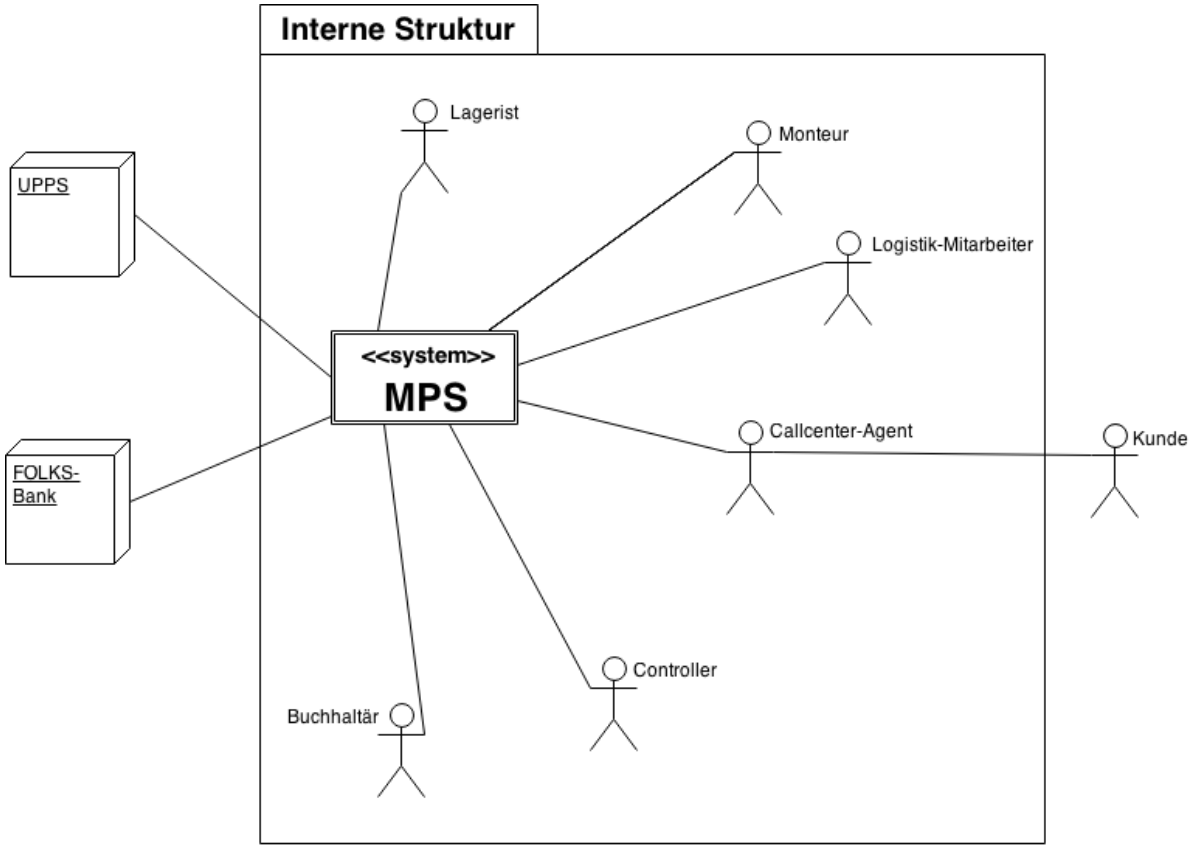
Aufgabenblatt 2

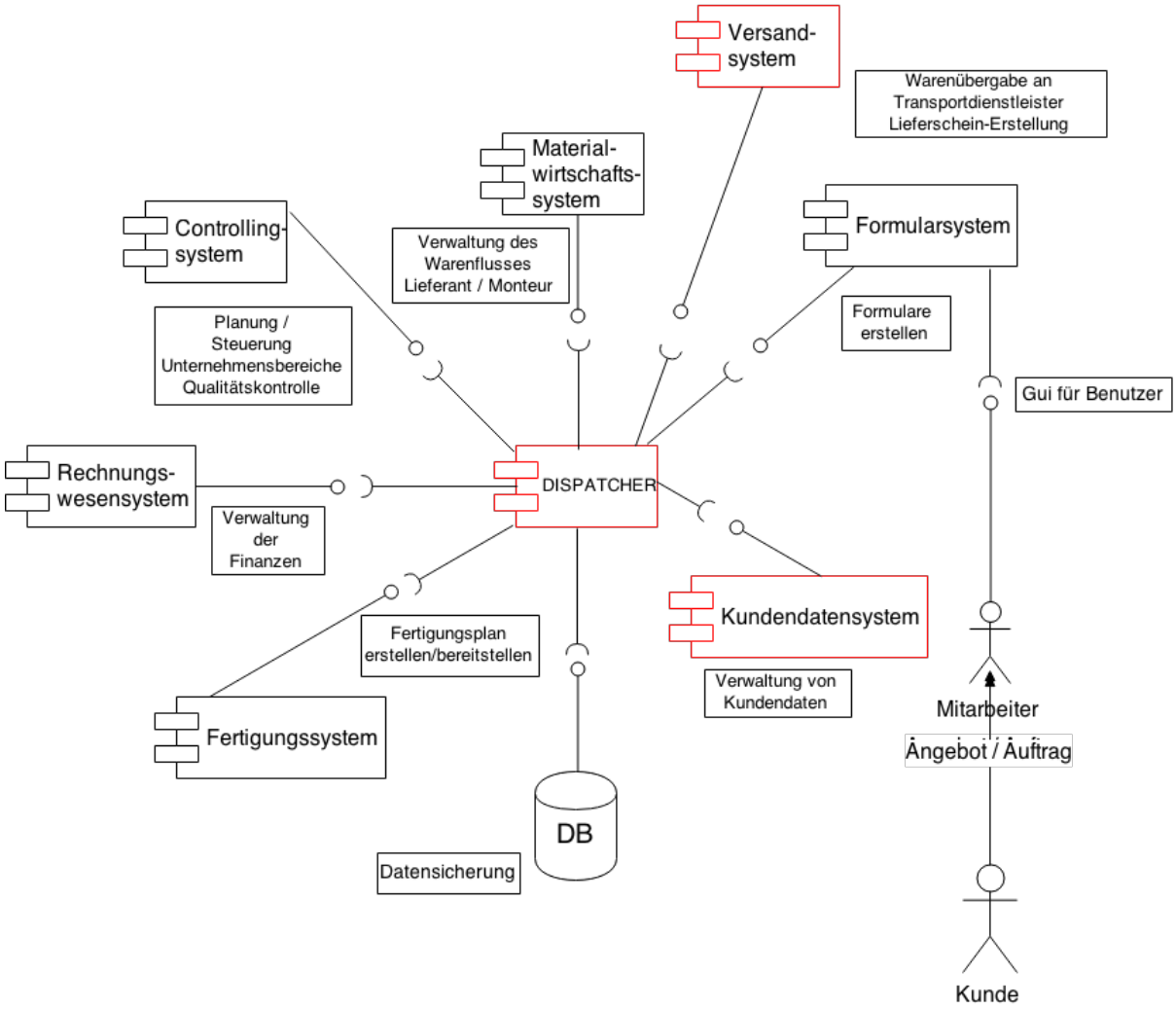
Aufgabe 4



Entscheidungen/Annahmen: Jeder Mitarbeiter ist einer Komponente/Modul zugeordnet.
Die Komponenten spiegeln auch die einzelnen Abteilungen/Bereiche des Unternehmens wieder.

- *Zuordnung der Mitarbeiter zu den Komponenten/Modulen*
  - ➢ Callcenter-Agent → FormularSystem
  - ➢ Monteur → FertigungsSystem
  - ➢ Lagerist → VersandSystem
  - ➢ Buchhaltär → RechnungswesenSystem
  - ➢ Controller → ControllingSystem
  - ➢ Logistik-Mitarbeiter → MaterialwirtschaftsSystem

MPS

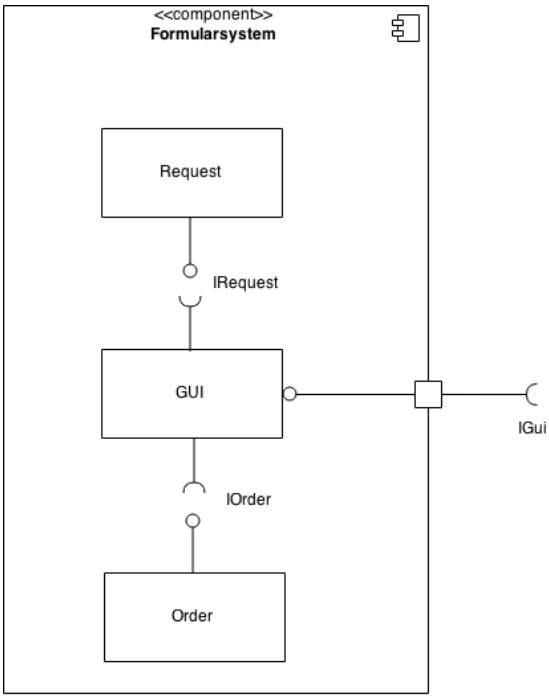


Entscheidungen/Annahmen: Im Mittelpunkt gibt es nun bei uns einen Dispatcher/Verteiler, der alle Komponenten miteinander verbindet. Ausserdem wurde das Versandsystem von der Material-Wirtschafts-System-Komponente getrennt und ist nun direkt am Dispatcher angebunden. Zusätzlich wurde noch die Kundendatensystem-Komponente hinzugefügt, da vorher noch keiner Verwaltung der Kundendaten zur Verfügung stand. Die Oberfläche-Komponente wurde entfernt, da der Mitarbeiter über eine GUI direkt mit dem Formaularsystem interagiert.

- *Anforderungen/Begründung der Mitarbeiter vom MPS*

Mitarbeiter	Anforderung	Begründung
Callcenter-Agent	leichte Benutzung	-neue Mitarbeiter können schneller eingearbeitet werden -weniger Fehlerpotenzial
Monteur	Das System muss Fertigungspläne zur Verfügung stellen	Somit weiß der Mitarbeiter welche Teile er benötigt
Lagerist	Das System muss Lieferscheine zur Verfügung stellen	Damit kann gewährleistet werden, dass die richtige Ware auch beim richtigen Empfänger ankommt
Buchhaltär	100% genaue Berechnungen vom System	Falsche Rechnungen würden den Kunden irritieren und unnötigen Ärger erzeugen
Controller	Meldung vom System bei falschen/unzulässigen Eingaben von Benutzern	So können gravierende Fehler frühzeitig entdeckt werden
Logistik-Mitarbeiter	Meldung vom System bei geringen Bestand	Lieferverzögerungen können so vermieden werden

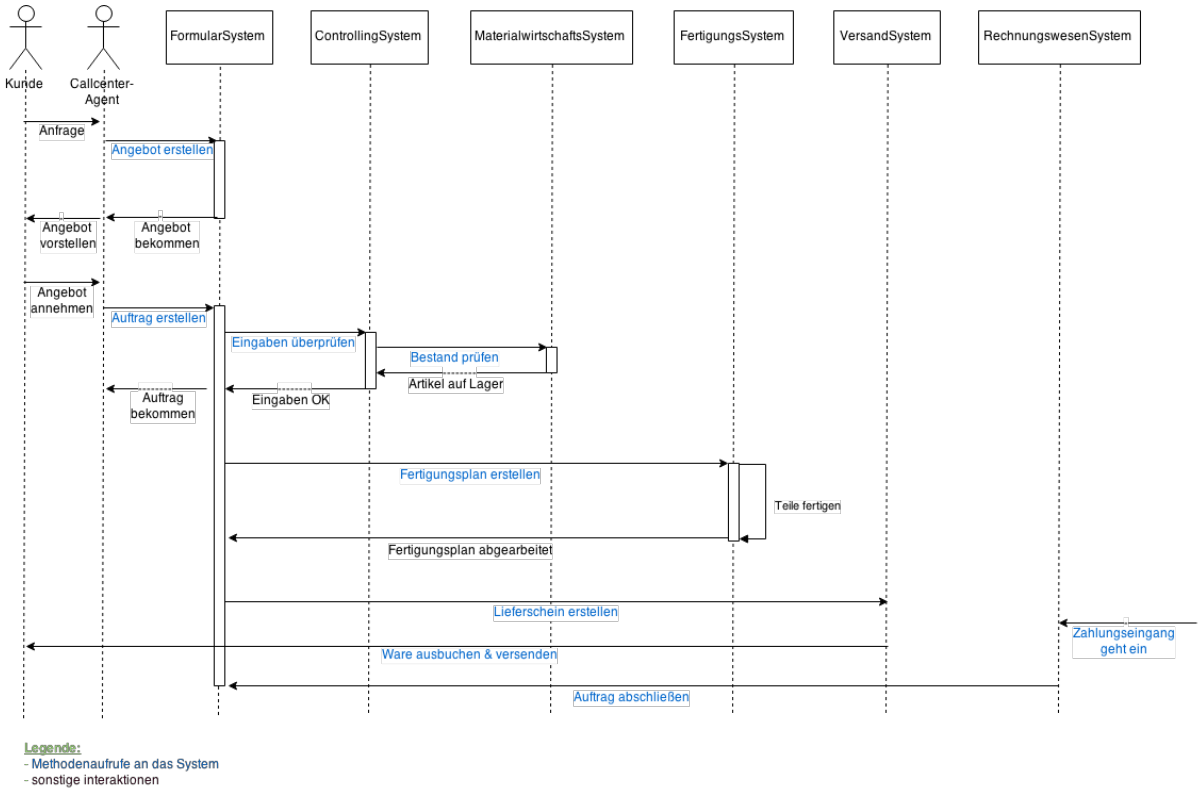
- *Interne Aufbau (Innensicht) der Formularsystem-Komponente*



Szenario für das Sequenzdiagramm

1. Kunde ruft den Callcenter-Agent an und will ein Angebot
2. Callcenter-Agent erstellt Angebot
3. Kunde entscheidet, ob er das Angebot wahrnimmt oder nicht
4. Callcenter-Agent erstellt aus Angebot einen Auftrag
5. MPS erstellt einen Fertigungsplan
6. Schleife zum Fertigen des Bauteils (komplexes;- oder Einfaches Bauteil)
7. Auftrag wird ausgeliefert. Der Versand vermerkt ,,Lieferung" erfolgt.
8. MPS erhält Zahlungseingang vom Kunden
9. Buchhaltung: Rechnung bezahlt -----> MPS -> Auftrag abgeschlossen markieren

,,Wir gehen davon aus, dass der Kunde schon im System existiert und ein Angebot haben möchte und sich für dieses entscheidet".



Parameter für das Sequenzdiagramm

1. createRequest (int customerNumber, int productNumber) -> Request request
2. createOrder (int requestNumber) -> Order order
   2.1 checkParameter (Request request) -> Bool
   2.2 checkMaterialIsAvailable (Request request) -> Bool
3. createManufacturingSchedule (int orderNumberr) -> ManufacturingSchedule schedule
4. createDeliveryNote (int orderNumber) -> DeliveryNote deliveryNote
5. setPaymentReceipt (int orderNumber)
6. bookOutProduct (Collection productList)
7. setOrderClose (int orderNumber)

Objekte

- ***R*equest**
  I. int requestNumber
  II. int customerNumber
  III. int productNumber
- **Order**
  I. int orderNumber
  II. int requestNumber
  III. Collection productList
  IV. Bool paymentReceipt
- **ManufacturingSchedule**
  I. int manufacturingNumber
  II. Date deadline
  III. int orderNumber

IV. Collection parts

- **DeliveryNote**
  I. int deliveryNoteNumber
  II. int orderNumber
  III. int customerNumber
  IV. Collection productList

Korrektheit
- Eigenschaft einer Spezifikation zu genügen => Anforderungen werden erfüllt

Usability
- Verständlichkeit und schnelle Erlernbarkeit des Systems
- Effiziente Nutzbarkeit des Systems

Verfügbarkeit
- Zeit der Systemverfügbarkeit

Interoperabilität
- Nutzung von klar definierten Schnittstellen beim Datenaustausch zwischen Systemen

Erweiterbarkeit
- Planung zukünftiger Veränderungen am System

Performanz
- Beschreibt die Antwortzeit eines Systems

Sicherheit

- Vertraulichkeit
- Verfügbarkeit
- Integrität
- Authentizität
- Verbindlichkeit
- Zurechnbarkeit
- Fehlertoleranz => versehentlicher Systemshutdown
- Unterscheidung zwischen vertrauenswürdigen und nicht vertrauenswürdigen Systemteilen / Datenquellen
- Akteuere identifizieren
- Zugriff auf Daten protokollieren (verschlüsselt)

	Call-Center Agent	Buchhalter	Versand	Fertigung	IT-Abteilung
Korrektheit	5	5	5	5	5
Usability	3	4	3	4	3
Verfügbarkeit	4	3	2	2	5
Interoperabilität	0	0	0	0	4
Erweiterbarkeit	0	0	0	0	4
Performanz	3	3	2	2	3
Sicherheit	2	5	2	2	5
Wartbarkeit	0	0	0	0	4

Korrektheit

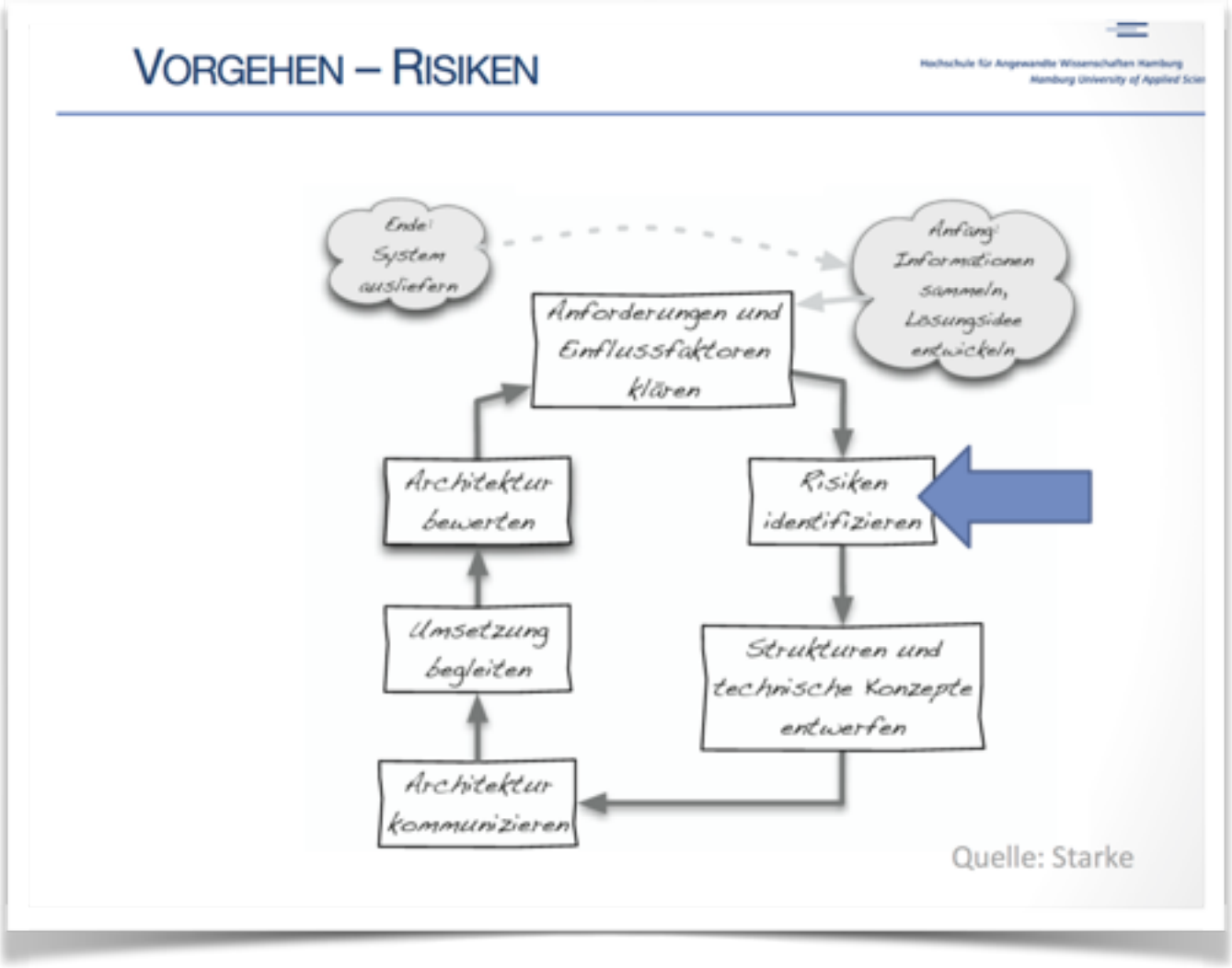
- Testautomatisierung (Regressionstests)
  - Jeden Tag zu definierten Zeiten die Software testen
  - Nebeneffekt von vorgenommenen Änderungen erkennen

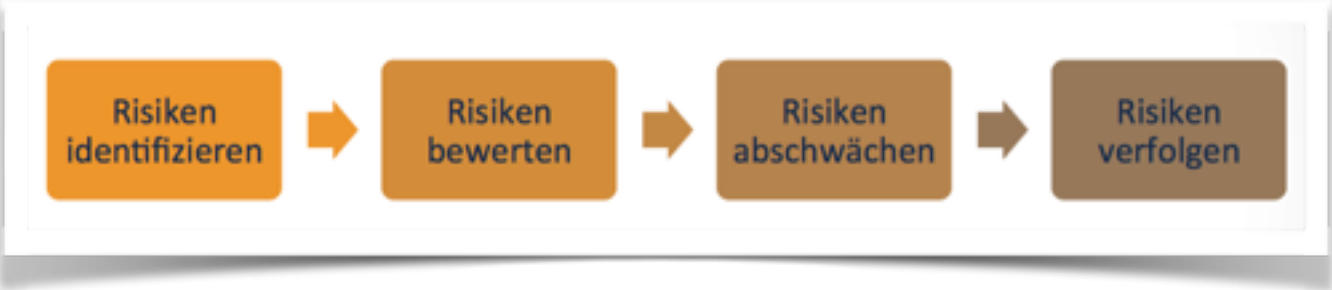
Usability

- Usability-tests
  - Versuchspersonen veranlassen typische Aufgaben mit der Software zu lösen (bsp.: Call-Center Agent erstellt ein Angebot)
  - Prüfen wo Schwierigkeiten bei der Benutzung auftreten

Sicherheit

- Alle sensiblen Daten müssen verschlüsselt gespeichert werden
- Verschlüsselter Netzwerkverkehr
- Sicherheitstests
  - Programmierfehler erkennen und beseitigen (Buffer Overflow, SQL-Injection, etc.)


VORGEHEN – RISIKEN
Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences
Ende!
System
ausliefern
Anfang!
Informationen
sammeln,
Lösungsidee
entwickeln
Anforderungen und
Einflussfaktoren
klären
Architektur
bewerten
Risiken
identifizieren
Umsetzung
begleiten
Strukturen und
technische Konzepte
entwerfen
Architektur
kommunizieren
Quelle: Starke


Risiken identifizieren
Risiken bewerten
Risiken abschwächen
Risiken verfolgen

Indizierung der möglichen Architekturrisiken im Projekt

Risiken durch Projektrahmen

Jetzt zu treffende Entscheidungen kann später voraussichtlich nicht mehr zurück genommen werden.
Unsere Gründe sind dafür die Zeit.

Variabilitätsrisiken

Anforderungen sind unrealistisch bezüglich der Fremdsysteme. Da man sich in der Realität mit den Anbieter der Fremdsysteme auf eine Schnittstelle einigen müsste. Wir denken uns irgendwas aus, da unsere Anbieter der Fremdsysteme Fiktiv sind.

Anforderungen wurde nicht priorisiert, dass kann auch zu einen Problem führen, wenn z.B. mehr Zeit in nicht so wichtige Dinge gesteckt wird.

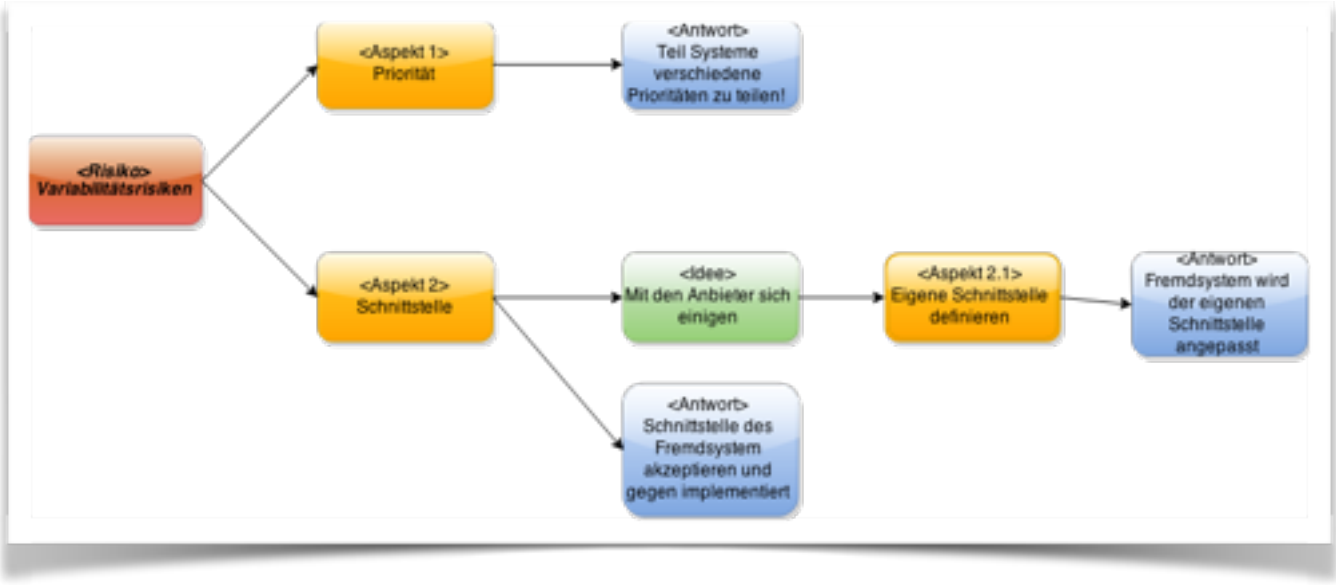
Know-how-Risiken

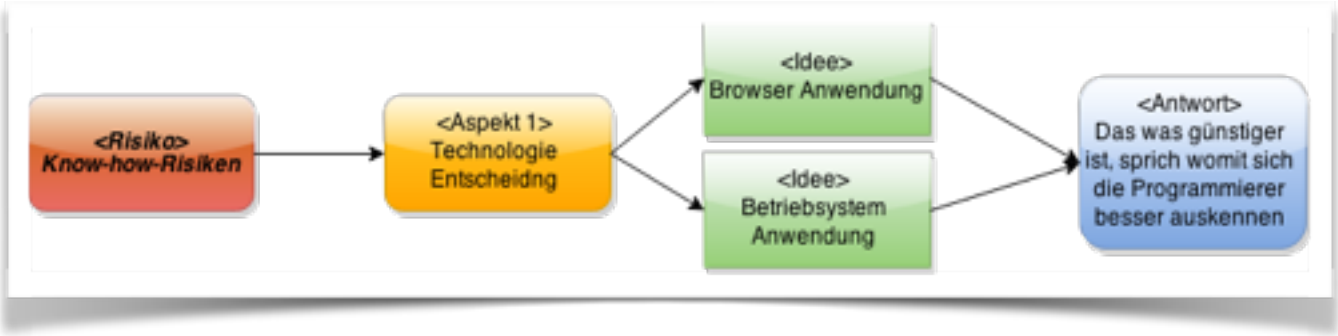
Wir haben uns noch nicht entschieden ob unser MPS über eine Application oder eine Browser Oberfläche benutzt werden soll, dass könnte zu einen Problem führen, wenn man hier nicht schnellstmöglich eine Entscheidung trifft.

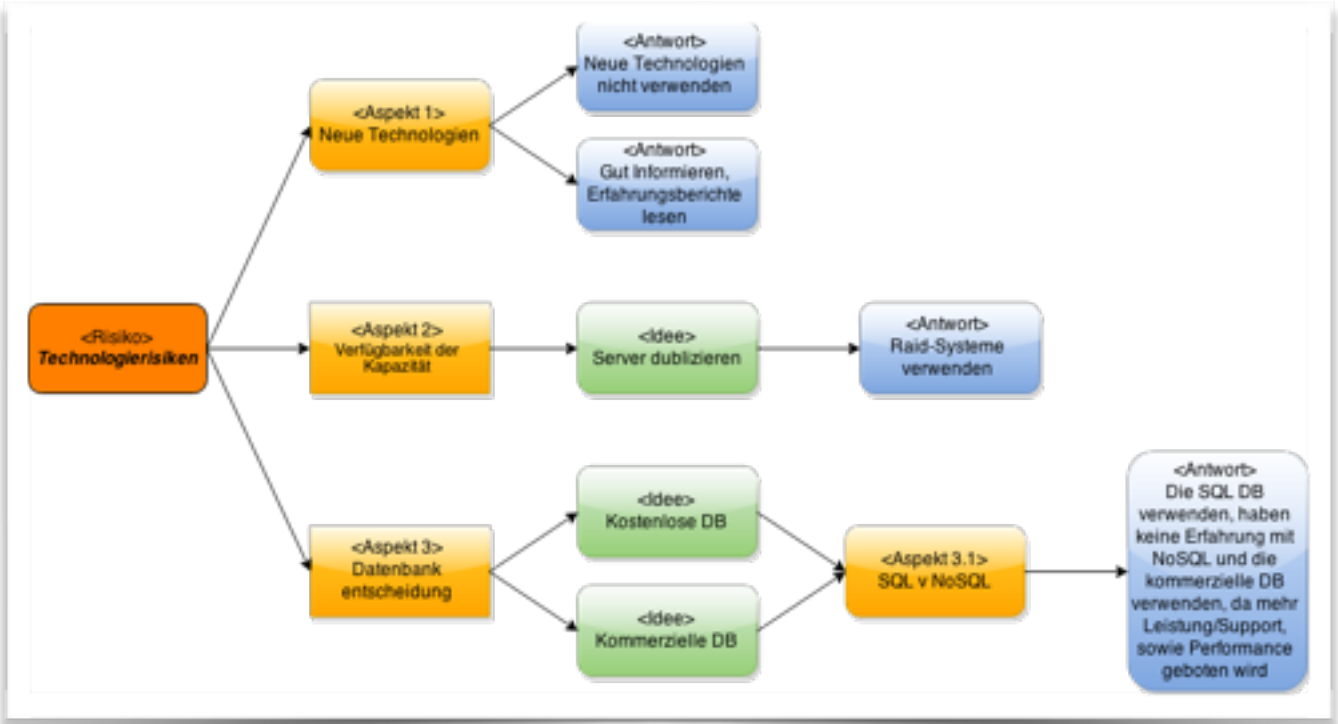
Technologierisiken

Neue Technologien, Leute müssen evtl. eingearbeitet werden.
Verfügbarkeit der Kapazität & Verfügbare Datenbankleistung, in beiden sollte man vorher schon abwiegen wie die Komplexität des Projektes ist und ob die Technologien für das MPS-System geeignet sind.

SQUID-Diagramme







Bewertung der Architekturrisiken

ID	Name	Auswirkung (0 - 9)	Auswirkung Beschreibung	Eintritt (0 - 9)	Eintritt Beschreibung	Maßnahme
1	***Risiken durch Projektrahmen***	8	Große Geld und Zeit Verluste können dadurch entstehen	3	Mehr Daten als geplant kommen dazu, Kapazität wird ausgeschöpft	Neue Technologien oder Erweiterung der alten Technologien
2	***Variabilitätsrisiken***	9	Systeme funktionieren nicht	1	Schlechte Kommunikation. Streit mit der anderen Partei	Verhandeln, gut kommunizieren
3	***Know-how-Risiken***	4	Mitarbeiter sind unzufrieden mit den System	4	Man entscheidet sich für eins der Technologien	Neue GUI bauen
4	***Technologierisiken***	9	System wird nicht fertig. Kann nicht erweitert werden	7	Man sucht sich die falsche Technologie aus	Andere Technologie benutzen

Risiken die gemanagt werden müssen

Die Risiken mit den ID’s 1, 3, 4 sollten gemanagt werden, da die Eintrittswahrscheinlichkeit relativ hoch ist und die Auswirkung relativ stark ist.

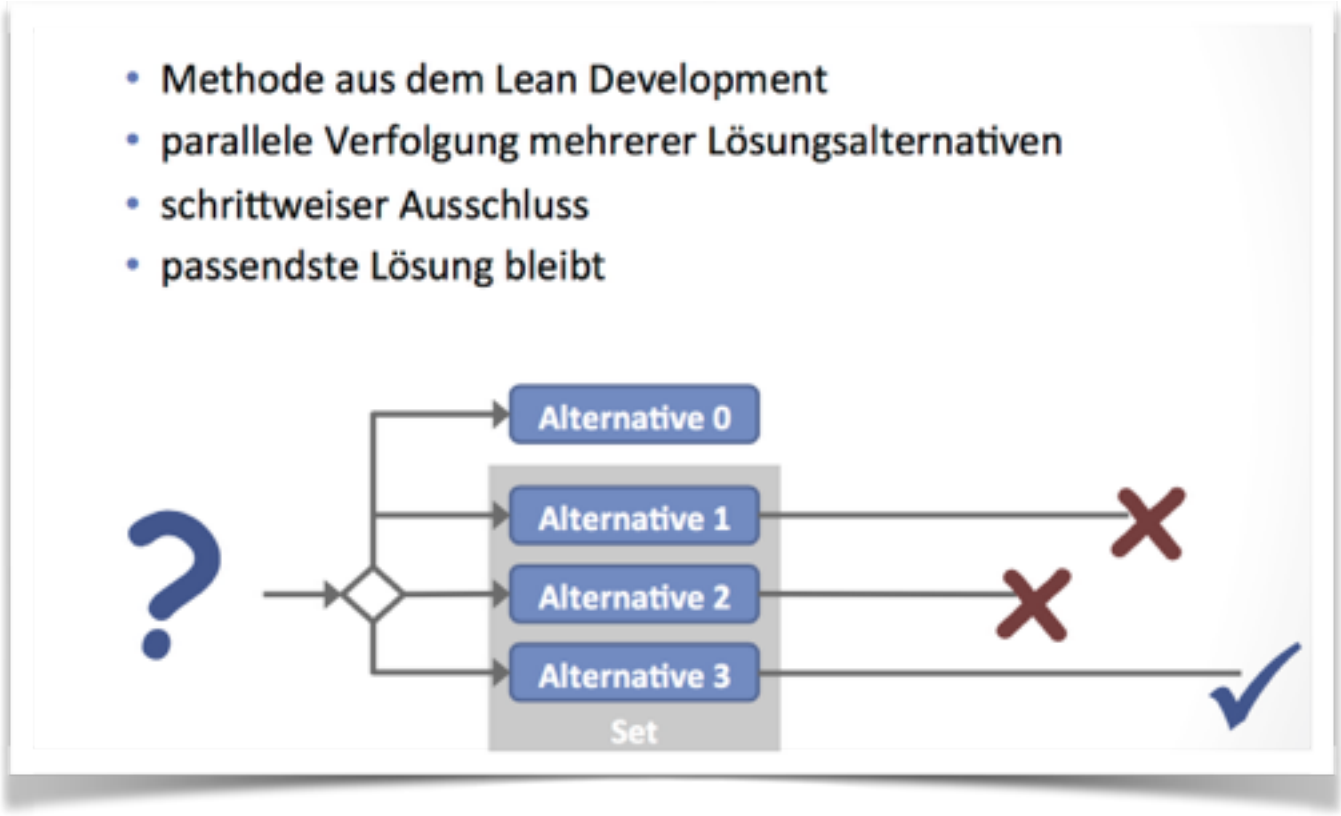
Maßnahmen zur Abschwächung von Risiken

Risiken durch Projektrahmen: Gut überlegen für welchen Design / Architektur man sich entscheidet. Gut überlegt planen, bevor man anfängt Technologie etc. einzukaufen.

Know-how-Risiken: Oft überlegt entscheiden und erst dann entscheiden, wenn es wirklich notwenig ist und nicht schon vorher.
SQUID Diagramm erstellen, durch gehen. Dies kann bei Entscheidungen sehr Hilfreich sein.

Technologierisiken: Prototypen verwenden wenn man sich nicht sicher ist ob es mit dieser bestimmten Technologie so klappen kann oder nicht, sprich experimentieren.

Beispiel:



Dies kann sich jedoch als kostenpflichtig erweisen!

Erfahrungsberichte lesen oder sich etwas bei der Konkurrenz anschauen, sofern die Konkurrenz erfolgreich mit ihren Produkt ist.