**System Design**



|  |  |
| --- | --- |
| *Version:* | *1.0* |
| *Datum:* | 05.11.2016 |
| *Autoren:* | *Jana Dreiling, Philip Waritschlager, Marius Koch, Tina Brunner* |

**Änderungsverzeichnis:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Datum** | **Autor(en)** | **Änderung** |
| 1.0 | 16.11.16 | Alle | Setup |
|  |  |  |  |

Inhaltsverzeichnis

1 Einleitung 3

1.1 Ziel dieses Dokuments 3

1.2 Referenzierte Dokumente 3

2 Überblick 4

3 Funktionales Modell 5

4 Dialoglandkarte 6

5 IT Architektur 7

5.1 Kontextabgrenzung 8

5.2 Bausteinsicht 8

5.3 Laufzeitsicht 8

5.4 Verteilungssicht 9

6 Schnittstellen und Integration 10

6.1 Schnittstellenspezifikation 10

6.2 Schnittstellenformat 10

7 Datenmodell 11

7.1 Datenspeicherungskonzept 11

7.2 Physikalische Datenbankstruktur 11

7.3 Datenmodell 11

8 Testspezifikation 12

8.1 Teststrategie 12

8.2 Testfälle 12

9 Projektmanagement 13

9.1 Projektplan 13

9.2 Aufgabenverteilung 13

10 Bibliographie 14

11 Glossar 14

# Einleitung

## Ziel dieses Dokuments

In diesem System Design soll

Das Dokument ist strukturiert in X Kapitel ….

In Kapitel x geht es um …

## Referenzierte Dokumente

Falls es weitere Dokumentreferenzen gibt (z.B. Schnittstellenkontrakte, Dialog-Konzepte, etc.) dann dürfen diese hier referenziert werden.

# Überblick

Im vorliegenden Dokument wird eine Verwaltungssoftware für Kühlschränke beschrieben. Außerdem wird das Gegenstück, die Bestellungsannahme eines Supermarktes, entwickelt.

Mit dem "*Freezer Management*" kann jeder Kühlschrankbesitzer den Bestand seines Gerätes automatisch überwachen und auffüllen lassen.

Jedes Lebensmittel, das der Nutzer seinem Kühlschrank hinzufügt, wird von der Software erfasst. Es ist also möglich, den Bestand jederzeit online im Webbrowser anzeigen zu lassen. Hierzu wird davon ausgegangen, dass neue Produkte beim Hinzufügen oder Entnehmen automatisch gescannt werden und das Softwaresystem eine entsprechende Benachrichtigung enthält.

Die Nutzung des Kühlschranks ist in keinster Weise beeinträchtigt. Es kommen jedoch neue Features hinzu:

* Der Nutzer kann das Sortiment des Supermarktes einsehen. Für jede Produktart kann eine Mindestbestandsmenge eingestellt werden. Wird diese unterschritten, so wird das Produkt auf einem Einkaufszettel vorgemerkt.
* Für jede Produktart kann der Nutzer eine Wunschmenge einstellen.
* In einstellbaren, regelmäßigen Intervallen generiert die Software hieraus eine Bestellung, die vom Nutzer noch manuell erweiterbar oder veränderbar ist. Die Menge der Produkte orentiert sich an den definierten Wunschmengen.
* Die Bestellung wird entweder automatisch oder nach Nutzeraufforderung an den Supermarkt gesendet. Der Liefertermin wird entweder automatisch oder nach Nutzereingabe mit dem Supermarkt vereinbart.
* Beim Eingang der Bestellung nimmt die Software die Produkte in den Bestand auf und der Nutzer kann den neuen Bestand seines Kühlschranks begutachten.
* Die Kosten der Bestellungen sind den Rechnungen zu entnehmen, die ebenfalls online einzusehen sind.

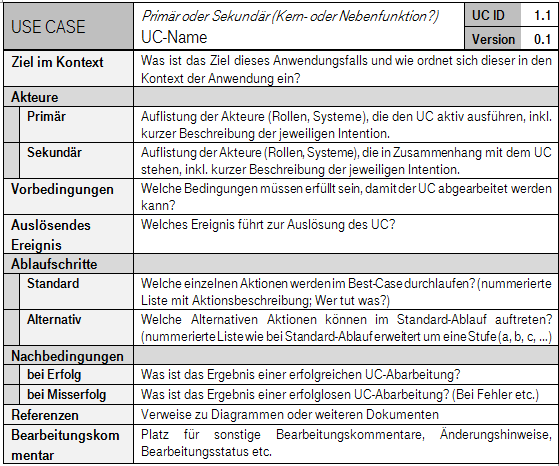
Diese Features können in einem Online-Tool konfiguriert werden.

Zweck des Projektes ist ein mobiler Zugriff auf den Kühlschrank. Es ist möglich, von überall aus den Bestand einzusehen, Bestellungen abzuschicken oder Wunschmengen oder Mindestbestandsmengen zu verändern, ohne vor Ort sein zu müssen. Des Weiteren erspart man dem Nutzer der Software den Weg zum Supermarkt und das manuelle Erstellen eines Warenkorbs. Außerdem geht der Prozess bargeldlos vonstatten.

Die Kühlschrank-Verwaltung interagiert mit der Schnittstelle des Supermarktes. Der Supermarkt gibt das Sortiment zur Einsicht aus, nimmt Bestellungen an, vereinbart Liefertermine, versendet Bestellungen und stellt Rechnungen aus.

# Funktionales Modell

Hier sollen in tabellarischer Form die Use Cases (UML-Use Case Beschreibungen) des Projekts dargestellt werden.

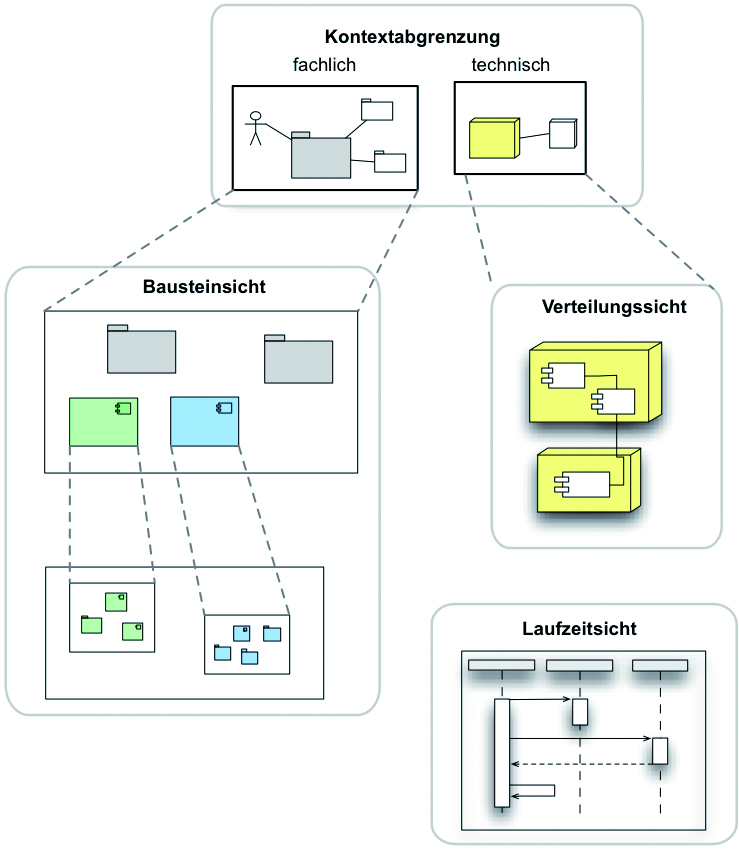


# Dialoglandkarte

Hier kann eine Dialoglandkarte mit ein paar kurzen Beschreibungen eingefügt werden. Es empfehlen sich hierfür Mockup-Tools wie Balsamiq.

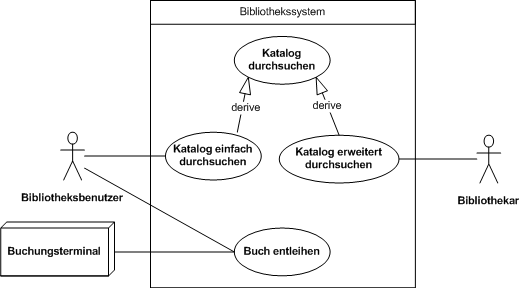
# IT Architektur

Hier sollen mehrere Architekturbilder (Kontextabgrenzung, Bausteinsicht, Verteilungssicht, Laufzeitsicht) entstehen. Idealerweise erstellt jedes Teammitglied eine Sicht auf das System.

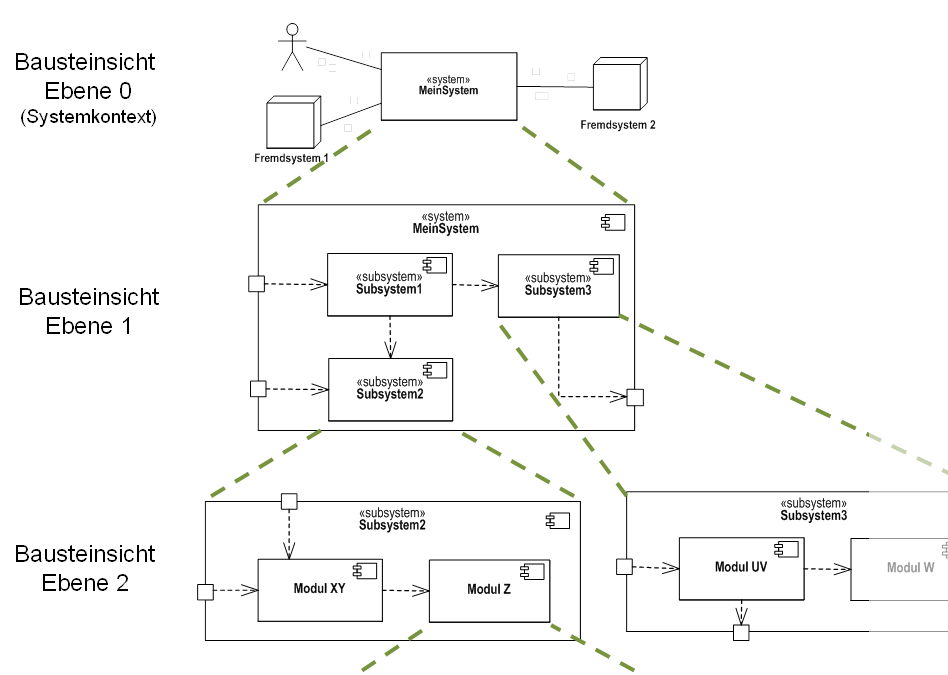


## Kontextabgrenzung

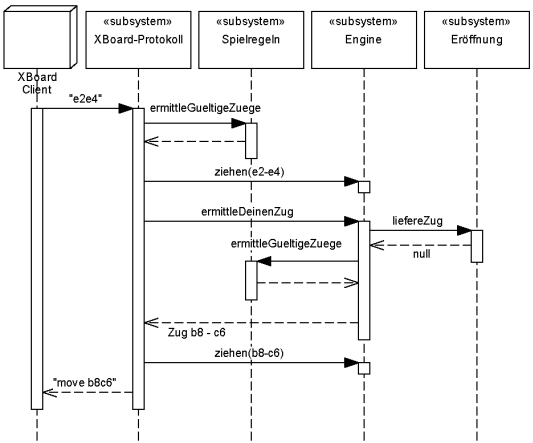
Die Kontextabgrenzung entspricht der Bausteinsicht Ebene 0 und wird in der Regel in Form von Use-Case Diagrammen skizziert.



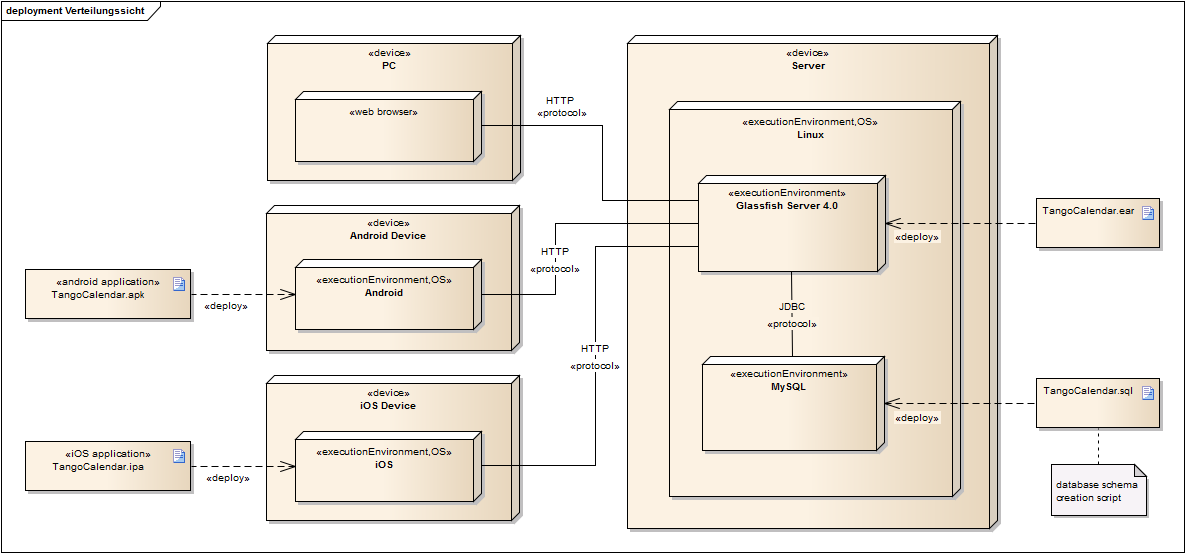
## Bausteinsicht



## Laufzeitsicht



## Verteilungssicht



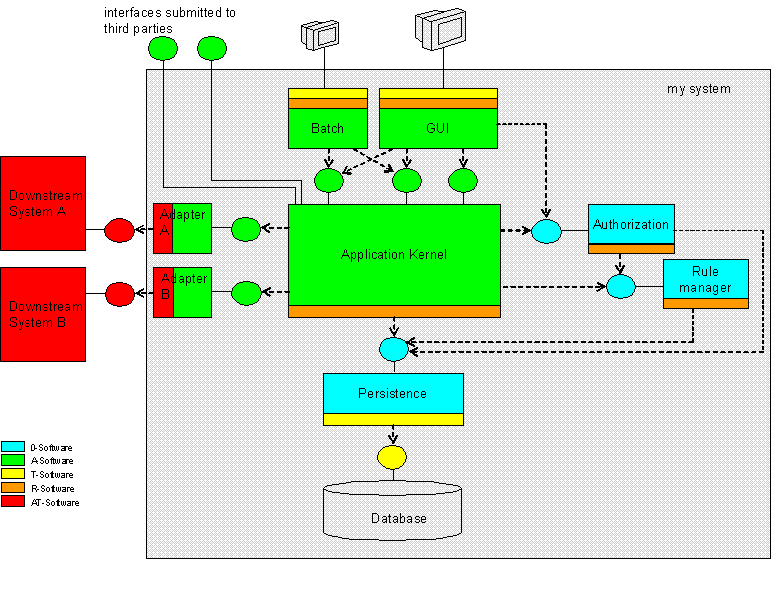
# Schnittstellen und Integration

## Schnittstellenspezifikation

Welche Schnittstellen bietet das System an? D.h. welche Services bietet unser System an? (Textuelle Beschreibung der Schnittstelle)

Hier kann eine explizite Aufschlüsselung der Schnittstellen erfolgen. (Auflistung mit sinnvoller Kategorisierung z.B. ob Inbound oder Outbound, Synchron oder asynchron, Request oder Batch, etc.)

Des Weiteren sollte hier in jedem Fall ein Architekturbild hinterlegt werden, auf dem die nach außen angebotenen Schnittstellen unseres Systems oder aber auch die von unserem System genutzten Schnittstellen ersichtlich sind. (Architekturbild mit Schnittstellen)



## Schnittstellenformat

Hier sollte das Format der Schnittstelle (XML, JSON) und auch die Struktur hinterlegt werden.

Z.B. wenn die Struktur einer Fahrzeugkonfiguration per XML übertragen werden soll, dann kann das hier hinterlegt werden.

# Datenmodell

## Datenspeicherungskonzept

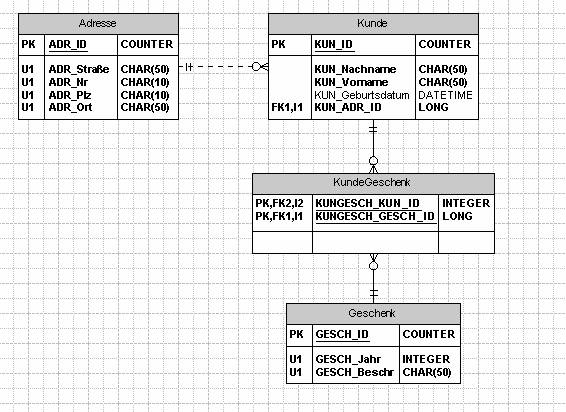
Hier kann kurz beschrieben werden, wie der Persistenzmechanismus des Moduls / der Komponente bzw. des Systems aussehen soll.

## Physikalische Datenbankstruktur

Hier kann eine tabellarische Darstellung und / oder eine grafische Darstellung der Datenbanktabellen erfolgen (mit Datentypen, Primary Keys, Foreign Keys, etc.)

## Datenmodell

Hier soll dargestellt werden, wie das Datenmodell der Erweiterung aussehen wird.



# Testspezifikation

## Teststrategie

Hier kann die grobe Teststrategie (manuelle Tests, automatisierte Tests, halb automatisierte Tests), das Testvorgehen beschrieben werden.

## Testfälle

Hier kann eine Beschreibung der Testfälle (Testschritte, Testergebnisse, ggfs. Varianten bei Tests) erfolgen.

# Projektmanagement

## Projektplan

* Projektstart – Kick-Off
  + Themenfestlegung
  + Grobe Use-Case-Definition
  + Meilensteine definieren
  + Issues erstellen und zuweisen
  + Logo und Dokumentendesign
  + Glossar erstellen
* Konzept / Projektbeschreibung
  + Projektüberblick schreiben
  + Use-Cases ausformulieren
  + Dialoglandkarte entwerfen
  + IT-Architektur visualisieren
  + Use-Case-Diagramm / Kontextabgrenzung erstellen
  + Schnittstellen und Integration definieren
  + Datenmodell entwerfen
  + Abnahme des Konzepts
* Implementierung
  + Technische Umsetzung
  + Dokumentation
  + Tests durchführen
* Präsentation
  + Erstellung und Vorbereitung der Präsentation
  + Abgabe einer digitalen Version des Projekts, sowie notwendiger Dokumente
  + Druchführung der Präsentation und Abschluss des Projekts

## Aufgabenverteilung

Hier sollte explizit beschrieben werden, wer für welche Funktionen / Aufgaben zuständig ist. Es soll am Schluss möglich sein, dass man die entsprechenden Aufgaben der jeweiligen Person zweifelsfrei zuordnen kann. Die Verteilung der Aufgaben soll klar getrennt (funktional oder über unterschiedliche Schichten) sein.

# Bibliographie

Links zu anderen referenzierten Dokumenten (z.B. Schnittstellenverträge, etc.) mit kurzer Beschreibung des Inhalts.

# Glossar

Siehe GitHub