C:\Users\Manfred Stueber\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Logo_Hauptcampus_Informatik.eps

Topology-Editor System Design Dokument

English Title

Autor

Ghislain Zeus

Master-Abschlussarbeit

Betreuer:

Ort, 17.9.2018

Vorwort

TODO? Was soll man hier schreiben

Kurzfassung

Während des Systemdesigns identifizieren wir Designziele, zerlegen das System in Subsysteme und verfeinern die Subsystemzerlegung, bis alle Designziele erreicht sind. Im vorherigen Kapitel haben wir die Konzepte von Entwurfszielen und Systemzerlegung beschrieben. In diesem Kapitel stellen wir die Systemdesign-Aktivitäten vor, die sich mit den Designzielen befassen. Insbesondere untersuchen wir einige Aspekte, die für die Umsetzung von Bedeutung sind.

Auswahl von Standard- und Legacy-Komponenten. Ältere Komponenten realisieren spezifische Subsysteme wirtschaftlicher. Die anfängliche Subsystemzerlegung wird angepasst, um sie anzupassen.

Zuordnung von Subsystem zu Hardware. Wenn das System auf mehreren Knoten implementiert wird, sind zusätzliche Subsysteme erforderlich, um Zuverlässigkeits- oder Leistungsprobleme zu beheben.

Design einer persistenten Datenmanagement-Infrastruktur. Die Verwaltung der Zustände, die eine einzelne Ausführung des Systems überdauern, wirkt sich auf die Gesamtsystemleistung aus und führt zur Identifizierung eines oder mehrerer Speichersubsysteme.

Spezifikation einer Zugriffskontrollrichtlinie gemeinsame Objekte sind geschützt, sodass der Benutzerzugriff auf sie gesteuert wird. Die Zugriffssteuerung beeinflusst die Verteilung von Objekten innerhalb von Subsystemen.

Design des globalen Kontrollflusses. Die Festlegung der Reihenfolge der Operationen hat Auswirkungen auf die Schnittstelle der Subsysteme.

Umgang mit Randbedingungen sobald alle Subsysteme identifiziert wurden, entscheiden die Entwickler über die Reihenfolge, in der einzelne Komponenten gestartet und heruntergefahren werden.

Abstract

The same in English (option).

Inhaltsverzeichnis

[1 Einleitung 1](#_Toc524950994)

[1.1 Zweck des Systems 1](#_Toc524950995)

[1.2 Wieso entwerfen? (Designziel) 1](#_Toc524950996)

[1.3 Definitionen, Akronyme und Abkürzungen 1](#_Toc524950997)

[1.4 Sämtliche Verweise 1](#_Toc524950998)

[2 Aktuelle Softwarearchitektur 2](#_Toc524950999)

[3 Vorgeschlagene Softwarearchitektur 3](#_Toc524951000)

[3.1 Überblick 3](#_Toc524951001)

[3.2 Subsystemzerlegung (Subsystem decomposition) 3](#_Toc524951002)

[3.2.1 Client oder UI 3](#_Toc524951003)

[3.3 Hardware / Software-Zuordnung (Hardware/software mapping) 4](#_Toc524951004)

[3.4 Persistente Datenverwaltung 4](#_Toc524951005)

[3.5 Zugriffskontrolle und Sicherheit 4](#_Toc524951006)

[3.6 Globale Softwaresteuerung 4](#_Toc524951007)

[3.7 Randbedingungen (boundary conditions) 4](#_Toc524951008)

[3.8 Subsystemdienste (Subsystem services) 5](#_Toc524951009)

[4 Zusammenfassung und Ausblick 6](#_Toc524951010)

[Literatur 7](#_Toc524951011)

[Index 8](#_Toc524951012)

[Glossar 9](#_Toc524951013)

Abbildungsverzeichnis

[Abbildung 1: Überblick des Topology-Editors 3](file:///C:\Users\GhislainZeleu\Desktop\Workspace\G_SVN\my-first-angular5\Topology-Editor%20SDD.docx#_Toc524951014)

[Abbildung 2: Topology-Editor Client Komponentendiagramm 4](#_Toc524951015)

Tabellenverzeichnis

Es konnten keine Einträge für ein Abbildungsverzeichnis gefunden werden.Verantwortliche

# Einleitung

Begonnen werden soll mit einer Einleitung zum Thema, also Hintergrund und Ziel erläutert werden.

Weiterhin wird das vorliegende Problem diskutiert: Was ist zu lösen, warum ist es wichtig, dass man dieses Problem löst und welche Lösungsansätze gibt es bereits. Der Bezug auf vorhandene oder eben bisher fehlende Lösungen begründet auch die Intention und Bedeutung dieser Arbeit. Dies können allgemeine Gesichtspunkte sein: Man liefert einen Beitrag für ein generelles Problem oder man hat eine spezielle Systemumgebung oder ein spezielles Produkt (z.B. in einem Unternehmen), woraus sich dieses noch zu lösende Problem ergibt.

Im weiteren Verlauf wird die Problemstellung konkret dargestellt: Was ist spezifisch zu lösen? Welche Randbedingungen sind gegeben und was ist die Zielsetzung? Letztere soll das beschreiben, was man mit dieser Arbeit (mindestens) erreichen möchte.

## Zweck des Systems

## Wieso entwerfen? (Designziel)

## Definitionen, Akronyme und Abkürzungen

## Sämtliche Verweise

# Aktuelle Softwarearchitektur

# Vorgeschlagene Softwarearchitektur

In diesem Kapitel wird beschrieben, warum es unterschiedliche Konsistenzmodelle gibt. Außerdem werden die Unterschiede zwischen strengen Konsistenzmodellen (Linearisierbarkeit, sequentielle Konsistenz und schwachen Konsistenzmodellen (schwache Konsistenz, Freigabekonsistenz) erläutert. Es wird geklärt, was Strenge und Kosten (billig, teuer) in Zusammenhang mit Konsistenzmodellen bedeuten.

## Überblick

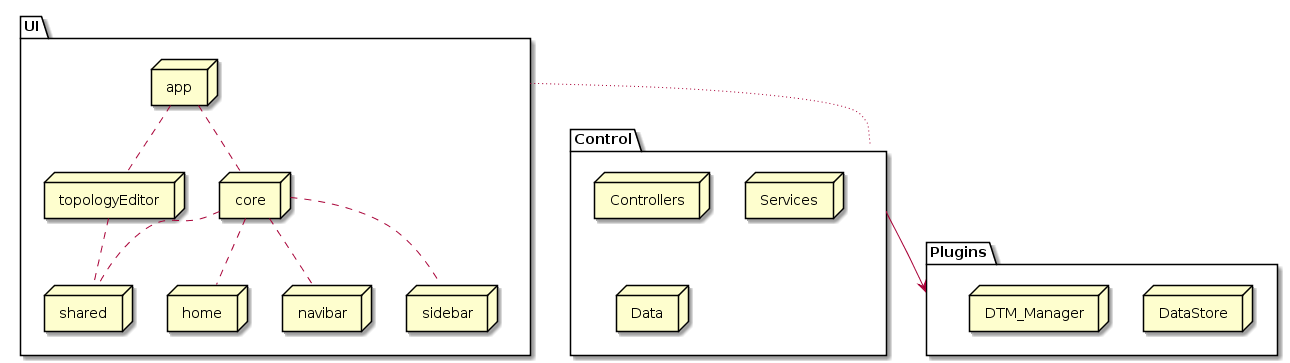
hier wird der Überblick der zu implementiertem System darzustellen.

Abbildung 1: Überblick des Topology-Editors

## Subsystemzerlegung (Subsystem decomposition)

Es handelt sich um eine verteilte Anwendung. Somit wird das System 2 Subsystem unterteilt nämlich Client und Server.

### Client oder UI

Das Subsystem besteht aus Komponenten, die für die grafische Benutzeroberfläche der Topologie zuständig sind. Dazu gehören das Hauptfenster des Topology-Editors, die Werkzeugleiste (navbar und sidebar), die die Manipulationstasten unterstützt, ein Home-Dialog, in dem der Endanwender die Einstellungen des Topology-Editors und das Tologogy-Informationsfenster steuern kann. Die restlichen Komponenten dienen hier nur für die Struktur der Applikation z.B. *Core-Komponente* beinhaltet die 3 Modules und die *Shared-Komponente* kapselt die Daten und Service, die in der gesamten Anwendung zur Verfügung stehen sollen.

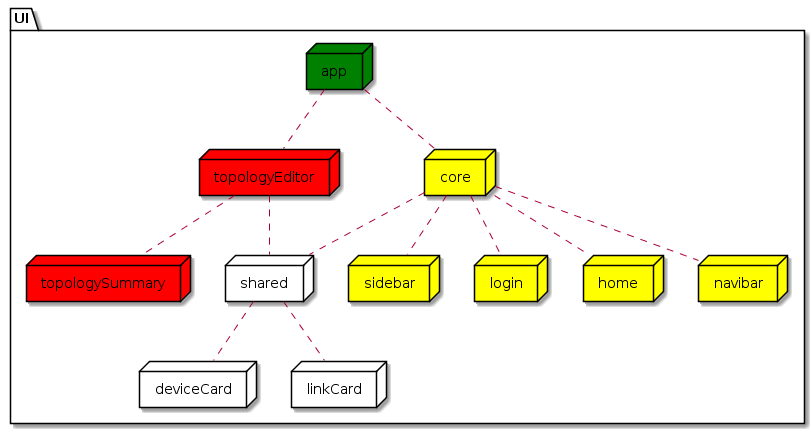


Abbildung 2: Topology-Editor Client Komponentendiagramm

## Hardware / Software-Zuordnung (Hardware/software mapping)

Tofo todod todo

## Persistente Datenverwaltung

Tofo

## Zugriffskontrolle und Sicherheit

Tofo

## Globale Softwaresteuerung

Tofo

## Randbedingungen (boundary conditions)

Tofo

## Subsystemdienste (Subsystem services)

Die Linearisierbarkeit im Zusammenhang mit DSM kann wie folgt definiert werden:

# Zusammenfassung und Ausblick

In diesem Kapitel soll die Arbeit noch einmal kurz zusammengefasst werden. Insbesondere sollen die wesentlichen Ergebnisse Ihrer Arbeit herausgehoben werden. Erfahrungen, die z.B. Benutzer mit der Mensch-Maschine-Schnittstelle gemacht haben oder Ergebnisse von Leistungsmessungen sollen an dieser Stelle präsentiert werden. Sie können in diesem Kapitel auch die Ergebnisse oder das Arbeitsumfeld Ihrer Arbeit kritisch bewerten. Wünschenswerte Erweiterungen sollen als Hinweise auf weiterführende Arbeiten erwähnt werden.

Literatur

Cheriton, D. R. (1985). Preliminary Thoughts on Problem-oriented Shared Memory: A Decentralized Approach to Distributed Systems. (S. University, Ed.) *Operating Systems Review, 19*(4), pp. 26-33.

CodePlex. (2013, 2). *BibWord*. (CodePlex) Retrieved 2 26, 2013, from http://bibword.codeplex.com/releases/view/15852

Coulouris, G., Dollimore, J., & Kindberg, T. (2002). *Verteilte Systeme: Konzepte und Design.* Addison-Wesley-Verlag.

Malte, P. (1997). (A. Grosse, & D. Kottmann, Editors) Retrieved from Architektur vernetzter Systeme. Seminar SS 1996 & WS 1996/97: Replikation in Mobil Computing: http://www.ubka.uni-karlsruhe.de/cgi-bin/psview?document=/ira/1997/31

Mosberger, D. (1993). *Memory Consistency Models.* Technical Report, University of Arizona.

Index

F

Freigabekonsitenz 3

K

Konsitenz 3

schwach 3

Konsitenzmodelle 3

S

schwach 3

sequentiell

Konsitenz 3

streng 3

Glossar

|  |  |
| --- | --- |
| DisASTer | DisASTer (Distributed Algorithms Simulation Terrain) A platform for the Implementation of Distributed Algorithms |
| DSM | Distributed Shared Memory |
| AC | Linearisierbarkeit (atomic consistency) |
| SC | Sequentielle Konsistenz (sequential consistency) |
| WC | Schwache Konsistenz (weak consistency) |
| RC | Freigabekonsistenz (release consistency) |

Subdiagram für Client: planuml.

@startuml

folder Client{

node app

node shared

node topologyEditor

node core

node home

node navibar

node sidebar

}

app .. core

app .. topologyEditor

core .. shared

core .. home

core .. navibar

core .. sidebar

topologyEditor .. shared

@enduml

Abb2:

@startuml

folder UI{

node app

node shared

node topologyEditor

node core

node home

node navibar

node sidebar

}

node app #green

node topologyEditor #red

node core #yellow

node home #yellow

node navibar #yellow

node sidebar #yellow

node shared #bluelight

app .. core

app .. topologyEditor

core .. shared

core .. home

core .. navibar

core .. sidebar

topologyE