

Verteilte Systeme und Komponenten

Architekturbeschreibung

Martin Bättig

Letzte Aktualisierung: 12. Oktober 2022

FH Zentralschweiz

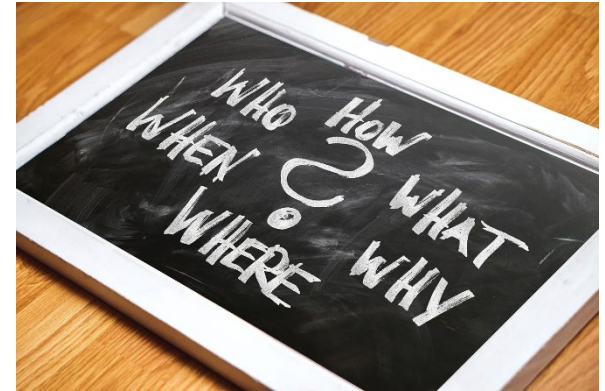


Inhalt

- Architekturbeschreibung
- Vorlagen
- Sichten auf ein System

Lernziele

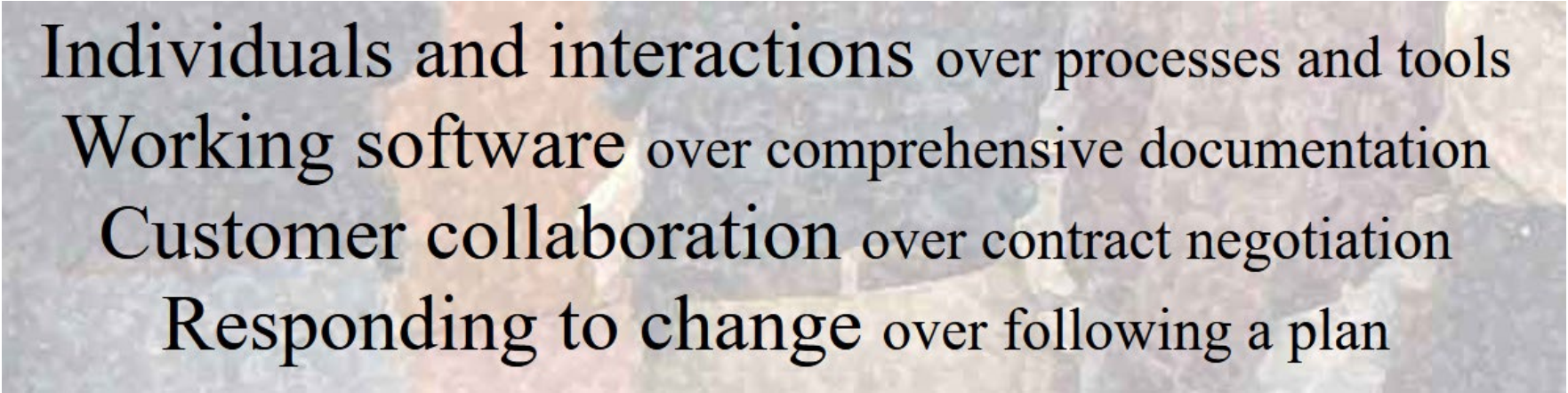
- Sie kennen das Konzept der Software-Komponenten und können Komponenten gemäss Spezifikation erstellen, dokumentieren, testen und überarbeiten.



Architekturbeschreibung

Bildquelle: pixabay.com

Manifesto for Agile Software Development



Individuals and interactions over processes and tools
Working software over comprehensive documentation
Customer collaboration over contract negotiation
Responding to change over following a plan

Quelle: <http://agilemanifesto.org>

Ziel der Architekturbeschreibung

Beschreibt die Umsetzung der funktionalen und nicht-funktionalen Anforderungen, welche an ein System gestellt werden.

Anforderungen abgeleitet aus:

- übergeordneten Anforderungsdokumenten (z.B. Lastenheft).
- übergeordneten Systemen (z.B. Schnittstellen der umgebenden Systeme).

Alternative Namen:

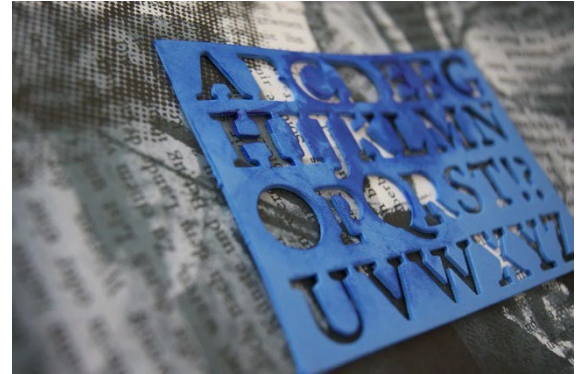
- Systemspezifikation
- Systembeschreibung

Nutzen der Architekturbeschreibung

- Entwurf und Dekomposition der Architektur.
- Grobdesigns der Komponenten.
- Kommunikation der Informationen an die am Projekt beteiligten Akteure.

Kompatibilität mit agilen Vorgehensmodellen

- Initiale Version (Vorprojekt / Initialisierungsphase / Sprint #0).
 - Bildet bisher erkannte Anforderungen ab.
- Anpassungen pro Sprint, sobald Anforderungen besser bekannt sind.
- Änderungen der Architekturbeschreibung in Sprint-Review kommunizieren.



Vorlagen

Bildquelle: pixabay.com

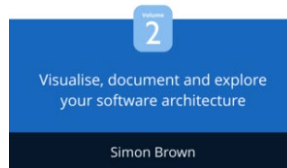
Vorlagen (Auswahl)

Auswahl an verschiedenen Vorlagen:



arc42

**Software
Architecture**
for Developers



SA4D/C4-Modell



Firmeneigene Vorlage

Warum Vorlagen verwenden?

arc42

- Gängiger Standard im deutschsprachigen Raum.
- Beantwortet zentrale Fragen:
 - **Was** sollen wir über unsere Architektur kommunizieren / dokumentieren?
 - **Wie** sollen wir kommunizieren / dokumentieren?

1. Einführung und Ziele 1.1 Aufgabenstellung 1.2 Qualitätsziele 1.3 Stakeholder	7. Verteilungssicht 7.1 Infrastruktur Ebene 1 7.2 Infrastruktur Ebene 2
2. Randbedingungen 2.1 Technische Randbedingungen 2.2 Organisatorische Randbedingungen 2.3 Konventionen	8. Querschnittliche Konzepte 8.1 Fachliche Struktur und Modelle 8.2 Architektur- und Entwurfsmuster 8.3 Unter-der-Haube 8.4 User Experience
3. Kontextabgrenzung 3.1 Fachlicher Kontext 3.2 Technischer- oder Verteilungskontext	9. Entwurfsentscheidungen 9.1 Entwurfsentscheidung 1 9.2 Entwurfsentscheidung 2
4. Lösungsstrategie	10. Qualitätsanforderungen 10.1 Qualitätsbaum 10.2 Qualitätsszenarien
5. Bausteinsicht 5.1 Ebene 1 5.2 Ebene 2	11. Risiken und technische Schulden
6. Laufzeitsicht 6.1 Laufzeitszenario 1 6.2 Laufzeitszenario 2	12. Glossar

Inhaltsverzeichnis, wobei blaue Kapitel den Kern des Dokuments bilden.

Quelle: <https://www.arc42.de>

arc42 Beispiele

- https://hsc.aim42.org/documentation/hsc_arc42.html
 - Verbose example for the documentation of a Gradle plugin, created by Dr. Gernot Starke.
- <https://biking.michael-simons.eu/docs/index.html>
 - (English) A real world example for a bike activity tracker, created by Michael Simons.

SA4D

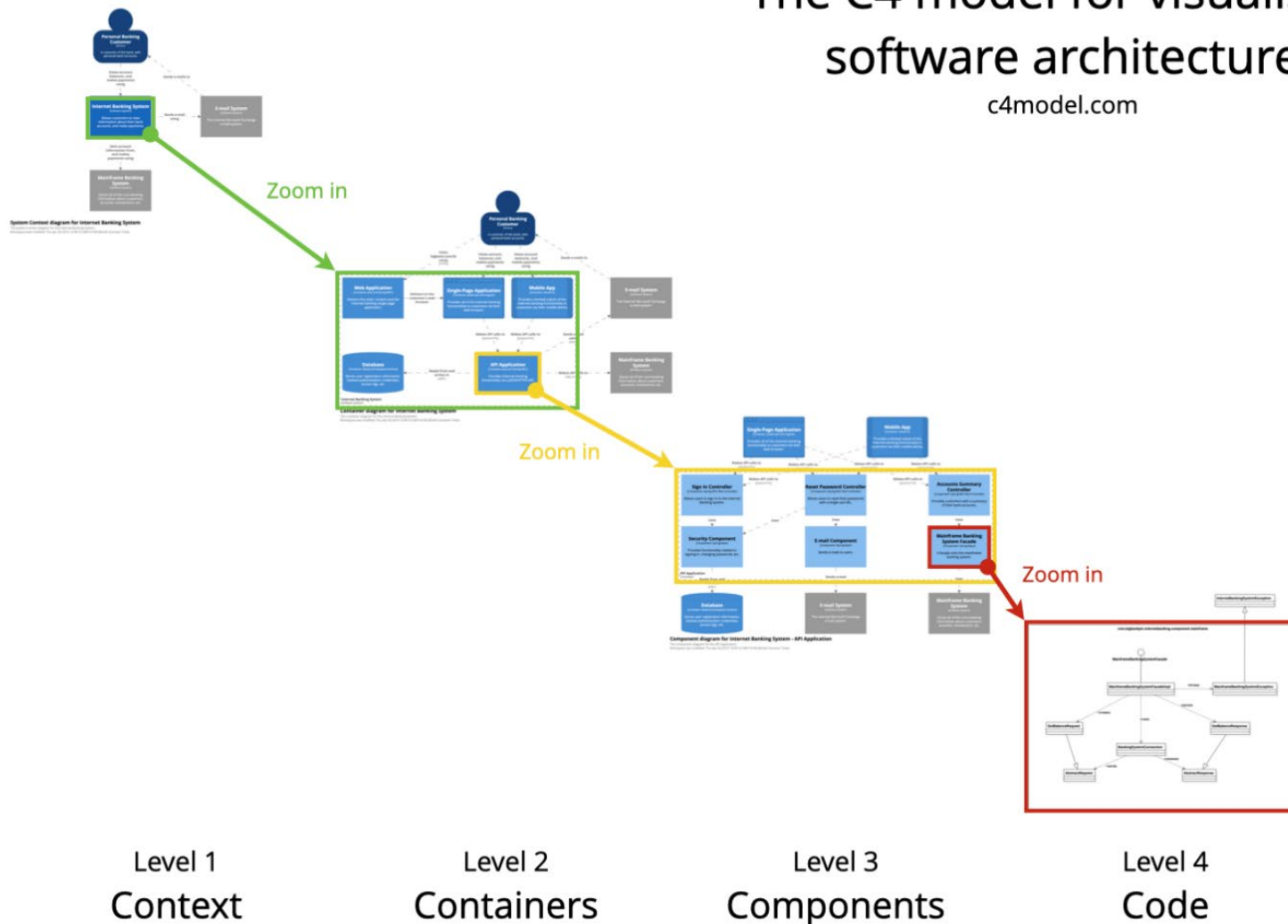
- Motto: "Beschreibe was nicht offensichtlich aus dem Code erkennbar ist."
- Eher Produkt als Projekt-Dokumentation.
- Kombiniert mit dem C4-Architektur-visualisierungsmodell.



C4 Modell

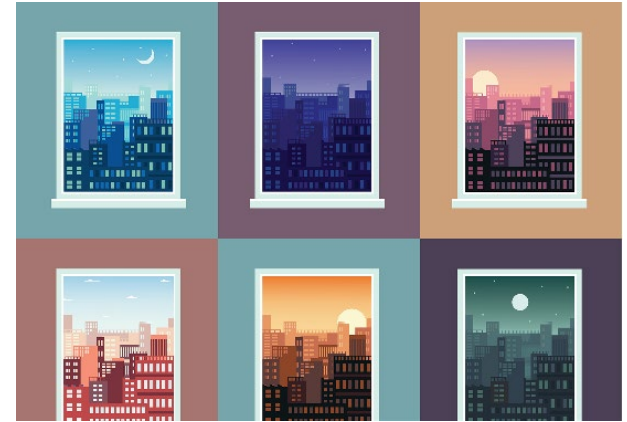
The C4 model for visualising software architecture

c4model.com



Firmeneigene Dokumente

- Viele grössere Firmen haben eigene Vorlagen.
- Auf Produkt, Vorgehen und/oder Branche zugeschnitten.



Sichten auf ein System

Sichten auf ein System

- Systemübersicht
- Schnittstellen
- Architektonische Sichten
- Systemumgebung
- Designentscheide

Systemübersicht

- Welches Problem löst das System?
- Wie löst das System das Problem?
- Wer benutzt das System?
- Annahmen und Einschränkungen.

1. Einführung und Ziele

- 1.1 Aufgabenstellung
- 1.2 Qualitätsziele
- 1.3 Stakeholder

2. Randbedingungen

- 2.1 Technische Randbedingungen
- 2.2 Organisatorische Randbedingungen
- 2.3 Konventionen

3. Kontextabgrenzung

- 3.1 Fachlicher Kontext
- 3.2 Technischer- oder Verteilungskontext

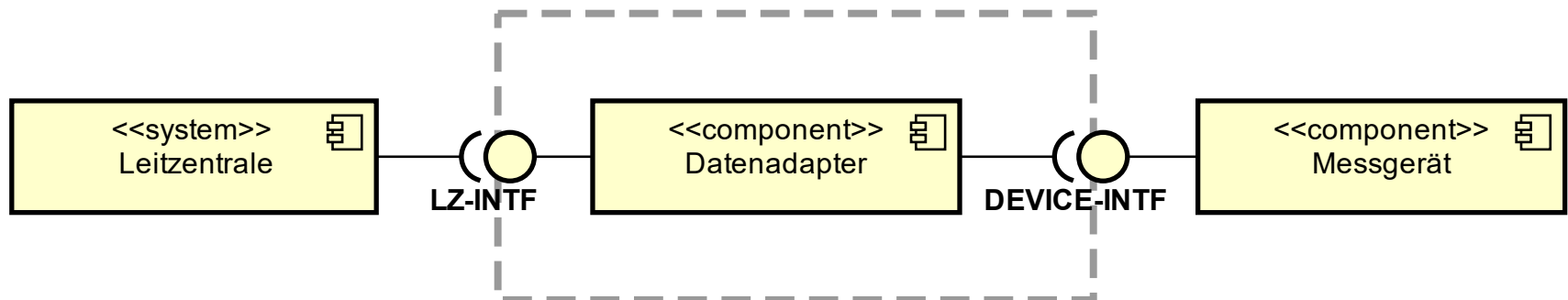
4. Lösungsstrategie

Kontextabgrenzung

Ziel: Zeige das zu entwickelnde System in seinem Kontext (geschäftlich / technisch).

3. Kontextabgrenzung
3.1 Fachlicher Kontext
3.2 Technischer- oder Verteilungskontext

Beispiel für technischen Kontext: Kontextdiagramm eines Datenadapters



Wichtig: Jedes Diagramm muss mit Text beschrieben werden!

Schnittstellen

- Externe Schnittstellen.
 - Schnittstellen nach aussen.
 - Sowohl exportierte als auch importierte Schnittstellen.
- Benutzerschnittstellen.

3. Kontextabgrenzung

3.1 Fachlicher Kontext

3.2 Technischer- oder Verteilungskontext

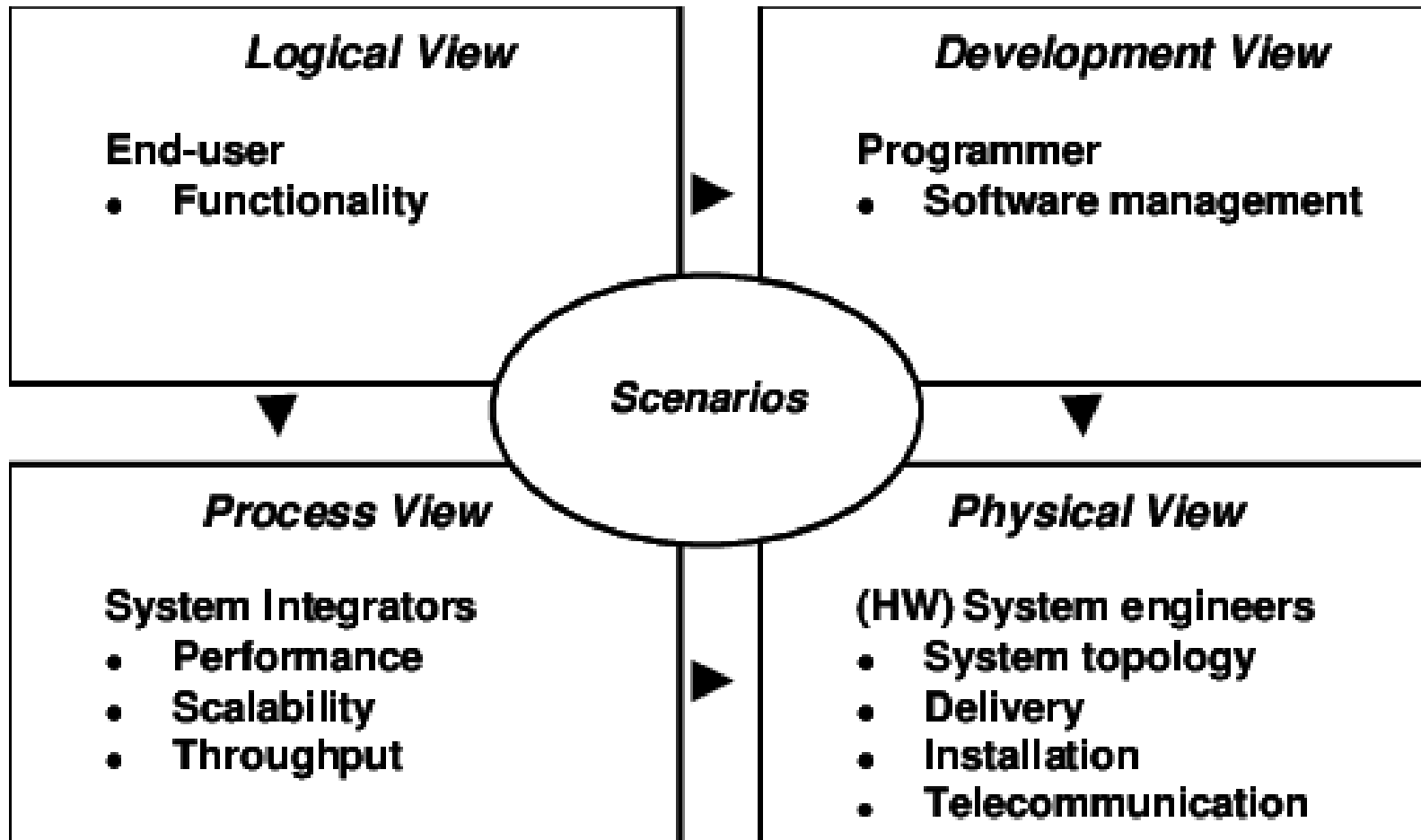
Beispiel: StringPersistor-Schnittstelle (Material auf ILIAS).

Softwarearchitektur

- Unterschiedliche Sichten auf ein System aus unterschiedlichen Perspektiven auf hohem Abstraktionsniveau.
- Zuhilfenahme von Architekturbeschreibungsmodellen, z.B.
 - arc42 (Bausteinebenen).
 - 4 + 1-Sichten-Modell von Philippe Kruchten.
 - C4-Modell von Simon Brown (Fokus auf statische Sicht).

4+1-Sichten-Modell von Philipp Kruchten (1995)

Weit verbreitetes Modell:



Statische Sicht und Laufzeitsicht (Logical View)

- Funktionalität des Systems auf unterschiedlichen Flughöhen zeigen z.B. arc42 (Bausteinebenen) oder C4-Modell.
- Pro System mehrere Ebenen (falls zutreffend):
 - Teilsystem / Services: z.B. Block- oder Komponentendiagramme.
 - Komponenten: z.B. Komponentendiagramme.
 - Code: i.d.R. Klassendiagramm oder Sequenzdiagramme.
 - **Richtlinie 1: Kein Reverse-Engineering von Diagrammen aus dem Code!**
 - **Richtlinie 2: Nur interessante / hilfreiche Stellen dokumentieren.**
- Dokumentation auf das Notwendigste reduzieren.
 - nur relevante Klassen und Attribute.

5. Bausteinsicht

5.1 Ebene 1

5.2 Ebene 2

....

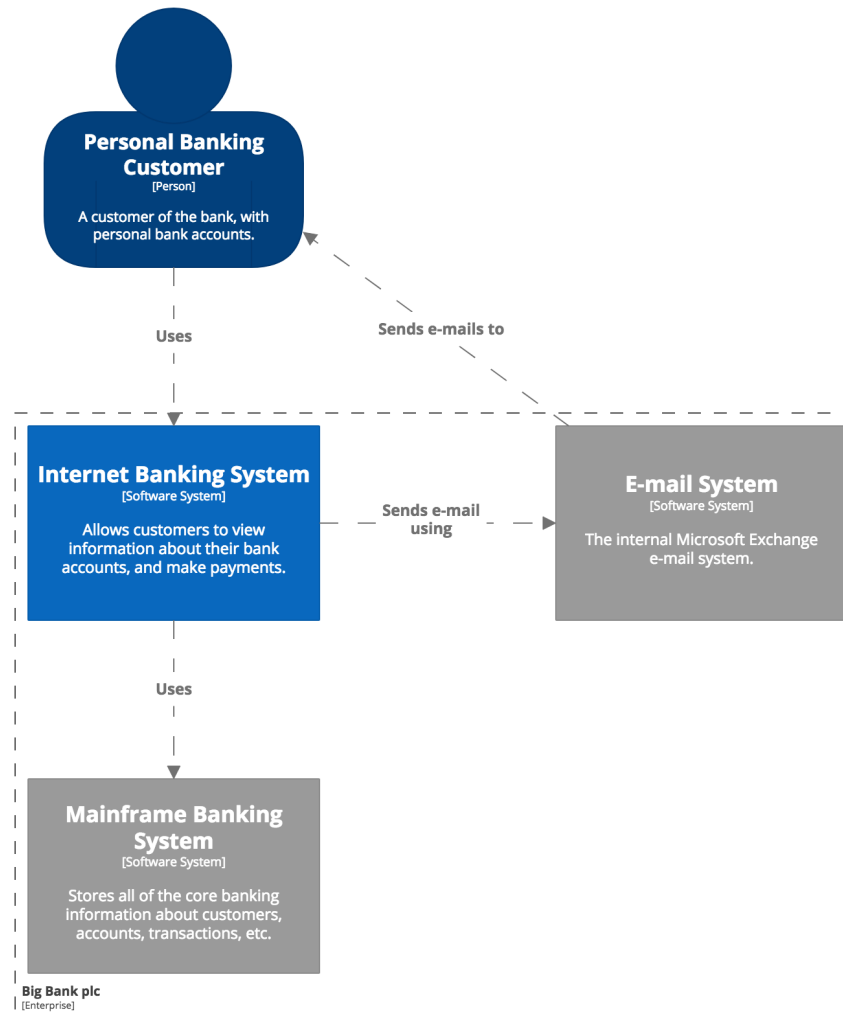
6. Laufzeitsicht

6.1 Laufzeitszenario 1

6.2 Laufzeitszenario 2

....

Statische Sicht am Beispiel des C4-Modells (Ebene 1)

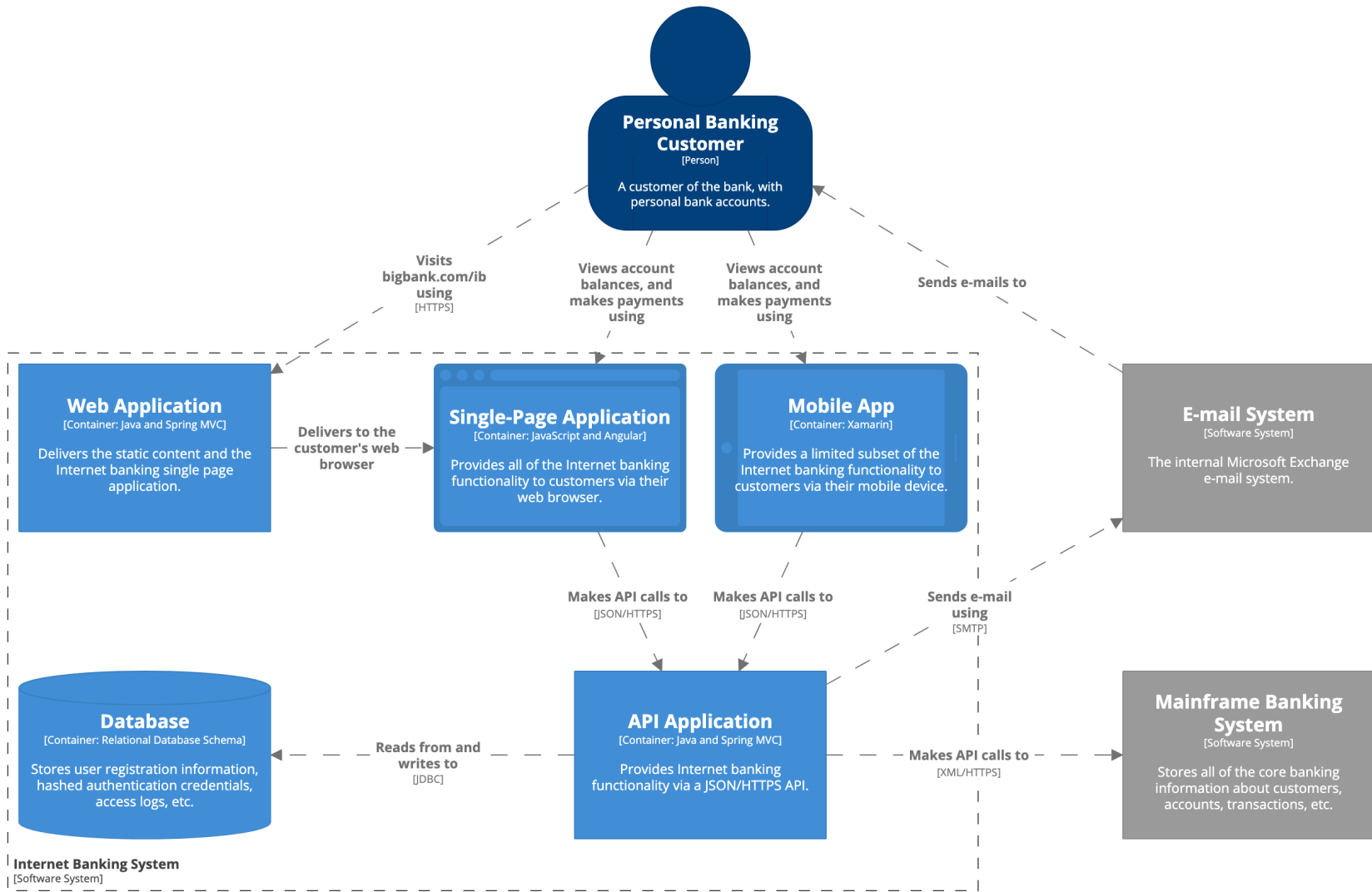


System Context diagram for Internet Banking System

The system context diagram for the Internet Banking System.
Last modified: Wednesday 02 May 2018 13:41 BST

Quelle: Software Architecture for Developers Vol. 2, Simon Brown, Leanpub, 2021.

Statische Sicht am Beispiel des C4-Modells (Ebene 2)



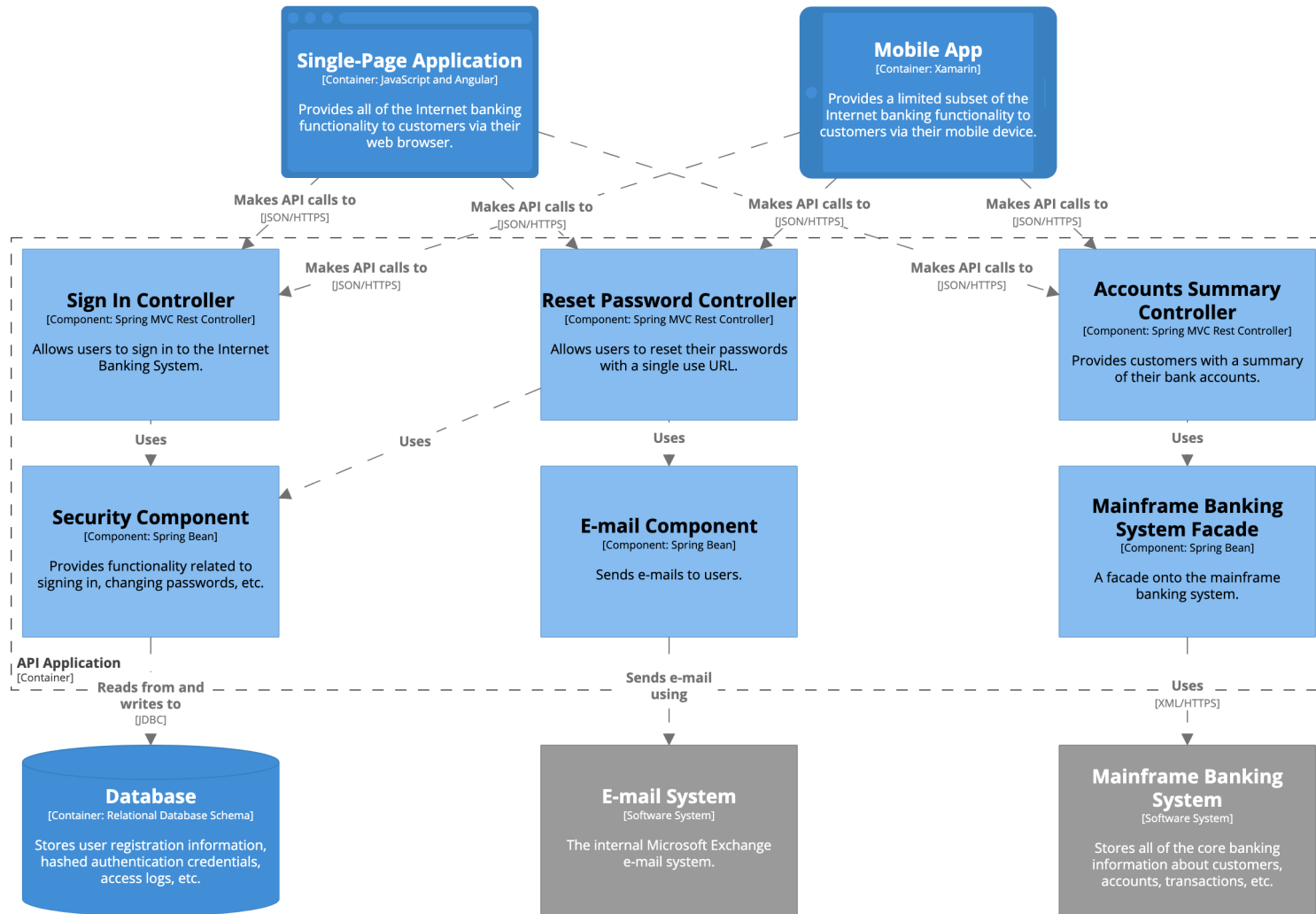
Container diagram for Internet Banking System

The container diagram for the Internet Banking System.

Workspace last modified: Thu Apr 04 2019 13:09:10 GMT+0100 (British Summer Time)

Quelle: Software Architecture for Developers Vol. 2, Simon Brown, Leanpub, 2021.

Statische Sicht am Beispiel des C4-Modells (Ebene 3)



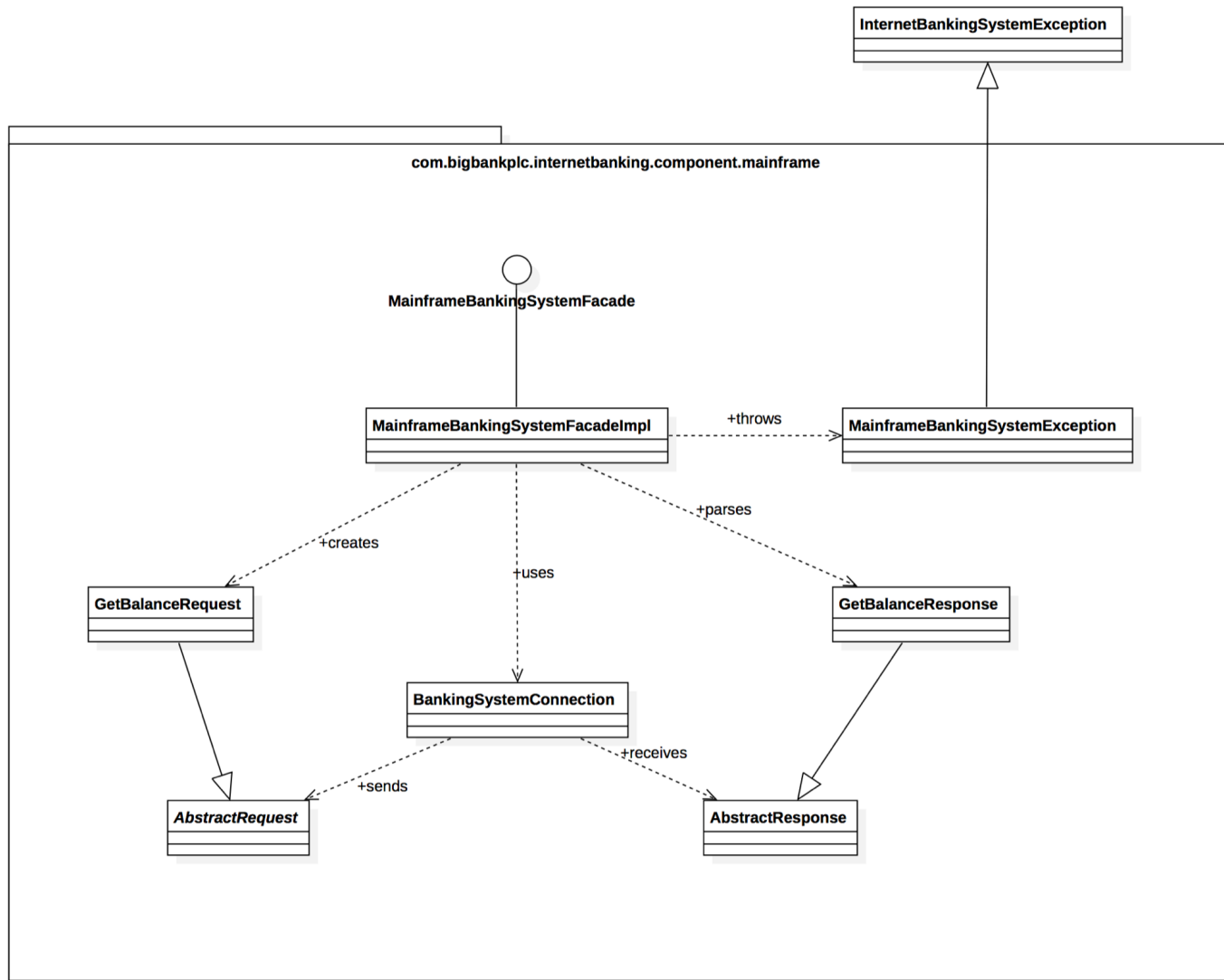
Component diagram for Internet Banking System - API Application

The component diagram for the API Application.

Workspace last modified: Thu Apr 04 2019 13:09:10 GMT+0100 (British Summer Time)

Quelle: Software Architecture for Developers Vol. 2, Simon Brown, Leanpub, 2021.

Statische Sicht am Beispiel des C4-Modells (Ebene 4)



Quelle: Software Architecture for Developers Vol. 2, Simon Brown, Leanpub, 2021.

Systemumgebung (Physical View)

- Sind mehrere physische Systeme involviert?
Welche?
- Auf welchem System wird welche Komponente eingesetzt?
Ergeben sich unterschiedliche Einsatzszenarien?
- Was sind die Anforderungen an das jeweilige System (Hardware, Betriebssystem, installierte Software, Netzwerkfreigaben / Firewall-Regeln, Netzwerkbandbreite, etc.).
- Wie werden die Komponenten in Betrieb genommen?
- Müssen die Komponenten konfiguriert werden? Wie?
- Praktisch: Angabe einer Beispielkonfiguration.

7. Verteilungssicht

7.1 Infrastruktur Ebene 1

7.2 Infrastruktur Ebene 2

....

Verarbeitungsansicht (Process View)

Datenverarbeitung:

- Persistente Daten und deren Strukturierung (z.B. Benutzerkonten, Transaktionsdaten).
- Beziehungen zwischen den Daten (z.B. ER-Modell).
- Wie werden die Daten gespeichert? Datenbank (relational, NoSQL, eigenes Fileformat, via Webservice, in der Cloud, etc.).
- (Wie) wird die Konsistenz sichergestellt? Wird dies überprüft?
- u.s.w.

8. Querschnittliche Konzepte

- 8.1 Fachliche Struktur und Modelle
- 8.2 Architektur- und Entwurfsmuster
- 8.3 Unter-der-Haube
- 8.4 User Experience
-

Wichtige Schnittstellen:

- Interne Schnittstellen zwischen Komponenten.

Verarbeitungsansicht (Process View, forts.)

Qualitätsanforderungen:

- Wie viele Daten pro Zeiteinheit muss das System bzw. einzelne Komponenten verarbeiten können?
- (Wie) werden die Daten wieder gelöscht?
- Maintainability: Wie werden die Daten gesichert (Backup)?
Wie schnell kann ein System wiederhergestellt werden?
- u.s.w.

10. Qualitätsanforderungen

10.1 Qualitätsbaum

10.2 Qualitätsszenarien

Designentscheide

- Was sind die wesentlichen Überlegungen, welche Sie beim Design des Systems gemacht haben?
- Sind bestimmte Randfälle absichtlich nicht unterstützt? Welche?
- Sollen bestimmte Techniken (nicht) verwendet werden? Welche?
- Bestimmte Programmierparadigmen, Patterns?
- Bestimmte Libraries, Frameworks, Laufzeitumgebungen?
- usw.

9. Entwurfsentscheidungen

9.1 Entwurfsentscheidung 1

9.2 Entwurfsentscheidung 2

....

Zusammenfassung

- Grobdesign des Systems (verschiedene Sichten, Daten & Mengengerüste, sowie Designentscheide).
- Kommunikation mit beteiligten Akteuren (Kunden, Entwickler, Tester, Betrieb, Wartung).
- Schnittstellen: Extern, wichtige interne (Komponentengrenzen), Benutzerschnittstellen.
- Anforderungen an die Systemumgebung.

Literatur

- arc42: better software architectures von Gernot Starke, Peter Hruschka, Ralf D. Müller.

<https://arc42.org/>

- Software Architecture for Developers Vol. 2 von Simon Brown, Leanpub, 2021.

Fragen?