|  |
| --- |
| Elmer Lukas, Heidt Christina, Steiner Diego, Treichler Delia, Waltenspül Remo  11. April 2011 |

|  |
| --- |
| SE2 Projekt MRT |
| Software Architektur |
|  |

****

# Dokumentinformationen

## Änderungsgeschichte

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Version | Änderung | Autor |
| 28.03.2011 | 1.0 | Erste Version des Dokuments | HC |

## Inhaltsverzeichnis

[1 Dokumentinformationen 1](#_Toc290303354)

[1.1 Änderungsgeschichte 1](#_Toc290303355)

[1.2 Inhaltsverzeichnis 1](#_Toc290303356)

[1.3 Abbildungsverzeichnis 1](#_Toc290303357)

[2 Einführung 3](#_Toc290303358)

[2.1 Zweck 3](#_Toc290303359)

[2.2 Gültigkeitsbereich 3](#_Toc290303360)

[2.3 Definitionen und Abkürzungen 3](#_Toc290303361)

[2.4 Referenzen 3](#_Toc290303362)

[3 Software Architektur 3](#_Toc290303363)

[3.1 Architektonische Ziele & Einschränkungen 3](#_Toc290303364)

[3.1.1 Ziele 3](#_Toc290303365)

[3.1.2 Einschränkungen 3](#_Toc290303366)

[3.2 Architekturübersicht 4](#_Toc290303367)

[3.3 Systemstruktur 4](#_Toc290303368)

[3.3.1 Physische Sicht 4](#_Toc290303369)

[3.3.2 Logische Sicht Rails 5](#_Toc290303370)

[3.3.3 Logische Sicht Android 6](#_Toc290303371)

[3.4 Design Pakete Rails 7](#_Toc290303372)

[3.5 Design Pakete Android 7](#_Toc290303373)

[3.5.1 Package GUI 7](#_Toc290303374)

[3.5.2 Package Problem Domain 7](#_Toc290303375)

[3.5.3 Package Datenhaltung 8](#_Toc290303376)

[4 Prozesse und Threads 8](#_Toc290303377)

[5 Datenspeicherung 8](#_Toc290303378)

[6 Grössen und Leistung 8](#_Toc290303379)

## Abbildungsverzeichnis

[Abbildung 1 - Deployment Diagram MRT 3](#_Toc290303339)

[Abbildung 2 - Architektur von Ruby on Rails 4](#_Toc290303340)

[Abbildung 3 - Ablauf eines Requests 5](#_Toc290303341)

[Abbildung 4 - Architektur in Ruby on Rails 6](#_Toc290303342)

# Einführung

## Zweck

Dieses Dokument beschreibt die Software Architektur für das Projekt MRT (Mobile Reporting Tool).

## Gültigkeitsbereich

Dieses Dokument ist während der gesamten Projektdauer gültig (21.02 bis 03.06.2011).

## Definitionen und Abkürzungen

Die Definitionen und Abkürzungen befinden sich in der ausgelagerten Datei doc/01\_Projektplan/glossar.docx.

## Referenzen

* doc/03\_Anforderungsspezifikationen/\*
* doc/templates/template.dotx
* doc/templates/java\_formatting.xml
* doc/templates/ruby\_formatting\_settings.zip
* doc/media/logo.png

# Software Architektur

## Architektonische Ziele & Einschränkungen

<Beschreibt die Softwareanforderungen und Objekte, welche einen Einfluss auf die Architektur haben [Bspl: Safety, Security, Privacy, Distribution, …] Beinhaltet auch eine Beschreibung von Design und Implementationsstrategie, Teamstruktur, Entwicklungstools, Zeitplan, etc…>

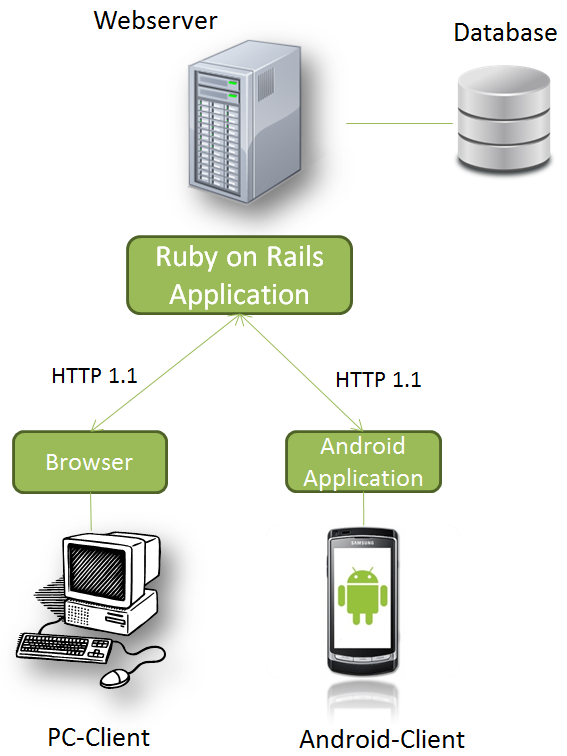
### Ziele

* Es soll möglichst einfach möglich sein, den Android Client durch einen anderen Client (z.B. iPhone, Windows7 Phone) zu ersetzen. Deshalb darf die Architektur keine Android-spezifischen Konzepte enthalten.
* Die Benutzeroberfläche auf dem Android Client soll sehr einfach und übersichtlich gestaltet werden. Deshalb stellt der Android Client nur die wichtigsten Funktionalitäten zur Verfügung.
* Mehrere Mitarbeiter müssen gleichzeitig Stundeneinträge machen können. Darum müssen die einzelnen Einträge voneinander getrennt werden.
* Die Daten müssen konsistent bleiben, auch wenn die Datenverbindung zwischen Client und Server unterbrochen wird. Das soll gezielt durch geplante Abläufe erreicht werden.
* Die Schnittstellen zwischen Client und Server sollen genau definiert werden, damit die zwei Teile unabhängig voneinander entwickelt werden könnten.

### Einschränkungen

* Der Android Client wird über Touchscreen bedient. Deshalb muss darauf geachtet werden, dass z.B. die Buttons gross genug sind.
* Da für die Implementation nur wenig Zeit zur Verfügung steht und es in diesem Projekt nicht um grafisches Design geht, wird das grafische Design nicht ausgereift sein.

## Architekturübersicht



## Systemstruktur

### Physische Sicht



Abbildung 1 - Deployment Diagram MRT

Da der Server die Rails Applikation hostet, wird er als Server Execution Node dargestellt. Der Client verbindet sich mit dem Server über http. Deshalb wird der Client als Client Execution Node dargestellt.

Zu beachten ist, dass auf dem Server und dem Client je eine andere Applikation läuft. Auf dem Server ist dies eine Rails Applikation (rails\_app), auf dem Client eine Android Applikation (mrt.apk).

Server und Client kommunizieren über HTTP/HTTPS. Natürlich läuft HTTP/HTTPS über TCP/IP. Eine klare Abgrenzung ist hier wichtig, demzufolge ist für das Projekt die Schicht HTTP / HTTPS tiefste Schicht.

### Logische Sicht Rails

Die Architektur der Rails Applikation ist zu einem grossen Teil vorgegeben.

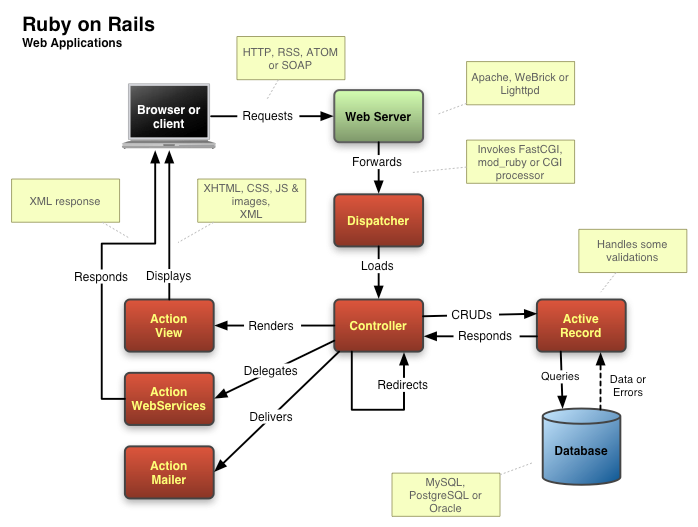


Abbildung - Architektur von Ruby on Rails

Abbildung 3 - Ablauf eines Requests

### Logische Sicht Android

## Design Pakete Rails

Die Architektur in Ruby on Rails ist grundsätzlich bereits vorgegeben. Deshalb ist dieses Kapitel sehr kurz gehalten. Die Architektur in Ruby on Rails sieht folgendermassen aus:

Abbildung 4 - Architektur in Ruby on Rails

## Design Pakete Android

### Package GUI

#### Beschreibung des Package

#### Diagramme

#### Schnittstellen

#### Operationen

##### Interne Operation 1

##### Interne Operation 2

### Package Problem Domain

#### Beschreibung des Package

#### Diagramme

#### Schnittstellen

#### Operationen

##### Interne Operation 1

##### Interne Operation 2

### Package Datenhaltung

#### Beschreibung des Package

#### Diagramme

#### Schnittstellen

#### Operationen

##### Interne Operation 1

##### Interne Operation 2

# Prozesse und Threads

<Wenn mehrere Prozesse oder Threads eingesetzt werden wird hier beschrieben, wie diese ablaufen, miteinander funktionieren, Daten austauschen, sich synchronisieren, etc...>

# Datenspeicherung

<Beschreibung mit Diagramm der Datenspeicherung [Data Model]. (zum Beispiel: Datenbank)>

# Grössen und Leistung

<Einschränkungen der Applikation bezüglich Speicher, Leistung, etc…. (zum Beispiel: Verwaltung unterstützt maximal 20'000 Einträge)>