



TECHNISCHE
UNIVERSITÄT
DRESDEN

Praktikum Softwaretechnologie

Abschlusspräsentation

Gruppe 3

Fahrgastinformationssystem Eisenbahnlabor

Dresden, 02.02.2016



DRESDEN
concept
Exzellenz aus
Wissenschaft
und Kultur

Gliederung

- 1 Zielstellung
- 2 Objektorientierte Analyse
- 3 Objektorientierter Entwurf
- 4 Implementierung
- 5 Rückblick

- 6 Demonstration

Gruppe 3

Eric Schölzel

3. Semester

Diplom Informatik

Oliver Schmidt

3. Semester

Diplom Informatik

Robert Mörseburg

3. Semester

Diplom Informatik

Jonas Schenke

3. Semester

Bachelor Informatik

Zdravko Yanakiev

3. Semester

Bachelor Informatik

Betreuer

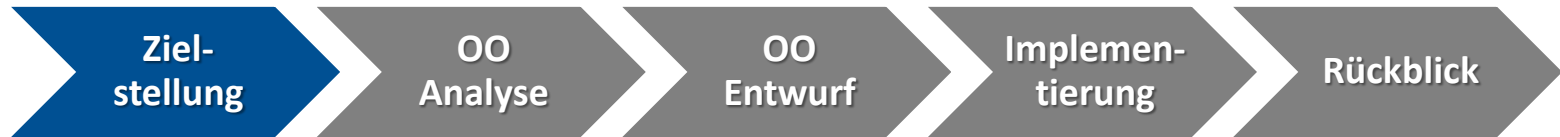


Dipl.-Medieninf. Ronny Kaiser

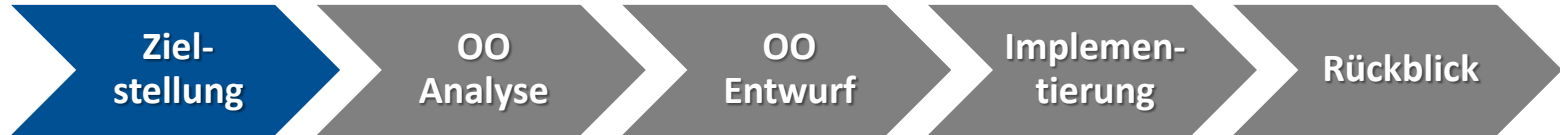
Fakultät Informatik

Institut für Software- und Multimediatechnik

Lehrstuhl Softwaretechnologie



- Webbasierte Ankunfts-, Abfahrts-, und Zuglaufanzeige
- Auswahl nach Bahnhof und Uhrzeit
- Optional Auswahl der Zuggattung
- Anzeige von Zwischenhalten
- Anzeige von Echtzeitinformationen (Verspätung, etc.) vom Fahrplanserver



- Orientierung: Online-Abfahrtsanzeige der Bahn

DB BAHN Startseite | Kontakt | Häufige Fragen | A A A

Angebotsberatung Fahrplan & Buchung Services BahnCard Geschäftsreisen Urlaub Meine Bahn

Abfahrt und Ankunft

Bahnhof / Haltestelle

Datum / Zeit

☒ Abfahrt ☐ Ankunft

Linie / Zugnummer (optional)

Verkehrsmittel ☒ ICE ☒ EC ☒ D ☒ NV ☒ S ☐ U ☐ B

Aktueller Abfahrtsplan von Dresden Hbf um 17:28 Uhr

[Aktualisieren](#)

Zeit	Zug	Richtung / Unterwegshaltestellen	Gleis	Aktuelles
↑ früher				
17:28	aktuelle Uhrzeit			
17:29	S 1	Schöna Dresden Hbf 17:29 - Dresden-Strehlen 17:31 - Heidenau 17:42 - Königstein(Sächs Schw) 18:07 - Bad Schandau 18:13 - Krippen 18:16 - Schöna 18:23	18	+0
17:30	S 1	Meißen Triebischtal Dresden Hbf 17:30 - Dresden Freiburger Straße 17:31 - Dresden-Neustadt 17:36 - Coswig(b Dresden) 17:55 - Neusörnewitz 17:59 - Meißen 18:04 - Meißen Triebischtal 18:09	19	+1
17:35	TL 74875	Zittau Dresden Hbf 17:35 - Dresden Mitte 17:38 - Dresden-Neustadt 17:40 - Dresden Industriegelände 17:45 - Arnsdorf(Dresden) 18:04 - Großharthau 18:11 - Taubenheim(Spree) 18:48 - Ebersbach(Sachs) 18:57 - Mittelherwigsdorf 19:22 - Zittau 19:28	11	+0



Muss-Kriterien

- Erreichbarkeit unter angegebener URL
- Kopfzeile (mit Logo, etc.)
- Fußzeile (mit Programmversion, etc.)
- Abfahrts- / Ankunfts- / Zuglaufanzeige
- Konfigurierbar
- Interaktiv (z.B. Anklicken eines Stops im Zuglauf)



Kann-Kriterien

- Verbindungsstatus
- Zuglaufanzeige als Perlenschnur
- Ausblenden der Uhrzeit bei Verbindungsproblemen
- Vor- / Zurückfunktion im Browser verwendbar
- Logo konfigurierbar



Zusätzlicher Kundenwunsch

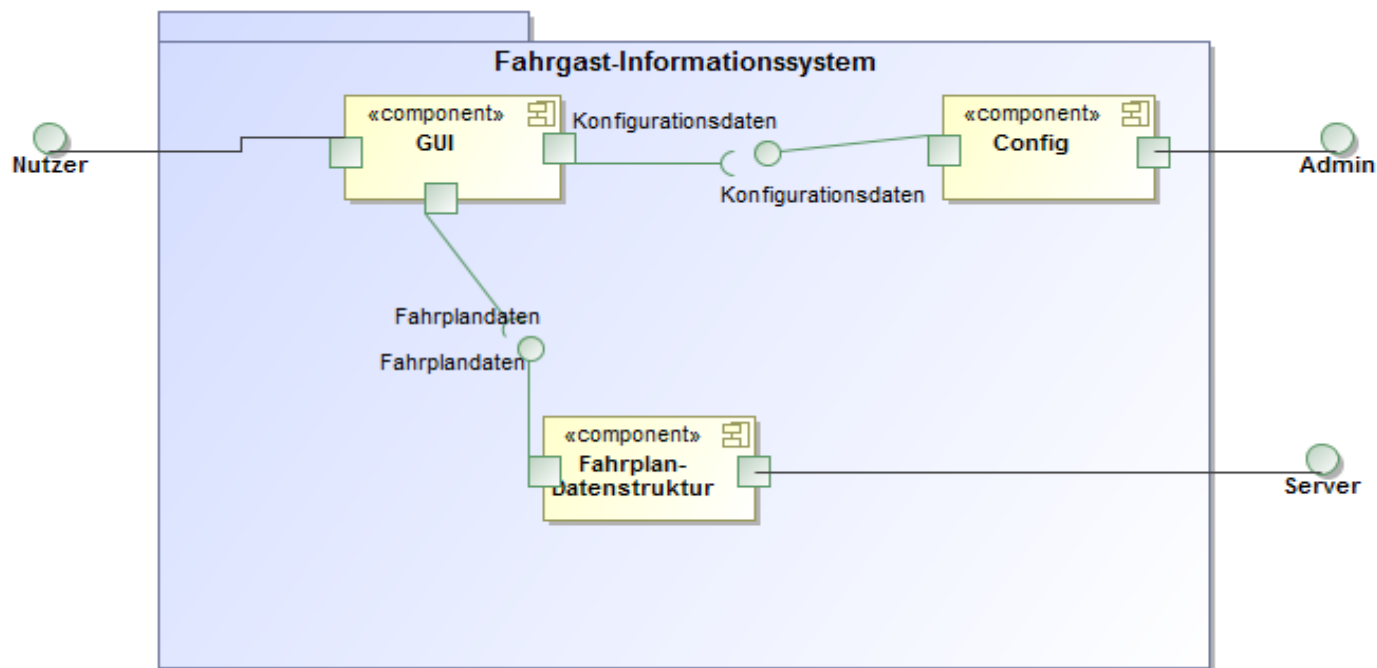
- Anzeige der Bahnhöfe auf interaktiver Karte
- Markieren des momentanen Bahnhofs

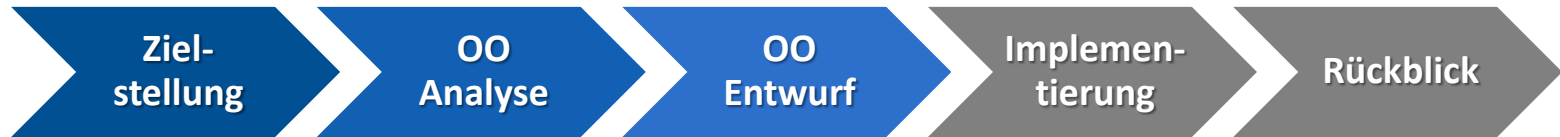


- GUI-Mockup
- Erstellung des Pflichtenheftes
- Erstellen eines Prototyps
- Planung mithilfe von UML-Diagrammen



Komponentendiagramm



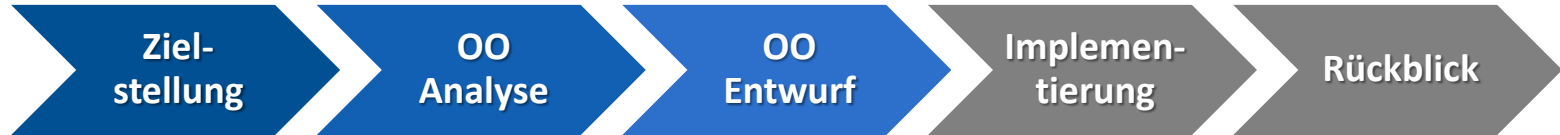


Web

- Anzeige, Nutzerinteraktion, Filterung
- Entwurf mithilfe von Mockups
 - funktional
 - Nutzerfreundlich, optisch ansprechend

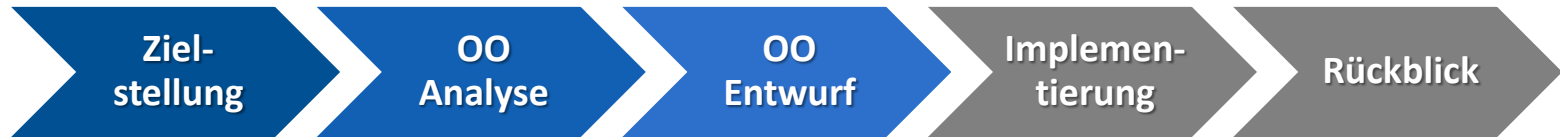
Datenstruktur

- Speichert Daten unabhängig von Datenquelle
- Wichtig: Datenintegrität
- Zentrales Bindeglied → Controller-Fassade



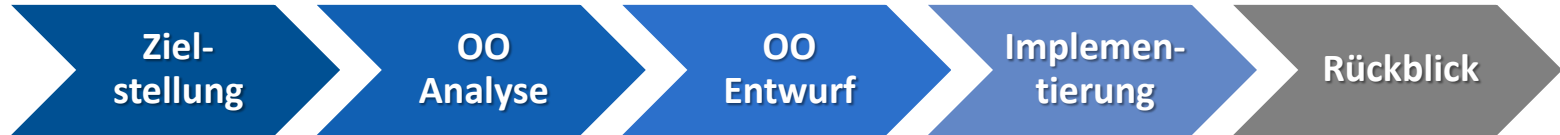
RailML[®]

- XML-Format zum Datenaustausch im Schienenverkehr
- Enthält Daten über die Infrastruktur, die Schienenfahrzeuge und den Fahrplan eines Eisenbahnsystems
- Der RailML-Parser stellt den anderen Programmkomponenten die Daten aus der RailML-Datei bereit
- Vorgehensweise: Trennung von XML-Parser und Datenstruktur → mehr Flexibilität



Telegramm-Struktur

- Entgegennehmen & Weiterreichen von Telegrammen
- Verschiedene Telegrammarten → Strategy
- Details während Entwurf unbekannt



- Java 1.8
- Spring Framework

Verlauf

1. Prototyp → nur kleine RailML
2. große RailML
3. Telegramme
4. Tests im Labor
5. Kundenwunsch

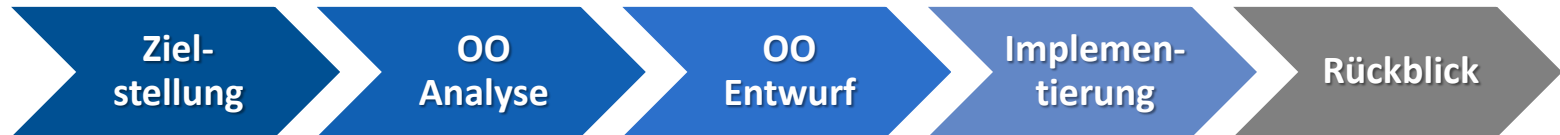


GUI

- Thymeleaf
- Datenweitergabe und Filterung im FisController

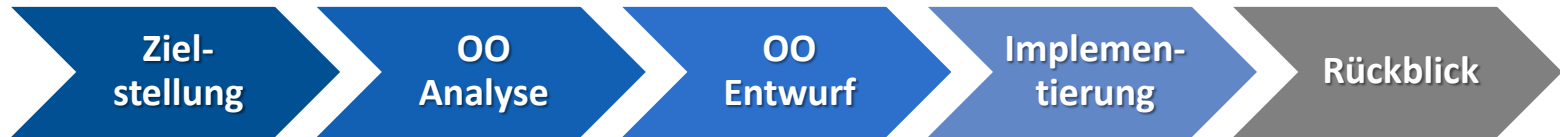
Datenstruktur

- Selbst unabhängig vom Spring-Framework
- TimetableController wertet Telegram-Objekte aus oder lädt Offline Fahrplan



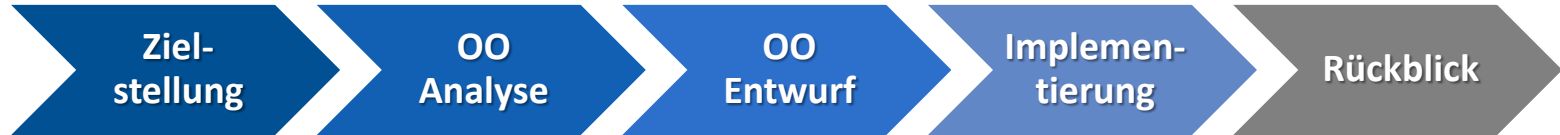
RailML[®]-Parser

- Implementierung unter Benutzung des Spring Frameworks (Spring OXM)
- Laden der ganzen Datei in den Speicher
- Anschließende Verarbeitung in die interne Datenstruktur



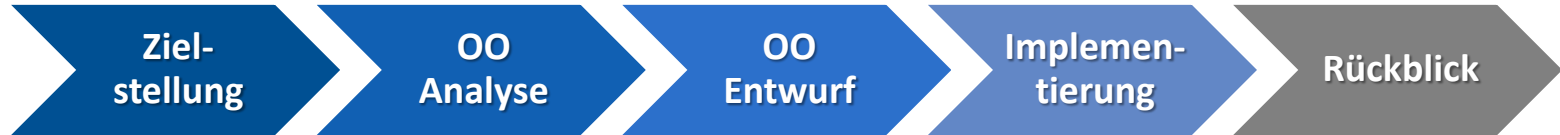
Telegramm-Struktur

- Asynchrones Empfangen
- Parsen der Byte-Arrays → Little Endian, Spezifikation
- Synchrone Verarbeitung
- Modellierung der Telegram-Datentypen in Java
- Übergabe an Datenstruktur durch Events



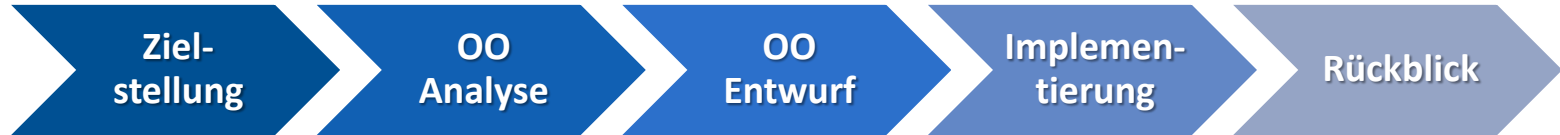
Kundenwunsch

- Karte des Eisenbahnnetzes
- Realisiert mithilfe von HTML5/JavaScript (Canvas)



Probleme bei der Implementierung

- RailML nicht korrekt
- Telegrammspezifikation
- Telegrammteil sehr schwierig zu testen



- Aufgabenstellung erfüllt
- Enge Rückkopplung mit Kunden
- Kommunikation im Team
 - Git, GitHub
 - Wöchentliche Treffen
- Frameworks
 - Fokussierung
 - Hoher Einarbeitungsaufwand



6 Demonstration der Anwendung



Vielen Dank für Ihre Aufmerksamkeit!