskonzepte, wie sie bereits im ersten Schritt oberflächlich betrachtet wurden, im Zuge von Fallstudien miteinander verglichen werden. Der zweite Projektschritt endet mit der Abschlusspräsentation vor dem Dozenten sowie den anderen Teams am 23.01.2017. Dies ist gleichzeitig das Ende der Entwicklung für das erste Semester.

### 4.1 Änderungen in Bezug auf Projektschritt 1

Trotz gründlicher Planung in Projektschritt 1, musste das Team im Laufe des Projektschritts 2 ein paar Änderungen vornehmen, welche im Folgenden beschrieben werden.

#### 4.1.1 Review der Ergebnisse des ersten Projektschrittes

Zu Beginn des zweiten Projektschrittes fand eine gemeinsame Betrachtung der vom Team gelieferten Ergebnisse mit dem Dozenten und Auftraggeber statt. Hierbei erhielt das Team grundsätzlich positives Feedback und die Genehmigung für die gewählte Vorgehensweise. Die Besprechung der als Ausblick für den zweiten Schritt entworfenen Visualisierungskonzepte, lieferte unter Betrachtung der Erkenntnisse aus dem Technologie-Scouting das Ergebnis, dass ein Großteil der Konzepte im Rahmen des Projekts aufgrund mangelnder Unterstützung seitens der vorhandenen Visualisierungs-Frameworks nicht umsetzbar sind. In Absprache mit dem Dozenten wurden zwei neue Visualisierungskonzepte entworfen, die von nun an in zwei separaten Fallstudien entwickelt und miteinander verglichen werden sollten. Zudem sollten beide Visualisierungsformen sowohl im Rahmen einer klassischen App als auch in Form einer Erweiterung in Hangouts integriert werden. Als Fazit auf Projektschritt 1 zog das Team noch, dass eine Entwicklung als klassische App auf Grund des größeren Platzangebots stattfinden würde. Eine Vorstellung der einzelnen Konzepte erfolgt jeweils zu Beginn der Fallstudien, welche zu einem späteren Zeitpunkt in dieser Dokumentation aufgeführt sind. Diese Änderung der Visualisierungskonzepte hatte ebenfalls Auswirkungen auf die verwendeten Visualisierungs-Frameworks. Auf diese wird in den einzelnen Fallstudien eingegangen, da sich der Frameworks-Einsatz je nach Fallstudie unterscheidet.

#### 4.1.2 Abschaltung der Hangouts-API

Bevor allerdings auf diese Fallstudien eingegangen werden kann, muss eine nötig gewordene, grundlegende Änderung des Projektes dargelegt werden. So stand als Fazit des ersten Projektschrittes fest, dass das System als entsprechende Applikation in Hangouts integriert werden würde, was auch nach den Review mit dem Dozenten zu Beginn des zweiten Schrittes als gewählte Vorgehensweise bestehen blieb. Dieses Vorhaben wurde allerdings kurzfristig durch eine Email von



Google unmöglich. Google hat am Abend des 6.01.17 deutscher Zeit bekannt gegeben, dass die Google Hangouts API, auf der der gesamte Projektplan basierte, nicht länger zur Verfügung steht und mit sofortiger Wirkung eingestellt wird. Es konnten mit sofortiger Wirkung keine neuen Applikationen mehr veröffentlicht werden und bestehende würden nur noch bis 25.04.17 unterstützt. Begründet wurde das Vorgehen durch eine Refokussierung von Hangouts in Richtung B2B-Segment.<sup>11</sup> Zwar hatte das Team zu diesem Zeitpunkt bereits eine erste Applikation für Google Hangouts als Test veröffentlicht, allerdings wurde eine Weiterentwicklung dieser in Hinblick auf die verbleibende Laufzeit von etwas mehr als drei Monaten vom Team abgelehnt.

Beide Fallstudien befanden sich zu diesem Zeitpunkt bereits in einem fortgeschrittenen Stadium und mussten als Konsequenz an die neuen Gegebenheiten angepasst werden. Als Lösung des Problems entschied sich das Team in Absprache mit dem Dozenten dazu, den ursprünglich abgelehnten Ansatz der Trennung von Hangouts und Applikation wieder aufzugreifen. Dieser war aber der einzig verbliebene mögliche Ansatz und brachte neben der geringer eingestuften Ergonomie keine weiteren Nachteile mit sich. Ebenso konnten die Fallstudien ohne zu große Anpassungen oder neuen Problemstellungen fortgeführt werden, da diese bereits in HTML, CSS und JavaScript entwickelt wurden, was sich leicht in eine alleinstehende Webanwendung überführen ließ. Um ein ansprechendes Bild zu erhalten entschloss sich das Team dazu, die Applikationen direkt in die ursprünglich für die Verlinkung angedachte Landingpage zu implementieren. Als Schlussfolgerungen wurden beide Fallstudien als alleinstehende Webapplikationen unter Verwendung von HTML, CSS und JavaScript entwickelt. Beide Fallstudien greifen außerdem auf jQuery als Code-Framework zurück.

#### **4.1.3 Gestaltung Landingpage**

Mit der Abschaltung der Google Hangouts API erhielt auch die anfangs nur als Einstiegspunkt für die richtige Hangouts-Version geplante Landingpage eine erweiterte Funktion. Sie dient nun nicht mehr nur als Startseite zur Applikation, sondern enthält die Anwendung selbst. Somit stellt die erweiterte Landingpage das komplette GUI dar und kann im Browser geöffnet parallel zu Google Hangouts, Skype oder anderen Videokonferenz-Tool genutzt werden.

Die Landingpage selbst wurde über Bootstrap verwirklicht, um dadurch erneut auf bestehendes Wissen und vorhandene Methoden zurückgreifen zu können. Bootstrap wurde deshalb ausgewählt, da es den Teammitgliedern bereits bekannt war und das Team sich in Anbetracht der Kurzfristigkeit hier die schnellsten Erfolge erhoffte. Bootstrap unterstützt dabei die Webseite responsive darzustellen und stellt sicher, dass die grundlegenden Informationen der Webseite für alle Geräte und Browser zugänglich sind. Das verwendete Theme Grayscale ist durch die MIT-Lizenz lizenziert und damit Open Source, was bedeutet, dass es frei verwendet und verändert werden darf.

---

<sup>11</sup> Ein Screenshot besagter Mail kann dem Anhang dieser Dokumentation entnommen werden.

Die Folgen für den Aufbau des Systems werden in den folgenden Abbildungen dargestellt. Abbildung 15 zeigt die ursprünglich geplante Umsetzung als Hangouts-App. Zu erkennen ist, dass bei dieser Umsetzung die Rolle des Hangouts Client und des Recommender Clients nahezu identisch sind, da unser zu entwickelndes System eine Erweiterung von Hangouts darstellt.

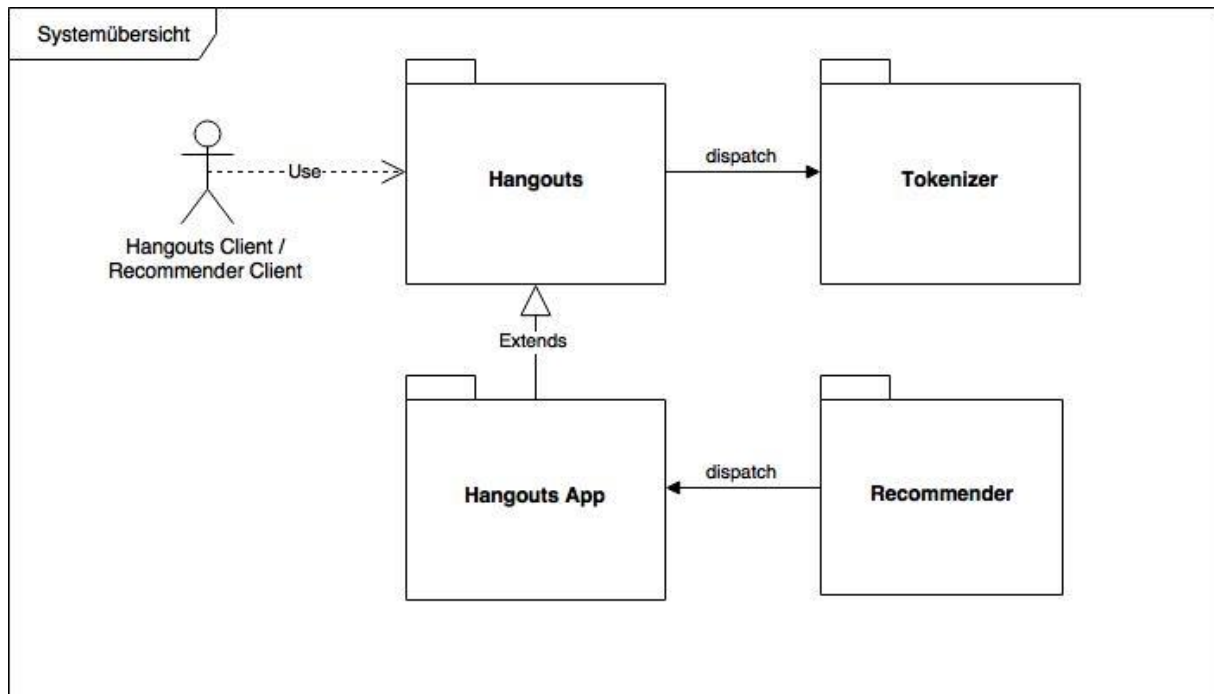


Abbildung 15: Systemübersicht Hangouts App

Abbildung 16 hingegen zeigt den neu gestalteten Systemkontext als Reaktion auf die Sperrung der Hangouts-API (vgl. 4.1.2). Im Vergleich zu dem in Abbildung 15 dargestellten Systemkontext sind die Rollen des Hangout Clients und des Recommender Clients unterschiedlich, da die GUI unseres Gesamtsystems über eine separate Website realisiert wird.

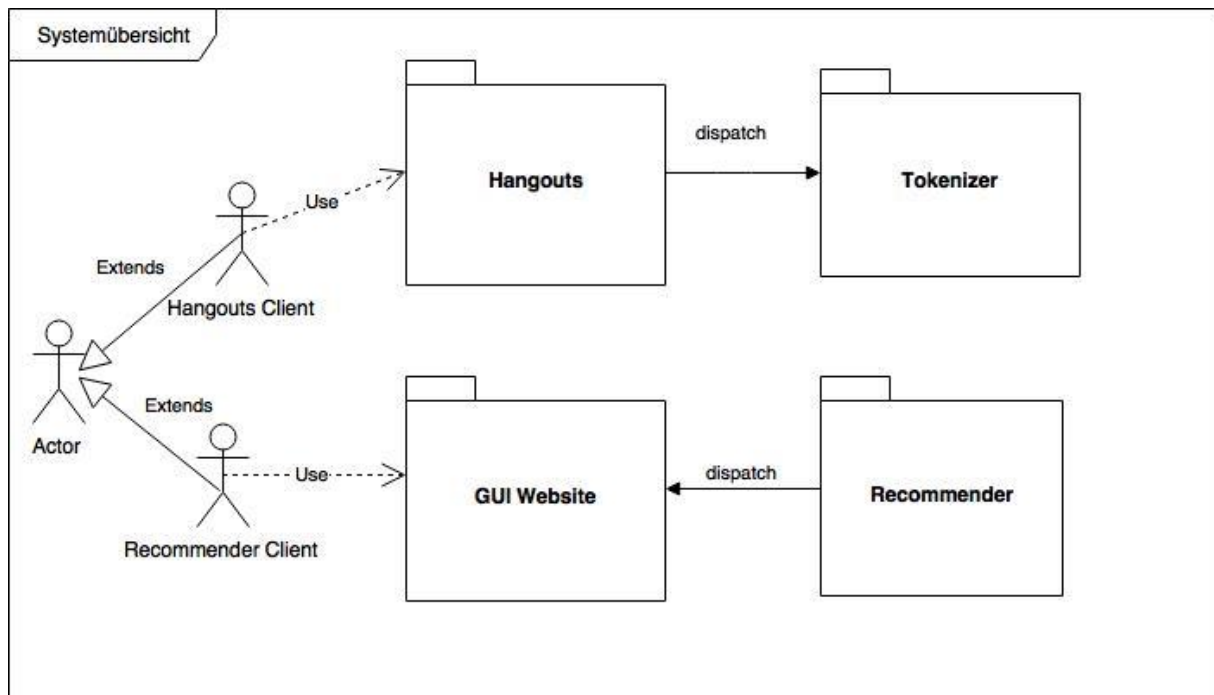


Abbildung 16: Systemübersicht GUI Website

#### 4.1.4 Einführung von User Stories

Als weitere Änderung wurde im Team hinsichtlich der Formulierung und Dokumentation der Anforderungen ein Umstieg auf User Stories beschlossen. Grund hierfür war eine Diskussion des Teams zu Beginn des Projektschrittes, wie die Fallstudien entworfen und konzipiert werden sollten. Hierbei kam die Gruppe zum Ergebnis, dass User Stories am geeignetsten sind, da diese zur gewählten agilen Vorgehensweise passen und gleichzeitig direkt so formuliert werden können, dass diese einen Testfall beinhalten. Das Team stellte daraufhin folgende User Stories als Grundlage für die weitere Entwicklung der Fallstudien auf, welche auch unter dem Punkt 6.1 auf der Projektseite vorgefunden werden können.

1. Dem User werden die **Dokument Vorschläge laufend aktualisiert** durch die App/ Erweiterung dargestellt, damit für User nachvollziehbar ist welche Dokumente vom System mit welchen Eigenschaften versehen wurden.
2. Dem User wird der **Erwähnungs-Zeitpunkt** der Vorschläge angezeigt, damit er nachvollziehen kann, zu welchem Zeitpunkt des Gesprächs die vorgeschlagenen Dokumente erwähnt wurden.

3. Der User hat die Möglichkeit, auf die gesamte Darstellung des **Gesprächsverlaufs zuzugreifen**, um bereits erwähnte Dokumente zu öffnen.
4. Der User hat die Möglichkeit **vom Gesprächsverlauf zum aktuellen Stand** des Gesprächs zurückzukehren, damit der aktuellen Stand des Gesprächs stets zu erreichen ist.
5. Dem User werden die Dateien **nach ihrer Priorität geordnet** angezeigt, damit er nachvollziehen kann welches Dokument vom System als wichtiger oder als unwichtiger im verhältnis zu den Anderen Dokumenten eingeschätzt wurde.
6. Der User kann die **Priorisierung des Systems überstimmen** und neu anordnen, um
  - 1.) seine eigenen Bedürfnisse zu erfüllen und somit die Benutzerfreundlichkeit zu gewährleisten.
  - 2.) dem System ein Lernprozess zu ermöglichen.
7. Der User kann direkt erkennen um welche **Kategorie von Dokument** es sich bei dem Dokument Vorschlag handelt, um dem User die Entscheidung über die tatsächliche Relevanz des Dokument Vorschlags zu erhöhen.
8. Wählt der User ein vorgeschlagenes Dokument aus, hat er die Möglichkeit, sich dieses Dokument im Google Drive Ordner in der **File-Ansicht** anzeigen zu lassen, um so den Ablageort des betreffenden Dokuments einsehen zu können.
9. Wählt der User ein vorgeschlagenes Dokument aus, hat er die Möglichkeit, dieses **Dokument direkt zu öffnen**, um dieses Dokument einsehen und bearbeiten zu können.

Aus diesen User Stories wurden in Rücksprache mit Herrn Prof. Dr. Rathke die Anwendungsfälle für die zu entwickelnde GUI Website herausgearbeitet. Sie definieren im weiteren Entwicklungsprozess die umzusetzenden Funktionalitäten aus Sicht des Anwenders (Recommender Clients) und bilden somit die gemeinsame Grundlage der zwei Fallstudien. Als Übersicht wurde ein Use Case Diagramm nach UML Konvention angelegt, sowie jeder der darin enthaltenen Anwendungsfälle in den nachfolgenden Tabellen detailliert beschreiben.

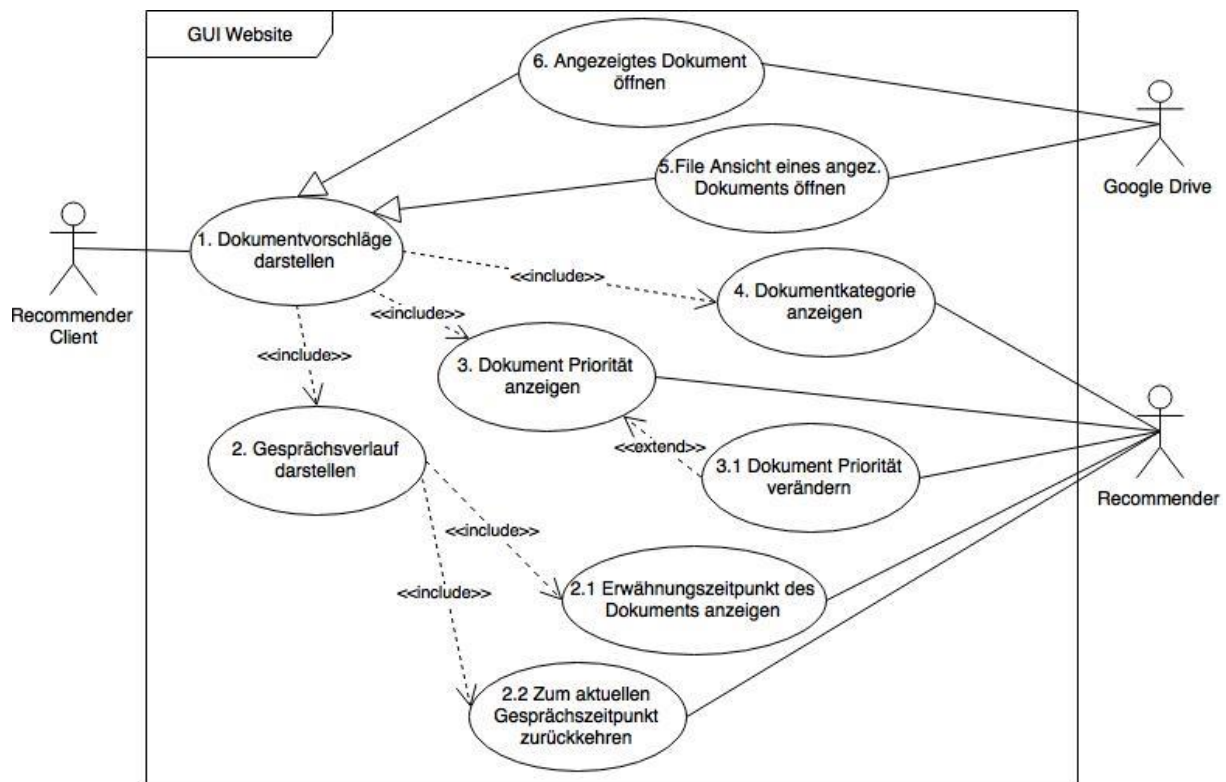


Abbildung 17: UML Use Case Diagramm

Beschreibung Anwendungsfall	
Name	1. Dokumentvorschläge darstellen
Kurzbeschreibung	Die Dokumentenvorschläge werden mit allen vom System vorgesehenen Attributen ausgegeben. Dieser Anwendungsfall ist als initialer Anwendungsfall der Einstieg in das System
Akteure	Recommender Client (RC), Recommender (R), Google Drive
Auslöser	Recommender Client(RC) möchte sich die Dokumentvorschläge des Recommenders (als Teil des Kobora-Systems) anzeigen lassen.
Ergebnis(se)	Die Use Cases 2-4 werden ausgeführt.(siehe 2-4)
Eingehende Daten	Der RC hat die Website aufgerufen
Vorbedingungen	Die Website hat eine Verbindung zum Recommender Die Website hat Zugriff auf den Google Drive Ordner der angesprochenen Dokumente

Nachbedingungen	(siehe Anwendungsfall 2-4)			
Essenzielle Schritte	1) RC ruft Website auf 2) Bekommt Dokumentvorschläge durchgehend dargestellt			
Offene Punkte	Login bzw. Identifizierung offen			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

Beschreibung Anwendungsfall	
Name	2. Gesprächsverlauf darstellen
Kurzbeschreibung	Die Dokumentvorschläge werden dem RC in einer Zeitlichen Abfolge dargestellt (Use Case 2.).  Der Erwährungszeitpunkt des Dokuments wird dem RC angezeigt (Use Case 2.1).  Der RC befindet sich innerhalb des Gesprächsverlaufs und kehrt zum aktuellen Stand des Gesprächs zurück.
Akteure	Recommender Client (RC), Recommender (R)
Auslöser	Der Recommender Client(RC) hat den Anwendungsfall 1 Dokumentvorschläge darstellen angewendet. Der Gesprächsverlauf wird in der Folge automatisch und ohne weitere Aktion des RC dargestellt.
Ergebnis(se)	Gesprächsverlauf wird von Beginn bis zum aktuellen Stand des Gesprächs dargestellt (Use Case 2.).  Jedes Dokument wird mit einem Erwährungszeitpunkt angezeigt (Use Case 2.1).  Der RC bekommt das zuletzt erwähnte Dokument angezeigt (Use Case 2.2).
Eingehende Daten	Dokumentvorschläge des Recommenders mit Timestamp

Vorbedingungen	Anwendungsfall „1. Dokumentvorschläge darstellen“ wurde durchlaufen Die Website hat eine Verbindung zum Recommender			
Nachbedingungen	Der Gesprächsverlauf kann vollständig und somit vom ersten bis zum letzten Dokumentvorschlag angezeigt werden (Use Case 2.).  Jedes Dokument wird mit einem Erwährungszeitpunkt des Dokuments angezeigt(Use Case 2.1).  Der RC befindet sich am zuletzt eingegangenen Dokumentvorschlag(Use Case 2.2).			
Essenzielle Schritte	Use Case 2: 1.) Der Recommender liefert seine Dokumentvorschläge kontinuierlich an die Website. 2.) Die Website stellt diese Vorschläge nach ihrem Erwährungszeitpunkt chronologisch dar Use Case 2.1: 1.) Die Erwährungszeitpunkte des Dokuments werden dargestellt Use Case 2.2: 1.) Der RC sucht im Gesprächsverlauf nach einem nicht mehr angezeigten Dokumentvorschlag dieses Gesprächs. 2.) Der RC kehrt zum zuletzt angezeigten Dokumentvorschlag zurück			
Offene Punkte	keine			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

Beschreibung Anwendungsfall	
Name	3. Dokumentpriorität anzeigen
Kurzbeschreibung	Die Dokumentpriorität, die vom System vergeben wurde, wird dem RC dargestellt (Use Case 3.).  Der RC kann diese vom System vergebene Dokumentpriorität verändern (Use Case 3.1).
Akteure	Recommender Client (RC), Recommender (R)
Auslöser	Der Recommender Client(RC) hat den Anwendungsfall 1 Dokumentvorschläge darstellen angewendet. Die Dokumentpriorität wird in

	der Folge automatisch und ohne weitere Aktion des RC dargestellt.			
Ergebnis(se)	<p>Dokumentpriorität wird von Beginn bis zum aktuellen Stand des Gesprächs für jedes Dokument dargestellt(Use Case 3.).</p> <p>Der RC hat die vom System vergebene Priorität des gewählten Dokuments verändert (Use Case 3.1).</p>			
Eingehende Daten	Priorität des Recommenders			
Vorbedingungen	<p>Anwendungsfall „1. Dokumentvorschläge darstellen“ wurde durchlaufen</p> <p>Die Website hat eine Verbindung zum Recommender</p>			
Nachbedingungen	<p>Die Dokumentpriorität kann vollständig und somit vom ersten bis zum letzten Dokumentvorschlag angezeigt werden (Use Case 3.).</p> <p>Das vom RC mit neuer Priorität versehene Dokument weist die Priorität des RC und nicht die des R aus (Use Case 3.1).</p>			
Essenzielle Schritte	<p>Use Case 3:</p> <ol style="list-style-type: none"> <li>1.) Der Recommender liefert seine Dokumentpriorität kontinuierlich an die Website.</li> <li>2.) Die Website stellt diese Vorschläge nach ihrem Priorität geordnet dar</li> </ol> <p>Use Case 3.1:</p> <ol style="list-style-type: none"> <li>1.) Der RC sucht im Gesprächsverlauf nach einem nicht mehr angezeigten Dokumentvorschlag dieses Gesprächs.</li> <li>2.) Der RC kehrt zum zuletzt angezeigten Dokumentvorschlag zurück</li> </ol>			
Offene Punkte	keine			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

Beschreibung Anwendungsfall	
Name	4. Dokumentkategorie anzeigen



Kurzbeschreibung	Die Dokumentkategorie der Dokumentenvorschläge werden angezeigt			
Akteure	Recommender Client (RC), Recommender (R),			
Auslöser	Der Recommender Client(RC) hat den Anwendungsfall 1 Dokumentvorschläge darstellen angewendet. Die Dokumentkategorie wird in der Folge automatisch und ohne weitere Aktion des RC dargestellt.			
Ergebnis(se)	Dokumentkategorie wird von Beginn bis zum aktuellen Stand des Gesprächs für jedes Dokument dargestellt.			
Eingehende Daten	Dokumentkategorie des Recommenders			
Vorbedingungen	Anwendungsfall „1. Dokumentvorschläge darstellen“ wurde durchlaufen Die Website hat eine Verbindung zum Recommender			
Nachbedingungen	Die Dokumentkategorie kann vollständig und somit vom ersten bis zum letzten Dokumentvorschlag angezeigt werden.			
Essenzielle Schritte	1.) Der Recommender liefert seine Dokumentkategorien kontinuierlich an die Website. 2.) Die Website stellt diese Vorschläge und ihre Kategorien dar			
Offene Punkte	keine			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

Beschreibung Anwendungsfall	
Name	5. File Ansicht eines angezeigten Dokuments öffnen
Kurzbeschreibung	Der RC kann einen Dokumentvorschlag auswählen und sich den Ablageort des dazugehörigen Dokuments im Google Drive Ordner anzeigen lassen
Akteure	Recommender Client (RC), Google Drive
Auslöser	Der Recommender Client(RC) wählt eine der ihm dargestellten Dokumentvorschläge aus.
Ergebnis(se)	Der Recommender Client(RC) bekommt zu seinem ausgewählten Dokumentvorschlag den Ablageort des dazugehörigen Dokuments im Google Drive Ordner angezeigt.

Eingehende Daten	Auswahl des Dokumentvorschlags durch den RC			
Vorbedingungen	Anwendungsfall „1. Dokumentvorschläge darstellen“ wurde durchlaufen Die Website ist der Ablageort des Dokuments bekannt			
Nachbedingungen	Der Ablageort des Dokument wird dem RC angezeigt			
Essenzielle Schritte	1.) Der RC wählt einen Dokumentvorschlag aus 2.) Der RC wählt die Funktion aus, die ihm das dazugehörige Dokument im Google Drive Ordner anzeigt. 3.) Dem Nutzer wird der Ablageort im Google Drive Ordner angezeigt.			
Offene Punkte	keine			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

Beschreibung Anwendungsfall	
Name	6. Angezeigtes Dokument öffnen
Kurzbeschreibung	Der RC kann einen Dokumentvorschlag auswählen und sich das dazugehörige öffnen und anzeigen lassen
Akteure	Recommender Client (RC), Google Drive
Auslöser	Der Recommender Client(RC) wählt eine der ihm dargestellten Dokumentvorschläge aus.
Ergebnis(se)	Der Recommender Client(RC) bekommt zu seinem ausgewählten Dokumentvorschlag das dazugehörige Dokument geöffnet angezeigt.
Eingehende Daten	Auswahl des Dokumentvorschlags durch den RC Ablageort des entsprechenden Dokuments im Google Drive Format des Dokuments Information welches Programm sich für dieses Dokumentformat eignet
Vorbedingungen	Anwendungsfall „1. Dokumentvorschläge darstellen“ wurde durchlaufen Die Website ist der Ablageort des Dokuments bekannt Format des Dokuments ist bekannt

	Information welches Programm sich für dieses Dokumentformat eignet ist bekannt			
Nachbedingungen	Das Dokument wird dem RC angezeigt			
Essenzielle Schritte	1.) Der RC wählt einen Dokumentvorschlag aus 2.) Der RC wählt die Funktion aus, die ihm das dazugehörige Dokument geöffnet anzeigt. 3.) Dem Nutzer wird das Dokument im Google Drive Ordner angezeigt.			
Offene Punkte	keine			
Änderungshistorie	15.01.17	Jh200	neuer Status	Was
Sonstiges, Anmerkungen	keine			

## 4.2 Fallstudie 1 - Explorer

Die erste der beiden Fallstudien trägt den Projektnamen "Explorer". Wie bereits erwähnt, dienen die beiden Fallstudien dazu zwei verschiedene Visualisierungsformen für das System prototypisch zu entwickeln und miteinander zu vergleichen. Die für die erste Fallstudie gewählte Visualisierung ist stark an eine listenartige Exploreransicht angelehnt, wie sie etwa in der Schnellzugriffsleiste des Windows Explorers integriert ist.

### 4.2.1 Entwurf

Ursprünglich war das Konzept aufgrund der schmalen Form vor allem für die Erweiterungsform der Hangouts-Applikation angedacht, erhielt allerdings im Zuge der Umstellung eine breitere Gesamtfläche. Der erste Entwurf der Visualisierung kann Abbildung 18 entnommen werden.

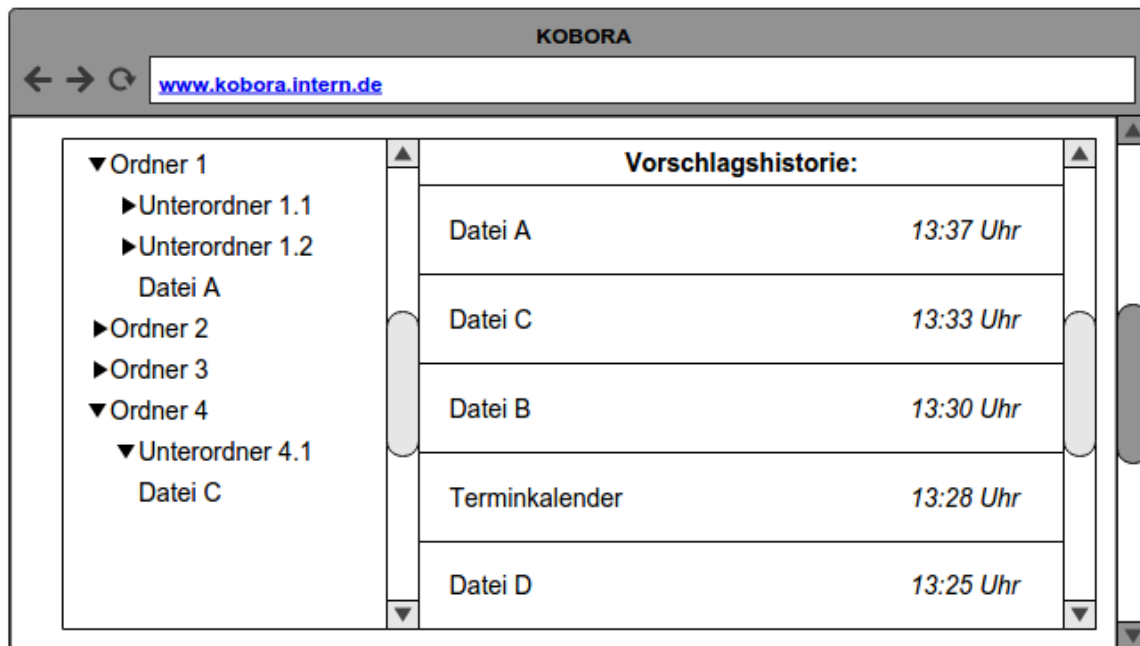


Abbildung 18: Erster Entwurf - Explorer

Im Entwurf des Explorers als alleinstehende Webanwendung wurde der Anwendungsbildschirm in zwei Hälften geteilt.

In der linken Hälfte ist die bereits für die Hangouts-Applikation geplante Exploreransicht vorzufinden, welche durch eine Historie der Vorschläge im rechten Fenster erweitert wurde. Die vom Recommender generierten Vorschläge werden in diesem Entwurf darüber kommuniziert, dass sich die Listenstruktur im Explorerfenster dynamisch auf- und zuklappt. So werden neue Vorschläge im Ordnersystem aufgeklappt und der Fokus der Anwendung auf diese ausgerichtet. Der Nutzer sieht also zu jedem Zeitpunkt den neuen Vorschlag, auch wenn dieser außerhalb des bisherigen Sichtfeldes lag. Er kann aber wie bereits in den grundlegenden Anforderungen definiert jederzeit selbst Kontrolle über das System übernehmen und frei in der Struktur navigieren und Dateien öffnen. Um die Usability zu gewährleisten, findet in Phasen der aktiven Nutzersteuerung keine automatische Neufokussierung des Systems statt.

Die Vorschlags-Historie wurde als Lösung für zwei durch die Ausrichtung auf den neuesten Vorschlag bzw. der Möglichkeit zur Eigennavigation entstehende Probleme implementiert. Sollten zwei Vorschläge, die sich nicht zusammen auf einer Sichtfenstergröße darstellen lassen, unmittelbar aufeinander folgen, hätte der User keine Möglichkeit den ersten Vorschlag wahrzunehmen, da dieser unmittelbar vom zweiten Vorschlag überblendet werden würde. Ein anderes Problem entsteht in dem Moment, wenn ein Vorschlag eintrifft, während der Nutzer aktiv im System navigiert und somit eine Neuausrichtung als optisches Signal für einen neuen Vorschlag nicht zur Verfügung steht. Es könnte somit unter Umständen sein, dass dem User der neue Vorschlag entgeht. Durch die Vorschlags-Historie kann der User jederzeit die zurückliegenden Vorschläge einsehen, ohne dass

diese sich gegenseitig blockieren. Während einer aktiven Navigation im Explorer kann die Historie dennoch aktualisiert werden, wodurch der User auf den neuen Vorschlag hingewiesen wird. Die Historie wird somit als Lösung für die beiden beschriebenen essentiellen Probleme benötigt und implementiert.

#### 4.2.2 Systemspezifikation

Für die Fallstudie "Explorer" wurde neben den bereits beschriebenen Technologien zusätzlich das Visualisierungs-Framework FancyTree verwendet. FancyTree<sup>12</sup> ist ein Plugin für jQuery, dass die für den Explorer nötige Baumstruktur bereitstellt und darüber hinaus einige einfach einzubindende Features hat, welche für das Projekt jedoch nicht genutzt wurden. FancyTree ist als openSource Projekt auf GitHub zu finden und steht unter der MIT-Lizenz Jedem zur Verfügung. Hierdurch lässt sich das Framework einfach mit den bereits im Projekt verwendeten Bibliotheken kombinieren. Durch die OpenSource-Lizenz entstehen zudem keine Kosten, was eine positive ökonomische Betrachtung nach sich zieht. Neben FancyTree wurden JavaScript mit jQuery für die funktionale, HTML und CSS für die grafische Umsetzung genutzt.

#### 4.2.3 Implementierung

Die EntryPage des Explorers wurde fancytree2.html benannt, sie basiert ebenfalls wie der zweite Ansatz BubbleChart auf einem Bootstrap-Theme Grayscale. In diesem Fall soll eine einheitliche Darstellung gewährleistet werden. Die fancytree2.html ist somit der erste Berührungspunkt der Anwendung Explorer. Sie integriert die Frameworks Bootstrap, Fancytree und somit auch jQuery und AJAX. jQuery ist ein JavaScript-Toolkit, dass eine einfache Schnittstelle zu Document Object Model bietet (Repräsentation eines Dokuments, auf die mit JavaScript zugegriffen werden kann). Es ist, auch dank der zahlreich verfügbaren Plugins, eines der meist genutzten Frameworks. AJAX steht für "Asynchronous JavaScript and XML" und bezeichnet ein Konzept der asynchronen Datenübertragung zwischen Browser und Server. Dieses ermöglicht es HTTP-Anfragen durchzuführen, während eine HTML-Seite angezeigt wird und die Seite zu verändern, ohne sie komplett neu zu laden.

Beim Laden der fancytree2.html initialisiert sie mit der function init() die beiden Buttons Starten und Stoppen. Durch einen Klick auf den Start-Button wird das Mitscrollen des Anzeigebereichs sowie ein sich alle drei Sekunden wiederholendes Intervall gestartet. Im Rahmen dieses Intervalls holt die starteTree()-Methode in den Dateien Test1.json, Test2.json und Test3.json mittels eines Zufallsprinzips die JSON-Daten ab und gibt diese an fancytree() mittels jQuery weiter. Die Daten

---

<sup>12</sup> <https://github.com/mar10/fancytree>

werden aus den JSON-Dateien anschließend in zwei verschiedenen <DIV> Bereichen geladen. Der Explorer (Informationsbaum/ Tree) wird auf der linken Seite dargestellt und die Liste mit den Informationen der Highlights auf der rechten Seite. Die Methode fillList() ermöglicht das Darstellen der zuvor im Informationsbaum markierten JSON-Daten (Highlight) und erstellt alle drei Sekunden eine Zeile, welche klickbar ist und auf den Link verweist. Das Beenden der Anwendung wurde mit der stoppeAnwendung() Methode umgesetzt, welche die Intervalle löscht und somit die Anwendung unterbricht.

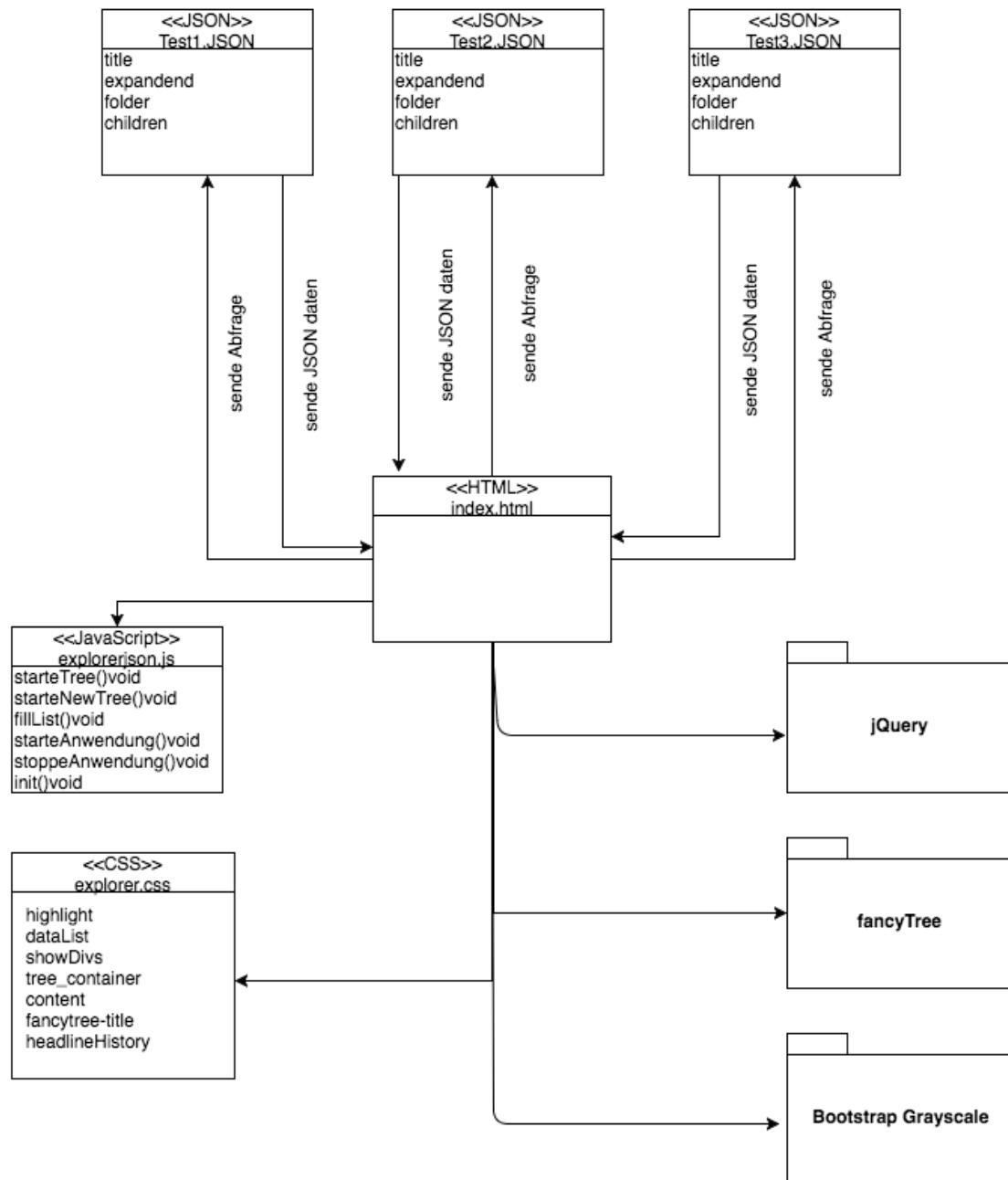


Abbildung 19: UML-Diagramm für den Explorer-Prototyp

#### 4.2.4 Test

Parallel zur Implementierung wurden ständig Tests durchgeführt, welche die Funktionalität auch in unterschiedlichen Browsern sichergestellt haben. Getestet wurde in den neuesten Versionen der folgenden Browser: Google Chrome, Mozilla Firefox und Microsoft Edge.

Zudem wurde abschließend überprüft, ob alle in Kapitel 4.1.4 eingeführten User Stories mit dem Prototypen umgesetzt werden können. Dies ist beim "Explorer"-Prototyp der Fall, weshalb der Test als erfolgreich bestanden angesehen wird.

#### **4.2.5 Fazit der Fallstudie**

Der Prototyp "Explorer" hatte bei seiner Entwicklung Vor- und Nachteile aufgezeigt. Eine strukturierte Darstellungsform (Highlight Liste) ermöglichte eine angenehme Informationsentnahme, aus der der Dateipfad des Vorschlags ersichtlich ist. Dennoch zählt die Explorer-Ansicht zu einer altmodischen Variante der Informationsdarstellung. Ein weiterer Nachteil ist, dass nicht alle Vorschlagstypen, vor allem Kalender-Verlinkungen, nicht einheitlich dargestellt werden können.

Alles in allem konnte der Prototyp mittels Frameworks, wie Bootstrap oder der Darstellungs-API FancyTree, gut umgesetzt werden.

### **4.3 Fallstudie 2 "Bubbles"**

Bei der zweiten Fallstudie handelt es sich um die "Bubbles"-Fallstudie. Wie bereits bei der ersten Fallstudie gilt es auch hier ein bestimmtes Visualisierungskonzept prototypisch umzusetzen und zu evaluieren. Hierbei handelt es sich in dieser Fallstudie um eine Darstellung via "Bubbles" bzw. Blasen, die über den Bildschirm laufen.

#### **4.3.1 Entwurf**

Bei ihrer ursprünglichen Konzeption als Visualisierung in einer Hangouts-Applikation waren die Bubbles für die breitere Darstellung in einer klassischen App ausgelegt. Diese breite Form wurde auch nach der Umstellung beibehalten, wie im ersten Entwurf in Abbildung 20 zu erkennen ist:



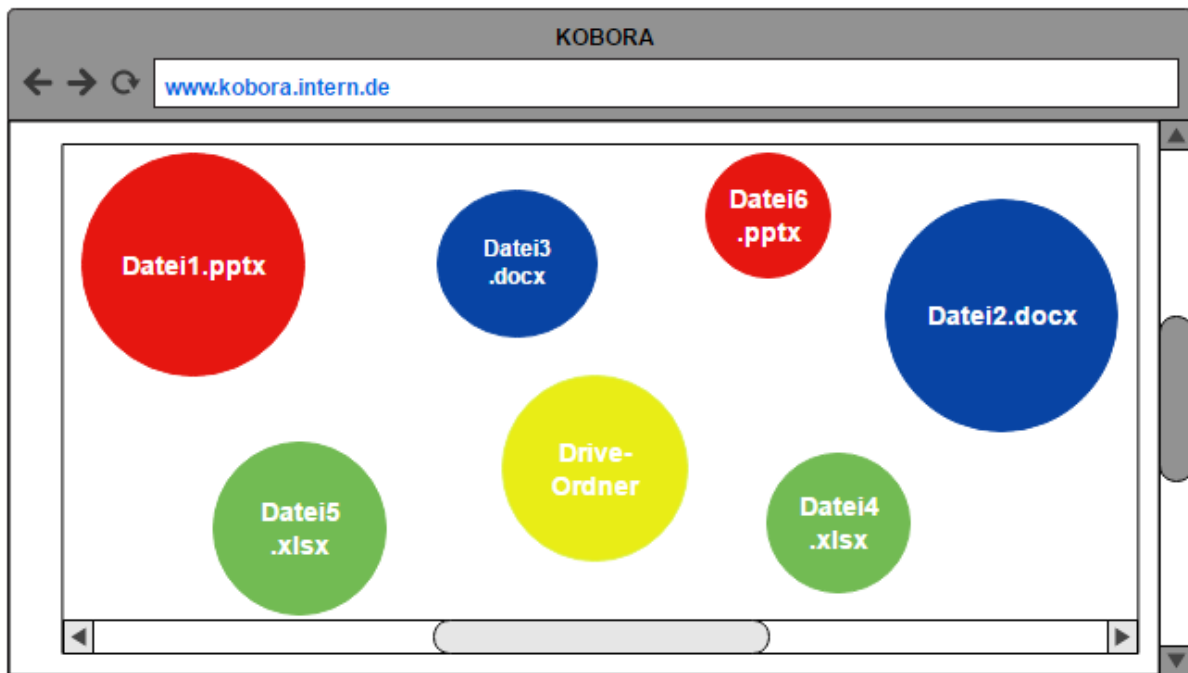


Abbildung 20: Erster Entwurf - Bubbles

Die Bubble-Visualisierung sieht ein großes, dynamisches Anwendungsfeld vor, durch das sich Blasen verschiedener Farben und Größen linear bewegen.

So gibt die Farbe einer Blase an, auf welchen Typ von Datei diese verlinkt. Hierbei wurde sich an den aus dem Microsoft Office und den Kollaborationstools der Google-Suite bekannten Farbmuster orientiert, wodurch etwa Textdokumente durch blaue und Tabellen über grüne Blasen dargestellt werden.

Die Größe der Blasen hingegen steht sinnbildlich für die Priorität bzw. Wahrscheinlichkeit einer Übereinstimmung des Vorschlages mit dem tatsächlichen Gespräch. Vom System als mit hoher Wahrscheinlichkeit treffende Vorschläge eingestufte Vorschläge werden somit größer als weniger wahrscheinliche dargestellt, um den User schnell und einfach eine visuelle Orientierung zu ermöglichen.

Die horizontale Ebene soll hierbei als Zeitachse verwendet werden, wodurch in diesem Konzept die Darstellung neuer Vorschläge intuitiv mit der Darstellung einer Historie verknüpft wird, da der User sich frei auf dieser Achse bewegen kann.

Die vertikale Achse besitzt keine tiefere Bedeutung, weshalb die Blasen randomisiert in dieser Dimension angeordnet werden. Zwar überlegte das Team mehrere Verwendungszwecke für die Y-Achse, entschied sich aber jeweils dagegen, weshalb am Ende eine Randomisierung als Mittel für eine ausgeglichene Darstellung gewählt wurde. Eine Verwendung als Darstellung des letzten Änderungszeitpunkts wurde abgelehnt, da eine Position damit nicht stetig wäre. Sollte eine neue Änderung stattfinden, müsste die Bubble an die Spitze der Achse geschrieben werden. Allgemein

müssten alle Blasen nach und nach absinken, um mit der fortschreitenden Zeit während dem Gespräch übereinzustimmen. Da dies zum einen für eine optische Unruhe sorgen und den Nutzer unnötig vom Gespräch ablenken würde und zum anderen entsprechende technologische Kapazitäten einfordern würde, wurde der Vorschlag vom Team abgelehnt. Vorschläge, die eine Verwendung der Y-Achse für die Sortierung nach Dateityp oder Priorität vorsahen, wurden abgelehnt, da diese Werte bereits über die Blasen vermittelt wurden und beide Anregungen ein durch Sortierung sehr monotones Bild erzeugen würden.

Wird nun ein neuer Vorschlag vom Recommender übermittelt, erzeugt das System eine dem Vorschlag entsprechende Blase und blendet diese am Rand des Anwendungsfelds ein. Die horizontale Achse ist hierbei einer permanenten Fortschreitung ausgesetzt, die den zeitlichen Fortschritt während des Gesprächs darstellt. Das System richtet, sofern der Nutzer nicht aktiv navigiert, die Horizontale stets so aus, dass sie aktuell gehalten wird. Hierdurch entsteht eine Wanderung der Blasen quer über den Bildschirm, bis diese schließlich am anderen Rand ausgeblendet werden. Der Nutzer kann über eine Scrollmöglichkeit jederzeit Kontrolle über die X-Achse erlangen, was eine Betrachtung der Vorschlags-Historie ermöglicht. Sollte der Nutzer die Kontrolle wieder abgeben wollen, muss das System wieder in den automatischen Modus zurückgeführt werden.

### 4.3.2 Systemspezifikation

Für die Fallstudie "Bubbles" wurden nur die bereits beschriebenen Technologien verwendet. Dabei wurde JavaScript mit jQuery für die funktionale, HTML und CSS für die grafische Umsetzung gewählt. Ein weiteres Visualisierungs-Framework wurde nicht genutzt. Die Bubbles sind eine Eigenentwicklung, angelehnt an die Google Bubble Chart, die in Kapitel 3.4.2.1 kurz vorgestellt wurde.

### 4.3.3 Implementierung

Die Landingpage des Prototyps ist die index.html, sie basiert auf dem Bootstrap-Theme Grayscale. Die index.html ist der Startpunkt der gesamten GUI, sie integriert neben den von Grayscale verwendeten Frameworks jQuery und Bootstrap die eigenentwickelten JavaScript Klassen xhrojekt.js und ajaxjson.js. Die Klasse xhrojekt.js erzeugt je nach Browser ein anderes XMLHttpRequest-Objekt das wiederum einen Grundbaustein der AJAX-Technik darstellt. AJAX steht für "Asynchronous Javascript and XML" und bezeichnet ein Konzept der asynchronen Datenübertragung zwischen Browser und Server. Dieses ermöglicht es HTTP-Anfragen durchzuführen, während eine HTML-Seite angezeigt wird und die Seite zu verändern, ohne sie komplett neu zu laden.

Die Klasse ajaxjson.js bestimmt das Verhalten der Webseite und steuert den gesamten Prozess. Beim

Laden der index.html initialisiert sie mit der function init() den mouseover() und den mouseleave() der Anzeige. Zudem initialisiert sie die Start- und Stop-Buttons mit onclick.

Beim Klick auf den Start-Button wird das Mitscrollen des Anzeigebereichs sowie ein sich alle zweieinhalb Sekunden wiederholendes Intervall gestartet. Im Rahmen dieses Intervalls holt die sndReq()-Methode in der Datei jsonDeliverer.php die JSON-Daten ab und gibt diese an handleResponse() weiter. Die handleResponse()-Methode überprüft, ob sich die gesendeten Daten im richtigen Stadium befinden und ruft im positiven Fall die Methode drawBubble() auf. In der drawbubble()-Methode werden die JSON-Daten mit JSON.parse() geparkt und anschließend auf die benötigten Variablen aufgeteilt. Mithilfe dieser Variablen werden die Bubbles erzeugt. Die fertigen Bubbles werden in einen BubbleRoom geschrieben. Dieses Paket wird, nachdem der Anzeigebereich verlängert wurde, am rechten Ende der Anzeige eingefügt. Anschließend scrollt der Anzeigebereich zur neu eingefügten Bubble.

Das Mouseover des Anzeigebereichs verhindert das Mitscrollen des Anzeigebereichs, wenn sich der Cursor innerhalb des Anzeigebereichs befindet. Der Mouseleave startet das Mitscrollen des Anzeigebereichs wieder, sobald der Cursor den Anzeigebereich verlässt. Ein Scrollen findet beim Einfügen der nächsten Bubble statt. Ein Klick auf den Stop-Button beendet das Intervall, sodass keine weiteren Daten und somit auch keine weiteren Bubbles mehr geholt und erzeugt werden.

Abbildung 21 zeigt das Zusammenspiel der Methoden.

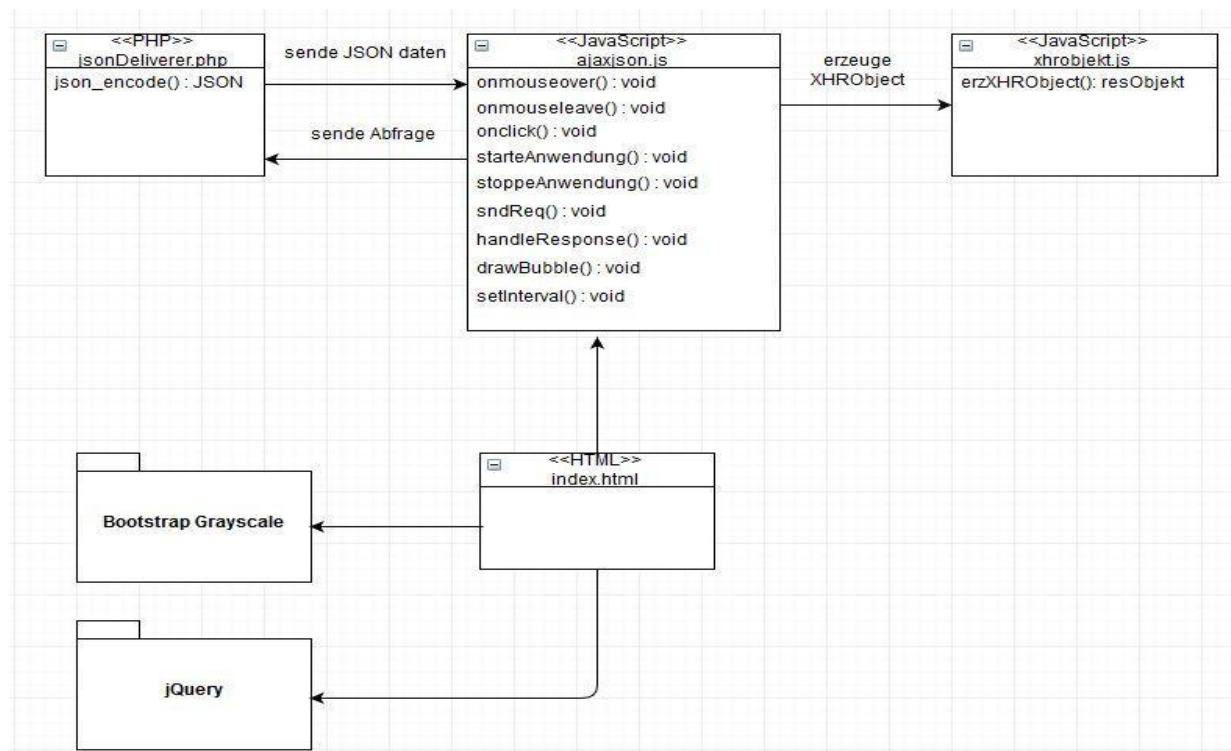


Abbildung 21: UML-Diagramm für den Bubble Prototyp

#### 4.3.4 Test

Parallel zur Implementierung wurden ständig Tests durchgeführt, welche die Funktionalität auch in unterschiedlichen Browsern sichergestellt haben. Getestet wurde in den neuesten Versionen der folgenden Browser: Google Chrome, Mozilla Firefox und Microsoft Edge.

Zudem wurde abschließend überprüft, ob alle in Kapitel 4.1.4 eingeführten User Stories mit dem Prototypen umgesetzt werden können. Dies ist beim "Bubbles"-Prototyp der Fall, weshalb der Test als erfolgreich bestanden angesehen wird.

#### 4.3.5 Fazit der Fallstudie

Es entstand ein gut umsetzbarer Prototyp, der eine intuitive Bedienbarkeit gewährleistet. Durch die farbige Unterscheidung wird ein schnelles Unterscheiden der Dokumententypen ermöglicht. Der gleichmäßige Fluss der einzelnen Bubbles sorgt dennoch dafür, dass eine Ablenkungsgefahr vom Gespräch minimiert wird. Mit Hilfe der Größe wird die Wichtigkeit auf eine intuitive Weise dargestellt. Durch die komplette Eigenentwicklung entstehen individuell anpassbare, uneingeschränkte Visualisierungsmöglichkeiten jenseits der klassischen Dateistruktur. Der zeitliche Verlauf wird ähnlich eines Videos dargestellt. Alle Dateitypen werden einheitlich visualisiert. Als Nachteil ist die fehlende Darstellung des Pfades anzubringen.

### 4 Projektfazit und Ausblick

Wir haben während der Entstehung des Projektes viel über Projekt-, Zeitmanagement und Cloud-Technologien gelernt. Auch den Softwareentwicklungsprozess an sich wurde als eine komplexe Sache dadurch trainiert. Auch haben wir gemerkt, wie wichtig es ist, sich sehr frühzeitig an die Bewältigung der Aufgaben zu machen, da das Testen und die Fehlersuche einen Großteil des Projekts in Anspruch nehmen. Es war nicht immer einfach, die Gruppenarbeit zu organisieren und alle Teammitglieder als Einzelnen zu berücksichtigen. Wir haben erkannt, dass eine Teamgröße von fünf Personen zu einer Herausforderung werden kann und somit als Obergrenze empfunden. Dennoch resultieren aus der Anzahl der Teammitglieder eine vorteilhafte Ansammlung an verschiedenen Kompetenzen und vereint Fähigkeiten von Programmierung, Organisation, Präsentation und Dokumentation. Die Vorgehensweise im Projekt entwickelte sich im Verlaufe der Durchführungszeit von agil-ähnlich zu einer agilen Softwareentwicklung. Seit der Zwischenpräsentation wurde ebenfalls in User Stories entwickelt. Die Vorteile dieser Vorgehensweise wurden durch die Konsequenzen der Email von Google (siehe Anhang) deutlich. Aufgrund der agilen Vorgehensweise, bei der die funktionalen Anforderungen aus Sicht des Anwenders beschrieben wurden, aber die technischen Lösungen erst

während der Entwicklung vollständig deutlich wurden, konnte schnell und mit wenig zusätzlichen Dokumentationsaufwand auf die veränderte Umweltbedingung reagiert werden. Diese geringen Anpassungen hätten sich, nach Einschätzung des Teams, vor allem in einem realen Unternehmens-Projekt in Bezug auf die Wirtschaftlichkeit positiv ausgewirkt. In Anbetracht der Wirtschaftlichkeit war es das Ziel, so wenig Kosten wie möglich zu verursachen. Der einzige Kostentreiber bei der Entwicklung der GUI war die reine Arbeitszeit der Studenten. Demnach sind keine Kosten für die Nutzung von Software, Tools, Frameworks etc. angefallen. Ebenfalls entstand durch die Unabhängigkeit zur Google-API der Anwendung eine höhere Investitionssicherheit bei der Weiterentwicklung des Gesamtsystems. Im Verlauf des Semesters wurden verschiedene Ansätze evaluiert, vorgestellt und die beste Möglichkeit gewählt. Dennoch kann, wie bereits erwähnt, auch selbst eine Entscheidung von externen Mächten eine Evaluierung nachträglich beeinflussen. In unserem Fall führte es zu einer offenen Umgebung in der mehr Freiheit in der Umsetzung besteht als in der Google Hangouts API. Zusätzlich entstand durch die Umstellung auf eine Webseite, eine Unabhängigkeit zur Videokonferenz Software. So können nun Hangouts, Facetime, Skype oder ähnliches verwendet werden.

Zuletzt kann gesagt werden, dass die Zusammenführung aller einzelnen Challenges zum Gesamtsystem nicht leicht werden wird und wir gespannt sind wie die konkrete Umsetzung sein wird. Das könnte eventuelle Anpassungen der Datenüberlieferung bedeuten, je nachdem wie die Bereitstellung durch das Team „Recommender“ definiert wurde.

## 5 Anhang

----- Forwarded message -----

From: **Platform Notifications** <[PlatformNotifications-noreply@google.com](mailto:PlatformNotifications-noreply@google.com)>

Date: 2017-01-06 21:52 GMT+01:00

Subject: Changes to Google+ Hangouts API

To: [markus.goetz91@gmail.com](mailto:markus.goetz91@gmail.com)

Hi,

At the Google Horizon event in September 2016, we previewed a new experience for Hangouts focused on meetings.

In order to streamline our efforts further, we will be retiring the [Google+ Hangouts API](#) that enables developers to build apps for the older version of Hangouts video calls. This API was originally intended to support social scenarios for consumer users as part of Google+, whereas Hangouts is now turning to focus on enterprise use cases.

Starting today, no new apps will be allowed, though existing apps will continue to work until April 25th, 2017. Users may continue to access existing apps through current entry points until that date.

We understand this will impact developers who have invested in our platform. We have carefully considered this change and believe that it allows us to give our users a more targeted Hangouts desktop video experience going forward.

We welcome your continued support.

Best regards,  
The Hangouts team

---

*Google Inc. 1600 Amphitheatre Parkway, Mountain View, CA 94043*

*You have received this mandatory email service announcement to update you about important changes to your Developer Console project.*