Template

2019-01-04

Über arc42

arc42, das Template zur Dokumentation von Software- und Systemarchitekturen.

Erstellt von Dr. Gernot Starke, Dr. Peter Hruschka und Mitwirkenden.

Template Revision: 7.0 DE (asciidoc-based), January 2017

> **Note**
>
> Diese Version des Templates enthält Hilfen und Erläuterungen. Sie dient der Einarbeitung in arc42 sowie dem Verständnis der Konzepte. Für die Dokumentation eigener System verwenden Sie besser die *plain* Version.

Einführung und Ziele

Beschreibt die wesentlichen Anforderungen und treibenden Kräfte, die bei der Umsetzung der Softwarearchitektur und Entwicklung des Systems berücksichtigt werden müssen.

Dazu gehören:

- zugrunde liegende Geschäftsziele,
- wesentliche Aufgabenstellungen und
- essenzielle fachliche Anforderungen an das System sowie
- Qualitätsziele für die Architektur und
- relevante Stakeholder und deren Erwartungshaltung.

Aufgabenstellung

Inhalt.

Kurzbeschreibung der fachlichen Aufgabenstellung, treibenden Kräfte, Extrakt (oder Abstract) der Anforderungen. Verweis auf (hoffentlich vorliegende) Anforderungsdokumente (mit Versionsbezeichnungen und Ablageorten).

Motivation.

Aus Sicht der späteren Nutzung ist die Unterstützung einer fachlichen Aufgabe oder Verbesserung der Qualität der eigentliche Beweggrund, ein neues System zu schaffen oder ein bestehendes zu modifizieren.

Form.

Kurze textuelle Beschreibung, eventuell in tabellarischer Use-Case Form. Sofern vorhanden, sollte die Aufgabenstellung Verweise auf die entsprechenden Anforderungsdokumente enthalten.

Halten Sie diese Auszüge so knapp wie möglich und wägen Sie Lesbarkeit und Redundanzfreiheit gegeneinander ab.

Qualitätsziele

Inhalt.

Die Top-3 bis Top-5 der Qualitätsziele für die Architektur, deren Erfüllung oder Einhaltung den maßgeblichen Stakeholdern besonders wichtig sind. Gemeint sind hier wirklich Qualitätsziele, die nicht unbedingt mit den Zielen des Projekts übereinstimmen. Beachten Sie den Unterschied.

Motivation.

Weil Qualitätsziele grundlegende Architekturentscheidungen oft maßgeblich beeinflussen, sollten Sie die für Ihre Stakeholder relevanten Qualitätsziele kennen, möglichst konkret und operationalisierbar.

Form.

Tabellarische Darstellung der Qualitätsziele mit möglichst konkreten Szenarien, geordnet nach Prioritäten.

Stakeholder

Rolle	Kontakt	Erwartungshaltung
CEO	*Dr. Michael Moore*	*Keine Vollautomatisierung, Segnet alle Boni manuell ab*
HR Senior Consultant	*Chantal Banks*	*Weniger manuelle Schritte, Keine Papierarbeit mehr*
IT-Admin	*Tom Foster*	*Keine Berührungspunkte mit dem gesamten Prozess*

Randbedingungen

Inhalt.

Beschreibung der zu nutzenden externen Systemen

Was ist die prinzipielle Aufgabe der Anwendung OpenCRX?

- CRM System
- Account management
- Product and Price Management
- Sales Pipeline
- Issue tracking
- Groupware
- mail, contact and calendar management

Was ist die prinzipielle Aufgabe der Anwendung OrangeHRM?

- an Resource Management
- mance Management
- Administration and Personal Information Management
- Recruitment etc.
- Employee Self Service (Time Tracking)

Welche grundlegenden Funktionen besitzen diese Anwendungen? OrangeHRM bietet Funktionalitäten zum Verwalten des Personals und alles was dazu gehört:

- Mitarbeiterverwaltung inkl. Mitarbeiter Self Service
- Dashboards
- Mitarbeiter Training
- Reiseplanung
- Dokumentenmanager
- OpenCRX bietet Funktionalitäten zur Verwaltung von Kunden
- Kundensupport
- Customer Success
- Marketing
- Analytics
- Sales Management

Welche Geschäftsobjekte werden in OpenCRX bzw. OrangeHRM verwaltet?

- Kunden
- Mitarbeiter

Welche Art Schnittstellen bieten OpenCRX bzw. OrangeHRM an?

- OpenCRX:
  - REST API
  - AirSync ActiveSync
  - User Interface (WebUI)
- OrangeHRM:
  - REST API
  - Mobile APP
  - User Interface (WebUI)

Kontextabgrenzung

Inhalt.

Die Kontextabgrenzung grenzt das System von allen Kommunikationsbeziehungen (Nachbarsystemen und Benutzerrollen) ab. Sie legt damit die externen Schnittstellen fest.

Differenzieren Sie fachliche (fachliche Ein- und Ausgaben) und technische Kontexte (Kanäle, Protokolle, Hardware), falls nötig.

Motivation.

Die fachlichen und technischen Schnittstellen zur Kommunikation gehören zu den kritischsten Aspekten eines Systems. Stellen Sie sicher, dass Sie diese komplett verstanden haben.

Form.

Verschiedene Optionen:

- Diverse Kontextdiagramme
- Listen von Kommunikationsbeziehungen mit deren Schnittstellen

Fachlicher Kontext

Inhalt.

Der Prozess kann von einem HR Senior innerhalb von der Camunda Plattform gestartet werden und beim Start wir direkt der entsprechende Mitarbeiter in das angebotene Formular-Feld eingetragen.

Weiterhin wird dem CEO ein entsprechendes Formular, in Camunda, mit einer Aufstellung der berechneten Boni zur Verfügung gestellt, in dem er Korrekturen an den einzelnen Boni vornehmen kann und entsprechend speichern, damit sie im fortlaufenden Prozess berücksichtigt werden.

Technischer Kontext

Die Camunda-Engine nutzt die bereitgestellte RESTful-Schnittstelle des Data-Colletors, um entsprechende Informationen über den zu bearbeitenden Salesman auszutauschen. Hierbei wird das sogenannte ClientInfoDTO ausgetauscht.

Data Collector - Objekte

```
data class ClientInfoDTO(
    var salesmanName: String = "",
    var salesInfos: ArrayList<SalesInfoDTO> = ArrayList()
)

data class SalesInfoDTO(
    var productName: String = "",
    var clientName: String = "",
    var clientRanking: Int = 0,
    var quantity: Double = 0.0,
    var bonus: Int = 0
```

```
)
```

Der Data-Collector kommuniziert entsprechend über RESTful Schnittstelle mit den Systemen OrangeHRM und OpenCRX, um die entsprechenden Daten zu erfassen.

OrangeHRM - Objekte

```
data class Employee(
    val firstName: String,
    val lastName: String,
    val employeeId: String,
    val jobTitle: String
)

data class Organization(
    val id: String,
    val name: String,
    val email: String?,
    val country: String,
    val numberOfEmployees: String
)
```

OpenCRX - Objekte

```
data class Account(
    val firstName: String? = "",
    val lastName: String? = "",
    val fullName: String? = "",
    val familyStatus: Int = 0,
    val organization: String? = "",
    val jobTitle: String? = "",
    val gender: Int = 0,
    val preferredSpokenLanguage: Int = 0,
    val accountRating: Int = 0,
    val industry: String? = "",
    val annualIncomeCurrency: Int = 0,
    @JsonProperty("@href") val accountUrl: String? = ""
)
```

Abbildung 1 zeigt die entsprechenden Kommunikationskanäle via HTTP zwischen den technischen Systemen.

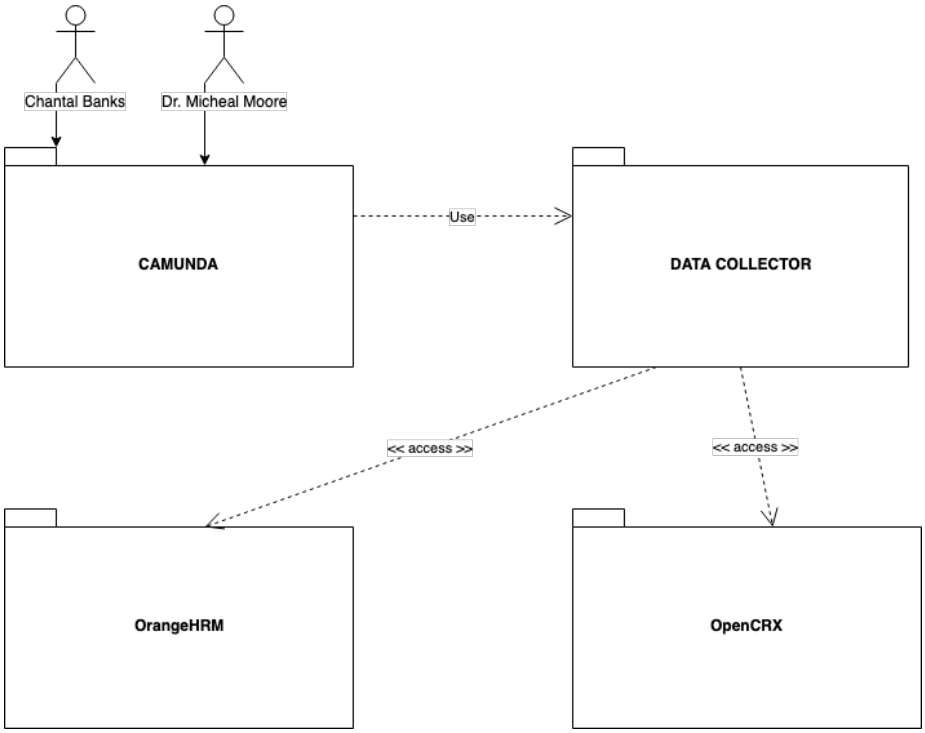


Figure 1: Kontext des Systems

Lösungsstrategie

Inhalt.

Kurzer Überblick über die grundlegenden Entscheidungen und Lösungsansätze, die Entwurf und Implementierung des Systems prägen. Hierzu gehören:

- Technologieentscheidungen
- Entscheidungen über die Top-Level-Zerlegung des Systems, beispielsweise die Verwendung gesamthaft prägender Entwurfs- oder Architekturmuster,
- Entscheidungen zur Erreichung der wichtigsten Qualitätsanforderungen sowie
- relevante organisatorische Entscheidungen, beispielsweise für bestimmte Entwicklungsprozesse oder Delegation bestimmter Aufgaben an andere Stakeholder.

Motivation.

Diese wichtigen Entscheidungen bilden wesentliche „Eckpfeiler“ der Architektur. Von ihnen hängen viele weitere Entscheidungen oder Implementierungsregeln ab.

Form.

Fassen Sie die zentralen Entwurfsentscheidungen **kurz** zusammen. Motivieren Sie, ausgehend von Aufgabenstellung, Qualitätszielen und Randbedingungen, was Sie entschieden haben und warum Sie so entschieden haben. Vermeiden Sie redundante Beschreibungen und verweisen Sie eher auf weitere Ausführungen in Folgeabschnitten.

Bausteinsicht

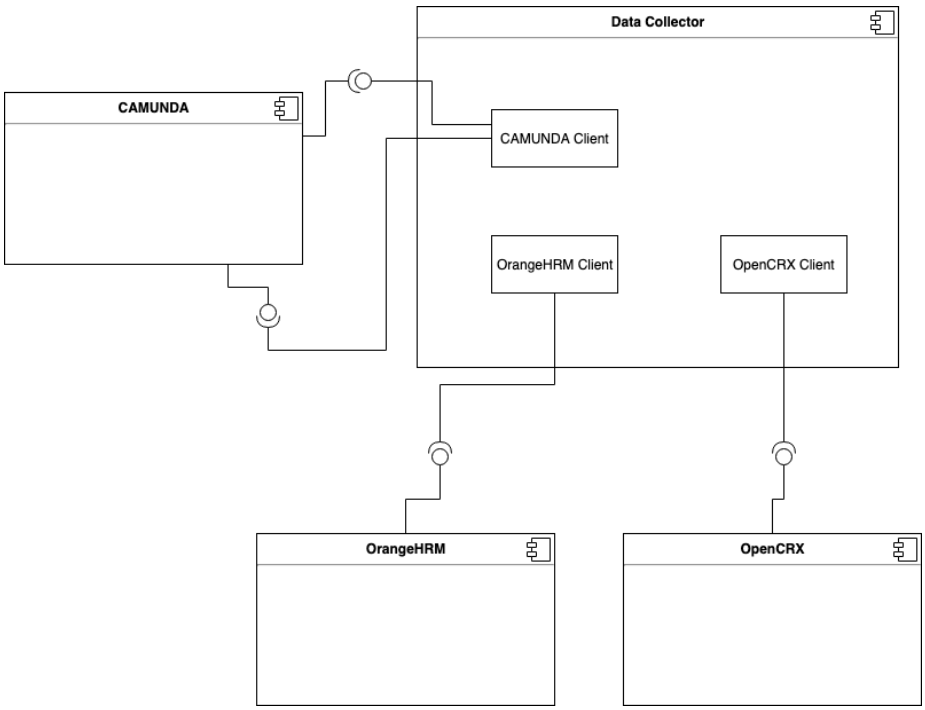


Figure 2: Bausteinsicht

Laufzeitsicht

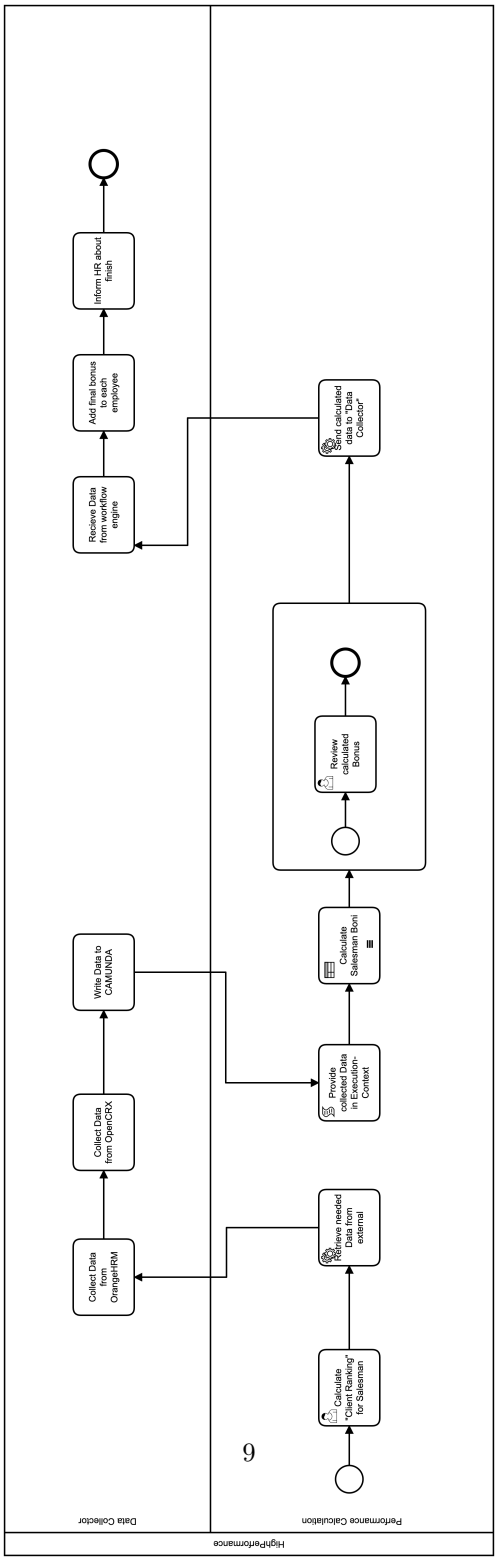


Figure 3: Laufzeitsicht

Verteilungssicht

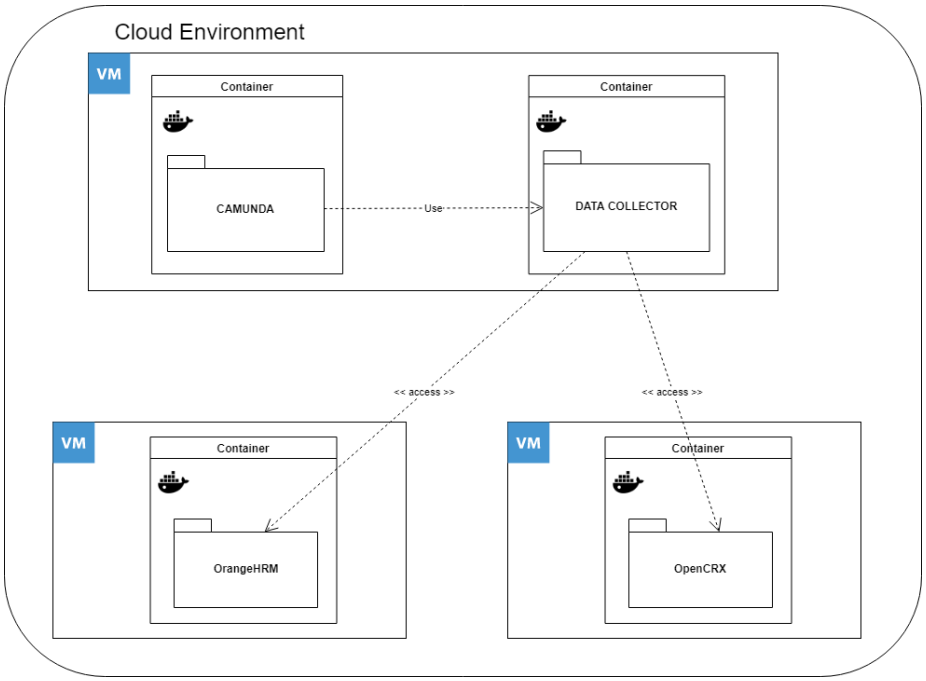


Figure 4: Verteilungssicht

Querschnittliche Konzepte

Inhalt.

Dieser Abschnitt beschreibt übergreifende, prinzipielle Regelungen und Lösungsansätze, die an mehreren Stellen (=*querschittlich*) relevant sind.

Solche Konzepte betreffen oft mehrere Bausteine. Dazu können vielerlei Themen gehören, beispielsweise:

- fachliche Modelle,
- eingesetzte Architektur- oder Entwurfsmuster,
- Regeln für den konkreten Einsatz von Technologien,
-
- Implementierungsregeln

Motivation.

Konzepte bilden die Grundlage für *konzeptionelle Integrität* (Konsistenz, Homogenität) der Architektur und damit eine wesentliche Grundlage für die innere Qualität Ihrer Systeme.

Manche dieser Themen lassen sich nur schwer als Baustein in der Architektur unterbringen (z.B. das Thema „Sicherheit“). Hier ist der Platz im Template, wo Sie derartige Themen geschlossen behandeln können.

Form.

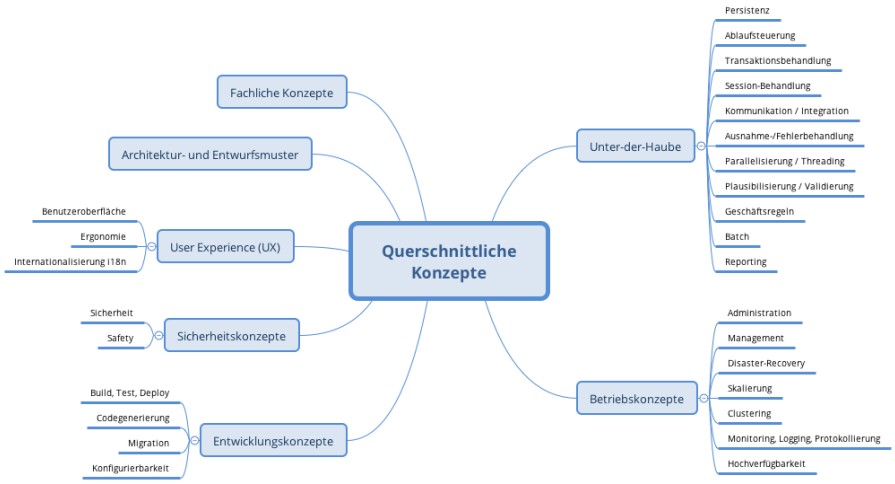
Kann vielfältig sein:

- Konzeptpapiere mit beliebiger Gliederung,
- übergreifende Modelle/Szenarien mit Notationen, die Sie auch in den Architektursichten nutzen,
- beispielhafte Implementierung speziell für technische Konzepte,
- Verweise auf „übliche“ Nutzung von Standard-Frameworks (beispielsweise die Nutzung von Hibernate als Object/Relational Mapper).

Struktur.

Eine mögliche (nicht aber notwendige!) Untergliederung dieses Abschnittes könnte wie folgt aussehen (wobei die Zuordnung von Themen zu den Gruppen nicht immer eindeutig ist):

- Fachliche Konzepte
- User Experience (UX)
- Sicherheitskonzepte (Safety und Security)
- Architektur- und Entwurfsmuster
- Unter-der-Haube
- Entwicklungskonzepte
- Betriebskonzepte



<Konzept 1>

<Erklärung>

<Konzept 2>

<Erklärung>

...

<Konzept n>

<Erklärung>

Entwurfsentscheidungen

Zur prototypischen Implementierung wurden verschiedene Sprachen und Technologien verwendet, welche in diesem Abschnitt erläutert werden.

Wie vorgegeben wurde zur modellierung des Workflows das Tool Camunda eingesetzt. Mithilfe von Camunda wurde der gesamte Prozess modeliert und ausgeführt. Das Camunda Cockpit und die Tasklist eignen sich hervorragend zum starten und verwalten der Prozesse. Da im verlaufe des Prozesses auch mit Fremdsystemen kommuniziert werden muss wurde ein Microservice in der Sprache Kotlin entwickelt. Dieser Service übernimmt die Kommunikation mit

OrangeHRM und OpenCRX über die dokumentierten REST API's. Um eine konsistene Architektur bereitzustellen bietet der Microservce ebenfalls REST-Schnittstellen an, welche aus der JavaDelegate Klasse aus Camunda heraus aufgerufen wird. Alle Anwendungen werden in eigenen, weitesgehend isolierten, Docker Containern betrieben.

Risiken und technische Schulden

Inhalt.

Eine nach Prioritäten geordnete Liste der erkannten Architekturrisiken und/oder technischen Schulden.

> Risikomanagement ist Projektmanagement für Erwachsene.
>
> — Tim Lister Atlantic Systems Guild

Unter diesem Motto sollten Sie Architekturrisiken und/oder technische Schulden gezielt ermitteln, bewerten und Ihren Management-Stakeholdern (z.B. Projektleitung, Product-Owner) transparent machen.

Form.

Liste oder Tabelle von Risiken und/oder technischen Schulden, eventuell mit vorgeschlagenen Maßnahmen zur Risikovermeidung, Risikominimierung oder dem Abbau der technischen Schulden.

Glossar

Inhalt.

Die wesentlichen fachlichen und technischen Begriffe, die Stakeholder im Zusammenhang mit dem System verwenden.

Nutzen Sie das Glossar ebenfalls als Übersetzungsreferenz, falls Sie in mehrsprachigen Teams arbeiten.

Motivation.

Sie sollten relevante Begriffe klar definieren, so dass alle Beteiligten

1. diese Begriffe identisch verstehen, und
2. vermeiden, mehrere Begriffe für die gleiche Sache zu haben.

- Zweispaltige Tabelle mit <Begriff> und <Definition>
- Eventuell weitere Spalten mit Übersetzungen, falls notwendig.

Begriff	Definition
<Begriff-1>	*<Definition-1>*
<Begriff-2	*<Definition-2>*