# Alarm Clock

Jonathan Hyams
Pascal Schmalz

28. Mai 2017

## Inhaltsverzeichnis

Zweck des Dokument Dieses Dokument soll dazu dienen, die Anforderungen an das Projekt AlarmClock zu definieren. Typische Leser sind die Entwickler und der Dozent im Requirements Engineering. Auch für den Projektbetreuer kann das Dokument von Interesse sein.

# 1 Kurzbeschreibung

Das Ziel des Projektes ist einen Ersatz zum Programm kAlarm zu entwickeln. Das Programm kAlarm erlaubt es dem User benutzerdefinierte Erinnerungen zu erstellen. Mittels Pop-Up Windows wird der User dann zur gegebenen Zeit daran erinnert. Im Gegensatz zu kAlarm soll das zu erstellende Produkt platformübergreifend verfügbar sein. Wie kAlarm soll dieses Produkt unter einer Open Source Lizenz entwickelt werden.

# 2 Projektziele

Unser Auftraggeber ist kein Unternehmen. Es werden somit auch keine Ziele innerhalb eines Unternehmens verfolgt. Ziele, welche ein Benutzer unsere Aplikation verfolgen könnte wären beispielsweise:

- An den Wäschetag erinnert weden.
- Die Wäsche rechtzeitig aus der Wäscheküche holen.
- Den Kuchen nicht im Backofen verbrennen lassen.
- Der Sekretärin zum Geburtstag Blumen schicken.
- usw.

# 3 Stakeholders

- Auftragsgeber: Prof. Claude Furrer
- Auftragsnehmer: Jonathan Hyams, Pascal Schmalz
- Benutzer: FOSS Communitiy

# 4 Systemabgrenzung

Es exisitert kein vordefiniertes System, welches mit unserem Programm interagiert. Das Programm soll auf einem Computer laufen, welcher Java 8.1 inklusive JavaFx installiert hat.

## 4.1 Geschäftsprozesse

Es existieren keine vor oder nachgelagerte Prozesse. Unsere Lösung lässt sich in beliebig viele Prozesse integrieren.

## 4.2 Systeme

Unser System beinhaltet einen Persistenz Layer um nach einem Neustart den vorherigen Zustand wiederherstellen zu können. Die Persistenz soll an einem zentralen Punkt sichergestellt werden, so dass mit wenig Aufwand eine andere Persistenzmethode, wie eine JPA Datenbank eingebunden werden kann.

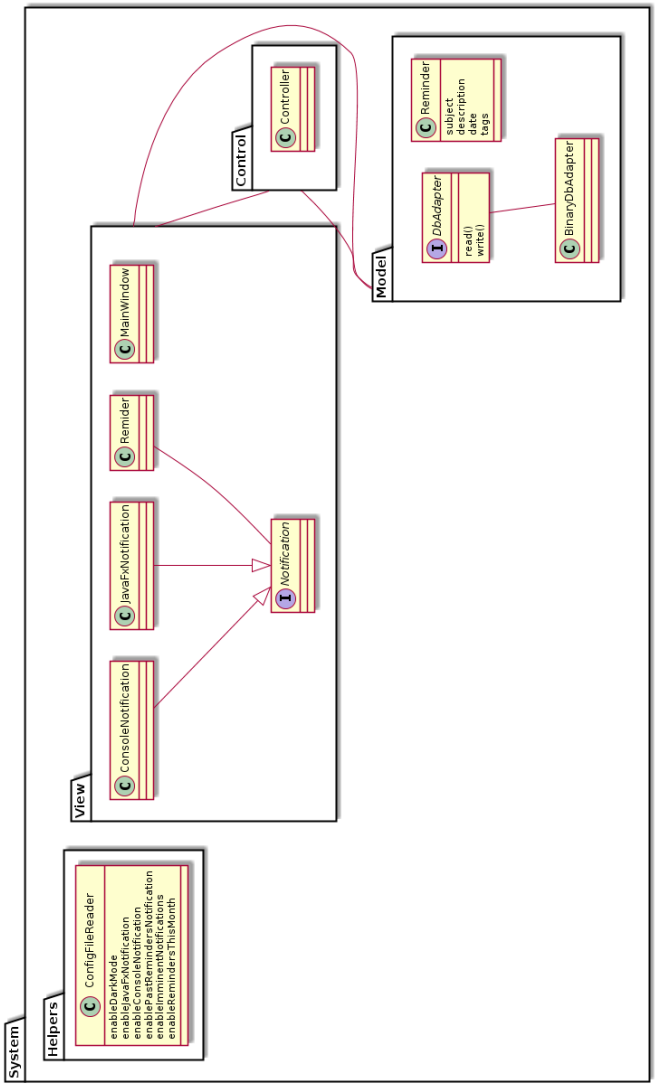


Abbildung 1: Systemübersicht und Abgrenzung

Die Systemübersicht und Abgrenzung Abb. 1 Seite 4 bedient sich der UML Notation, ohne die UML Spezifikation vollständig einzuhalten. Die Komponenten innerhalb des Kasten “System”, insbesondere ihre Beziehung zueinander, sind nicht Teil der Anforderungen, und somit nicht bindend. Sie veranschaulichen lediglich was noch innerhalb des Systems liegt und was ausserhalb angesiedelt ist. Abbildung:4

Falls die Funktionalität vom Auftraggeber verlangt wird, muss das Programm den Nutzer mit E-Mails, SMS und Systemnotifications erinnern. Dies bedingt, dass entsprechende Services / Server eingerichtet und über eine definierte Schnittstelle erreichbar sind, was nicht mehr Teil des Projektes ist.

### 4.3 Randbedingungen

Die Entwicklung soll mittels Java und JavaFX erfolgen. Das Programm soll Betriebssystem und Datenbank agnostisch sein. Das Projekt muss unter einer Opensource Lizenz veröffentlicht werden. Die Dokumentation muss in LaTeX erstellt werden.

## 5 Anforderungen

### 5.1 Basisfaktoren

- Die Schnittstelle zwischen dem Programmcode und der Datenbank soll so gestaltet sein, dass mit einem minimalen Aufwand eine beliebige gängige Datenabank genutzt werden kann.
- Die Bereitstellung eines JPA Drivers der Datenbank ist dabei nicht mehr im Scope des Projektes, sondern wird von den Datenbankherstellern oder von Dritten vorgenommen.
- Das System muss dem Benutzer die Möglichkeit bieten, in dem GUI ein Einzelevent zu erfassen.
- Das System kann anderen Programmen über eine API ermöglichen, Einzelevents zu erfassen.
- Das Löschen von Events soll wie oben beschrieben, ebenfalls möglich sein.
- Das System muss den Benutzer mittels einer Notification vorzeitig, am Tag des Events Zeit, auf ein Event hinweisen.
- Das System muss den Benutzer mittels einer Notification rechtzeitig, zur voreingestellten Zeit, auf ein Event hinweisen.
- Das Programm muss an einem Zentralen Ort konfiguriert werden.

## 5.2 Leistungsfaktoren

- Der Benutzer soll neue Kategorien hinzufügen, anpassen und löschen können.
- Events sollen durch den Benutzer in Kategorien eingeteilt werden können.
- Der Benutzer soll Events nach Kategorien gruppieren und filtern können.
- Das Programm soll vorkonfigurierte Dienste nutzen können um seine Notifications als SMS und Emails zu versenden.
- Das Programm soll vorkonfigurierte Dienste nutzen können um anstelle einer Notification ein Script laufen zu lassen.
- Die GUI soll Benutzerzentriert und Ergonomisch sein.

## 5.3 Begeisterungsfaktoren

- Das Programm soll vorkonfigurierte Dienste nutzen können um anstelle einer Notification ein IFTTT Event auszulösen.
- Das Programm soll zusätzlich zu den gängigen Computer-Betriebssystemen auch auf Android lauffähig sein.
- Das Programm soll auf den Betriebssystemen, welche eine Notificationinfrastruktur bereitstellen in der Lage sein die Notifications über diese vorzunehmen.(BSP KDE)
- Das Programm soll sich vom Look and Feel in die jeweilige Desktop Environment anpassen.
- Das System kann dem Benutzer die Möglichkeit bieten, in dem GUI wiederkehrende Events zu erfassen.
- Das Programm kann dem Benutzer die Möglichkeit bieten, die Konfiguration über die GUI vorzunehmen und im Configfile zu speichern.

## 5.4 Quellen und Vorgehen

Unsere wichtigste Quelle ist unser Auftraggeber Prof. Furrer. Um die Fragen, welche bei der Erstellung der Anforderungen aufgetaucht sind zu klären, werden wir uns mit ihm treffen. Besondere folgende Fragen gilt es abzuklären:

- Entspricht die bisherige Ausarbeitung der Vorstellung des Auftraggebers.
- Vollständigkeit der Anforderungen.
- Kategoriserung der Anforderungen.
- Priorisierung der Anforderungen.

- Was soll mit der Eventkategorisierung erreicht werden, welche weiteren Anforderungen ergeben sich daraus.
- Abgrenzung des Programmes gegenüber eines kAlarm, Kalenders und gegenüber des Cronjobs.

Da unser Auftraggeber als Professor der Informatik tätig ist, ist das Glossar für unseren Auftraggeber obsoloet. Der Vollständigkeihthalber wird es dennoch erstellt. Als weitere Diskusionsgrundlage sind wir daran neben diesem Dokument einen Prototyp des GUI zu erstellen. Da wir selber der potentiellen Nutzergruppe des Programmes angehören, könnten wir auch uns selber als Quelle nutzen, als Entwickler hat man jedoch oft einen anderen Blickwinkel als ein normaler Benutzer. Auch das Feedback von Komilitonen war uns ein anhaltspunkt. Im Sinne einer agilen Entwicklung wollen wir möglichst rasch präsentierbare Artefakte generieren, damit der Auftraggeber frühstmöglich Einfluss nehmen kann und Fehlentwicklungen vermieden werden. Treffen werden bei Bedarf mit dem Auftraggeber ausgemacht.

### 5.5 Technische Anforderungen

- Das Programm muss auf Linux, Windows und OSx lauffähig sein.
- Das Programm muss datenbankagnostisch sein.
- Das Programm muss auf dem Buildsystem kompiliert werden können.
- Das Programm muss auf dem Referenzsystem ohne weitere installationen kompiliert werden können.

### 5.6 Qualitätsanforderungen

- 80% der Benutzer sollen in der Lage sein, ohne vorgänige Schulung, bei der ersten Benutzung, folgende Aktionen jeweils innert 5 Minuten nach erfolgtem Programmstart, auszuführen:
  - Erstellen eines Events.
  - Löschen eines vorhandenen Events.
  - Editieren eines Events (Zeit, Datum, Notificationtext).
- 60% der Benutzer solle in der Lage sein, die Konfiguration des Programmes mittels ConfigFiles Vorzunhemen.

## 6 Glossar

## Abbildungsverzeichnis

# Tabellenverzeichnis

# 7 Anhang

## 7.1 Qualitätsaspekt Inhalt

Folgende Punke sind vorhanden:

- Vollständigkeit als Menge aller Anforderungen.
- Vollständig der einzelnen Anforderungen.
- Korrektheit der Anforderungen.
- Konsistezn
- Keine vorzeitigen Entwurfscentscheidungen.
- Die Überprüfbarkeit ist gegeben. Es werden jedoch nicht alle Anforderungen einzeln überprüft. Als Erfolgskriterien dienen verkürzten Userstotys mit genau angegebenen Metriken zur Überprüfbarkeit.
- Notwendigkeit

## 7.2 Qualitätsaspekt Dokumentation

Da wir nur einen Steakholder haben, ist es gegeben ,dass auch die relevanten Stakholder zugange sind. Laut Lehrbuch sollte es vermieden werden, dass die Anforderungen von den Erstellern überprüft werden. Da unser Betreuer sich aber verständlicherweise primär für die Artefakte des eigentlichen Entwicklungsprozesses interessierte, haben wir als Ersteller diese Dokumentation systematisch geprüft. Externe Überprüfer sind nicht angezeigt, da die Anforderungen keinem genügend hohen Qualitätsgrad genügen müssen.

Das Prinzip der Trennunb von Fhelersuche und Fehlerkorrektur haben wir nicht strikt eingehlten. Da die Fehlersucher und die Korrekture onehin dieselben Personen sind, war dies schwierig. Gerade bei kleinen Fehlern ist das korrigieren dieser rasch erledigt. Eine zusätzliche Dokumentation als Zwischenschritt hätte einen Mehraufwand bedeutet die angedeutete Gefahr beim Zusammenlegen der Schritte hielten wir in diesem Rahmen als überschaubar.

Die überprüfung aus unterschiedlichen Sichten konnten wir zu zweit durchspielen, indem eine Person aus der Sicht eines potentiellen Users argumentierte. Für den grossteil dieser Dokumentation haben wir uns auf Texte als Dokumentationsform beschränkt. Lediglich die Abbildung 1 auf Seite 4 haben wir schematisch dargestellt.

Entwicklungsartefakte als teil es Anforderungsanalyse spielen vorallem in der Agilen Entwicklung eine wichtige Rolle. Diese werden dort aber vorallem als teil der Entwicklung betrachtet. Dementsprechend haben wir hier auch keine solchen beigelgt, sondern

verweisen auf unse Github Code Repository, welches wir gerne für Sie freischalten, wenn Sie uns ihren Github Username angeben.

### 7.3 Qualitätsaspekt Abgestimmtheit der Anforderungen

Die Abstimmun der Anforderungen Unser Betreuer wollte uns im sinne einer Agilen Entwicklung eine grösstmögliche Freiheit lassen.Änderungswünsche an diesem Dokument wurden von ihm keine angebracht. Da es sich um keine Auftragsarbeit für ein komerzielles Unternehmen handelt, bildet dieses Dokument auch keine Vertragsgrundlage.

Obwohl wir Unterschiedliche Steakholder aufgelistet haben, gibt es in diesem Projekt einen klaren Entschiedungsträger, unser Betreuer. Prof. Furrer. Die Open Source Gemeinde selber ist so heterogen und verstreut, dass eine Abstimmung der Wünsche mit der gesamten Comunity onehin zum scheitern verurteilt wäre.

Ohne Zielkonflikte hatten wir auch keine Gelegenheit diese als Chance zu nutzen um neue Ideen zu entwickeln. Der Start der Entwicklungsphase dürfte laut dem Requirements Engineering Prozess erst erfolgen, nachdem die Abstimmung der Anforderungen erfolgt ist. Durch die Struktur des Unterrichts war es aber so gegeben, dass der Start der Entwicklungsphase mit dem Start der Anforderungsanalyse zusammen fielen.Für eine rein agile Vorgehensweise ist dies kein Problem. Will man das Requirements Engineering jedoch wie gefordert vor der Entwicklungsphase abschliessen, müsste man die Entwicklungsphase in einem späteren Semester starten.

Hier offenbart sich wohl ein Unterschied zwischen einem agilen Entwicklungsprozess und dem System Engineering.

### 7.4 Definition of Ready - Checklist

Wie bereits erwähnt, passt unser Projekt nicht optimal zur Vorgehensweise, welche wir im Requirements Engineering betrachtet haben. Wir haben mit unserem Betreuer auch keine vollumfängliche Definitino of Ready für die Anforderung ausgearbeitet. Die drei oben erwähnten Qualitätsaspekte haben wir wie beschrieben abgehandelt.

## 8 Versionskontrolle

Manuelle Version: 0.1.0

Automatische Versionierung: Last compiled: Sun May 28 08:27:31 CEST 2017
Git HEAD Version: 133