Modellbasierte Zuverlässigkeitsanalyse

Masterseminar SS 2014

Vitalij Stepanov

AIRBUS GROUP

Inhalt

- Einführung 
- Anwendungsfall
- Aktuell: Projekt 2
- Ziele der Masterarbeit
  - Vorgehen
- Chancen/Risiken
- Literatur

Einführung

- EU-gefördertes Forschungsprojekt
- Modelbasierte Entwicklung
- Integration von Werkzeugen in LifeCycle-Management Systeme
- Basierend auf offenen Standards wie OSLC

AIRBUS GROUP

Einführung

- Probleme der Integration
  - Viele dedizierte Werkzeuge
  - Verteilte Inhalte
  - Schnittstellen passen nicht zueinander
  - Inkonsistenz durch Datenkopien
- Verwendung von offenen Standards wie
  - Open Services for Lifecycle Collaboration (OSLC)

Einführung

Erwartungen

- Interoperabilität
- Traceability
- Datenkonsistenz
- Wiederverwendung
- Entscheidungsunterstützende Analysen
- Herstellerunabhängigkeit

Inhalt

- Einführung
- Anwendungsfall
- Aktuell: Projekt 2
- Ziele der Masterarbeit
  - Vorgehen
- Chancen/Risiken
- Literatur

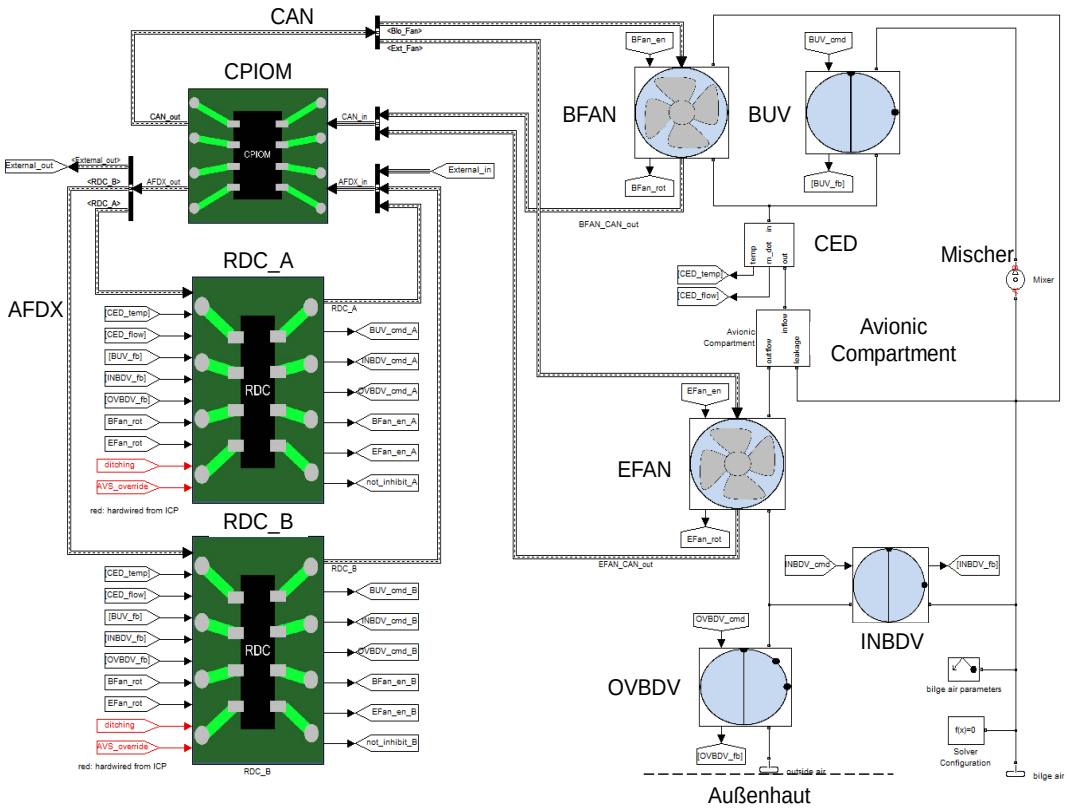
Anwendungsfall

- Environmental Control System (ECS)
  - Bereitstellung von Atemluft
  - Regelt die Temperatur
  - Kabinendruck
  - Kühlt Systeme
  - Saugt Rauch ab

- Sehr hohe Anforderungen an Ausfallsicherheit daher
  - Redundante Auslegung von Komponenten
  - Anwendung von formalen Methoden zur Verifikation

Environmental Control System

- ECS Simulink Modell von Marvin Tunnat [Tunnat2011]
- Beschreibt das Verhalten von Komponenten und Netzwerks
- Enthält Modellierung von
  - Funktionalen Anforderungen
  - Zuverlässigkeitsanforderungen
  - Sicherstellung von Alarmanzeige
- Ausfall von Komponenten durch boolesche Variable
- Entspricht nicht ganz der Wirklichkeit



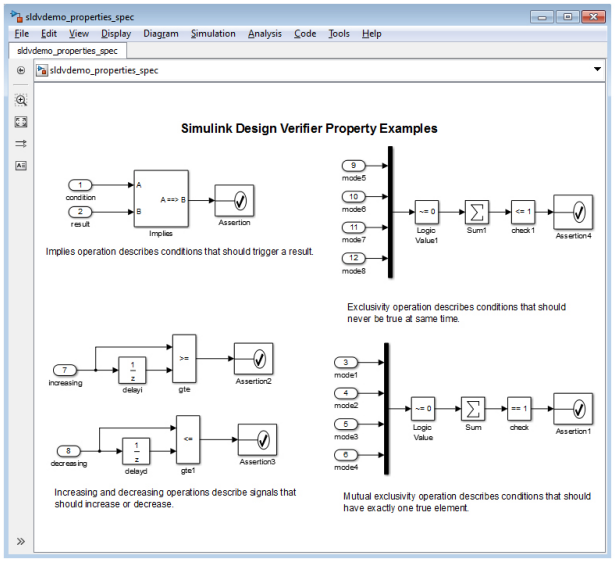
[1]

Simulink Design Verifier

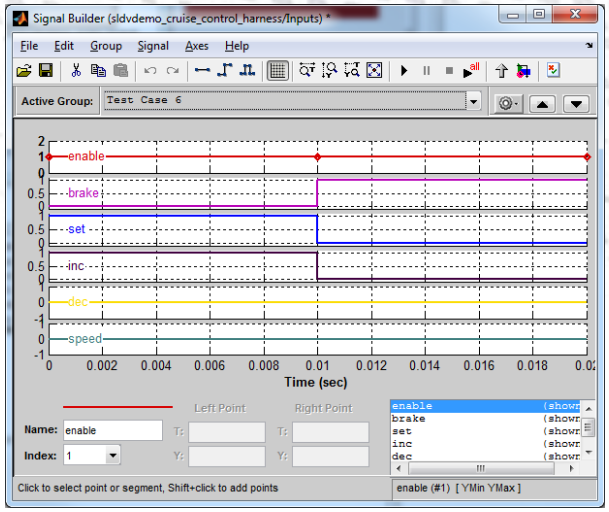
- Toolbox zur Modelverifikation
  - Blöcke und Funktionen zur Modellierung funktionaler und sicherheitsspezifischer Anforderungen
- Erkennung von
  - fehlerhafter Logik
  - Integer- und Festkommaüberläufen
  - Teilung durch Null
  - Verletzungen der Entwurfseigenschaften
- Verfahren zur formalen Analyse
- Eigenschaftenprüfung mit Generierung von Nachweisen durch Gegenbeispiel

Verifikation mit Simulink Design Verifier

- *Prove Objective* ist die negierte Anforderung
- Wird mit dem Gegenbeispiel widerlegt (Counter Example)
- Beispiele der Anforderungen:
  - Beim Ausfall von Ventilator soll der Backup-ventil innerhalb von 5 Sek. aufgehen
  - Wenn Flugzeug Notwasserung ausführt, soll die Aussenklappe geschlossen sein
- Unterbricht Verifikation bei der ersten Verletzung



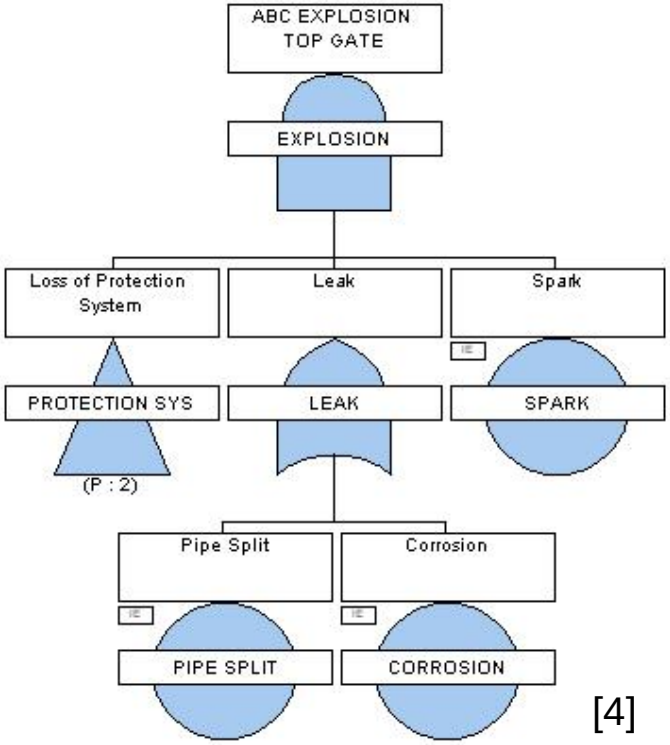
[2]



[3]

Zuverlässigkeitsanalyse

- Fehlerbaumanalyseverfahren eng. Fault Tree Analysis (FTA)
- Beschreibt alle mögliche Ausfälle der Funktion
- Bestimmt die Wahrscheinlichkeit eines Systemausfalls
- Wurzel steht für den unerwünschten Effekt, der nicht eintreten darf
- Minimal Cut Set enthält die Liste der Ereignisse, die zum Ausfall führen



[4]

Ist-Zustand

- Fehlerbäumen werden derzeit nur manuell von funktionalen Modellen abgeleitet
- Es gibt keine OSLC Implementierung für Matlab
- Keine Anbindung an FaultTree+
- Verlinkung von Daten nur unidirektional und eingeschränkt möglich

- Vorteile durch Automatisierung
  - Früherkennung von Designfehler
  - Weniger Iterationen
  - Schnellere Reife von Produkten

AIRBUS GROUP

Inhalt

- Einführung
- Anwendungsfall
- Aktuell: Projekt 2
- Ziele der Masterarbeit
  - Vorgehen
- Chancen/Risiken
- Literatur

AIRBUS GROUP

Aktuell Projekt 2

- Flexibler, einfacher Konzept zur Automatisierten Injektion von Fehlverhalten
- Möglichkeit der Stimulierung von Außen
- Design Verifier zur Überprüfung der Anforderungen
- Einfache Erweiterung von Simulink Bibliothek um zusätzlichen Fehlerarten
- Designer bestimmt die Stellen zur Fehlerinjektion
- Automatische Generierung von Fehlerbäumen

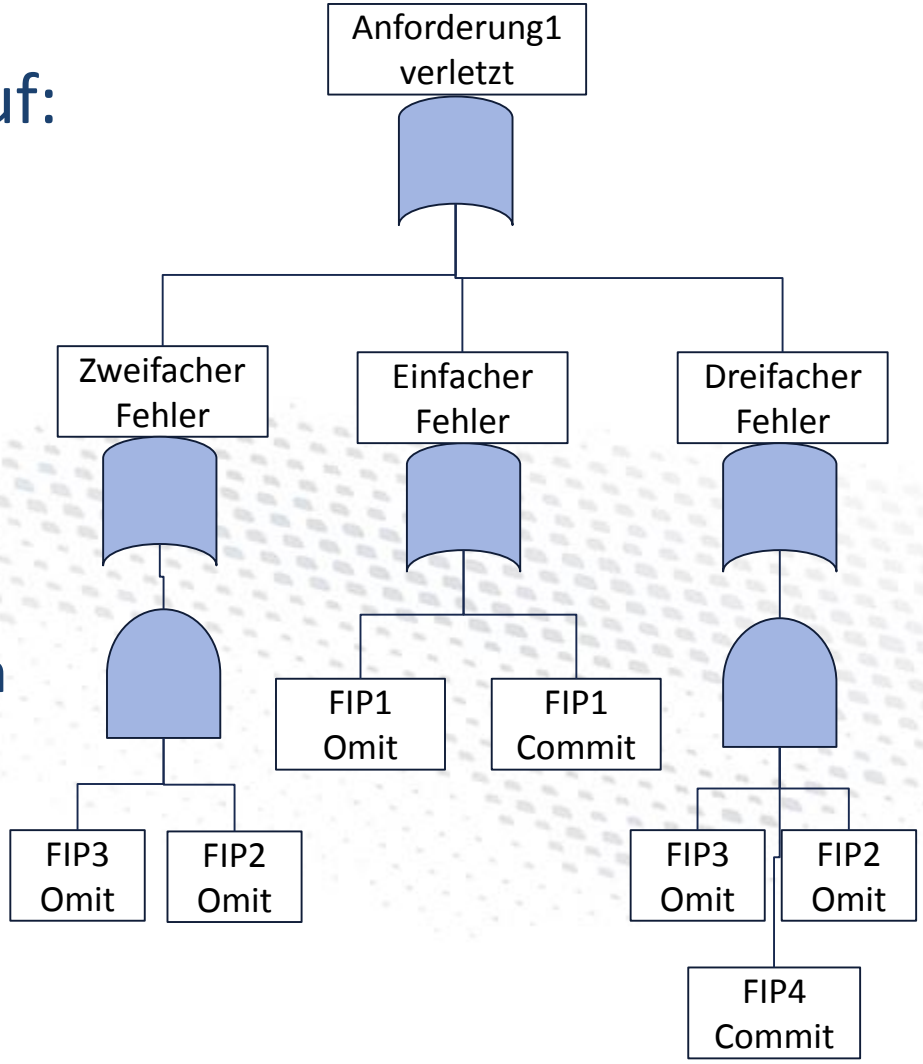
AIRBUS GROUP

Konzept der Fault Injection

- Modellierung des unerwünschten Verhaltens
- Durch die Erweiterung de Simulink Bibliothek mit:
  - Platzhalter für Injektion
    - Fault Injection Point (FIP)
  - Ausfallarten [Papadopolous&Maruhn2001]
    - Omit
    - Commit
    - Value Failure
- Designer definiert die Liste der möglichen Fehler für jeden FIP
- Fault Injection prüft jede Konfiguration durch die
- Verwendung von Design Verifier

Fehlerbaum Generierung

- Reduzierung des Zustandsraumes auf:
  - Einfache Fehler
  - Zweifache Fehler
  - Dreifache Fehler
- Keine Kombinationen mit bereits festgestellten Verletzungen
  - Ein einfacher Knoten hat niemals einen Nachfolgerknoten
- Zur Übersichtlichkeit Gruppierung nach der Größe
  - Failure of size X



Inhalt

- Einführung
- Anwendungsfall
- Aktuell: Projekt 2
- Ziele der Masterarbeit
  - Vorgehen
- Chancen/Risiken
- Literatur

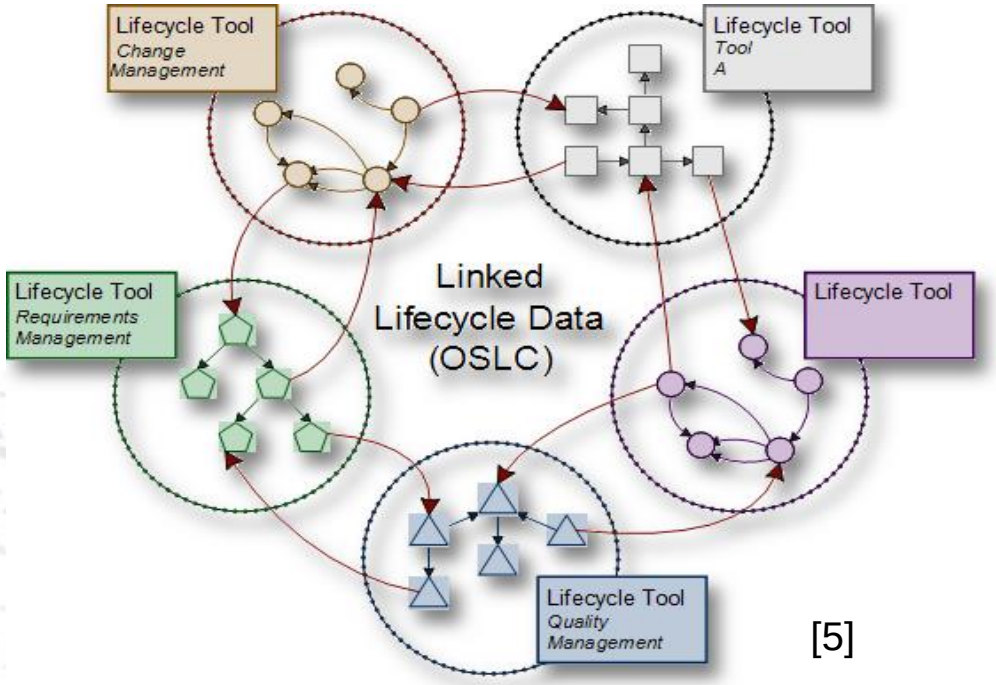
Ziele der Masterarbeit

- Definition der Ontologie für Safety-Analyse Domäne
- Implementierung des OSLC Adapters
- Implementierung des OSLC Plug-Ins für FaultTree+
- Realisierung von Stimulation der Fehlerinjektion von Außen
- Kompatibilität mit Jazz Architektur
  Zur Verlinkung mit Anforderungen

AIRBUS GROUP

Eigenschaften von OSLC

- Ist ein offenes Model
  - Verlinkung von unbekannten Inhalten [Berners-Lee2006]
- RESTful web Services
  - GET
  - POST
  - PUT
  - DELETE
- Verwendet Ressource Description Framework (RDF)
  - Ein Datenmodel im RDF/XML Format



[5]

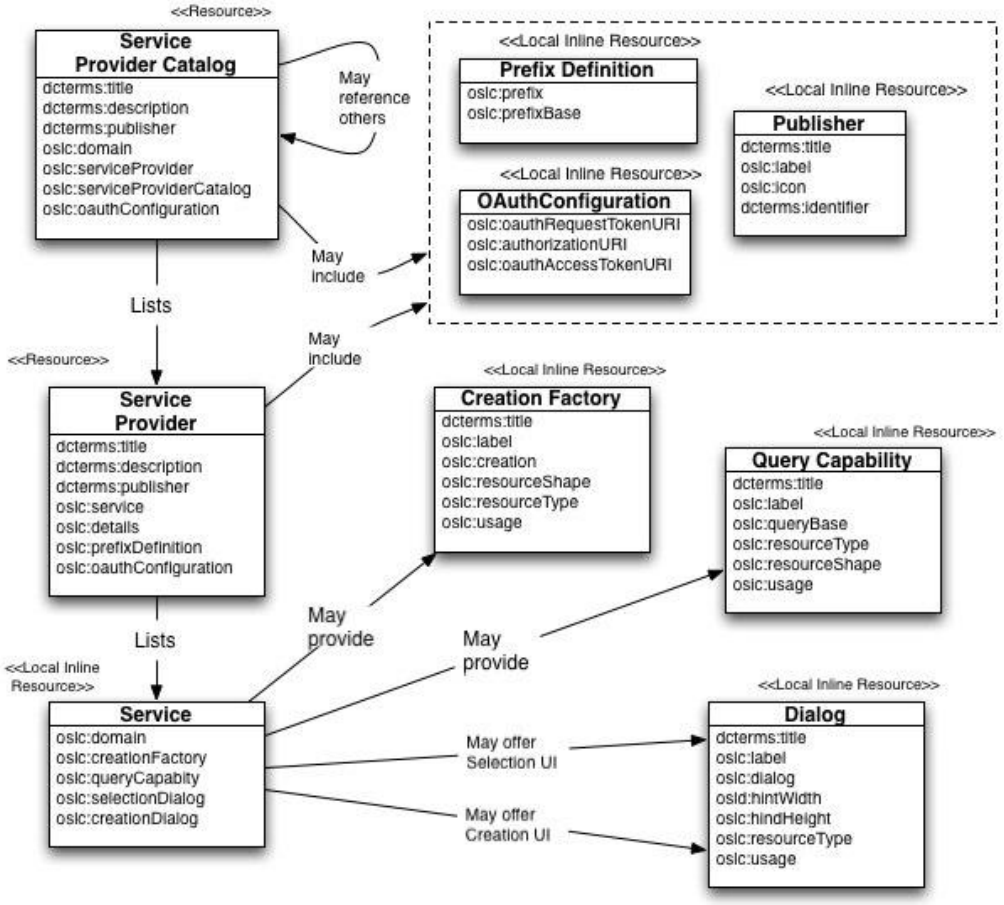
AIRBUS GROUP

OSLC Spezifikation

- OSLC Core Spezifikation
  - Beschreibt allgemein das Verfahren
- Domänenspezifische Spezifikationen
  - Requirement Management (RM)
  - Architecture Management (AM)
  - Change Management (CM)
  - Quality Management (QM)
- Keine Spezifikationen für:
  - Safety Analyse (SA)

OSLC Core 2.0 Spezifikation

- Service Provider
  - Hat eine eindeutigen URL
  - Ist ein Container mit Ressourcen
  - Clients können die enthaltenen Ressourcen auflisten
- HTTP GET auf Service Provider liefert die Metadaten mit URLs für:
  - oslc:queryBase Suchen von Ressourcen
  - oslc:creation Erstellung von Ressourcen
- Delegated UI Dialog
  - Zur Erstellung und
  - Auswahl von Ressourcen
- OSLC Ressource
  - Definiert durch RessourceShape in einer der Domainspezifikation [Ryman2013]



[2]

AIRBUS GROUP

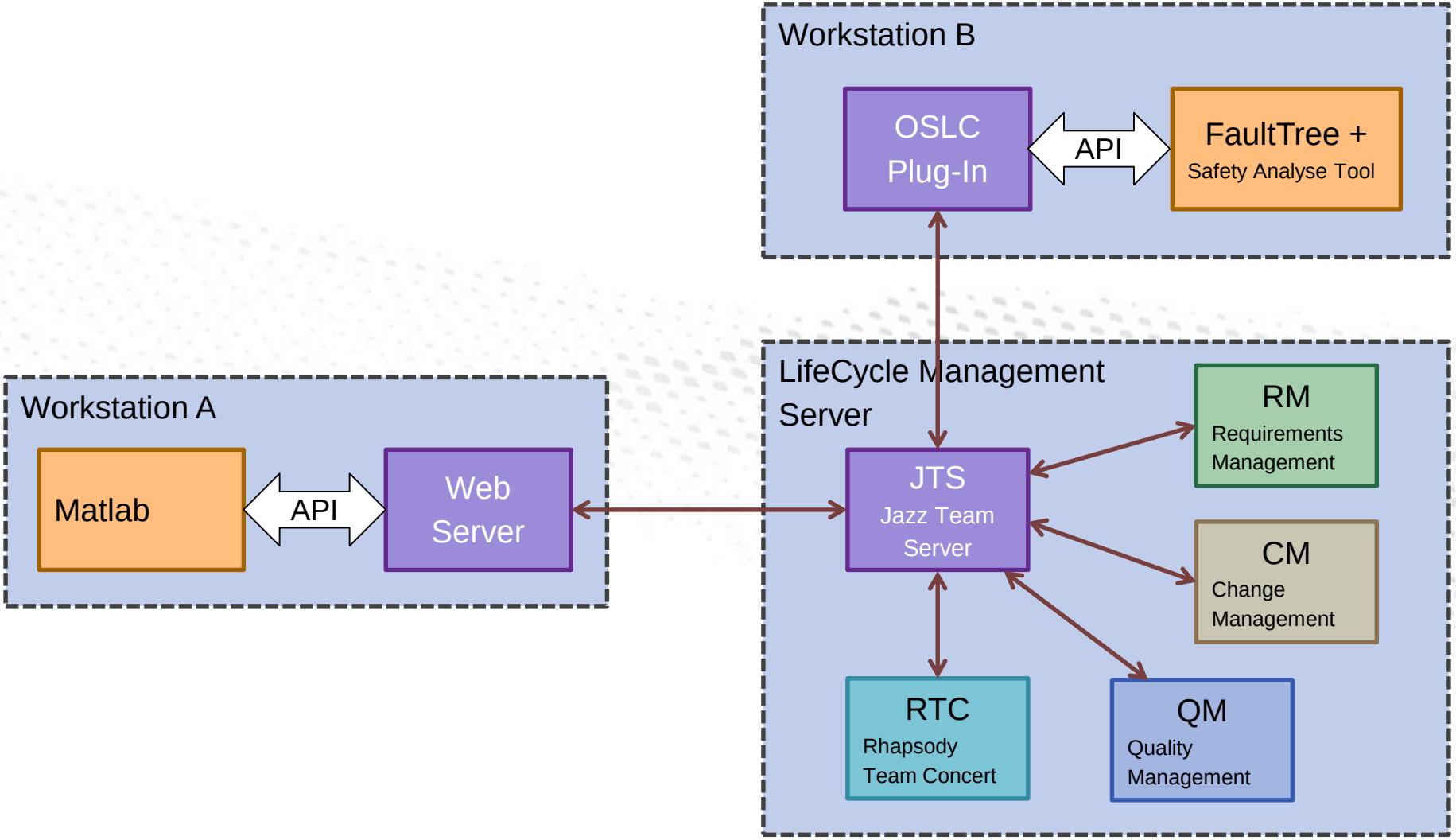
Vorgehen

- Entwickeln einer Systemarchitektur
- Integration von Komponenten
- Validieren des Produktes am Modell des Anwendungsfalls
- Integration mit Jazz LifeCycle Management System

OSLC Adapter

- Open-Source Referenzimplementierungen Existieren bereits
- Bildet die Projektstruktur hierarchisch auf OSLC Konzepte ab
- Stellt die Ressourcen OSLC-konform zur Verfügung
- Implementiert die Domäne Safety Analyse (SA)

Integration mit Jazz LifeCycle Management



Inhalt

- Einführung
- Anwendungsfall
- Aktuell: Projekt 2
- Ziele der Masterarbeit
  - Vorgehen
- Chancen/Risiken
- Literatur

Chancen

- Mehrere Lücken die ich mit meiner Arbeit schließen möchte

- Automatisierte Zuverlässigkeitsanalyse
  - Vereinfachung der Vorgänge
  - Reduzierung der Fehleranfälligkeit
  - Spezifikation von Safety Analyse (SA)

- Interoperabilität
  - Hohe Granularität der Datendarstellung bietet viele Möglichkeiten zur Wiederverwendung und Verlinkung
  - Datenkonsistenz durch Vermeidung von Kopien
  - Ermöglicht Plattformübergreifende Analysen

Risiken

- Performance
  - Kombinatorisches Problem bei vollständiger Überdeckung
  - Abhängig von der Modellgröße
- Umfang
  - Aufgrund der fehlenden Erfahrungen kann der Umfang nicht genau abgeschätzt werden
- Komplexität
  - Lässt sich sehr schwer einschätzen

Literatur

[Ryman2013]

Arthur G. Ryman, Arnaud J Le Hors, Steve Speicher:
OSLC Resource Shape A language for defining constraints on Linked Data

http://events.linkeddata.org/ldow2013/papers/ldow2013-paper-02.pdf

[Berners-Lee2006]

Tim Berners-Lee: Linked Data

http://www.w3.org/DesignIssues/LinkedData.html

[Papadopolous&Maruhn2001]

Yiannis Papadopolous & Mathias Maruhn:

Model-Based Synthesis of Fault Trees from Matlab-Simulink models.

In: 0-7695-1101-5/01 IEEE

[Tunnat2011]

Marvin Tunnat: Integration modellbasierter Methoden in den Entwicklungsprozess hybrider Flugzeugregelungssysteme am Beispiel des Ventilation-Control-System

Abbildungen

[1] Martin Tunnat [Tunnat2011]

[2] http://www.mathworks.de/cmsimages/69905_wl_simulinkdv_fig3_wl.jpg

[3] Design Verifier Manual

[4] http://www.reliabilityeducation.com/intro_ft.html

[5] http://en.wikipedia.org/wiki/Open_Services_for_Lifecycle_Collaboration

AIRBUS GROUP

Vielen Dank für die Aufmerksamkeit

Fragen?