



## Projektplan

Silvan Adrian  
Fabian Binna

## 1 Änderungshistorie

<b>Datum</b>	<b>Version</b>	<b>Änderung</b>	<b>Autor</b>
<b>17.09.15</b>	1.00	Erstellung des Dokuments	Gruppe
<b>18.09.15</b>	1.01	Organisationsstruktur + Einführung + Projektübersicht	Silvan Adrian
<b>18.09.15</b>	1.02	Qualitätsmassnahmen	Fabian Binna
<b>20.09.15</b>	1.03	Text verbessert und zusätzliche Infos eingefügt	Silvan Adrian

## Inhaltsverzeichnis

<b>1</b>	<b>Änderungshistorie</b>	<b>2</b>
<b>2</b>	<b>Einführung</b>	<b>5</b>
2.1	Zweck . . . . .	5
2.2	Gültigkeitsbereich . . . . .	5
2.3	Referenzen . . . . .	5
<b>3</b>	<b>Projektübersicht</b>	<b>5</b>
3.1	Zweck und Ziel . . . . .	5
3.2	Primäre Features . . . . .	5
3.3	Erweiterte Features . . . . .	5
3.4	Lieferumfang . . . . .	5
3.5	Annahmen und Einschränkungen . . . . .	6
<b>4</b>	<b>Projektorganisation</b>	<b>6</b>
4.1	Organisationsstruktur . . . . .	6
4.2	Externe Schnittstellen . . . . .	6
<b>5</b>	<b>Managment Abläufe</b>	<b>7</b>
5.1	Kostenvoranschlag . . . . .	7
5.2	Sprints . . . . .	7
5.3	Besprechungen . . . . .	7
<b>6</b>	<b>Risikomanagement</b>	<b>7</b>
6.1	Risiken . . . . .	7
6.2	Umgang mit Risiken . . . . .	7
<b>7</b>	<b>Arbeitspakete</b>	<b>7</b>
<b>8</b>	<b>Infrastruktur</b>	<b>8</b>
8.1	Entwicklungsinfrastruktur . . . . .	8
8.2	Tools/Software . . . . .	8
8.3	Kommunikationsmittel . . . . .	8
<b>9</b>	<b>Qualitätsmassnahmen</b>	<b>8</b>
9.1	Dokumentation . . . . .	8
9.2	Projektmanagement . . . . .	8
9.3	Entwicklung . . . . .	9
9.3.1	Unit Testing / Test-Driven Development . . . . .	9
9.3.2	Code Review . . . . .	9
9.3.3	Code Style Guidelines . . . . .	9

9.4	Testen . . . . .	9
9.4.1	Komponententest . . . . .	9
9.5	Systemtest . . . . .	9

## 2 Einführung

### 2.1 Zweck

Dieses Dokument beschreibt die Planung der SA SDDC.

### 2.2 Gültigkeitsbereich

Dieses Dokument ist während des ganzen Projekts gültig und wird laufend aktualisiert.

### 2.3 Referenzen

**Generische API:** <https://libcloud.apache.org/>

**OpenStack API:** <http://developer.openstack.org/>

## 3 Projektübersicht

Es soll eine generische API erstellt oder eine bestehende API verwendet werden, um über ein Dashboard direkt Services zu abonnieren, dabei wird für jede Virtuelle Maschine Compute, Storage und Network benötigt. Es sollen verschiedene Hersteller eingebunden werden können, welche modular der generischen API hinzugefügt werden können (z.B.: als Plugins).

### 3.1 Zweck und Ziel

Gelerntes aus verschiedenen Modulen anwenden, sich mit der gestellten Aufgabenstellung auseinandersetzen und diese umsetzen.

### 3.2 Primäre Features

- Generische API
- Dashboard

### 3.3 Erweiterte Features

- TBD

### 3.4 Lieferumfang

- Source-Code
- Dokumentation (Projekt und Software)

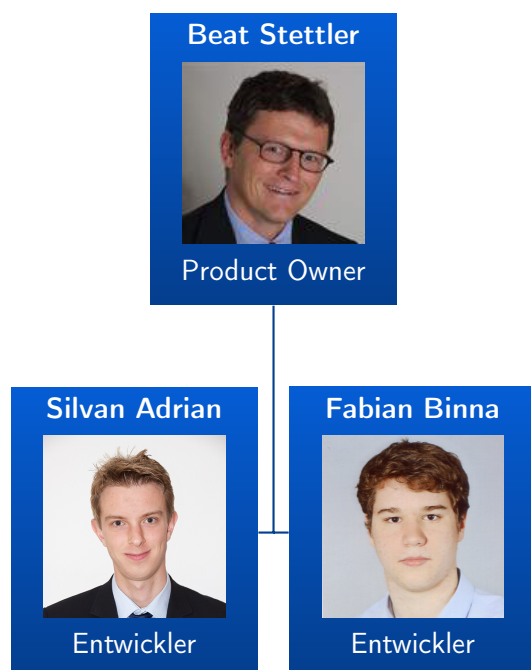
### 3.5 Annahmen und Einschränkungen

Die Applikation setzte sich aus einem Dashboard und einer generischen API zusammen, beides sollte dabei verteilt betrieben werden können. Die generische API soll dabei so viele Anbieter wie möglich abdecken.

## 4 Projektorganisation

Wir setzen beim Project auf Scrum, wodurch lediglich die Projektrollen Product Owner, Entwickler und Scrum-Master vorhanden sind. Es gibt keine Unterscheidung zwischen speziellen Fachgebieten und das Team als ganzes ist gefragt.

### 4.1 Organisationsstruktur



### 4.2 Externe Schnittstellen

#### Projektbetreuer:

Beat Stettler ([Beat.Stettler@ins.hsr.ch](mailto:Beat.Stettler@ins.hsr.ch))

Urs Baumann ([Urs.Baumann@ins.hsr.ch](mailto:Urs.Baumann@ins.hsr.ch))

## 5 Managment Abläufe

### 5.1 Kostenvoranschlag

Gemäss Vorgaben wird mit 240 Stunden gesamt gerechnet + die einberechneten Risiken von 41 Stunden.

**Gesamtaufwand:** 281 Stunden

**Risikoaufwand:** 41 Stunden

**Aufwand pro Woche:** ca.: 20 Stunden

**Aufwand pro Woche pro Teammitglied:** ca.: 10 Stunden

### 5.2 Sprints

Sprint	Beschreibung	Beginn	Ende
<b>Sprint 1</b>	Erste Version der API zum Testen	21.09.15	28.09.15
<b>Sprint 2</b>	tbd	tbd	tbd

### 5.3 Besprechungen

Daily Meetings (ca. 10 - 15min) werden während der Pause oder über Skype stattfinden, um sich gegenseitig über Probleme, abgeschlossene Arbeitspakete etc. zu informieren. Nach jedem Abschluss eines Sprints wird ein Sprintreview geplant und durchgeführt, um zu prüfen ob alles gemäss Plan erledigt wurde + die aktuellste Version der Software released.

## 6 Risikomanagement

### 6.1 Risiken

Risiken werden im Dokument TechnischeRisiken-JBomberman.pdf beschrieben

### 6.2 Umgang mit Risiken

Für die Risiken werden Reserven eingeplant. Die Reserven werden direkt in die einzelnen Tickets eingerechnet. Falls Risiken eintreffen werden diese sofort an einem der Daily-Meetings kommuniziert und mögliche Lösungen evaluiert.

## 7 Arbeitspakete

Die Arbeitspakete werden in Open Project erstellt und gepflegt. Lesender Zugriff ist anonym möglich, schreibender nur eingeloggt (Projekt ist öffentlich).

Link zur Open Project Instanz : <http://sddc.silvn.com>

## 8 Infrastruktur

### 8.1 Entwicklungsinfrastruktur

Name	Hardware	Betriebssystem	IDE
Silvan Adrian	MacBook Pro	OSX 10.10.5	N/A
Fabian Binna	Lenovo T430s	Windows 10	N/A

### 8.2 Tools/Software

- **BuildServer:** Travis-CI
- **Konfigurationsmanagement:** GIT
- **Notifications:** Slack

### 8.3 Kommunikationsmittel

- E-Mail
- Skype
- Open Project (Kommentare)
- GitHub (Issues, Kommentare)
- Whatsapp
- Slack

## 9 Qualitätsmassnahmen

### 9.1 Dokumentation

Die Dokumentation befindet sich auf einem privaten GitHub Repository. Die Texte werden in LaTeX geschrieben. Die Dokumente werden versioniert.

[https://github.com/silvanadrian/SDDC\\_Doku.git](https://github.com/silvanadrian/SDDC_Doku.git)

### 9.2 Projektmanagement

Für das Projektmanagement wird OpenProject verwendet.

[sddc.silvn.com](http://sddc.silvn.com)



## **9.3 Entwicklung**

### **9.3.1 Unit Testing / Test-Driven Development**

Die Unit Tests kommen in einen separaten Ordner "Test". Es wird eine möglichst hohe Code Coverage angestrebt. Die Code Coverage wird mit einem Tool (z.B. eclEmma) sichergestellt.

Die Klassen werden mit Hilfe von Test-Driven Development implementiert.

### **9.3.2 Code Review**

Nach jedem Sprint oder bei Abschluss grosser Arbeitspakete wird ein Code Review durchgeführt.

Die Review Protokolle werden auf dem GitHub Repository SDDC-Doku abgelegt.

### **9.3.3 Code Style Guidelines**

Editor Standard

## **9.4 Testen**

### **9.4.1 Komponententest**

## **9.5 Systemtest**