2019

Bahn 2.0

Frederik Duda (11126504)

John-Bryan Spieker (11125583)

29.8.2019

Inhaltsverzeichnis

Vorwort

Bei diesem Dokument handelt es sich um eine Arbeitsmappe, welche sich mit der Aufgabenstellung der Firma Capgemini befasst. Für die Bearbeitung der Aufgaben im Architekturdesign, wird ein UML-Tool benötigt. Hierfür wurde das UML-Tool Modelio verwendet, bei dem es sich um eine Open-Source Sofware handelt, welche die Standards UML2 und BPMN2 unterstützt[1]. Zweites ist für die Bearbeitung der Aufgabenstellung jedoch nicht weiter relevant.

Capgemini – das Unternehmen

Das Unternehmen Capgemini ist einer der weltweit führenden Anbieter von Technologie-Services, Digitaler Transformation sowie für Management- und IT-Beratung.

Capgemini wurde 1967 gegründet und hat seinen Hauptsitz in Paris (Frankreich). Die Unternehmensstruktur gliedert sich in vier Hauptgeschäftsfelder. Die Submarke Capgemini invent ist für die Consulting Services zuständig. Capgemini sogeti (Submarke) für Technology und Engineering Services. Die letzten beiden Felder Application Services und other Managed Services werden von Capgemini bearbeitet.[2] Außerdem arbeitet Capgemini mit vielen großen Unternehmen zusammen, wie z.B. Adobe, Microsoft und SAP.[3]

Aufgrund der regelmäßigen Besuche der Mitarbeiter von Capgemini wird dem Kunden der Projektfortschritt stetig vermittelt. Dadurch können eventuelle Missverständnisse frühzeitig beseitigt werden. Das führt dazu, dass das Unternehmen Capgemini besser agieren kann.

Aufgabenstellung

Das Bahnfahren soll sich verändern. Im neuen Zeitalter soll das Bahnfahren einfacher und unterhaltsamer werden. Dafür hat die Bahn 2.0 klare Ziele. Es sollen zwei neue Systeme eingeführt werden. Hierbei handelt es sich um das Ticket–Buchungssystem und das Sitzplatz-Reservierungssystem. Durch das Zusammenspiel der beiden neuen Systeme soll erreicht werden, dass es nicht mehr nötig ist, sein Ticket und den Sitzplatz manuell zu Buchen. Zusätzlich soll das Erlebnis im Zug verbessert werden. Um dies zu erreichen, müssen sowohl ein personalisiertes InTrain Infotainmentsystem, sowie Community- und Shopping-Möglichkeiten geschaffen werden.

[1] vgl. Wikimedia Foundation Inc.: „Modelio"
[2] vgl. Kaubisch/Klein: „Technische Architektur in der individuellen Softwareentwicklung" S.5
[3] vgl. Capgemini Service SAS: „Alle Partner"

Aufgabe 1 – Architekturskizzen

Die erste Aufgabe besteht darin, ein Architekturmodell nach der Vorgehensweise zu erstellen, wie sie in der Präsentation der Firma Capgemini vorgestellt worden ist. Dieses Modell beinhaltet eine sogenannte A-Architektur, sowie eine T- und TI-Architektur. Dieses Modell soll für das oben genannte „Bahn 2.0 System“ erstellt werden. Welches neben einer automatischen Sitzplatzzuweisung, auch eine Infotainmentsystem an den einzelnen Sitzplätzen zur Verfügung stellt und noch einige andere autonome Funktionen übernehmen soll.

A-Architektur

Die A-Architektur ist der Grundbaustein für ein Architekturmodell, da sich diese mit den Funktionen und Abläufen des zukünftigen Systems befassen. Dieser Abschnitt ist somit auch Technik frei und sehr Businessorientiert.

Abbildung 1 Komponenten des Systems für die Bahn 2.0

Beim ersten Schritt der A-Architektur, welche sich mit dem Analysieren der benötigten Komponenten beschäftigt, ergaben sich die sechs Komponenten, welche in Abbildung 1 dargestellt sind. Nachstehend ist beschrieben, aus welchen Anforderungen sich die jeweiligen Komponenten ergeben.

Das Bahn 2.0 System benötigt ein *Abrechnungssystem*, welches sich um die gesamte Verwaltung und Abwicklung von Ticketverkäufen und Kauftransaktionen von Infotainmentinhalten beschäftigt. Des Weiteren wird natürlich das *Infotainmentsystem* selbst benötigt, das sich ausschließlich um die Anforderungen in dem Medien- und Entertainmentbereich kümmert, die dem Passagier an seinem Platz zur Verfügung steht. Damit der Passagier überhaupt einen Sitzplatz zugeordnet bekommt, wird natürlich auch eine *Sitzplatzverwaltung* benötigt, welche aus der Anforderung der automatischen Sitzplatzzuweisung hervorgeht. Damit der Passagier auch genau weiß, welcher Platz ihm zugeteilt wurde und wo sich dieser befindet, ist ein Subsystem, welches speziell für das Informieren des Passagiers dient, unumgänglich. Dieses Subsystem steht den Passagieren in den einzelnen *Infoterminals* zur Verfügung. Damit die anderen Systeme wissen, auf welchen Passagier sie ihre GUI anpassen müssen, braucht man natürlich noch eine *Profilverwaltung* in der alle wichtigen Daten des Passagiers abgefragt und aktualisiert werden können. Abschließend wird noch das *RFID-System*, das dazu dient, zu erkennen, ob Beispielweise ein Passagier gerade den Zug betritt oder seinen zugewiesenen Sitzplatz einnimmt, von Nöten sein.

Im zweiten Schritt, der Erstellung der A-Architektur, wird geprüft, mit welchen umliegenden Systemen das zukünftige System interagiert (welches der Einfachheit, im weiteren Dokument immer als „Bahn 2.0 System“ bezeichnet wird). Hier wird sowohl analysiert welche Systeme benötigt werden, damit das Bahn 2.0 System lauffähig ist, also Abhängigkeiten unserer Seite aus und welche Fremdsysteme auf das Bahn 2.0 System zugreifen. Außerdem wird in diesem Abschnitt festgelegt, welche User das System hat.

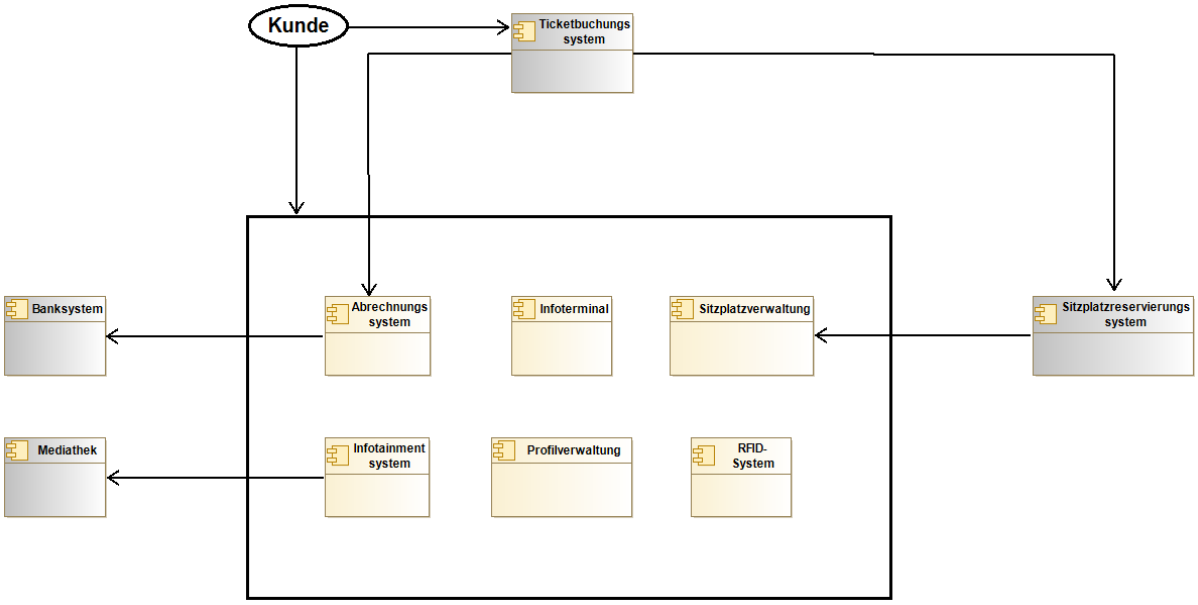

Abbildung 2 A-Architektur mit Fremdsystemen und Usern

Wie man aus Abbildung 2 entnehmen kann, hat das Bahn 2.0 System nur zwei Abhängigkeiten mit anderen Systemen. Dort wäre zum einen das *Banksystem*, welches als Abhängigkeit für das Abrechnungssystem dient. Diese Abhängigkeit entsteht dadurch, dass das System Überweisungstransaktionen anstoßen muss. Dies kann das System nur tun, wenn es eine Anbindung zu einer Schnittstelle einer Bank hat. Hier könnte beispielweise eine Verknüpfung mit einer Schnittstelle der Hausbank des Unternehmens sein. Diese wird genutzt, wenn das Unternehmen das Konto des Passagiers belastet möchte. Hierfür wird natürlich auch eine Einzugsermächtigung benötigt. Des Weiteren ist eine Abhängigkeit von dem Infotainmentsystem zu einem System, welches zum besseren Verständnis als *Mediathek* bezeichnet wird. Dieses Fremdsystem stellt die benötigten Filme und Serien zur Verfügung, welche dem Passagier angeboten werden können. Beispielweise könnte hier ein Fremdsystem wie Netflix oder Maxdome hinter verborgen sein, sollte die Bahn mit einem Drittanbieter einen Vertrag abgeschlossen haben. Jedoch ist es auch möglich, dass die Bahn selbst eine Mediathek auf einem Server zur Verfügung stellt. Grundsätzlich sollten die großen Mengen an Daten nicht direkt im Infotainmentsystem hinterlegt sein.

Da trotz der vielen Vorteile des einzuführenden Bahn 2.0 System, es dem Passagier trotzdem möglich sein soll ein Ticket zu kaufen, welches dem Passagier einen Sitzplatz im Zug reserviert (nicht eine freie Wahl eines Sitzplatzes), wird natürlich eine Schnittstelle nach außen benötigt. Auf diese Schnittstelle greifen die beiden Fremdsysteme *Ticketbuchungssystem* und *Sitzplatzreservierungssystem* zu. Zwischen diesen beiden Systemen besteht ebenfalls eine Abhängigkeit seitens des Ticketbuchungssystems. Diese entsteht daraus, dass nach einer erfolgreichen Abbuchung eine Reservierung im jeweiligen Zug hinterlegt wird. Wenn ein Kunde nun ein Ticket über das Internet oder einen Bahnschalter kauft, greifen diese GUI-Einstiegssysteme auf das Ticketbuchungssystem zu, um den Prozess einer Ticketbuchung anzustoßen. Der genaue Prozess, der sich hinter einer Ticketbuchung verbirgt, wird in Abschnitt „Use-Case Beschreibung“ von „Aufgabe 2 b) Analyse des Use-Case „Ticket Buchung““ noch einmal genauer beschrieben.

Aus den Anforderungen ergibt sich für das ganze System nur ein User. Dies ist der Zuggast, welcher in der Abbildung 2 als *Kunde* bezeichnet wird. Dieser User interagiert sowohl mit dem Bahn 2.0 System als auch mit dem Fremdsystem, welches auf das Bahn 2.0 System zugreift.

Selbstverständlich sind natürlich noch vereinzelt andere Nutzer mit dem System interagieren, wie Beispielsweise ein Administrator, der dafür sorgt, dass die Software einwandfrei installiert und gewartet wird oder ein Techniker, der Beispielweise Hardware austauschen muss, die für das System benötigt wird (siehe später Abschnitt T-Architektur). Jedoch sind dies vereinzelte Interaktionen, da sich die A-Architektur jedoch nur mit Power-Usern befasst, wurden diese User nicht mit in die A-Architektur aufgenommen.

T-Architektur

Im zweiten Teil der Erstellung einer Architekturskizze wird die sogenannte T-Architektur erstellt. Diese befasst sich mit den Abhängigkeiten innerhalb des Systems und den Geräten, welche später mit dem System interagieren.

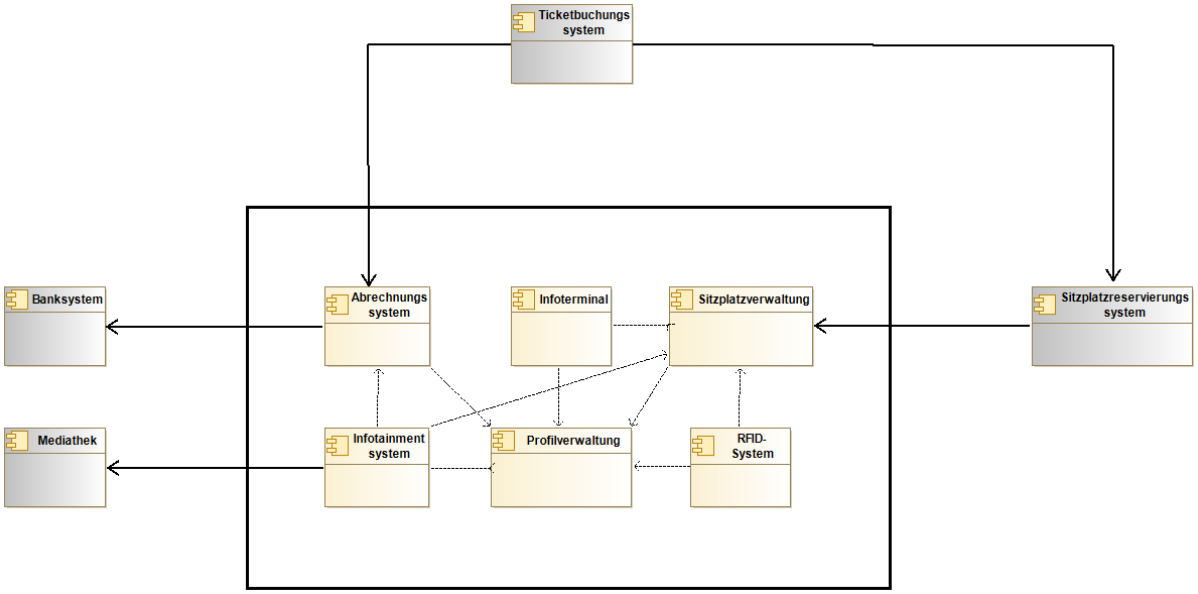

Abbildung 3 Abhängigkeiten innerhalb des Bahn 2.0 Systems

Im ersten Schritt werden die Abhängigkeiten innerhalb des Systems analysiert. Die aus dieser Analyse resultierenden Abhängigkeiten sind in Abbildung 3 zu sehen. Im nachstehenden Abschnitt wird erläutert wie es zu den Ergebnissen kommt.

Betrachtet wird zunächst das Infoterminal. Dies hat eine Abhängigkeit an die beiden Komponenten Profilverwaltung und Sitzplatzverwaltung. Diese resultieren aus der Anforderung, dass beim Einsteigen in den Zug das Ticket gescannt wird und dem Passagier sein zugeteilter Sitz angezeigt wird. Voraussetzung für das korrekte Anzeigen des Sitzplatzes ist, dass der Sitzplan geladen wird. Dieser wird aus der Sitzplatzverwaltung geladen. Des Weiteren hat der Passagier die Möglichkeit, sich sein aktuelles Guthaben/Kontostand sowie historische Daten (z.B. vergangene Fahrten) auf dem Infoterminal anzeigen zu lassen. Dafür müssen diese Daten aus der Profilverwaltung geladen werden.

Als Nächstes wird die Abhängigkeit zwischen dem Abrechnungssystem und der Profilverwaltung betrachtet. Diese entsteht durch das Abwickeln einer Kauftransaktion. Wird so eine Transaktion vom Infotainmentsystem angestoßen, werden zum weiteren Bearbeiten die Kontodaten bzw. das Guthaben vom Nutzer benötigt. Sonst weiß das Banksystem nicht, von welchem Konto das Geld eingezogen werden muss. Alternativ muss natürlich das Guthaben des jeweiligen Passagiers im Profil angepasst werden.

Auch das Infotainmentsystem benötigt die Profilverwaltung, da es sonst nicht möglich ist, dem Passagier eine für ihn zugeschnittene Empfehlung anzuzeigen. Sollte der Passagier dann etwas Kostenpflichtiges auswählen, muss das Infotainmentsystem natürlich auch die Möglichkeit haben, diese Aktivität dem Abrechnungssystem mitzuteilen. Dadurch entsteht die Abhängigkeit zu dieser Komponente.

Die Sitzplatzverwaltung ist für das organisieren der Sitzplätze innerhalb des Zuges verantwortlich. Um diese Funktion zu erfüllen, braucht sie, neben einer Abhängigkeit zu einer Datenbank (dies wird in einem späteren Abschnitt erläutert), die Profilverwaltung. Diese wird benötigt, da hinterlegt werden muss, welcher Passagier, sich auf welchem Platz befindet. Hierfür braucht die Sitzplatzverwaltung wenigstens eine ID, am besten auch den Namen. Auch bei einer Reservierung braucht die Sitzplatzverwaltung diese relevanten Daten.

Das RFID-System benötigt sowohl die Sitzplatzverwaltung als auch die Profilverwaltung. Dies geht daraus hervor, weil das RFID-System erkennt, ob ein Passagier ein- respektive aussteigt. Um der Sitzplatzverwaltung aber überhaupt mitteilen zu können, welcher Passagier gerade ein- beziehungsweise ausgestiegen ist, muss das RFID-System erst einmal erkennen, um welche Person es sich handelt. Dies erfolgt dann über die RFID. Sonst wüsste die Sitzplatzverwaltung nicht, wann ein Sitz frei bzw. belegt ist.

Abschließend ist noch die Profilverwaltung zu betrachten. Diese hat zu keiner anderen Komponente eine Abhängigkeit. Somit würde man bei einem Deployen des Systems beispielsweise folgende Reihenfolge haben:

1. Profilverwaltung
2. Sitzplatzverwaltung
3. Abrechnungssystem
4. RFID-System
5. Infoterminal
6. Infotainmentsystem

Der nächste Abschnitt beschäftigt sich mit der Einführung einer Datenbank und GUI (Graphical User Interface) in das System. Da Datenbanken in der heutigen Zeit eine gute und solide Möglichkeit sind, Daten zu speichern, wurde sich für ein gängiges RDBMS (relationales Datenbankmanagementsystem) entschieden, wie in Abbildung 4 deutlich zu sehen ist. Eine Speicherung der Daten in einer Datenbank macht auch dahingehend Sinn, weil Millionen von Daten erzeugt werden, da für jede Person ein Profil angelegt wird. Dies allein erzeugt nicht eine große Menge Daten, doch das Stichwort an dieser Stelle ist „Historisierung“. Es soll dem Reisenden auch die Möglichkeit geben, seine früheren Fahrten nachzuvollziehen. Weiterführend werden noch Daten benötigt, welche für die Sitzplatzzuordnung erforderlich sind. Hinzu kommen noch kleinere vereinzelte Datengruppierungen, wie Beispielweise der aktuelle Stand des verlassenen Platzes, da dieser Stand an einem anderen Platz wieder abgerufen werden kann.

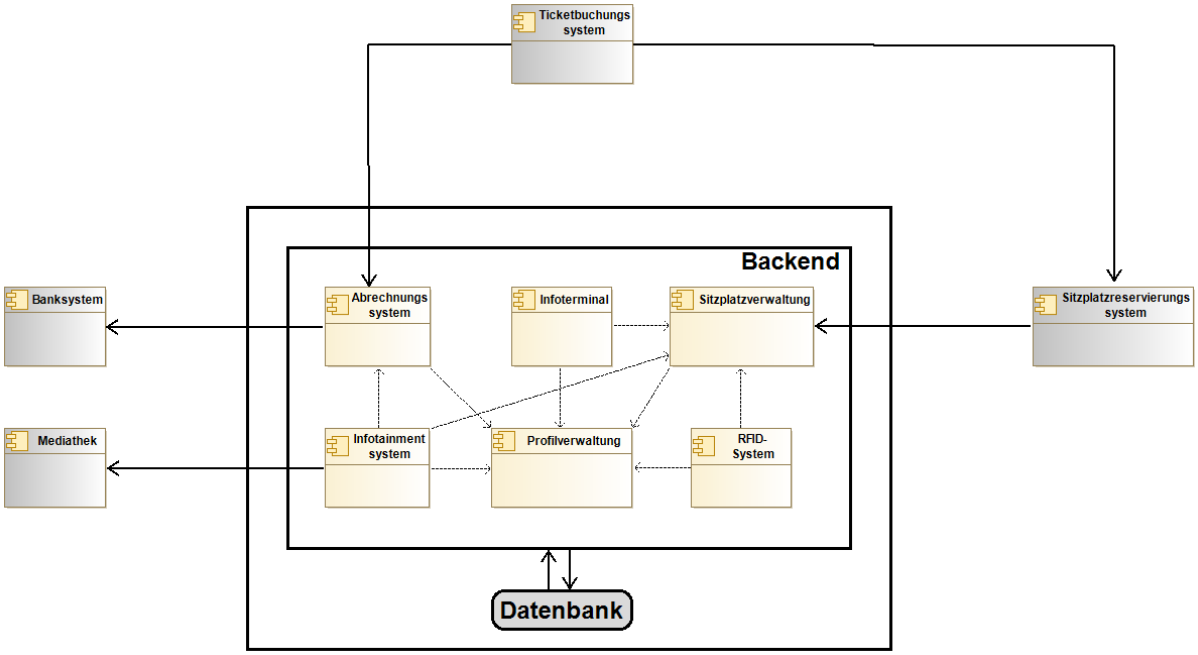

Abbildung 4 Einführung einer zentralen Datenbank für das System

Eine Einführung von embedded Datenbanken, wie beispielsweise HSQLDB, wäre an dieser Stelle auch zu empfehlen, da es einige Daten gibt, die relativ häufig geladen werden und somit immer erreichbar sein müssen. Ein Beispiel hierfür ist der Sitzplan des jeweiligen Zugmodels. Dieser sollte in der Sitzplatzverwaltung selbst vorhanden sein, damit diese nicht immer die Daten aus der zentralen Datenbank laden muss, wie viele Plätze und Waggons der Zug hat.

Eine genauere Analyse ergibt, dass es für das ganze Bahn 2.0 System keinen Sinn macht, ein GUI zu erstellen. Natürlich haben Komponenten, wie das Infoterminal und Infotainmentsystem, eine GUI, da der Passagier direkt mit diesen Komponenten interagiert, jedoch wären diese in den Komponenten selbst vorhanden. Da das Bahn 2.0 System als Ganzes jedoch nicht genutzt wird, sondern zu verschiedenen Zeitpunkten nur vereinzelte Komponenten, ergibt sich daraus keine Daseinsberechtigung für eine GUI.

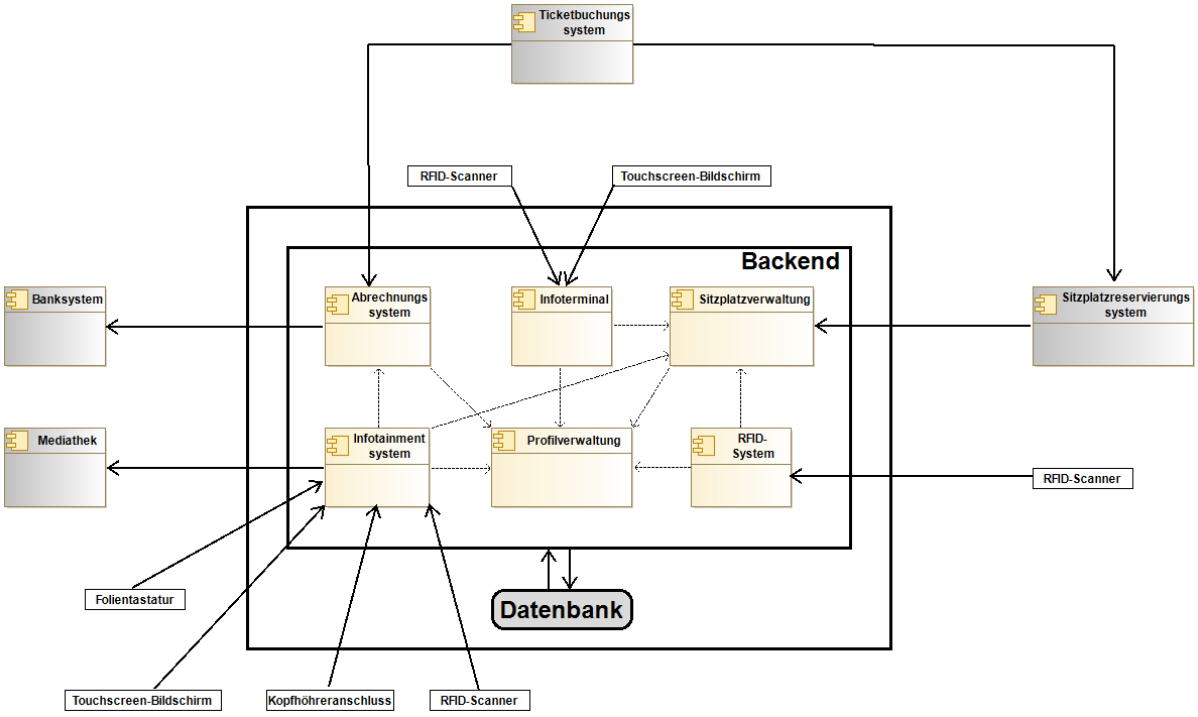

Abbildung 5 Benötigte Hardware für das Bahn 2.0 System

Der letzte Abschnitt der T-Architektur befasst sich mit der Festlegung der Hardware, welche dem System als Input dienen. In Abbildung 5 ist zu erkennen, dass mehrere Hardware mit dem System interagiert.

Zum einen wären da die RFID-Scanner. Diese sind die wichtigsten Komponenten im Zug, da durch den Scanner erkannt werden kann, wo sich ein Passagier gerade befindet. Also ob dieser gerade an der Schleuse, einem Infoterminal oder an seinem zugewiesenen Platz ist. Ohne diese Scanner, würde das System kaum bis gar nicht funktionieren. Wie in der Abbildung 5 zu sehen ist, gibt es drei Scanner bzw. handelt es sich hierbei eher um „Scanner-Typen“. Das bedeutet, dass es die Scanner für das RFID-System gibt, welche prüfen, ob ein Zuggast gerade in den Zug einsteigt oder aussteigt. Als Nächstes wären da die Scanner, welche auf das Infoterminal zugreifen, damit dieses erkennt, ob und welcher Passagier gerade vor dem Terminal steht. Abschließend gibt es noch die Scanner für das Infotainmentsystem. Diese sorgen dafür, dass das System weiß, dass sich der Passagier auf einen Platz gesetzt hat. Daher entfällt auch ein Scanner für die Sitzplatzverwaltung, da über das Infotainmentsystem eine Nachricht an die Sitzplatzverwaltung geschickt wird, dass der Passagier seinen zugewiesenen Platz eingenommen hat. Äquivalent gilt dies natürlich auch für das Verlassen des Sitzplatzes. Zusammenfassend bedeutet dies also, dass die Anzahl der Scanner im Zug, die Summe der Schleusen, Infoterminals und Sitzplätze sind.

Sowohl das Infotainmentsystem als auch das Infoterminal brauchen natürlich eine Möglichkeit, die programmierte GUI darzustellen. Dies erfolgt über Touchscreens. Außerdem dienen diese als

Möglichkeit, dem System ein Input zu geben. Das Infotainmentsystem benötigt weiterhin einen Kopfhöreranschluss, der dem Passagier die Möglichkeit bietet, seinen mitgebrachten Kopfhörer anzuschließen. Eine Anforderung das Bluetooth-Kopfhörer angeschlossen (verbunden) werden können, gibt es nicht. Es ist jedoch trotzdem sinnvoll, diesen Punkt im Hinterkopf zu haben, da in der heutigen Zeit, die meisten Menschen eher auf ein Funkgerät setzen als auf ein Kabelgerät (Stichwort: Erweiterbarkeit des Systems in Zukunft). Abschließend wird noch eine Folientastatur benötigt. Diese geht aus den Anforderungen des Kunden hervor. Dies ist auch sinnvoll, da es dem Passagier dann freisteht, auf dem Touchscreen die Eingabe zu tätigen, was bei einer kleinen Displaygröße schwierig werden könnte, oder ob er auf einer, ihm vertrauten Eingabemöglichkeit, Folientastatur den Input an das System übergibt.

TI-Architektur

Die TI-Architektur, welche auch Architektur der technischen Infrastruktur genannt wird, dient zur Darstellung und Festlegung welche Software benötigt wird, damit das System lauffähig ist.

Für das Bahn 2.0 System, wurde sich für drei eigenständige Laufzeitumgebungen entschieden (Abbildung 6). Zum einen eine Virtuelle Maschine (VM), auf der ein Datenbankserver läuft. Diese befindet sich nicht innerhalb des Zuges, sondern beispielsweise in einer Zentrale der Bahn. Dort befinden sich alle wichtigen Daten, wie z.B. die Informationen des Passagiers. Ein größeres Datenbanksystem im Zug laufen zu lassen wäre nicht der Sinnvoll. Da es der Bahn möglich ist, zu erkennen, ob ein Passagier gerade einsteigt oder den Zug verlässt, werden die Daten beim Betreten des Zuges temporär im Zug gespeichert. Beim Verlassen werden diese Daten dann wieder gelöscht. Dies hat den Vorteil, dass die Daten sehr schnell geladen werden können, da der Zug sich gerade an einem Bahnhof befindet. Außerdem ist die Datenbank nur dann ausgelastet, wenn Züge in einem Bahnhof sind. Somit greifen nicht alle Züge gleichzeitig auf die Datenbank zu.

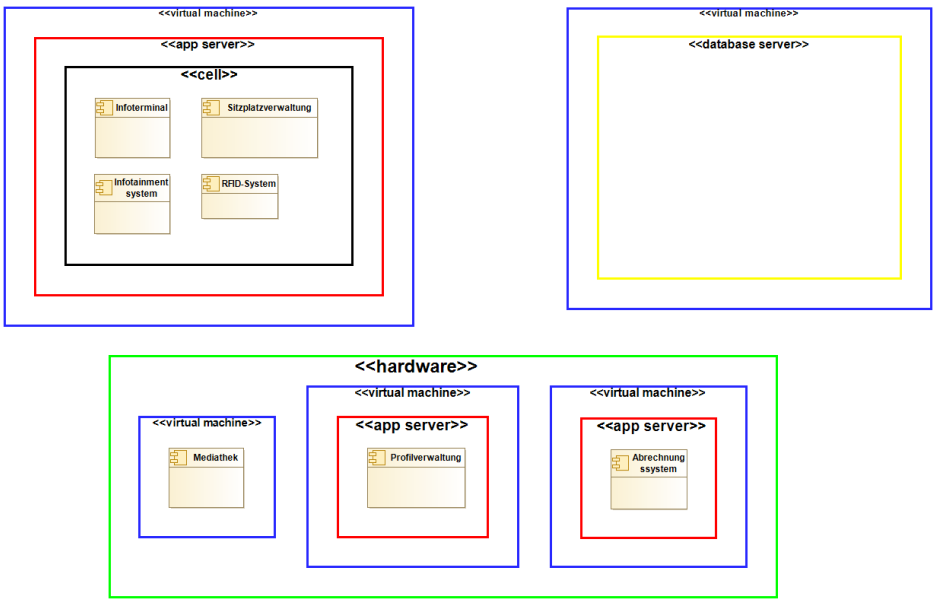

Abbildung 6 Infrastruktur des Bahn 2.0 Systems

Alle relevanten Systeme, die für die Funktionen im Zug selbst verantwortlich sind, benötigen eine VM mit einem laufenden Applikationsserver. Auf diesem werden dann die folgenden Komponenten deployt: Infoterminal, Sitzplatzverwaltung, Infotainmentsystem, RFID-System. Dies sind alles Komponenten, die nicht auf einem externen Server (also außerhalb der Bahn) laufen sollten, da sich ihre Funktion, immer an einen speziellen Gast anpasst, somit muss die Sitzplatzverwaltung nicht mehrere Gigabyte an Daten gleichzeitig verarbeiten, sondern nur die, die wichtig für den aktuellen Zug (und seiner Gäste) sind.

Abschließend bleiben noch Systeme bzw. Komponenten, die nicht im Zug selbst laufen müssen, da diese seltener benutzt werden. Diese können somit in einer Zentrale der Bahn laufen. Da diese Komponenten jedoch unterschiedliche Domänen abdecken, werden für die Komponenten Profilverwaltung und Abrechnungssystem jeweils eine unterschiedliche VM und Applikationsserver benötigt. Grundsätzlich könnten diese Systeme jedoch auf derselben Hardware laufen. Der Vollständigkeit halber, wurde die Mediathek-Komponente mit in diese Architektur aufgenommen. Da aus den Anforderungen nicht ersichtlich ist, ob diese über einen Drittanbieter bereitgestellt wird oder ob es sich um eine Mediathek der Bahn handeln wird. Ist erstes der Fall, wird diese Komponente aus der Abbildung 6 entfernt. Sollte jedoch Zweites der Fall sein, wird dieses System, welches größtenteils aus Video-Dateien bestehen wird, auf einer VM installiert sein. Da sich hinter dieser Komponente nicht sehr viel Logik und Komplexität verbergen wird, braucht man an dieser Stelle keinen Applikationsserver, da hier nur Daten abgefragt werden. Diese Menge an Daten sollte aber nicht dauerhaft in einem Zug gespeichert sein, da dies sehr Ressourcen aufwendig wäre.

Aufgabe 2 - Nicht funktionale Anforderungen

Nicht funktionale Anforderungen

Die nicht funktionalen Anforderungen (NFA) beziehen sich darauf, im Gegensatz zu den funktionalen Anforderungen, wie das System seine Leistung erbringen soll.[4] Die funktionalen Anforderungen beschreiben gewünschte Funktionalitäten (was soll das System tun / können) eines Systems bzw. des Produktes.[5] Deshalb werden die NFAs oft vom Auftraggeber vergessen, außer sie haben in der Vergangenheit in diesem Bereich schlechte Erfahrungen gemacht (z.B. zu geringe Verfügbarkeit oder schlechte Qualität). Zusätzlich wirken sich die nicht funktionalen Anforderungen auf die gesamte Architektur aus.

Jedoch stellt sich die Frage, wie man die Qualität einer Software bestimmt. Dafür wurde unteranderem die ISO-Norm 9126 erstellt. Es wird zwischen sechs Kriterien unterschieden. (Abbildung 7)

1. Anforderungen an die **Benutzbarkeit** des Systems (Welchen Aufwand fordert der Einsatz der Software von den Benutzern und wie wird die Software von diesen beurteilt?)
2. Anforderungen an die **Änderbarkeit** des Systems (Welchen Aufwand erfordert die Durchführung vorgegebener Änderungen an die Software?

Abbildung 7 Softwarequalität nach ISO-Norm 9126

3. Anforderungen an die **Effizienz** des Systems (Wie liegt das Verhältnis zwischen Leistungsniveau der Software und den eingesetzten Betriebsmitteln?
4. Anforderungen an die **Funktionalität** des Systems (Inwieweit besitzt die Software die geforderten Funktionen?)
5. Anforderungen an die **Übertragbarkeit** des Systems (Wie leicht lässt sich die Software in eine andere Umgebung übertragen?)

[4] vgl. Taentzer: „Einführung in die Softwaretechnik“
[5] vgl. Heini: „Funktionale, nicht funktionale Anforderungen“

6. Anforderungen an die **Zuverlässigkeit** des Systems (Kann die Software ein bestimmtes Leistungsniveau unter bestimmten Bedingungen über einen bestimmten Zeitraum aufrechterhalten?)[6]

Nichtsdestotrotz gibt es Probleme zwischen den einzelnen Merkmalen. Denn in der Regel ist es nicht möglich alle Qualitätskriterien gleichermaßen zu verbessern. Es gibt unteranderem Zielkonflikte der unterschiedlichen Kriterien. Dementsprechend ist es immer notwendig bei der Qualitätssicherung einer Software Schwerpunkte zu setzen und diese zu priorisieren.

Neben den beiden oben genannten Typen gibt es einen weiteren Typen, nämlich die Randbedingungen. Typische Randbedingungen sind eine Obergrenze von Kosten oder einzuhaltende Termine für den Abschluss des Projektes. [7]

Aufgabe 2 a) Analyse der Use Cases „Ein- / Aussteigen“ und „Film anschauen“

Einleitung

Dieser Abschnitt umfasst das Feststellen der nicht funktionalen Anforderungen (NFAs) für die beiden Use Cases „Ein- / Aussteigen“ und „Film anschauen“. Außerdem erfolgt eine Abschätzung der fachlichen Transaktionen bezüglich des Mengengerüstes (siehe unten) und deren Zugriff auf die beteiligten Geschäftsobjekte. Um die NFA für die beiden Use Cases bestimmen zu können, wird zunächst betrachtet, welche Komponenten benötigt werden und wie die beiden Use Cases ablaufen.

Use Case „Ein- und Aussteigen“ Beschreibung

Als Erstes wird der Use Case „Ein- / Aussteigen“ erläutert. In der Bahn 2.0 ändert sich für den Fahrgast beim Ein- bzw. Aussteigen kaum etwas. Statt dem klassischen Papierticket kauft sich der Fahrgast einmalig eine Bahn-Karte. Diese erhält er entweder am Automaten oder er beantragt sie übers Internet. Um dies zu erreichen, wird eine gültige Kreditkarte oder ein gültiges Girokonto benötigt. Anschließend werden wichtige Profildaten abgefragt und im Profil gespeichert. Ansonsten verändert sich für den Fahrgast nichts mehr. Er steigt in den Zug ein und wieder aus. Das RFID System wird vom RFID-Scanner benachrichtigt, dass der jeweilige Fahrgast ein- bzw. ausgestiegen ist. Beim Einsteigen erhält der Fahrgast einen Sitzplatz gemäß seinem Profil. Der Sitzplatz wird auf einem Display angezeigt. Deshalb ruft das RFID System beim Einsteigen die Sitzplatzverwaltung auf. Diese wiederum benötigt wichtige Daten, wie den letzten Sitzplatz, 1. oder 2. Klasse und ob der Fahrgast in der Handyzone sitzen möchte, die das System von der Profilverwaltung erhält.

[6] vgl. Wikimedia Foundation Inc.: „ISO / IEC 9126”
[7] vgl. Wikimedia Foundation Inc.: „Anforderung (Informatik)“

Nicht funktionale Anforderungen bezüglich des Use Case „Ein- und Aussteigen“

- Funktionalität:
  - Angemessenheit: Das System muss die Funktionen des Use Cases bereitstellen.
  - Richtigkeit: Das System darf jeden Platz im Zug nur einmal vergeben, solange der Reisende im Zug ist oder den Platz frei gibt.
  - Interoperabilität: Das System muss mit dem bisher bestehenden System (Reservierung) kompatibel sein.
  - Ordnungsmäßigkeit: Der Prozess muss sich an die gesetzlichen Bedingungen halten.
  - Sicherheit: Das System muss den unberechtigten Zugriff verhindern, damit persönliche Daten wie Banknummer, Name etc. nicht ausgelesen werden können (von der Bahn Karte).
  - Konformität: Für den Ablauf des Prozesses müssen ausschließlich gesetzliche Bedingungen eingehalten werden. Andere Konventionen gibt es an dieser Stelle nicht.
- Zuverlässigkeit:
  - Reife: Das System muss trotz auftretender Fehler den Fahrgast beim Ein- und Aussteigen erkennen. Es muss ein hoher Grad an Reife vorhanden sein.
  - Fehlertoleranz: Das System muss trotz Fehler in der Lage sein, den Reisenden beim Einstieg oder beim Ausstieg zu identifizieren.
  - Wiederherstellbarkeit: Falls das System versagt, muss das System spätestens am nächsten Halt wieder funktionieren. Damit die Reisenden beim Ein- und Ausstieg erkannt werden.
  - Konformität: Die Konformität bei diesem Prozess muss hoch sein, da dieser Prozess millionenfach am Tag benutzt wird.
- Benutzbarkeit:
  - Verständlichkeit: Die Verständlichkeit des Ein- und Aussteigens ist gering. Vom Reisenden wird keine Verständlichkeit für den Prozess verlangt, da er einzig die Bahn-Karte bei sich haben muss.
  - Erlernbarkeit: Der Prozess muss nicht neu erlernt werden, da dieser bereits ausgeführt wird.
  - Bedienbarkeit: Der Benutzer bedient den Prozess nicht.
  - Attraktivität: Die Attraktivität des Benutzers gegenüber dem Prozess ist gering, da der Reisende nur ein bzw. aussteigt.
  - Konformität: Der Grad der Konformität bezüglich der Benutzbarkeit muss nicht groß sein, da der Reisende kaum bis gar keinen Einfluss auf den Prozess hat.

- Effizienz:
    - Zeitverhalten: Das System muss unter 1s die Bahn Karte erkennen und dem Fahrgast ein Sitzplatz zuteilen.
    - Verbrauchsverhalten: Die Hardware muss in der Lage sein, auch bei höherer Auslastung (Spitzenzeiten) den Reisenden zu erkennen. Zusätzlich sollte der Prozess ressourcensparend umgesetzt werden, da sich der Prozess kaum noch ändern wird.
    - Konformität: Die Konformität bezüglich der Effizienz muss sich an dem Reisenden orientieren. Der Reisende möchte nicht länger als eine Sekunde auf die Identifizierung warten.
- Änderbarkeit:
    - Analysierbarkeit: Das Projekt Bahn 2.0 ist riesig. Dafür muss für die komplette Software eine Dokumentation angelegt werden, damit eine gute Analysierbarkeit gewährleistet ist. Wird in der Dokumentation ein Fehler begangen, hat dies eine starke Auswirkung auf die Analysierbarkeit.
    - Modifizierbarkeit: Das System muss nicht weiter modifiziert werden. Jedoch müssen auftretende Fehler, wie nicht Erkennung der Bahn-Karte, beseitigt werden.
    - Stabilität: Das System wird in jedem Zug eingesetzt und muss daher einen hohen Grad an Stabilität haben.
    - Testbarkeit: Der Aufwand für das Testen einer Änderung muss gering sein.
    - Konformität: Die Software muss einen geringen Grad an Konformität im Bereich der Änderbarkeit aufweisen. Eine Änderung des Systems sollte kaum vorkommen.
- Übertragbarkeit:
    - Anpassbarkeit: Das System läuft nur in den Zügen. Daher wird es nicht mit einer Umsiedlung konfrontiert.
    - Installierbarkeit: Die Software wird in jedem Zug (Zugsoftware) benötigt. Daher sollte die Installation einfach und schnell sein.
    - Austauschbarkeit: Es gibt bisher keine andere Software. Jedoch sollte man davon ausgehen, dass das System später austauschbar ist.
    - Koexistenz: Das System muss in der Lage sein auch in einem Dualbetrieb einwandfrei zu Laufen.
    - Konformität: Das System muss eine mittlere Konformität im Bereich der Übertragbarkeit bereitstellen.

Use Case „Film anschauen“ Beschreibung

Als Nächstes wird der Use Case „Film anschauen“ erläutert. Ein weiterer Schritt für die Bahn ist es eine Mediathek anzubieten. Dadurch wird das Bahnfahren komfortabler. Da dies eine neue Entwicklung für die Bahn ist, wird der Fahrgast vor neue Herausforderungen gestellt.

Zunächst nimmt der Fahrgast seinen zugewiesenen Sitzplatz ein. Der Fahrgast wird anschließend durch seine Bahn-Karte identifiziert. Nach der erfolgreichen Identifikation des Fahrgastes wird der Bildschirm freigeschaltet. Um den gewünschten Film in der Mediathek zu finden, kann der Fahrgast die Folientastatur und den Touchscreen benutzen. Sobald er den gewünschten Film gefunden hat, kann er ihn anschauen. Damit der Fahrgast auch den Ton hört, kann er den bereitgestellten Kopfhöreranschluss benutzen. Falls der Fahrgast einen Film bereits kostenpflichtig ausgeliehen und ihn noch nicht zu Ende gesehen hat, kann er diesen weiterschauen. Intern meldet der Scanner den Fahrgast beim Infotainmentsystem an. Dieses benötigt Daten von der Profilverwaltung und ruft die externe Mediathek auf.

Nicht funktionale Anforderungen bezüglich des Use Case „Film anschauen“

- Funktionalität:
    - Angemessenheit: Das System muss die Funktionen des Use Cases bereitstellen.
    - Richtigkeit: Das System muss dem richtigen Reisenden sein Programm anzeigen. Ein falsches Anzeigen des Programms muss zu 100% ausgeschlossen sein.
    - Interoperabilität: Das System muss mit der Mediathek funktionieren.
    - Ordnungsmäßigkeit: Der Prozess muss sich an die gesetzlichen Bedingungen halten. Bevor ein Film abgespielt werden darf, muss die Rechteverteilung des Filmes geklärt sein.
    - Sicherheit: Das System muss den unberechtigten Zugriff verhindern, damit die Daten nicht ausgelesen werden können.
    - Konformität: Während des Ablaufs des Prozesses müssen ausschließlich gesetzliche Bedingungen eingehalten werden.
- Zuverlässigkeit:
    - Reife: Das System muss auch bei auftretenden Fehlern in der Lage sein, Filme abzuspielen. Eventuell könnte man nur die bereits ausgeliehenen Filme ansehen.
    - Fehlertoleranz: Das System muss die ausgeliehenen Filme immer bereitstellen können.
    - Wiederherstellbarkeit: Damit der Reisende komfortabel reisen kann, muss das System spätestens beim nächsten Halt wieder funktionieren.

- Konformität: Die Konformität bei diesem Prozess muss hoch sein, da dadurch gewährleistet wird, dass das Bahnfahren komfortabler wird und dieser Prozess millionenfach am Tag ausgeführt wird.

- Benutzbarkeit:
  - Verständlichkeit: Wie ein Reisender einen Film auswählt, sollte einfach gehalten werden.
  - Erlernbarkeit: Der Benutzer muss sich einmal mit dem System (Mediathek) auseinandersetzen, damit er es verstehen kann.
  - Bedienbarkeit: Der Benutzer sollte schnell erkennen, wie er das System benutzt.
  - Attraktivität: Das System sollte sehr attraktiv für den Reisenden sein, da dadurch das Reisen angenehmer wird.
  - Konformität: Der Grad der Konformität bezüglich der Benutzbarkeit muss mittel groß sein, da der Reisende sich einmalig mit der Mediathek vertraut machen muss.
- Effizienz:
  - Zeitverhalten: Das System sollte auf Eingaben vom Benutzer nicht mehr als 1s benötigen.
  - Verbrauchsverhalten: Die Hardware muss in der Lage sein, auch bei höherer Auslastung (Spitzenzeiten) die Reisenden mit Filmen zu versorgen. Zusätzlich sollte der Prozess ressourcensparend umgesetzt werden, da sich der Prozess kaum noch ändern wird. Einzig die Filmauswahl wird sich stetig verändern.
  - Konformität: Die Konformität bezüglich der Effizienz muss sich an dem Reisenden orientieren. Der Reisende möchte nicht länger als eine Sekunde auf das System warten.
- Änderbarkeit:
  - Analysierbarkeit: Das Projekt Bahn 2.0 ist riesig. Dafür muss für die komplette Software eine Dokumentation angelegt werden, damit eine gute Analysierbarkeit gewährleistet ist. Wird in der Dokumentation ein Fehler begangen, hat dies eine starke Auswirkung auf die Analysierbarkeit.
  - Modifizierbarkeit: Das System muss beim fahrenden Zug eventuell mit einer schwachen Netzwerkverbindung arbeiten können. Im Bahnhof ist die Netzverbindung stärker.
  - Stabilität: Das System muss über einen hohen Grad an Stabilität verfügen, da diese Software millionenfach von Reisenden benutzt werden.
  - Testbarkeit: Das System sollte nicht während der Fahrt eines Zuges getestet werden. Vor dem Release sollte das System intern getestet werden.

- Konformität: Die Software muss einen geringen Grad an Konformität im Bereich der Änderbarkeit aufweisen. Eine Änderung des Systems sollte kaum vorkommen.

- Übertragbarkeit:
  - Anpassbarkeit: Das System läuft nur in den Zügen. Daher wird es nicht mit einer Umsiedlung konfrontiert.
  - Installierbarkeit: Die Software wird in jedem Zug (Zugsoftware) benötigt. Daher sollte die Installation einfach und schnell sein.
  - Austauschbarkeit: Es gibt bisher keine andere Software. Jedoch sollte man davon ausgehen, dass das System später austauschbar ist.
  - Koexistenz: Das System muss in der Lage sein auch in einem Dualbetrieb einwandfrei zu Laufen.
  - Konformität: Das System muss eine mittlere Konformität im Bereich der Übertragbarkeit bereitstellen.

Abschätzung der fachlichen Transaktionen bezüglich des Mengengerüstes

Schätzungen der fachlichen Transaktionen

Nachdem im oberen Abschnitt eine Analyse durchgeführt haben, welche NFAs relevant für die Use Cases sind, werden nun die fachlichen Transaktionen anhand des Mengengerüstes und deren Zugriff auf die beteiligten Geschäftsobjekte geschätzt. Zunächst wird das Mengengerüst, laut Capgemini (Praxisbeispiel), aufgelistet.

Bahn 2.0 Mengengerüste:

- Initial 250 Züge pro Tag
  - Mittelfristiges Wachstum auf bis zu 1000 Züge vorsehen
- Durchschnittliche 400 Fahrgäste im Zug anwesend
- Durchschnittlich 80 Einstiege und 80 Ausstiege je Halt
- Durchschnittlich alle 30 Minuten ein Halt, durchschnittlich 12 Stationen pro Fahrt
- Spitzenzeiten mit Faktor 3 kalkulieren

Für ein besseres Verständnis von einer fachlichen Transaktion folgt nun eine Definition. „Fachliche Transaktionen beschreiben den fachlichen Ablauf mit mehreren zusammengefassten Aktionen.“[8]

[8] vgl. Al-Zoubi: „Kurz und Gut RESTful Web Services – Transaktionen“ S1

Schätzungen der fachlichen Transaktionen des Use Cases „Ein- und Aussteigen“

Als Erstes werden die fachlichen Transaktionen vom Use Case „Ein- / Aussteigen“ geschätzt. Dafür wird der Use Case aufgeteilt. Einmal in Einsteigen und einmal in Aussteigen. Zunächst werden die fachlichen Transaktionen vom Teil Use Case „Einsteigen“ geschätzt.

Um eine Schätzung zu erhalten, muss analysiert werden, wie viele fachliche Transaktionen für einen Fahrgast benötigt werden. Als Erstes wird die Bahn-Karte vom Scanner erkannt. Daraufhin meldet der Scanner dies beim RFID-System. Anschließend ruft das RFID-System die Sitzplatzverwaltung auf, damit der Fahrgast einen Sitzplatz zugewiesen bekommt. Da die Sitzplatzverwaltung den Sitzplatz gemäß dem Profil des Fahrgastes vergibt, ruft die Sitzplatzverwaltung die Profilverwaltung auf. Aufgrund der hinterlegten Profildaten wird der Sitzplatz auf dem Infoterminal angezeigt. Daher werden beim Einsteigen eines Fahrgastes fünf fachliche Transaktionen durchgeführt.

Bevor eine Schätzung erfolgen kann, muss analysiert werden, wie viele fachliche Transaktionen beim Aussteigen erfolgen müssen. Auch hier wird die Bahn-Karte vom Scanner erkannt. Der Scanner meldet dies dem RFID-System. Ebenfalls ruft das RFID-System wieder die Sitzplatzverwaltung auf, damit diese den Ausstieg des Fahrgastes erhält. Zum Schluss lässt die Sitzplatzverwaltung die Daten, wie den Sitzplatz oder welche Klasse der Fahrgast eingenommen hat, in die Profilverwaltung speichern. Dadurch kann der Fahrgast bei der nächsten Fahrt einen erneuten passenden Sitzplatz zugewiesen bekommen. In der Zwischenzeit ist der Fahrgast bereits ausgestiegen. Das bedeutet, dass beim Aussteigen eines Fahrgastes vier fachliche Transaktionen durchgeführt werden müssen.

Als Nächstes wird das Mengengerüst (siehe oben) benötigt. Dort wird beschrieben, dass durchschnittlich 400 Fahrgäste bereits in einem Zug sind. Das heißt, dass 400 Fahrgäste bereits eingestiegen sind. Dadurch erhält man für die bereits eingestiegenen Fahrgäste 2.000 fachliche Transaktionen (fT).

$$fT\ für\ einen\ Fahrgast\ beim\ Einstieg: 1\ Fahrgast = 5\ fT$$

$$fT\ für\ einen\ Fahrgast\ beim\ Ausstieg: 1\ Fahrgast = 4\ fT$$

$$Fahrgäste\ bereits\ im\ Zug: 400 * 5\ fT = 2.000\ fT$$

Rechnung 1: fT für einen Fahrgast beim Ein- und Aussteigen, sowie für die bereits eingestiegenen Fahrgäste

Nun hält der Zug an einem Bahnhof und es steigen durchschnittlich 80 Personen ein und dieselbe Anzahl auch wieder aus. Dadurch werden insgesamt 720 fachliche Transaktionen pro Halt für einen Zug benötigt.

$$Einstieg\ Fahrgäste: 80\ Fahrgäste * 5\ fT = 400\ fT$$

$$Ausstieg\ Fahrgäste: 80\ Fahrgäste * 4\ fT = 320\ fT$$

Rechnung 2: fT für einsteigende und aussteigende Fahrgäste für einen Halt

Im Durschnitt hat der Zug 12 Stationen pro Fahrt. Da der Zug an der ersten Station keine Fahrgäste hat, kann dort auch kein Fahrgast aussteigen. Folglich darf auch kein Fahrgast an der letzten Station einsteigen. Das bedeutet, dass es für eine komplette Fahrt eines Zuges durchschnittlich 7.920 fachliche Transaktionen, bezüglich des Ein- und Aussteigens, durchgeführt werden müssen.

$$Einstieg\ der\ Fahrgäste\ an\ allen\ Stationen: 400\ fT * 11\ Stationen = 4.400\ fT$$

$$Ausstieg\ der\ Fahrgäste\ an\ allen\ Stationen: 320\ fT * 11\ Stationen = 3.520\ fT$$

Rechnung 3: fT für einsteigende und aussteigende Fahrgäste aller Stationen

Um die Schätzung nun vollständig zu haben, werden spätestens beim letzten Halt des Zuges alle Fahrgäste den Zug verlassen müssen. Dadurch werden durchschnittlich 400 Fahrgäste zusätzlich den Zug verlassen. Damit werden weitere 1.600 fachliche Transaktionen benötigt.

$$Ausstieg\ restlicher\ Fahrgäste: 400\ Fahrgäste * 4\ fT = 1.600\ fT$$

$$fT\ des\ gesamten\ Zuges: 2.000\ fT\ + 7.920\ fT + 1.600\ fT\ = 11.520\ fT\ für\ einen\ Zug$$

Rechnung 4: fT für restlich aussteigende Fahrgäste und fT für einen Zug

Zusammenfassend werden also für eine komplette Fahrt eines Zuges 11.520 fachliche Transaktionen durchgeführt.

Zu guter Letzt werden vom Mengengerüst initial 250 Züge pro Tag vorgegeben. Daher wird die Schätzung für die fachlichen Transaktionen eines kompletten Tages aktuell auf 2.880.000 geschätzt.

$$fT\ für\ alle\ Züge\ am\ Tag: 11.520\ fT * 250\ Züge = 2.880.000\ fT\ pro\ Tag$$

$$Wachstum\ der\ Bahn\ auf\ 1000\ Züge: 11.520 * 1000 = 11.520.000\ fT\ pro\ Tag$$

Rechnung 5: fT für initial Züge und Wachstum auf 1000 Züge

Zusätzlich möchte die Bahn 2.0 mittelfristig ein Wachstum auf bis zu 1000 Züge vorsehen. Dadurch werden die fachlichen Transaktionen weiter steigen. Mittelfristig muss das System durchschnittlich 11.520.000 fachlichen Transaktionen pro Tag verarbeiten.

Weiterhin wurden noch keine Spitzenzeiten mit kalkuliert. Zum Beispiel an Feiertagen oder in den Ferien möchten mehr Personen mit der Bahn fahren. Das Mengengerüst listet bei Spitzenzeiten das Dreifache an Fahrgästen auf. Aufgrund der Schienenkapazität kann die Anzahl der Züge pro Tag nicht erhöht werden.

$$Spitzenzeiten\ Fahrgäste\ im\ Zug: 1200 * 5\ fT = 6.000\ fT$$

$$Spitzenzeiten\ Einstieg\ Fahrgäste: 240 * 5\ fT = 1.200\ fT$$

$$Spitzenzeiten\ Ausstieg\ Fahrgäste: 240 * 4\ fT = 960\ fT$$

$$Einstieg\ der\ Fahrgäste\ an\ allen\ Stationen: 1.200\ fT * 11\ Stationen = 13.200\ fT$$

$$Ausstieg\ der\ Fahrgäste\ an\ allen\ Stationen: 960\ fT * 11\ Stationen = 10.560\ fT$$

$$Ausstieg\ restlicher\ Fahrgäste: 1.200 * 4\ fT = 4.800\ fT$$

$$fT\ des\ gesamten\ Zuges: 6.000\ fT + 23.760\ fT + 10.560\ fT = 40.320\ fT$$

$$fT\ für\ alle\ Züge\ am\ Tag: 40.320\ fT * 250 = 10.080.000\ fT\ pro\ Tag$$

$$Wachstum\ der\ Bahn\ auf\ 1000\ Züge: 40.320\ fT * 1000 = 40.320.000\ fT\ pro\ Tag$$

Rechnung 6: fT für Spitzenzeiten

Die bereits vorhandenen Fahrgäste steigen auf 1200. Die Ein- und Ausstiege steigen auf 240 Fahrgäste pro Halt. Damit werden durchschnittlich 40.320 fachliche Transaktionen pro Zug in Spitzenzeiten durchgeführt. Das hat zur Folge, dass über zehn Millionen fachliche Transaktionen pro Tag durchgeführt werden müssen. Mittelfristig möchte die Bahn, wie bereits oben erwähnt, die Züge auf 1000 erhöhen. Daher muss das System mit über 40 Millionen fachlichen Transaktionen zu Recht kommen.

Schätzungen der fachlichen Transaktionen des Use Cases „Film anschauen“

In diesem Abschnitt werden nun die fachlichen Transaktionen des Use Cases „Film anschauen“ geschätzt. Erneut werden erst die fachlichen Transaktionen für einen Fahrgast geschätzt. Wie auch im Use Case „Ein- / Aussteigen“ wird die Karte vom RFID Scanner erkannt. Der Sitzplatzscanner meldet die Karte beim Infotainmentsystem an. Das Infotainmentsystem benötigt von der Profilverwaltung die letzten Infotainment-Nutzungen. Dadurch erhält der Fahrgast an seinem ausgewählten Platz Zugriff auf den Touchscreen-Bildschirm. Gleichzeitig meldet sich das Infotainmentsystem bei der Mediathek, um Zugriff auf die Filme zu erhalten. Diese fachlichen Transaktionen werden auf jeden Fall ausgeführt.

Möchte nun der Fahrgast einen Film anschauen, werden die nächsten fachlichen Transaktionen ebenfalls ausgeführt. Da der Use Case „Film anschauen“ geschätzt werden soll, wird im Folgenden dieses Szenario erwartet.

Falls der Fahrgast den Film bereits ausgeliehen und noch nicht zu Ende geschaut hat, kann er diesen kostenfrei weiteranschauen. Anderenfalls sucht der Fahrgast in der Mediathek nach einem passenden Film und schaut sich diesen an. Daher übermittelt das Infotainmentsystem die erhobenen Gebühren für den Film an das Abrechnungssystem. Das Abrechnungssystem veranlasst die Zahlung

des Betrages an die Bahn. In der Zwischenzeit wird der Film von der Mediathek abgespielt. Das Infotainmentsystem speichert das Filmgenre in der Profilverwaltung, damit der Fahrgast bei der nächsten Fahrt Filme aus diesem Genre vorgeschlagen bekommt. Daher werden pro Fahrgast geschätzte acht fachliche Transaktionen durchgeführt.

Nun wird erneut auf das Mengengerüst verwiesen. Es sind immer noch 400 Fahrgäste aktuell im Zug, die potenziell alle einen Film anschauen. An jedem Halt steigen 80 Personen ein und auch aus. Da die Fahrgäste, die Aussteigen, keinen Film über das Infotainmentsystem anschauen können, werden diese nun vernachlässigt. Zusätzlich muss man davon ausgehen, dass insbesondere am Anfang dieses Projektes kaum bis gar kein Fahrgast einen Film bereits ausgeliehen hat. Der Einfachheit halber wird davon ausgegangen, dass bisher kein Fahrgast einen Film ausgeliehen hat.

$$Fahrgäste\ bereits\ im\ Zug: 400 * 8\ fT = 3.200\ fT$$

$$Einstieg\ Fahrgäste: 80\ Fahrgäste * 8\ fT = 640\ fT$$

$$Einstieg\ der\ Fahrgäste\ an\ allen\ Stationen: 640\ fT * 11\ Stationen = 7.040\ fT$$

$$fT\ des\ gesamten\ Zuges: 7.040\ fT\ + 3.200\ fT = 10.240\ fT\ für\ einen\ Zug$$

$$fT\ für\ alle\ Züge\ am\ Tag: 10.240\ fT * 250\ Züge = 2.560.000\ fT\ pro\ Tag$$

Rechnung 7: fT für alle Züge an einem Tag, wo kein Fahrgast einen Film bereits ausgeliehen hat.

Dadurch ergeben sich die oberen Zahlen. Für die bereits eingestiegenen Fahrgäste (400) werden insgesamt 3.200 fachliche Transaktionen benötigt. Für die Fahrgäste, die Einsteigen, werden weitere 640 fachliche Transaktionen benötigt. Dadurch kann man die fachlichen Transaktionen für alle Stationen auf ca. sieben Tausend schätzen. Der gesamte Zug würde damit ca. zehn Tausend fachliche Transaktionen pro Fahrt je Zug benötigen. Damit kommt man auf 2.560.000 fachlichen Transaktionen pro Tag für alle Züge.

Diese Zahlen werden sich nach und nach verkleinern, da immer mehr Fahrgäste die Mediathek benutzen werden. Sobald die Hälfte der Fahrgäste einen Film bereits ausgeliehen hat, sinkt die Schätzung für alle Züge am Tag auf 1.840.000 fachlichen Transaktionen. Außerdem kommt dazu, dass nur sechs fachliche Transaktionen durchgeführt werden müssen, da das Infotainmentsystem den Betrag nicht an das Abrechnungssystem übergeben muss und die Zahlung muss nicht veranlasst werden.

$$Fahrgäste\ bereits\ im\ Zug\ mit\ Film: 200\ Fahrgäste * 6\ fT = 1.200\ fT$$

$$Fahrgäste\ bereits\ im\ Zug\ ohne\ Film: 200\ Fahrgäste * 8\ fT = 1.600\ fT$$

$$Einstieg\ Fahrgäste\ mit\ Film: 40\ * 6\ fT = 240\ fT\ für\ einen\ Halt$$

$$Einstieg\ Fahrgäste\ ohne\ Film: 40\ * 8\ fT = 320\ fT\ für\ einen\ Halt$$

$$Alle\ Stationen\ mit\ Fahrgästen\ mit\ Film: 240\ fT * 11\ Stationen = 2.640\ fT\ je\ Fahrt$$

$$Alle\ Stationen\ mit\ Fahrgästen\ ohne\ Film: 320\ fT * 11\ Stationen = 3.520\ fT\ je\ Fahrt$$

$$fT\ des\ gesamten\ Zuges: 2.640 + 1.200 + 3.520 = 7.360\ fT\ je\ Fahrt\ pro\ Zug$$

$$fT\ für\ alle\ Züge\ am\ Tag: 7.360 * 250\ Züge = 1.840.000\ fT\ pro\ Tag$$

Rechnung 8: fT für alle Züge an einem Tag, wo die Hälft der Fahrgäste bereits einen Film ausgeliehen haben

Da die Bahn mittelfristig ein Wachstum der Züge auf bis zu 1000 vorsieht, muss dies ebenso berücksichtigt werden. Jedoch muss man nicht davon ausgehen, dass bei dem Wachstum die kompletten acht fachlichen Transaktionen durchlaufen werden müssen. In den 1000 Zügen werden auch Fahrgäste mitfahren, die bereits die Mediathek benutzt und Filme offen haben. Auch hier wird von der Hälfte der Fahrgäste ausgegangen.

$$Hälfte\ Fahrgäste: 7.360 * 1000\ Züge = 7.360.000\ fT\ pro\ Tag$$

Rechnung 9: fT des Wachstums auf 1000 Züge

Damit erhält man eine Schätzung für das Wachstum von ca. 7,3 Millionen fachlichen Transaktionen für alle Züge an einem Tag.

Zum Abschluss des Abschnittes müssen auch hier die Spitzenzeiten der Bahn mit einkalkuliert werden. Auch hier gibt das Mengengerüst den Faktor drei vor. Zunächst wird eine Schätzung durchgeführt, wo jeder Fahrgast ein Film anschaut und keinen Film vorher ausgeliehen hat. Anschließend wird davon ausgegangen, dass die Hälfte der Fahrgäste bereits einen Film ausgeliehen hat und nur die andere Hälfte einen neuen Film ausleiht.

$$Fahrgäste\ bereits\ im\ Zug: 1200\ Fahrgäste * 8\ fT = 9.600\ fT$$

$$Einstieg\ Fahrgäste: 240\ * 8\ fT = 1.920\ fT\ für\ einen\ Halt$$

$$Alle\ Stationen\ mit\ Fahrgästen: 1.920\ fT * 11\ Stationen = 21.120\ fT\ je\ Fahrt$$

$$fT\ des\ gesamten\ Zuges: 9.600 + 21.120 = 30.720\ fT\ je\ Fahrt\ pro\ Zug$$

$$fT\ für\ alle\ Züge\ am\ Tag: 30.720 * 250\ Züge = 7.680.000\ fT\ pro\ Tag$$

Rechnung 10: fT für Spitzenzeiten

Durch die Schätzung von oben erhält man einen Wert von 7.680.000 fachlichen Transaktionen pro Tag für alle Züge, wenn jeder Fahrgast einen neuen Film ausleiht. Da jedoch das System nicht in den

Spitzenzeiten eingeführt werden sollte, muss man davon ausgehen, dass einige Fahrgäste bereits eine Erfahrung mit der Mediathek und dem Infotainmentsystem gemacht haben.

Dafür wird die untenstehende Schätzung verwendet:

$$Fahrgäste\ bereits\ im\ Zug\ mit\ Film: 600\ Fahrgäste * 6\ fT = 3.600\ fT$$

$$Fahrgäste\ bereits\ im\ Zug\ ohne\ Film: 600\ Fahrgäste * 8\ fT = 4800\ fT$$

$$Einstieg\ Fahrgäste\ mit\ Film: 120\ * 6\ fT = 720\ fT\ für\ einen\ Halt$$

$$Einstieg\ Fahrgäste\ ohne\ Film: 120\ * 8\ fT = 960\ fT\ für\ einen\ Halt$$

$$Alle\ Stationen\ mit\ Fahrgästen\ mit\ Film: 720\ fT * 11\ Stationen = 7.920\ fT\ je\ Fahrt$$

$$Alle\ Stationen\ mit\ Fahrgästen\ ohne\ Film: 960\ fT * 11\ Stationen = 10.560\ fT\ je\ Fahrt$$

$$fT\ des\ gesamten\ Zuges: 8.400 + 18.480 = 26.880\ fT\ je\ Fahrt\ pro\ Zug$$

$$fT\ für\ alle\ Züge\ am\ Tag: 26.880 * 250\ Züge = 6.720.000\ fT\ pro\ Tag$$

Rechnung 11: fT für Spitzenzeiten, wo die Hälfte der Fahrgäste bereits einen Film ausgeliehen haben

Während der Spitzenzeiten kann man davon ausgehen, dass mindestens die Hälfte der Fahrgäste mit der Mediathek vertraut ist und einen Film ausgeliehen haben. Damit kann man die fachlichen Transaktionen auf 26.880 pro Zug je Fahrt schätzen. Rechnet man nun noch die Anzahl der initial Züge dazu, kommt man auf einen Wert von 6.720.000 fachlichen Transaktionen pro Tag für 250 Züge.

Auch bei den Spitzenzeiten muss das System, dem Wachstum der Bahn auf 1000 Züge, standhalten können. Wie bereits weiter oben beschrieben, kann man nicht davon ausgehen, dass es keinen Fahrgast gibt, der bereits einen Film ausgeliehen hat. Um jedoch eine Schätzung zu erhalten, wird davon ausgegangen, dass die Hälfte der Fahrgäste bereits einen Film ausgeliehen und offen zum Sehen haben. Dadurch erhält man den Wert von 26.880.000 fachlichen Transaktionen pro Tag für 1000 Züge.

$$Hälfte\ Fahrgäste: 26.880 * 1000\ Züge = 26.880.000\ fT\ pro\ Tag$$

Rechnung 12: fT des Wachstums auf 1000 Züge während Spitzenzeiten. Hälfte der Fahrgäste haben einen Film ausgeliehen

Abschließend für den Abschnitt kann festgehalten werden, dass man noch weitere Faktoren mitberücksichtigen könnte. Zum Beispiel kann ein Fahrgast beim ersten Halt einsteigen und beim letzten Halt aussteigen. Dadurch kann der jeweilige Fahrgast auch mehrere Filme anschauen, die er jeweils bezahlen muss. Oder einem Fahrgast gefällt der aktuelle Film nicht und entschließt sich einen weiteren Film auszuleihen. Weiterhin kann es sein, dass der Fahrgast gar keinen Film anschauen

möchte. Damit würden sich die fachlichen Transaktionen erhöhen bzw. verringern. Jedoch wurde hier vom realistischsten Ereignis ausgegangen, und zwar das jeder Fahrgast nur einen Film anschaut.

Aufgabe 2 b) Analyse des Use-Case „Ticket Buchung“

Einleitung

Dieser Abschnitt umfasst das Feststellen der NFAs für den Use-Case „Ticket Buchung“, Kommunikationsarten für Systeme, welche Kommunikationsart für das geforderte System am sinnvollsten ist und wie sich die ausgewählte Kommunikationsart auf die NFAs auswirkt. Um die NFAs für diesen Use-Case bestimmen zu können, wird zunächst betrachtet, welche Komponenten benötigt werden und wie überhaupt eine „Ticket Buchung“ abläuft.

Use-Case Beschreibung

Durch die Anforderungen an das System, ist das klassische kaufen eines Tickets nicht mehr notwendig, da das System eine bzw. mehrere Fahrten in einem bestimmten Zeitraum zusammenrechnet und somit eine optimale Abrechnung für den Fahrgast bereitstellt[9]. Es steht dem Fahrgast jedoch frei, eine Reservierung über das Internet oder einem Bahnschalter zu tätigen. Mit diesen beiden Fällen ist der eigentliche Use-Case gemeint, da hier eine vorab Buchung stattfindet.

Die beiden Fälle unterscheiden sich jedoch nicht wirklich. Nur der Einstiegspunkt für den Prozess ist anders. Bei einem der beiden Fälle nutzt der Fahrgast die Möglichkeit über die Internetseite der Bahn, eine Fahrt zu reservieren, was eine Buchung der Strecke automatisch beinhaltet. Dies findet also auf einem externen Gerät statt, welches nicht von der Bahn betrieben wird. Da die Internetseite jedoch auf Hardware der Bahn gehostet ist, wird der Prozess dennoch Bahn intern angestoßen. Ähnlich findet der Prozess statt, wenn der Fahrgast eine Reservierung über einen Bahnschalter tätigt. Dieser wird auch von der Bahn selbst verwaltet und ist somit intern. Daraus folgt, dass der Prozess immer auf interner Seite angestoßen wird.

Nachdem der Fahrgast also eine Reservierung/Buchung vorgenommen hat, wird das Ticket-Buchungssystem angesteuert. Dort wird dann das Abrechnungssystem angesteuert, damit auch sichergestellt wird, dass das gebuchte Ticket bezahlt wird, da eine weitere Bearbeitung des Prozesses sonst keinen Sinn machen würde. Nachdem sichergestellt worden ist, dass das Ticket bezahlt ist, wird das Sitzplatz-Reservierungssystem darüber informiert, dass es in der Sitzplatzverwaltung eine neue Reservierung hinterlegen muss und somit ein Platz weniger belegt werden kann, durch einen Fahrgast, welcher keine Reservierung/Buchung abgeschlossen hat. Zum gleichen Zeitpunkt kann

[9] vgl. Huschke: „Aufgabe Capgemini (zur Veranstaltung vom 18.06.2019)“ S.1

jedoch dem Fahrgast bereits ein Feedback gegeben werden, dass die Abbuchung erfolgreich war und sein Ticket im System hinterlegt ist. Natürlich muss das System in der Lage sein, bei einer Ticketbuchung, bei der mehr als ein Ticket gekauft worden ist, die Sitzplätze beim Einsteigen dann so zu verteilen, das alle Personen zumindest in unmittelbarer Nähe sitzen. Es sollte nicht zu dem Fall kommen, das beispielweise eine Mutter in einem Zugabschnitt sitzt und ihre Kinder in einem anderen.

NFAs bezüglich Use-Case

Dieser Abschnitt beschäftigt sich mit den NFAs die speziell für diesen Use-Case sind.

- Funktionalität:
    - Angemessenheit: Das System muss die Funktionen bereitstellen, damit dieser Prozess/Use-Case durchgeführt werden kann.
    - Richtigkeit: Es muss gewährleistest werden, dass die Daten, die genutzt werden, bei allen Komponenten identisch sind. Das Abbuchen bei einer anderen Person muss zu 100% ausgeschlossen sein.
    - Interoperabilität: Für diesen Prozess muss die Interoperabilität nicht ausgeprägt sein, da das System sich stark nach diesem Prozess richten muss und nicht anders herum.
    - Konformität: Es müssen für den Ablauf des Prozesses ausschließlich gesetzliche Bedingungen eingehalten werden. Konventionen gibt es an dieser Stelle keine.
    - Ordnungsmäßigkeit: Der Prozess muss sich an die gesetzlichen Bedingungen halten. Weitere anwendungsspezifischen Regelungen gibt es nicht.
    - Sicherheit: Es muss sichergestellt werden, das während der ganzen Transaktion keine Daten abgefangen werden können. Dafür sollte eine geeignete Verschlüsselung genutzt werden, weil besonders bei dieser Transaktion Bankdaten des jeweiligen Fahrgastes involviert sind. Die Daten müssen zu 100% geschützt sein.
- Zuverlässigkeit:
    - Reife: Es muss möglich sein, auch bei teilweise auftretenden Fehlern ein abschließen des Prozesses zu garantieren. Es muss ein hoher Grad an Reife vorhanden sein.
    - Konformität: Die Konformität bei diesem Prozess muss hoch sein, denn dieser ist ein häufiger Prozess, sobald das System in der Produktion eingesetzt wird.
    - Fehlertoleranz: Der Prozess hat eine geringe Fehlertoleranz, daher müssen die für den Prozess wichtigen Komponenten zu jedem Zeitpunkt funktionieren.
    - Wiederherstellbarkeit: Bei einer Wiederherstellung des Systems nach einem Fehler, muss das System in der Lage sein, die betroffenen Daten wieder aufzunehmen um den Prozess abzuschließen

- Benutzbarkeit:
    - Verständlichkeit: Muss gering sein. Bezieht sich auf das TBs und nicht das Bahn 2.0 System.
    - Erlernbarkeit: Die Erlernbarkeit muss leicht sein. Diese hängt von anderen Faktoren ab. Dieser Faktor ist jedoch nicht messbar für unser System, sondern das Fremdsystem TBs.
    - Konformität: Der Grad der Konformität bezüglich der Benutzbarkeit muss nicht groß sein, da im System nur der Prozess abgewickelt wird und die GUI auf dem TBs.
    - Attraktivität: Nicht relevant für das Bahn 2.0 System. Wichtiger für NFAs des TBs.
    - Bedienbarkeit: Siehe „Attraktivität“

- Effizienz:
    - Zeitverhalten: Das System muss in der Lage sein, dem Fremdsystem eine Rückmeldung in unter fünf Sekunden zu geben.
    - Konformität: Die Konformität im Bereich Effizienz muss sich an die Vereinbarungen mit dem Kunden orientieren. Beispielsweise unter fünf Sekunden Antwortzeit.
    - Verbrauchsverhalten: Die Hardware muss in der Lage sein, auch bei höherer Auslastung einen schnellen Lese-/Schreibzugriff zu gewährleisten. Der Prozess muss auch ressourcensparend umgesetzt sein, da hier keine größeren Änderungen von Nöten ist.

- Änderbarkeit:
    - Analysierbarkeit: Es muss eine Dokumentation über die gesamte Software angelegt und verwaltet werden, damit eine gute Analysierbarkeit gewährleistet werden kann. Es handelt sich um eine größere Software, die über Generationen gewartet wird. Wird ein Fehler in der Dokumentation gemacht, hat dies starke Auswirkungen auf die Analysierbarkeit. Die Analysierbarkeit des Systems muss groß sein, was bedeutet, dass ein geringer Aufwand, für das durchsuchen des Quellcodes, von Nöten ist.
    - Konformität: Die Software muss einen mittleren Grad an Konformität im Bereich der Änderbarkeit aufweisen. Eine Änderung des Systems sollte nicht oft vorkommen.
    - Modifizierbarkeit: Eine gute Modifizierbarkeit im Bereich der Optimierung muss nicht gewährleistet werden, jedoch im Bereich der Fehlerbeseitigung. Grundsätzlich sollte für so etwas ein ausführlicher Testplan geschrieben werden.
    - Stabilität: Das System muss über einen hohen Grad an Stabilität verfügen, da es sich hier um eine Software handelt, ohne die der Konzern nicht arbeiten kann.

- Testbarkeit: Der Aufwand für das Testen einer Änderung muss gering sein.

- Übertragbarkeit:
  - Anpassbarkeit: Die Software muss in der Lage sein, auch bei einer Umsiedelung oder einer Datenbankmigration, schnellst möglich wieder einsatzbereit zu sein. Ein maximaler Ausfall von acht Stunden muss gewährleistet sein (Umzug über Nacht).
  - Installierbarkeit: Der Aufwand zur Installation der Software muss einfach und unkompliziert sein.
  - Austauschbarkeit: Das System muss austauschbar sein.
  - Konformität: Das System muss eine hohe Konformität im Bereich der Übertragbarkeit bereitstellen.
  - Koexistenz: Das System muss in der Lage sein, auch in einem Dualbetrieb einwandfrei zu Laufen.

Kommunikationsarten

Nachdem im oberen Abschnitt eine Analyse durchgeführt wurde, welche NFAs wichtig für den Use-Case sind, wird nun betrachtet, wie sich das System unter den jeweiligen Kommunikationsarten verhalten würde. Zum besseren Verständnis, folgt zunächst aber eine Erläuterung zu den drei Kommunikationsarten synchron, asynchron und Caching.

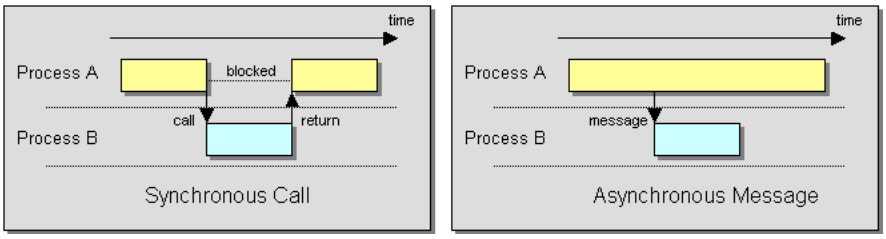

Abbildung 8 Unterschied zwischen Synchroner und Asynchroner Kommunikation

Anhand von Abbildung 8 kann man gut den Unterschied von synchroner und asynchroner Kommunikation erklären. Synchrone und asynchrone Kommunikation funktionieren so, dass ein Prozess (Prozess A) beispielweise einen anderen Prozess (Prozess B) anstößt. Bis hier hin unterscheiden sich synchron und asynchron noch nicht. Der entscheidende Unterschied folgt nun, denn synchrone Kommunikation unterbricht Prozess A und wartet bis Prozess B abgeschlossen ist und gegebenenfalls einen Rückgabewert liefert. Das bedeutet also, sollte Prozess B einen längeren Zeitraum benötigen, so wartet Prozess A auch so lange. Daraus ergibt sich dann folgende Laufzeit:

$$Prozess\ A_{gesamt} = Prozess\ A_{selbst} + Prozess\ B_{gesamt}$$

Bei der asynchronen Kommunikation hingegen, wird Prozess A nicht unterbrochen und läuft weiter durch. Also:

$$Prozess\ A_{gesamt} = Prozess\ A_{selbst}$$

Wann sollte man nun eine synchrone und wann eine asynchrone Kommunikation verwenden? Grundsätzlich dient als grober Leitfaden, ob sich beide Prozesse im selben System befinden. Ist dies nicht der Fall, sollte man asynchrone Kommunikation wählen, weil man nicht abschätzen kann, wie viel Zeit der Prozess des Fremdsystems benötigt und somit das eigene System unterbrochen wird. Ist im Gegenzug der andere Prozess im selben System (schließt Subsysteme mit ein), kann man im Normalfall abschätzen wie lange der Prozess dauert. Meist wird hier dann auf eine synchrone Kommunikation gesetzt, wenn man weiß, dass der Prozess keine längere Zeit benötigt und der aufrufende Prozess diese Zeit warten kann.

Als weitere Kommunikationsart gibt es noch Caching, welches so funktioniert, dass die Daten beim ersten Aufruf initial in einen temporären Speicher (Cache) geladen werden. Das hat zur Folge, dass die Daten nicht immer neu geladen werden müssen, sondern aus dem Cache geladen werden können. Nach einer gewissen Zeit verfallen dann diese Daten, da man sonst nicht garantieren kann, dass diese immer auf dem aktuellen Stand sind. Natürlich kann man diese Art der Kommunikation nur nutzen, wenn auf Daten zugegriffen werden, die sich selten ändern, da man sonst die Daten über keinen längeren Zeitraum (>= 30 Min.) zwischen speichern könnte, ohne dass das andere System die Änderungen mit bekommt. Nutzbare Daten wären beispielsweise die Profildaten, da sich diese zwar ändern können, dieser Fall jedoch eher selten vorkommt bzw. der Zeitraum zwischen zwei Änderungen relativ groß ist (>=1 Tag).

Kommunikationsart für das System

Als nächstes wird untersucht wie die Umsetzung in dem Bahn 2.0 System aussehen würde. Vorab aber nochmal eine Kurzfassung des Use-Cases.

Der Kunde interagiert mit einer Webseite/Bahnschalter, welche/r auf Seiten der Bahn gehostet wird. Diese Systeme steuern das Ticket-Buchungssystem (TBs) an. Von dort wird eine Nachricht an das Abrechnungssystem (As) versendet. Nach Rückmeldung des Systems, wird dem Sitzplatz-Reservierungssystem (S-Rs) gesagt, dass ein Platz geblockt werden soll. Dieses System lässt dies dann in der Sitzplatzverwaltung (Sv) des jeweiligen Zuges eintragen.

Zunächst wird die Kommunikation zwischen dem Einstiegspunkt und dem TBs betrachtet. Entscheidet man sich an dieser Stelle für eine Caching Kommunikation, könnte man nicht garantieren, dass die Daten auf beiden Seiten Konsistent sind, da das TBs in einem bestimmten Zeitpunkt nur die Daten mit der Webseite/Bahnschalter synchronisiert. Daher macht Caching an dieser Stelle keinen Sinn. Als nächstes gäbe es die asynchrone Kommunikation. Dies würde bedeuten, dass das System weiterlaufen würde, auch wenn keine Rückmeldung des TBs käme. Prinzipiell wäre

dies möglich, aber an dieser Stelle sehr gefährlich und fahrlässig, da zu diesem Zeitpunkt nicht garantiert werden kann, ob die Abbuchung erfolgreich war. Daher muss an dieser Stelle eine synchrone Kommunikation genutzt werden. Dadurch, dass es sich bei beiden Systemen um ein internes System handelt, kann man an dieser Stelle auch von einer schnellen Antwortzeit ausgehen bzw. von einer geeigneten Fehlerbehandlung bei einer Zeitüberschreitung (>5 Sek.)

Danach muss zwischen dem TBs und dem As kommuniziert werden. Auch an dieser Stelle macht Caching wenig Sinn, da ein dauerhafter Abgleich von Daten notwendig ist und das TBs ein schnelles (< 5 Sek.) Feedback braucht, welches es an den Kunden weitergeben kann. Wie bei der Kommunikation zwischen dem Einstiegspunkt und dem TBs, ist es an dieser Stelle ratsam, eine synchrone Kommunikation zu verwenden, da beide Systeme intern sind und mithilfe der Rückmeldung erst entschieden werden kann, ob eine weitere Bearbeitung des Prozesses überhaupt möglich ist.

Nachdem das TBs nun das Feedback von dem As bekommen hat, gibt es dies an die Internetseite bzw. den Bahnschalter weiter, damit der Kunde weiß, ob seine Fahrt abgebucht worden ist oder nicht. Der Prozess ist an dieser Stelle jedoch noch nicht zu Ende und das TBs interagiert mit dem S-Rs. Hier kann sowohl eine asynchrone als eine synchrone Kommunikation verwendet werden, weil es sich um interne Systeme handelt, setzt man im Normallfall auch auf die synchrone Kommunikation. Sollte das S-Rs nämlich einen Fehler zurückgeben, muss das TBs über eine geeignete Fehlerbehandlung Verfügen wie beispielweise einen Neuversuch oder das Versenden einer E-Mail, welche an den Fahrgast raus geht, dass dessen Ticket im System hinterlegt ist, jedoch kein Platz geblockt wurde und er sich bei einem Zugmitarbeiter melden soll. Diese Fehlerbehandlung kann jedoch auch bei einer asynchronen Kommunikation eingebaut werden. Caching macht auch bei dieser Kommunikation zwischen den Backends wenig Sinn, da eine Inkonsistenz fatale Folgen haben könnte.

Zu guter Letzt wäre noch die Kommunikation zwischen dem S-Rs und der Sv zu analysieren. Zur Erinnerung, im Abschnitt der TI-Architektur wurde entschieden, dass in jedem Zug eine eigene Sv vorhanden ist. Dieser Punkt ist ausschlaggebend für die Eignung der Kommunikationswahl zwischen den Systemen. Würde man sich für eine synchrone Kommunikation entscheiden, würde sich das S-Rs in der Zeit wo es auf die Sv wartet, in einem Wartemodus befinden. Das bedeutet, dass in dieser Zeit keine anderen Prozesse, wie beispielweise weitere Buchungen, angestoßen werden könnten. Aus den Eigenschaften der Bandbreite im Zug, kann man entnehmen, dass diese außerhalb eines Bahnhofs, nicht als ausreichend garantiert werden kann. Die in diesem Fall beste Variante wäre die Kommunikationsart Caching. Diese Kommunikation würde stattfinden, sobald der Zug in einen Bahnhof einfährt. Dann würde die Sv, des jeweiligen Zugs, seine Daten mit der S-Rs abgleichen, damit

die Daten wieder konsistent sind. Ausschließlich zu diesem Zeitpunkt ist es nämlich notwendig, dass der Zug weiß, wie viele Sitzplätze (nicht jedoch welche) er für Reservierungen blocken muss. Während einer Fahrt muss die Sv des jeweiligen Zugs dies noch nicht zu wissen, da während der Fahrt keine Fahrgäste ein- oder aussteigen. Alternativ könnte man sich auch für eine asynchron Kommunikation entscheiden, welche jedoch nicht so effektiv wie Caching wäre, da die Sv bereits während der Fahrt mit Daten versorgt wird, welche sie aber noch nicht braucht, woraus eine höhere Auslastung des Speichers im Zug entsteht.

Abschließend kommt man also zu dem Ergebnis, dass man bei den internen Systemen auf eine synchrone Kommunikation setzt, weil man sowohl Feedback von den Systemen braucht und eine kurze Antwortzeit (< 5 Sek.) garantieren kann und alle Systeme sich intern befinden. Ausschließlich die Kommunikation zwischen dem S-Rs und dem Sv des jeweiligen Zugs wird auf Caching gesetzt, da hier eine dauerhafte Synchronisation nicht von Nöten ist.

In Abbildung 9 ist der gesamte Use-Case noch einmal in einem Sequenzdiagramm. Pfeile mit einer schwarz ausgefüllten Spitze bedeuten eine synchrone Kommunikation. Gestrichelte Linien deuten einen Rückgabewert für einer synchrone Nachricht an. Dünne Pfeile bedeuten eine asynchrone Kommunikation. Die Kommunikation zwischen dem S-Rs und dem Sv wurde mit asynchronen Pfeilen gezeichnet, soll jedoch als Caching Kommunikation betrachtet werden.

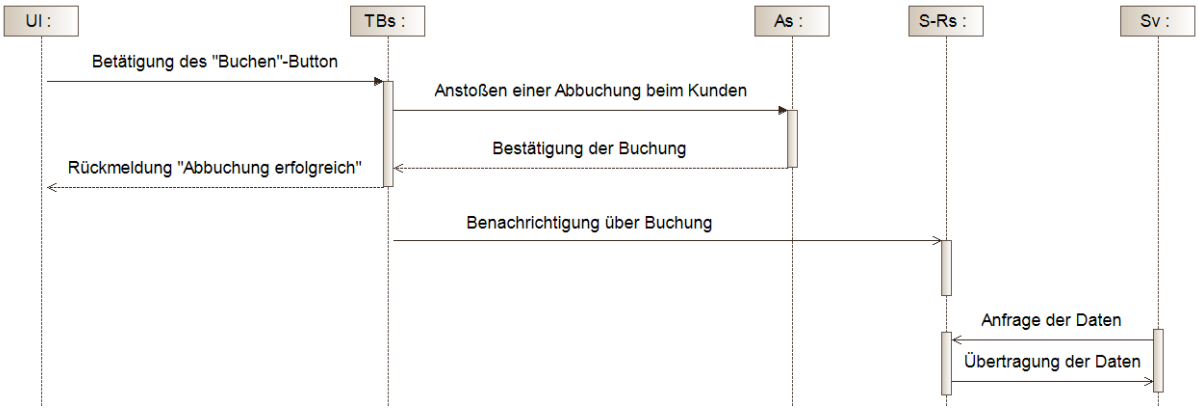

Abbildung 9 Sequenzdiagramm für den Use-Case „Ticket buchen“

NFAs bezüglich der Lösungsalternativen

Zum Abschluss muss noch eine Analyse der NFAs bezüglich der unterschiedlichen Kommunikationsarten durchgeführt werden. Zum besseren Verständnis werden bei der Analyse für die passende Kommunikationsart die einzelnen Abschnitte betrachtet.

Den Anfang macht der Aufruf vom Einstiegspunkt zum TBs. Würde man sich für eine Caching Kommunikation entscheiden, würde man besonders das Zeitverhalten (Effizienz) negativ beeinflussen, da bei Caching die Daten nur zu einem bestimmten Intervall abgeglichen werden. Bei

den beiden Kommunikationsarten synchron und asynchron würde dieser Punkt nicht verletzt werden, hingegen muss bei diesen beiden Kommunikationsarten auf die Fehlertoleranz (Zuverlässigkeit) geachtet werden. Dies gilt selbstverständlich auf für Caching. Trotzdem ist an dieser Stelle nur eine synchrone oder asynchrone Kommunikation möglich und somit Lösungsalternativen.

Als nächstes folgt der Aufruf des As aus dem TBs. Genau wie beim Aufruf des TBs aus dem GUI-Einstiegspunkt, würde man bei einer Caching Kommunikation gegen das Zeitverhalten verstoßen. Ebenfalls müsste die Fehlertoleranz bei allen Kommunikationsarten beachtet werden. Auch hier gelten als Lösungsalternativen nur synchrone oder asynchrone Kommunikation.

Im Gegensatz zu den vorherigen Aufrufen, ist beim Aufruf des S-Rs das Zeitverhalten nicht so entscheidend, denn der Nutzer wartet auf keine Rückmeldung mehr. Wichtiger an dieser Kommunikation ist das Einhalten der Fehlertoleranz und der Reife (Zuverlässigkeit). Dies resultiert daraus, dass die Buchung bis zur Sitzplatzverwaltung des Zuges sichergestellt wird. Folgendes Szenario muss jedoch bei der Entscheidung der Lösungsalternativen bedacht werden. Bucht ein Kunde ein Ticket eine kurze Zeit (< 5 Min.) vor dem Einsteigen in den jeweiligen Zug, muss der Kunde davon ausgehen können, dass seine Buchung im System hinterlegt ist. Daher muss sichergestellt sein, dass die Daten vom TBs zum S-Rs, in einer angemessenen Zeit, erfolgreich übertragen worden sind. Daher stehen für diesen Abschnitt nur die Lösungsalternativen synchron und asynchron zur Verfügung.

Zum Abschluss muss noch die Kommunikation zwischen dem S-Rs und dem Sv des jeweiligen Zuges betrachtet werden. Bei dieser Kommunikation liegt das Hauptaugenmerk auf der NFA bezüglich des Verbrauchsverhaltens (Effizienz). Es sollte eine schnelle und Ressourcen sparende Kommunikation gewählt werden, weil es sich bei dieser Kommunikation um ein kurzes Zeitfenster handelt. Durch das abgleichen der Daten beim Einfahren in den Bahnhof, werden bereits viele neue Daten abgerufen, welche bereits verarbeitet werden können. Auch hier muss natürlich die geforderte Fehlertoleranz und Reife berücksichtigt werden. Als beste Lösung eignet sich daher, wie in der obigen Analyse, eine Caching Kommunikation. Eine synchrone und asynchrone Kommunikation wären an dieser Stelle auch alternativen, jedoch wären diese zu übertrieben.

Abschließend muss natürlich auch besonders auf die NFAs bezüglich der Sicherheit und der Richtigkeit (beides Funktionalität) geachtet werden. Diese können aber bei allen drei Kommunikationsarten gewährleitstet werden.

Schlusswort

Die Bahn 2.0 würde für die Reisenden die Fahrt erheblich komfortabler machen. Dafür sollen personalisierte Sitze, Filme und ein teilweise kostenloser Internetzugang sorgen. Jedoch ist der Umbruch für die Bahn gewaltig. Jeder Zug benötigt neue Hardware. Dazu muss das System immer stabil laufen, da ansonsten Geldeinnahmen verloren gehen können. Dennoch wäre es ein wichtiger Schritt für die Bahn das Projekt zu meistern. Wie machbar das Projekt für die Bahn ist, kann man aus dieser Ausarbeitung nicht schlüssig erläutern, da hier nur ein kleiner Teil analysiert worden ist.

Das präsentierte Unternehmen Capgemini zeigte bei ihrer Präsentation einen kleinen Teil ihrer Aufgaben. Diese Aufgaben spiegelten den Aufgabenbereich eines Software Engineer wider. Zum Beispiel erstellt der Software Engineer den Aufbau von Softwaresystemen und trifft grundlegende Entscheidungen über das Zusammenspiel der diversen Komponenten. Weitere Berufsfelder im Unternehmen Capgemini sind Business Analyst, Application Consultant oder Project Manager. Damit sieht man, dass Capgemini ein großes Spektrum bedient.

Abschließend lässt sich sagen, dass der Vortrag sehr informativ war. Durch den Vortrag erhielt man einen kleinen Einblick in den Berufsalltag der Präsentierenden. Die Präsentation war insbesondere, wie oben beschrieben, für angehende bzw. interessierte Software Engineer Pflicht.

Anhang

Quellen- / Literaturverzeichnis

Internet:

Al-Zoubi, Ahmad: Kurz und Gut RESTful Web Services – Transaktionen
URL: http://www.se.uni-hannover.de/pub/File/kurz-und-gut/ws2011-labor-restlab/RESTLab-Transaktionen-Ahmad-Al-Zoubi-kurz-und-gut.pdf S1 (Stand: 20.12.11) (letzte Nutzung: 21.08.19)

Capgemini Service SAS, Hermelin, M. Paul: Alle Partner
URL: https://www.capgemini.com/de-de/partners/ (letzte Nutzung: 20.08.19)

Heini, R.: Funktionale, nicht funktionale Anforderungen
URL:http://www.anforderungsmanagement.ch/in_depth_vertiefung/funktionale_nicht_funktionale_anforderungen/index.html (letzte Nutzung: 19.08.19)

Huschke, Sabine: Aufgabe Capgemini (zur Veranstaltung vom 18.06.2019)
URL: https://ilias.th-koeln.de/goto.php?target=file_1328704_download&client_id=ILIAS_FH_Koeln (Stand: 19.07.19) (letzte Nutzung: 19.08.19)

Kaubisch, Christian / Klein, Norbert: Technische Architektur in der individuellen Softwareentwicklung
URL: https://ilias.th-koeln.de/goto.php?target=file_1305749_download&client_id=ILIAS_FH_Koeln S5, S54 (Stand: 18.06.19) (letzte Nutzung: 20.08.19)

Prof. Dr. Bente, Stefan / Prof. Dr. Winter, Mario: Serviceschicht Kommunikationsprotokolle
URL: https://ilias.th-koeln.de/goto.php?target=file_1288771_download&client_id=ILIAS_FH_Koeln S18 (Stand: 05.05.19) (letzte Nutzung: 15.08.19)

Taentzer, Gabriele: Einführung in die Softwaretechnik
URL: https://www.uni-marburg.de/fb12/arbeitsgruppen/swt/lehre/files/est1415/EST141028.pdf S88 (Stand: 28.10.14) (letzte Nutzung: 19.08.19)

Wikimedia Foundation Inc.: Anforderung (Informatik)
URL: https://de.wikipedia.org/wiki/Anforderung_(Informatik) (Stand: 05.01.19) (letzte Nutzung: 19.08.19)

Wikimedia Foundation Inc.: ISO / IEC 9126
URL: https://de.wikipedia.org/wiki/ISO/IEC_9126 (Stand: 28.05.18) (letzte Nutzung: 19.08.19)

Wikimedia Foundation Inc.: Modelio
URL: https://de.wikipedia.org/wiki/Modelio (Stand: 09.05.19) (letzte Nutzung: 05.08.19)

Abbildungsverzeichnis

Tätigkeitsmatrix

Abschnitt	**Frederik Duda**	**John-Bryan Spieker**
Vorwort		**X**
Capgemini – das Unternehmen	**X**	
Aufgabenstellung	**X**	
Aufgabe 1 – Architekturskizzen	**X (Bilder)**	**X (Bild & Text)**
A-Architektur	**X (Bilder)**	**X (Bild & Text)**
T-Architektur	**X (Bilder)**	**X (Bild & Text)**
TI-Architektur	**X (Bilder)**	**X (Bild & Text)**
Aufgabe 2 - Nicht funktionale Anforderungen	**X**	
Nicht funktionale Anforderungen	**X**	
Aufgabe 2 a) Analyse der Use Cases „Ein- / Aussteigen“ und „Film anschauen“	**X**	
Einleitung	**X**	
Use Case „Ein- und Aussteigen“ Beschreibung	**X**	
Nicht funktionale Anforderungen bezüglich des Use Case „Ein- und Aussteigen“	**X**	
Use Case „Film anschauen“ Beschreibung	**X**	
Nicht funktionale Anforderungen bezüglich des Use Case „Film anschauen“	**X**	
Abschätzung der fachlichen Transaktionen bezüglich des Mengengerüstes	**X**	
Schätzungen der fachlichen Transaktionen	**X**	

Schätzungen der fachlichen Transaktionen des Use Cases „Ein- und Aussteigen“	**X**	
Schätzungen der fachlichen Transaktionen des Use Cases „Film anschauen“	**X**	
Aufgabe 2 b) Analyse des Use-Case „Ticket Buchung“		**X**
Einleitung		**X**
Use-Case Beschreibung		**X**
NFAs bezüglich Use-Case		**X**
Kommunikationsarten		**X**
Kommunikationsart für das System		**X**
NFAs bezüglich der Lösungsalternativen		**X**
Schlusswort	**X**	