Master Thesis

Zukunftsfähige Informationssystem-Architektur

---

von

Lukas Fuchs

Studiengang

Master of Science in Wirtschaftsinformatik

Zürcher Hochschule für Angewandte Wissenschaften ZHAW

Matrikelnummer:	06-530-745
E-Mail:	lukas.fuchs@gmail.com
Telefon:	+41 79 240 01 68
Erstgutachter:	Prof. Dr. Andreas Spichiger
	Berner Fachhochschule, Wirtschaft
Zweitgutachter:	Prof. Dr. Thomas Jarchow
	Berner Fachhochschule, Wirtschaft
Abgabe:	Zürich, 24. Mai 2018

«Wer das Ziel kennt, kann entscheiden.
Wer entscheidet, findet Ruhe.
Wer Ruhe findet, ist sicher.
Wer sicher ist, kann überlegen.
Wer überlegt, kann verbessern.»

Konfuzius

Wahrheitserklärung

«Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig, ohne Mithilfe Dritter und nur unter Benützung der angegebenen Quellen verfasst habe und dass ich ohne schriftliche Zustimmung der Studiengangleitung keine Kopien dieser Arbeit an Dritte aushändigen werde.»

Gleichzeitig werden sämtliche Rechte am Werk an die Zürcher Hochschule für Angewandte Wissenschaften (ZHAW) abgetreten. Das Recht auf Nennung der Urheberschaft bleibt davon unberührt.

Name/Vorname Student/in (Druckbuchstaben)

Lukas Fuchs

Unterschrift (Student/in)

Danksagung

An dieser Stelle möchte ich mich bei all jenen bedanken, die mich während der Erstellung dieser Masterarbeit unterstützt haben.

Zuerst gebührt mein Dank Herr Prof. Dr. Andreas Spichiger, der durch sein Interesse und seinem Engagement in der Betreuung diese wissenschaftliche Abschlussarbeit erst ermöglicht hat.

Auch möchte ich mich bei meiner Verlobten Melanie Böllenrücher, meiner Familie Daniel, Silvia und Janik Fuchs, meinem Kommilitonen Matthias Imhof und meinem Arbeitgeber Trivadis bedanken, die mich während der Erarbeitung der vorliegenden Masterarbeit motiviert und unterstützt haben.

Zürich, 24. Mai 2018

Lukas Fuchs

Management Summary

Disruptive Geschäftsmodelle und globale Mitbewerber können Unternehmen vor existenzielle Herausforderungen stellen. Hierauf muss zeitnah mit innovativen Produkten und Services geantwortet werden können. Als Fundament für diese neuen Lösungen braucht ein Unternehmen eine zukunftsfähige Informationssystem-Architektur, die sowohl eine digitale Service Platform als auch ein starkes betriebliches Rückgrat (Operational Backbone) bietet. Dies kann erreicht werden, indem die IS-Architektur sowohl wandlungsfähig als auch nachhaltig, beständig und resilient gestaltet wird. Dadurch wird es z.B möglich spezialisierte Partnerdienstleistungen interoperabel, lose gekoppelt und mit geringem Aufwand in den eigenen Geschäftsprozess zu integrieren.

Ziel dieser Master-Thesis ist daher die Entwicklung eines Informationssystem-Referenzarchitektur-Artefakts, das auf die Förderung der Zukunfsfähigkeit ausgerichtet ist. Dieses Artefakt wird basierend auf einer deduktiven Design-Science-Methodologie erstellt und szenariobasiert, angelehnt an eine Fallstudie, validiert.

Zuerst werden hierbei die internen und externen Einflussfaktoren auf ein Unternehmen und seine Informationssystem-Architektur definiert. Anschliessend wird eine Prinzipien-Baustein-Pyramide konstruiert, die sich nach den Perspektiven «Wandlungsfähigkeit» und «Nachhaltigkeit» richtet, welche beide die Zukunftsfähigkeit des Unternehmens fördern. Danach werden prinzipienunterstützende Paradigmen – wie Verfügbarkeit und Multimodalität – hergeleitet und durch entsprechende Patterns und Lösungstechnologie-Beispiele gestützt. Diese aufgegriffenen Architektur-Patterns werden im Anschluss analysiert und hinsichtlich der Unterstützung der Zukunftsfähigkeitsparadigmen bewertet.

Ein wichtiges Mittel für das Vermitteln von Wissen über die komplexen Zusammenhänge einer Informationssystem-Architektur sind strukturierte Abbildungen von Anwendungslandschaften. Hierfür wird die Informationssystem-Architektur – ausgehend von der Geschäftsfähigkeiten-basierten Unternehmensarchitektur – entlang einer sowohl ressourcenunabhängigen als auch ressourcenspezifischen Sichtweise entwickelt. Die zuvor bewerteten Informationssystem-Architektur-Patterns werden darauf zu einem generischen, zukunftsfähigen Referenzarchitektur-Artefakt zusammengefügt und szenariobasiert – anhand einer integrierten Einzelfallstudie – validiert sowie entsprechende Verbesserungsmassnahmen ausgearbeitet.

Folgende essenzielle Architektur-Patterns sind hierbei benutzt worden: die Gestaltung digitaler Organisationen nach Ross et al., die Geschäftsinteroperabilitätsschnittstellen zwischen Partnern nach Ziemann, Event-basierte Architekturkonzepte sowie Microservice-, Serverless Functions- und Big Data-Architekturmodelle.

Eine Weiterführung dieser Master-Thesis könnte darin bestehen, das Artefakt mit zusätzlichen Szenarios der Fallstudie zu prüfen, einen Prototypen zu realisieren und auf Experteninterviews gestützte Erkenntnisse einfliessen zu lassen.

Inhaltsverzeichnis

1 Einleitung **1**

1.1 Ausgangslage . . . 1
1.2 Problemstellung . . . 2
1.3 Zielsetzung . . . 4
1.4 Forschungsfragen . . . 5
1.5 Forschungsdesign und methodisches Vorgehen . . . 5
1.5.1 Forschungs-Guidelines . . . 6
1.5.2 Conceptual Model . . . 8
1.5.3 Argumentation zum gewählten Vorgehen . . . 9
1.6 State of the Art . . . 9

2 Geschäftsfähigkeiten-basierte Unternehmensarchitektur (GFbUA) **12**

2.1 Ressource Independent Model (RIM) . . . 12
2.2 Ressource Specific Model (RSM) . . . 13

3 Informationssystem-Architektur (ISA) **16**

3.1 ISA-Definition . . . 17
3.2 ISA-Zielsetzung - Der Idealzustand oder «der Weg ist das Ziel» . . . 17
3.3 ISA-Strukturierung . . . 19
3.3.1 ISA-Komplexitätsbeherrschung . . . 19
3.3.2 ISA-RSM-Verknüpfung . . . 21
3.4 ISA-Einflussfaktoren und -Prinzipien . . . 24
3.5 ISA-Paradigmen . . . 29
3.5.1 WP1 - Modularität . . . 31
3.5.2 WP2 - Interoperabilität . . . 34
3.5.3 WP3 - Mobilität . . . 35
3.5.4 NP1 - Verfügbarkeit . . . 37
3.5.5 NP2 - Zustandslosigkeit . . . 39
3.5.6 NP3 - Reaktivität . . . 40
3.5.7 NP4 - Selbstorganisation und Selbstähnlichkeit . . . 43
3.5.8 FP1 - Skalierbarkeit . . . 43
3.5.9 FP2 - Wissen über die IS-Architektur . . . 45
3.5.10 FP3 - Multimodale IS-Architektur . . . 46
3.6 ISA-Patterns . . . 50
3.6.1 Pattern A - Digital Organisation Design . . . 51
3.6.2 Pattern B - Architecture of Interoperable Information Systems . . . 56
3.6.3 Pattern C - Microservice-Architektur . . . 61
3.6.4 Pattern D - Quasar IS-Architektur . . . 66

3.6.5 Pattern E - Serverless Functions-Architektur 69
3.6.6 Pattern F - Big Data-Architektur 72
3.6.7 Zusammenfassung und Vergleich der ISA-Patterns A-F 77
3.7 Kombination der ISA-Patterns zu einer RSM-ISA-Referenzarchitektur . . 79

4 Anwendung des RSM-ISA-Referenzarchitektur-Artefakts 81
4.1 Fallstudie «Arthur Reisen» (AR) 81
4.1.1 Ableitung der ISA-Leistungsanforderungen 82
4.1.2 RSM-ISA-Validierungsszenarios 89
4.1.3 Validierung anhand Szenario «Geschäftsprozess Reise buchen» . 90
4.2 Beurteilung und Verbesserungsmassnahmen 91
4.2.1 Forschungs-Guideline 3 (Design Evaluation) 91
4.2.2 Forschungs-Guideline 6 (Design as a Search Process) 92

5 Ergebnisse 93
5.1 Beantwortung der Forschungsfragen 93
5.2 Diskussion 94
5.2.1 Beurteilung zur Einhaltung der Forschungs-Guidelines 94
5.2.2 Fazit 95
5.3 Ausblick und weiterführende Forschungen 96

Literaturverzeichnis 97

Anhang 106
A Qualitätsmodell für externe und interne Qualität nach ISO 25010 106
B Beispiel einer Pace Layered-Architektur 106
C Alternative ISA-RSM-Validierungsszenarios nach Wertangebot AR . . . 107
D Geschäftsprozess «Reise buchen» nach GFbUA 108

Abbildungsverzeichnis

1 Information Systems Research Framework; Quelle: Hevner et al. [HMR04, S.80]) . . . 6
2 Grundtypen eines Fallstudienaufbaus; Quelle: Yin [Yin09] . . . 7
3 Der Entwicklungs-/Test-Kreislauf; Quelle: Simon [Sim96] . . . 8
4 Conceptual Model zum Vorgehen . . . 9
5 Konzeptionelles Interdependenzenmodell zur Herleitung der zukunftsfähigen ISA-Referenzarchitektur . . . 11
6 Ordnungsraster für Teilmodelle der Unternehmensarchitektur; Quelle: Spichiger & Noser [SN, S.5] . . . 12
7 Metamodell der Geschäftsfähigkeit; Quelle: Spichiger & Noser [SN, S.4] 13
8 Eingliederung von RIM, RSM und Ressourcen nach GFbUA; Quelle: Spichiger & Noser [SN, S.5] . . . 14
9 Von der Geschäftsstrategie zur IS-Architektur; Quelle: Engels et al. [EJH+08, S.79] . . . 15
10 Deduktives Ableiten und iteratives Validieren . . . 16
11 Ist, Soll und Ideal im Rahmen einer gesteuerten Evolution; Quelle: Engels et al. [EJH+08, S.8] . . . 19
12 Begriffe und Zusammenhänge von Anwendungslandschaften in der Übersicht; in Anlehnung an Engels et al. [EJH+08, S.102] . . . 20
13 ISA-Architekturebenen; Quelle; Engels et al. [EJH+08, S.2] . . . 21
14 Logische ISA-Gliederung; Quelle; Engels et al. [EJH+08, S.2] . . . 21
15 ISA-RSM-Verknüpfung, in Anlehnung an Spichiger & Noser [SN, S.52] . 22
16 IS-Architektur im GFbUA-Kontext von RIM und RSM . . . 24
17 Abhängigkeiten von externen und internen Einflussfaktoren mit ihren exemplarischen Zeithorizonten . . . 27
18 Prinzipien-Pyramide der Zukunftsfähigkeit; in Anlehnung an Spath et al. [SBBD01, S.9] . . . 29
19 Monolythische Architektur und Microservice-Architektur im Vergleich; in Anlehnung an Fowler [Fow14] . . . 32
20 Ausbaustufen der verschiedenen «as-a-Service»-Angebote . . . 37
21 Charakteristiken eines reaktiven Systems; in Anlehnung an Bonér et al. [BFKT14] . . . 42
22 Ausschnitt aus dem Gestaltungsmodell digitaler Organisationen; Quelle: Ross et al. [RSBJ17] . . . 46
23 Zusammenhang zwischen multimodalen Ansprüchen und den verschiedenen Funktionsprinzipien; Quelle: Gaughan [Gau18] . . . 48
24 Architekturebenen-Indikator . . . 50

25 Gestaltungsmodell digitaler Organisationen; Quelle: Ross et al. [RSBJ17, S.4] . . . 52
26 Netzdiagramm zur Bewertung von Pattern A - Digital Organization Design 56
27 Architecture of Interoperable Information Systems; Quelle: Ziemann et al. [Zie10] . . . 59
28 Netzdiagramm zur Bewertung von Pattern B - Architecture of Interoperable Information Systems . . . 61
29 Schematische Microservice-Architektur; Quelle: Hert [Her16] . . . 62
30 Resiliente Microservice-Architektur; Quelle: Tresh [Tre18] . . . 63
31 Netzdiagramm zur Bewertung von Pattern C - Microservice-Architektur . 66
32 Begriffe und Zusammenhänge von Anwendungslandschaften in der Übersicht; Quelle: Engels et al. [EJH+08, S.79] . . . 67
33 Netzdiagramm zur Bewertung von Pattern D - Quasar-IS-Architektur . . . 69
34 Netzdiagramm zur Bewertung von Pattern E - Serverless Functions-Architektur . . . 72
35 Lambda-Architektur für Big Data; Quelle: Schmutz [Sch17a] . . . 74
36 Kappa-Architektur für Big Data; Quelle: Schmutz [Sch17a] . . . 74
37 Unified-Architektur für Big Data; Quelle: Schmutz [Sch17a] . . . 75
38 Netzdiagramm zur Bewertung von Pattern F - Big Data-Architektur . . . 77
39 ISA-Patterns Netzdiagramm . . . 78
40 ISA-Patterns Balkendiagramm . . . 78
41 RSM-ISA-Referenzarchitektur . . . 80
42 Business Model Canvas nach Osterwalder [OPW11] zum Geschäftsmodell Arthur Reisen; Quelle: Atilgang & Hurst [AH16]. . . . 82
43 Zusammenhang zwischen Strategie und Achitektur; Quelle: Engels et al. [EJH+08, S.79] . . . 83
44 (RSM-)IS-Architektur aus Sicht der Leistungsanforderungen AR . . . 89
45 RSM-ISA-Artefakt für den Geschäftsprozess «Reise buchen» von AR . . 91
46 Qualitätsmodell für externe und interne Qualität nach ISO 25010; Quelle: Herzwurm & Mikusz [HM16] . . . 106
47 Beispiel einer Pace Layered-Architektur; Quelle: Gaughan [Gau18] . . . 106
48 Überblick über den Geschäftsprozess «Reise buchen»; Quelle: Spichiger & Noser [SN, S.31] . . . 108
49 Geschäftstransaktionen des Geschäftsprozesses «Reise buchen»; Quelle: Spichiger & Noser [SN, S.32] . . . 108
50 Zustandsdiagramm von «Reisebuchung»; Quelle: Spichiger & Noser [SN, S.32] . . . 109
51 Geschäftstransaktion «Reise auf Option buchen»; Quelle: Spichiger & Noser [SN, S.32] . . . 109

52 Objektdiagramm nach der Geschäftstransaktion «Reise auf Option buchen»; Quelle: Spichiger & Noser [SN, S.33] 109
53 Geschäftstransaktion «Reise fix buchen»; Quelle: Spichiger & Noser [SN, S.33] 110
54 Objektdiagramm nach der Geschäftstransaktion «Reise fix buchen»; Quelle: Spichiger & Noser [SN, S.34] 110
55 Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.34] 110
56 Objektdiagramm nach der Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.35] 111
57 Umsetzung der Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.35] 111

Tabellenverzeichnis

1 ISA-Prinzipien zu -Paradigmen Verknüpfung 30
2 Bewertung Pattern A - Digital Organization Design 55
3 Bewertung Pattern B - Architecture of Interoperable Information Systems 60
4 Bewertung Pattern C - Microservice-Architektur 65
5 Bewertung Pattern D - Quasar IS-Architektur 68
6 Bewertung Pattern E - Serverless Functions-Architektur 71
7 Bewertung Pattern F - Big Data-Architektur 76
8 ISA-Leistungsanforderungen abgeleitet vom Geschäftsmodell AR 88
9 Geschäftsfähigkeiten, -prozesse und -objekte von Arthur Reisen; Quelle: Spichiger & Noser [SN, S.24] 90
10 Abgeleitete ISA-RSM-Validierungsszenarios nach Wertangebot von Arthur Reisen 107

Akronyme und Glossar

AIOS

Architektur interoperabler Informationssysteme; Referenzarchitektur für die Entwicklung von interoperablen Informationssystemen nach Ziemann [Zie10]. 3, 7, 10, 56

AL

Anwendungslandschaft; Bezeichnet die Gesamtheit der Anwendungssysteme, die ein Unternehmen zur Organisation und Abwicklung seines Geschäfts betreibt. Die Anwendungssysteme stehen meistens nicht für sich alleine, sondern sind über gemeinsame Datenbanken oder Schnittstellen miteinander vernetzt. Diese Abhängigkeiten gehören ebenfalls zur Anwendungslandschaft; nach Engels et al. [EJH+08]. 17, 19, 21

AR

Arthur Reisen; Ein fiktives Reiseanbieterunternehmen aus einer Fallstudie nach Spichiger & Noser [SN]. 4, 81

BII

Eine umfassende Schnittstelle, die das Interaktionspotenzial einer Organisation aus betriebswirtschaftlicher und technischer Sicht beschreibt und als eine Schnittstelle für interne und externe Prozesse dient; nach Ziemann [Zie10]. 3, 10, 35, 46, 57, 79

CKR

Christoph Kolumbus Reisen; Ein fiktives Reiseanbieterunternehmen aus einer Fallstudie nach Engels et al. [EJH+08]. 81, 83

FaaS

Function as a Service; Bei Function as a Service wird ausschliesslich die Geschäftslogik selbst verwaltet, während diese bei Software as a Service vom SaaS-Anbieter gemanagt wird. Als Abgrenzung zu Backend as a Service werden bei FaaS nur einzelne Funktionen, nicht ganze Anwendungen selbst implementiert. 3, 36, 37, 39, 43, 45, 69

GFbUA

Geschäftsfähigkeiten-basierte Unternehmensarchitektur nach Spichiger & Noser [SN]. 4, 12, 16, 59, 80, 81

IS

Informationssystem; Ist ein soziotechnisches System, das die Deckung von Informationsnachfrage zur Aufgabe hat. 2, 17, 56

ISA

Informationssystem-Architektur. 1, 16

MSA

Microservice-Architektur; Ein Architekturmuster der Informationstechnik, bei dem komplexe Anwendungssoftware aus unabhängigen Prozessen komponiert wird, die untereinander mit sprachunabhängigen Programmierschnittstellen kommunizieren. Die Dienste sind weitgehend entkoppelt und erledigen eine kleine Aufgabe. 2, 31, 33, 36, 37, 39, 43–45, 55, 61

RIM

Resource Independent Model; Ein ressourcenunabhängiges Modell nach Spichiger & Noser [SN]. 12, 21, 59

RSM

Resource Specific Model; Ein ressourcenspezifisches Modell nach Spichiger & Noser [SN]. 4, 12, 16, 19, 21, 59

SOA

Service Oriented Architecture; Architekturmuster der Informationstechnik aus dem Bereich der verteilten Systeme, um Dienste von IT-Systemen zu strukturieren und zu nutzen. 2, 31, 44, 56

SWOT

Analysemodell mit einer Unterteilung nach Strengths (Stärken), Weaknesses (Schwächen), Opportunities (Chancen), Threats (Bedrohungen). 4, 54, 60, 64, 67, 70, 75, 93

1 Einleitung

Dieses Kapitel beschreibt unter Abschnitt 1.1 die Ausgangslage für diese Master-Thesis. Danach wird unter 1.2 die eigentliche Problemstellung erläutert und unter 1.3 die damit einhergehende Zielsetzung definiert. Abgeleitet von der Zielsetzung werden unter 1.4 die Forschungsfragen dieser Arbeit beschrieben. Unter Abschnitt Forschungsdesign 1.5 werden das gewählte Vorgehen aus Sicht der Forschung sowie die angewendete Forschungsmethodologie erläutert. Zum Abschluss wird unter 1.6 der aktuelle Stand der Forschung unter Angabe der wichtigsten verwendeten Literatur geschildert.

1.1 Ausgangslage

«Die Daten über uns sind das Erdöl der Neuzeit» vermeldet «Der Bund» [Bü14], auch der Deutsche Bundesminister des Innern sieht «Daten als Rohstoff der Zukunft» [Inn17]. Dies zeigt klar: das Thema «Daten» beschäftigt Gesellschaft und Politik. Auch erschliessen immer mehr Unternehmen neue, datenzentrierte Geschäftsmodelle [Kau15], seien es «Big Data» Lösungen für «Predictive Maintenance» von Flugzeug-Turbinen [SP14], «Adword» Algorithmen auf Suchmaschinen [WC12] oder «Internet of Things» Sensordaten in Windenergieanlagen [IWB17]. Die entsprechende Forschung beschäftigt sich daher damit, wie sich diese internen und externen Daten und die daraus gewonnenen Informationen im eigenen Unternehmen möglichst effektiv, effizient und nachhaltig mittels Informationssystem-Architekturen (ISA) einsetzen und beherrschen lassen.

Durch die unaufhaltsame Verbreitung des Internets und die damit einhergehende, stetig zunehmende Vernetzung, Globalisierung und Verbreitung des Wissens durchläuft die Informationstechnologie eine kontinuierliche Weiterentwicklung und erfindet sich quasi laufend neu. «Ideal» [EJH+08, S.143] Informationssystem-Architekturen sollen in dieser dynamischen Welt durch zuverlässige, robuste Strukturen und Standards als Leuchtturm dienen, nach denen Anwendungslandschaften ausgerichtet werden. So kann sichergestellt werden, dass Informationssysteme die benötigten Informationen trotz stetigem Wandel einfach und zuverlässig an ihre Bezüger liefern – seien dies Applikationen, die Geschäftsfunktionen bedienen, Prozessinformationen für Mitarbeiter bereitstellen oder Auswertungen für den Finanzabschluss ausführen. Informationssystem-Architekturen sind dabei als ausgeklügeltes «Ökosystem» zu betrachten, welches Zachmann [Zac02, S.4] als ein vielschichtiges Zusammenspiel von Daten, menschlichen Akteuren, Geschäftsfunktionen, Prozessen, Organisationsstrukturen sowie weiteren Komponenten definiert, die orchestriert werden müssen.

Im übertragenen Sinn können Unternehmen als menschlicher Körper interpretiert werden, wobei die Informationssystem-Architekturen einen wesentlichen Teil zum längerfristigen Dasein eines Wesens beitragen, da sie quasi das Gesamtkonzept für die Zellen

liefern, die sich einerseits ständig erneuern und andererseits in ähnlicher Form wieder in ein Gesamtsystem integrieren müssen. Organe (Informationssysteme), welche im Grossen und Ganzen nach einem «Soll» ISA-Bauplan ausgerichtet sind, unterstützen so nachhaltiges Wachstum. Auch wenn vereinzelt Krebszellen (Systemanomalien) auftreten sollten, ist das Gesamtsystem robust genug, um ein möglichst langes Überleben sicherzustellen, sofern sich der (ISA-) Bauplan an einer soliden (Überlebens-) Strategie orientiert.

Diese Analogie zeigt auf, dass es im Kern von Informationssystemen (IS) immer um einen möglichst effektiven Informationskreislauf in einem Gesamtsystem geht. Das IS soll durch zukunftsfähige ISA-Patterns optimal für interne und externe Veränderungen sowie wechselnde Anforderungen vorbereitet sein.

1.2 Problemstellung

Die durch eBusiness und den globalen Marktzugang herbeigeführte Preissensitivität der heutigen Kunden geht zu Lasten von Kundenloyalität [CKKK06, S.925]. Dies bedeutet für Unternehmen unter anderem, dass sie vermehrt Services durch spezialisierte, kosteneffiziente Dienstleister beziehen müssen. Hierdurch lassen sich einerseits Kostensenkungen realisieren und durch das föderative Zusammenarbeiten auch innovative Produktideen schneller am Markt platzieren. In der Quintessenz lässt sich somit auch besser auf die im globalen Kontext vermehrt auftretenden, disruptiven Geschäftsmodelle von Marktbegleitern antworten [Ter17].

In Bezug zur Informationssystem-Architektur bedeutet dies, dass neben den aufbau- und ablauforganisatorischen Fragen vor allem die Informationssysteme gefordert sind. Sie müssen einerseits architektonischen Vorgaben Rechnung tragen und gleichzeitig die benötigte Time-to-Market erfüllen. Dies erfordert Services, welche die saubere Integration von neuen oder geänderten Prozessen und Daten unterstützen. Es zeigt sich aber, dass gerade im Bereich der Datenintegration grosse Herausforderungen bestehen [EFNV16, S.3]. Schliesslich brauchen neue Produkte und damit zusammenhängende Prozesse spezifische Daten aus vorhandenen Systemen und müssen die (angereicherten) Daten wiederum sauber zurückspeisen. Erweiterungen innerhalb von traditionellen, monolithischen Systemen sind sehr zeitaufwändig, teuer und wenig zukunftsoffen. So wird offensichtlich: solche Ansätze sind nicht länger zielführend. Informationssystem-Architekturen müssen im Sinne einer eventorientierten Architektur möglichst modular aufgebaut sein. Zudem muss die funktionale Erweiterbarkeit, Skalierbarkeit und Wiederverwendbarkeit von Lösungskomponenten gewährleistet sein. Dies erfordert, dass Schnittstellen rasch an neue Anforderungen angepasst werden können. Hierzu bietet sich beispielsweise der in der Praxis verbreitete Service Oriented Architecture (SOA) [EJH+08, S.91] Standard an, der durch weitergehende Ansätze wie Microservice-Architekturen (MSA) eine kontinuierliche Weiterentwicklung der Systeme in Bezug auf Funktion und Architektur ermöglicht und die tech-

nischen Komponenten als Open Source zur Verfügung stellt [Sch17b, S.4]. Im Zusammenhang mit (proprietären) Cloudlösungen kann hierbei auch von Function as a Service (FaaS) [Avr16] gesprochen werden. Ross et al. [RWR06] empfiehlt als anzustrebendes Architektur-Paradigma im globalisierten Umfeld gar einen «Dynamic Venturing» Ansatz. Dieser basiert auf einer organisationsübergreifenden Zusammenarbeit mit den jeweils weltweit führenden Lösungsanbietern, welche an jedem Punkt der Wertschöpfungskette anhand von dynamisch gekoppelten (Teil-) Services agieren. Durch das damit verbundene Outsourcing stellen sich ganz neue Herausforderungen an die Informationssystem-Architektur eines Unternehmens, denn die Lösungskomponenten müssen vorausschauend entwickelt werden (in Anlehnung an Jackson [Jac82, S.2]), damit sie bei Bedarf rasch, föderativ und über die Organisationsgrenzen hinaus genutzt werden können. Ein wichtiger Aspekt einer zukunftsfähigen Architektur ist folglich die Interoperabilität und die damit einhergehende Transparenz der Services und ihrer Schnittstellen. Einen Ansatz hierfür bietet das sogenannte Business Interoperability Interface (BII), welches auf der Referenzarchitektur Interoperabler Informationssysteme (AIOS) aufbaut [Zie10, S.232]. Dass Anforderungen wie Wiederverwendbarkeit, Transparenz und Interoperabilität zentral sind, unterstreicht auch ein wegweisendes Manifest des Council of EU [EU17], welches kürzlich auf interstaatlicher Ebene ratifiziert wurde. Darin wurden die folgenden Grundprinzipien für öffentliche Institutionen hinsichtlich eGovernment festgehalten:

1. «Digital-by-default, inclusiveness and accessibility» - Standardmässig digitale Interaktion / Zugriff auf Daten und Prozesse ermöglichen.
2. «Once only» - Redundanzen vermeiden, eine Kultur der Wiederverwendbarkeit anstreben.
3. «Trustworthiness and Security» - Sicherheitsaspekte wie eine elektronische Identifikation der Benutzer.
4. «Openness and Transparency» - Digitale Datenverwaltung der eigenen Daten, Transparenz und Verfügbarkeit über Schnittstellen.
5. «Interoperability by default» - Neue Systeme sollen interoperabel sein und auf offenen Standards beruhen.
6. «Horizontal enabling policy steps» - Unterstützung von datengetriebenen Services und «decision making» mit Hilfe von Analytics.

Diese Deklaration verdeutlicht den «Common Sense», dass eine effiziente Datenhaltung und die dadurch ermöglichten Business-Intelligence Capabilities im immer stärker datengetriebenen Unternehmenskontext an Relevanz gewinnen. Die sich mit diesen Prinzipien deckenden Business Intelligence-Anforderungen an eine IS-Architektur wie «Single Point of Truth», «Once Only», Redundanzbeherrschung und eine «Big Data» Unterstützung

werden daher im Rahmen dieser Arbeit geprüft. Die in diesem Kontext in den Fokus gelangenden Themen wie das Spannungsfeld rund um das «Brewer's CAP Theorem»[1] [GL, S.1] werden in der Master-Thesis ebenfalls ausgeleuchtet, da diese einen massgebenden Einfluss auf die relevanten ISA-Patterns haben.

Die in diesem Abschnitt geschilderten Zusammenhänge verdeutlichen, dass ISA-Vorgaben langfristige und wegweisende Auswirkungen auf Unternehmen haben. Folglich müssen sie sowohl mit der Business- und IT-Strategie als auch mit der Businessarchitektur abgeglichen sein [EJH+08, S.89]. Zentrale Anforderungen sind daher, dass IS-Architekturen sowohl zukunftsfähig ausgerichtet als auch genügend flexibel sind, um kontinuierlich und zielgerichtet genutzt und weiterentwickelt werden zu können.

1.3 Zielsetzung

Ziel dieser Arbeit ist es, die sich oft in einem Spannungsfeld befindenden Anforderungen und Voraussetzungen für IS-Architekturen «von morgen» zu identifizieren, daraus sinnvolle architektonische Patterns (Muster und Schemata) und Lösungstechnologien abzuleiten, deren Potential zu erheben und in einem sinnvollen Gesamtkonzept respektive Artefakt zu verknüpfen. Auf der Basis von verschiedenen Fallszenarien soll dabei das erzeugte Artefakt iterativ überprüft und anschliessend zu einer generischen «Ideal» ISA-Referenzarchitektur als Design-Science-Artefakt verdichtet werden, welches sich nach der Best-Practice-Architektur von Engels et al. orientiert [EJH+08, S.82]. In dieser Forschung werden folglich die bestehenden und neuen architektonischen ISA-Patterns hinsichtlich Zukunftsfähigkeit und sinnvollen Einsatzszenarien analysiert und verglichen. Dabei werden Stärken, Schwächen, Chancen und Risiken (SWOT) der ISA-Patterns analysiert und aufgezeigt. Ein spezielles Augenmerk gilt zudem den analytischen Anforderungen (Business Intelligence Capabilities) und der Einhaltung der Prinzipien von Tallinn [EU17] appliziert auf den Unternehmenskontext. Das im Kontext des Resource Specific Models (RSM) einer Geschäftsfähigkeiten-basierten Unternehmensarchitektur (GFbUA) nach Spichiger & Noser [SN] ausgearbeitete Artefakt wird anschliessend anhand des Fallbeispiels von Arthur Reisen (AR) [SN] validiert und gegebenenfalls weiter optimiert. Kurzum wird ein ISA-Design-Science-Artefakt «RSM-ISA» im Sinne einer Referenzarchitektur erzeugt, die hilft, Geschäftsfähigkeiten anforderungsbasiert mittels zukunftsfähiger ISA-Patterns umzusetzen.

Abgrenzend wird definiert, dass sich diese Forschung nach dem RSM der Unternehmensarchitekturmodells nach Spichiger & Noser [SN, S.5] orientiert. Der Fokus liegt hierbei auf den Gesichtspunkten der logischen Informationssystem-Architekturmodellie-

[1] Das CAP Theorem besagt, dass es in einem verteilten System unmöglich ist, gleichzeitig die drei Eigenschaften Consistency (Konsistenz), Availability (Verfügbarkeit) und Partition Tolerance (Ausfalltoleranz) zu garantieren.

rung mit Berührungspunkten zur Aufbau- und Ablauforganisation. Rechtliche Aspekte sowie Daten- und Zugriffssicherheit werden abgegrenzt.

1.4 Forschungsfragen

Aufgrund der in den vorhergehenden Abschnitten 1.2 und 1.3 erläuterten Problemfeldern werden die nachfolgenden Forschungsfragen abgeleitet:

FF 1: Was müssen zukunftsfähige «Ideal» IS-Architekturen im Rahmen der an sie gestellten Anforderungen leisten?

FF 2: Wie lassen sich zukunftsfähige ISA-Patterns aufgrund ihrer Charakteristiken optimal zu einem ISA-Referenzarchitektur-Artefakt kombinieren?

FF 3: Wie lässt sich anhand der Geschäftsfähigkeiten aus der Fallstudie «Arthur Reisen» nach Spichiger & Noser [SN] das «Ideal» RSM-ISA-Artefakt für Arthur Reisen gestalten?

1.5 Forschungsdesign und methodisches Vorgehen

Die beiden zentralen Forschungsparadigmen der Wirtschaftsinformatik sind die gestaltungsorientierte und die erklärungsorientierte Forschung [BJPS06, S.6]. Die gestaltungsorientierte Forschung (Design Science Research) verfolgt dabei die Gestaltung nützlicher Artefakte in Bezug auf Informationssysteme als Lösung für praxisrelevante Probleme [HMR04, S.7], während die gestaltungsorientierte Forschung auf die Wissensbasis von sowohl gestaltungsorientierter als auch erklärungsorientierter Forschung zurückgreift, um neben der Problemrelevanz (Problem Relevance) auch die geforderte wissenschaftliche Stringenz (Research Rigor) sicherzustellen [HMR04, S.84] (vgl. Abbildung 1 auf Seite 6). Die Master-Thesis, welche den Anspruch hat, ein Artefakt zur Lösung der unter den Forschungsfragen geschilderten Problemstellung zu entwickeln, ist folglich der Prämisse der Forschungsfragen der Wirtschaftsinformatik, insbesondere der gestaltungsorientierten Forschung (Design Science), zuzuordnen. Hierbei wird eine deduktive Wissenserzeugung unter Anwendung von anerkannten Frameworks, Modellen und Methodiken angewendet, was die wissenschaftliche Stringenz gewährleistet.

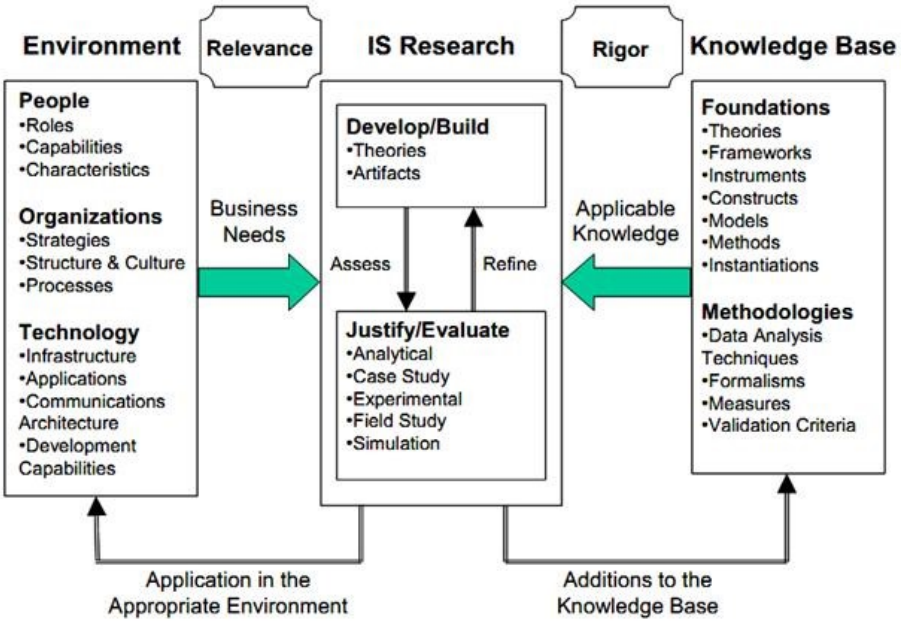


Abbildung 1: Information Systems Research Framework; Quelle: Hevner et al. [HMR04, S.80])

1.5.1 Forschungs-Guidelines

Das gewählte Vorgehen und die definierten Ziele stellen die Einhaltung von Hevners [HMR04, S.83] Design-Science Research Guidelines sicher durch:

Guideline 1: Design as an Artefact (Design als Artefakt)

Es wird ein ISA-Artefakt erzeugt, welches die beschriebene Problemstellung der zukunftsfähigen IS-Architektur und die damit zusammenhängenden Forschungsfragen (FF1-3) klärt.

Guideline 2: Problem Relevance (Relevanz der Problemstellung)

Ziel von Design Science Research ist es, eine technologiebasierte Lösung zu entwickeln, welche ein relevantes Problem im Business-Kontext löst. Die Relevanz ergibt sich einerseits aus der Fragestellung bezüglich IS-Architektur für AR aus GFbUA [SN] und neuen, im ISA-Kontext relevanten Anforderungen wie der Deklarationen von Tallinn [EU17] und einem «Dynamic Venturing» Ansatz von Ross et al. [RWR06].

Guideline 3: Design Evaluation (Design Bewertung)

Eine stringente Design Bewertung wird durch die zwei nachfolgenden Design-Evaluationsmethoden [HMR04, S.86] gewährleistet:

1. Analytische Design Bewertung
   Auf der Basis einer ISA-Pattern-Analyse und -Bewertung und einer

anschliessenden Kombination der Patterns wird eine den Anforderungen entsprechende Architektur sichergestellt.

2. Deskriptive Design Bewertung
   Das Artefakt wird empirisch anhand der integrierten Einzelfallstudie AR (vgl. Abbildung 2 nach Yin [Yin09]) und verschiedenen ausgearbeiteten Szenarios (Leistungsanforderungen AR), hinsichtlich deren Anwendbarkeit überprüft.

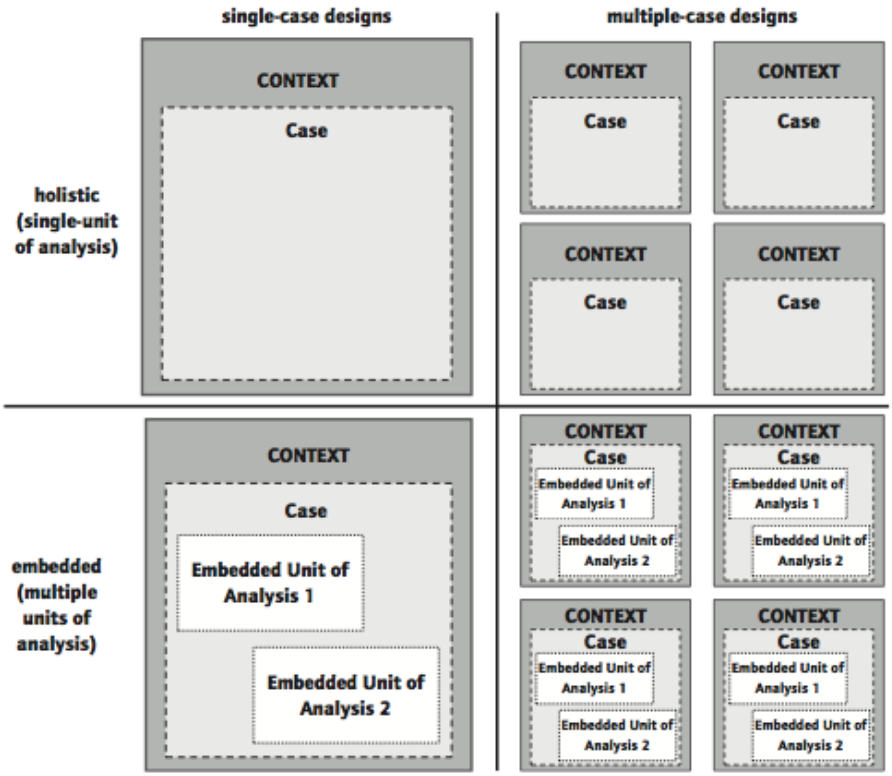


Abbildung 2: Grundtypen eines Fallstudienaufbaus; Quelle: Yin [Yin09]

Guideline 4: Research Contributions (Beitrag zur Forschung)

Der Beitrag zur Forschung und der Rückfluss der Erkenntnisse in die Forschungsgemeinde werden einerseits durch die bewerteten ISA-Patterns und andererseits auch durch das gebildete RSM-ISA-Referenzarchitektur-Artefakt sichergestellt. Weiter wird angestrebt, dass das erzeugte RSM-ISA-Artefakt für AR in das GFbUA [SN] integriert wird.

Guideline 5: Research Rigor (Wissenschaftlich stringente Forschung)

Für die Forschung relevante und nachvollziehbare Beiträge werden auf Basis von wissenschaftlich fundierten Architektur-Modellen wie Ziemanns AIOS [Zie10], dem Enterprise Architecture Maturity Model nach Ross et

al. [RWR06] und dem Designing Digital Organisations Model von Ross et al. [RSBJ17] erarbeitet. Ergänzend fliessen «Best-Practice» Ansätze wie von Quasar Enterprise nach Engels et al. [EJH+08] in das Artefakt ein. Im Anschluss werden diese mittels den unter «Guideline 3» vorgestellten Methoden bewertet, was die wissenschaftliche Stringenz gewährleistet.

Guideline 6: Design as a Search Process (Design als interativer Suchvorgang)

Simon [Sim96] beschreibt die Natur eines Design-Prozesses als einen Entwicklungs-/Test-Kreislauf (vgl. Abbildung 3). Dieser Ansatz wird im Rahmen des Testings iterativ und szenariobasiert durchgeführt, wobei Erkenntnisse in ein verbessertes Artefakt einfliessen.

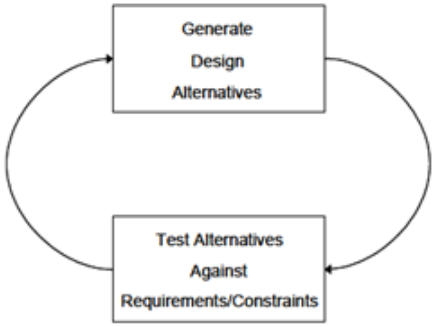


Abbildung 3: Der Entwicklungs-/Test-Kreislauf; Quelle: Simon [Sim96]

Guideline 7: Communication of Research (Forschung kommunizieren)

Ziel der Forschung ist nebst dem Klären der Forschungsfragen anhand eines Design Science-Artefakts auch die spätere Verankerung im GFbUA [SN], was durch das Veröffentlichen des Werkes eine Kommunikation der Forschungsergebnisse sicherstellt. Zudem wird die Arbeit in den Bibliotheken der Zürcher Hochschule für Angewandte Wissenschaften und der Berner Fachhochschule zur Verfügung gestellt.

1.5.2 Conceptual Model

Für die Master-Thesis wird das unter Abbildung 10 auf Seite 10 dargestellte Conceptual Model verfolgt. Es verdeutlicht das unter Abschnitt 1.5.1 erläuterte Vorgehen und die Zusammenhänge bei der Erarbeitung der Artefakte im Rahmen dieser Arbeit.

Ausgehend von einer umfassenden Literaturanalyse (Desk-Research) wird so nach der deduktiv erarbeiteten ISA-Pattern-Analyse und Bewertung das RSM-ISA-Referenzarchitektur-Artefakt erzeugt. Dieses wird analytisch und deskriptiv getestet. Als Produkt dieser Arbeit wird so einerseits das RSM-ISA-Artefakt für AR und das validierte RSM-ISA-Referenzarchitektur-Artefakt erzeugt.

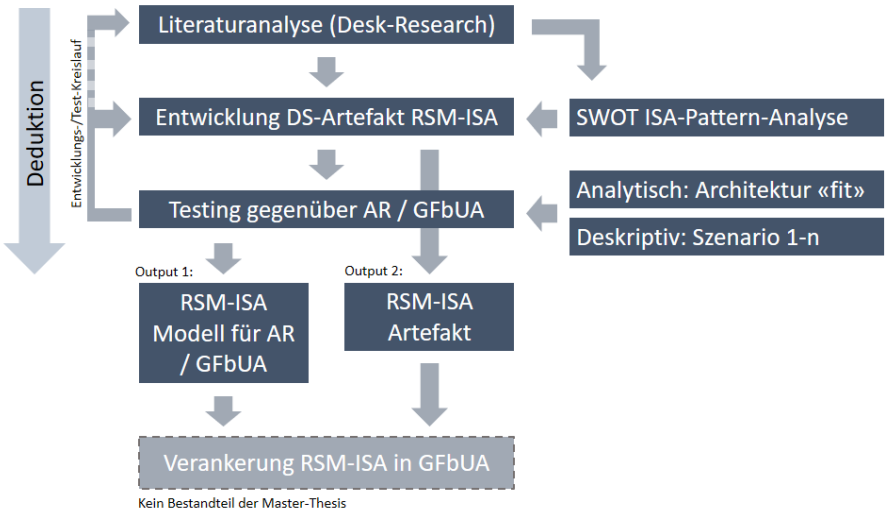


Abbildung 4: Conceptual Model zum Vorgehen

1.5.3 Argumentation zum gewählten Vorgehen

Der qualitative Design Science-Ansatz wird gewählt, da es sich um die erste Entwicklung des Artefakts zum Lösen der spezifischen Problemstellung aus der Praxis unter der Prämisse der Forschungsfragen handelt. In einer nachfolgenden Forschung erscheint es sinnvoll, das Artefakt qualitativ mittels einem Field Research-Ansatz, beispielsweise anhand von Experten-Interviews durch Validierung respektive Falsifizierung zu testen und anhand der neuen Erkenntnisse weiterzuentwickeln.

1.6 State of the Art

In diesem Abschnitt wird der aktuelle Stand der Forschung verdeutlicht, der Beitrag der wichtigsten Literaturquellen aufgezeigt und die im Rahmen dieser Arbeit identifizierten Forschungslücken erläutert.

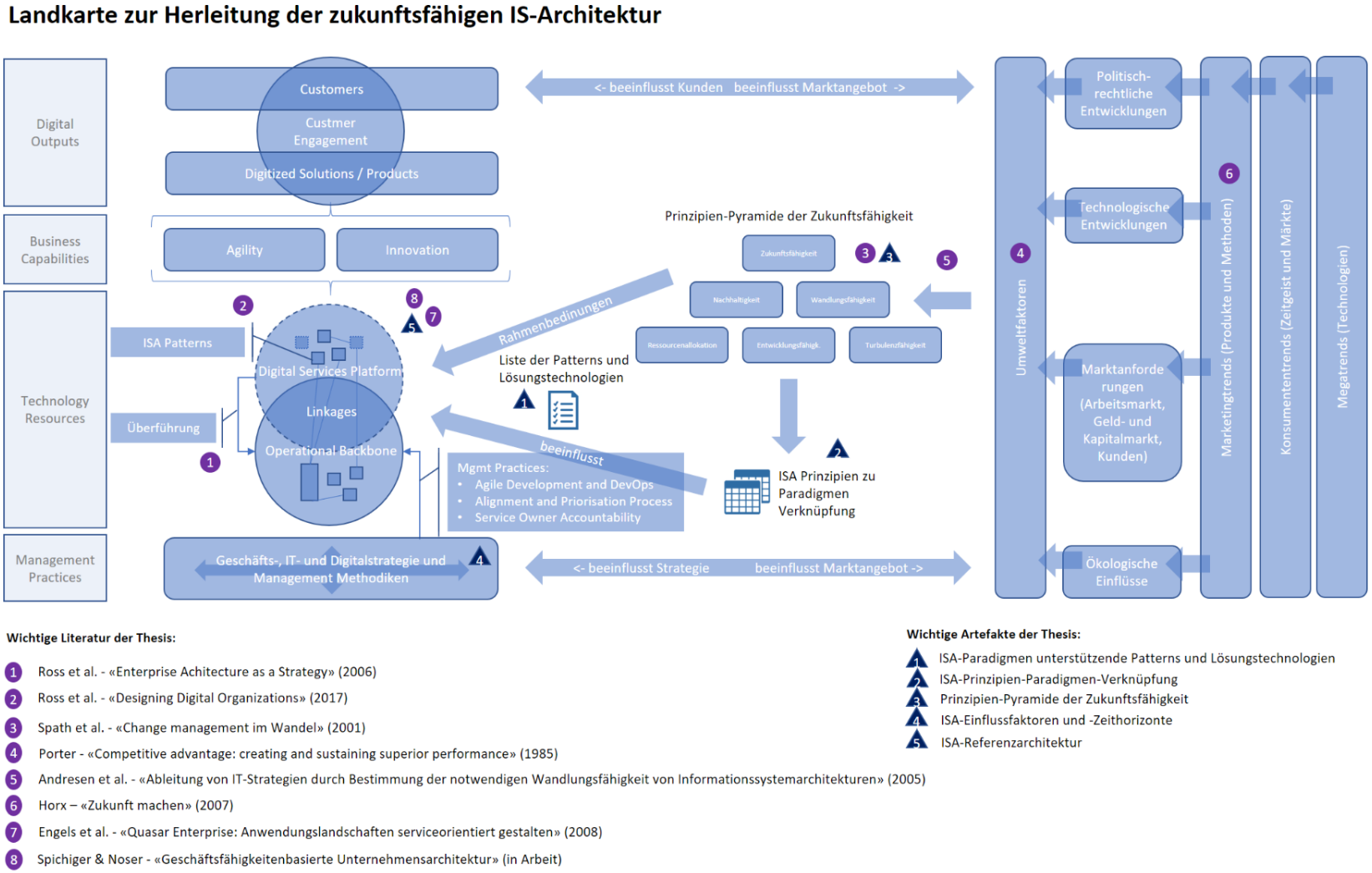
IS-Architekturen sind stark vom technologischen Wandel getrieben und müssen sich kontinuierlich weiterentwickeln. Aus Kostengründen werden sie in bestehenden Unternehmen aber oft nur opportunistisch und nicht holistisch implementiert. Es gibt daher aufgrund der unterschiedlichen unternehmensspezifischen Umständen nicht die *beste* zukunftsfähige IS-Architektur, wohl aber eine Ideal IS-Architektur, welche unter Berücksichtigung der Gegebenheiten in einer gesteuerten Evolution angestrebt werden sollte. Eine zukunftsfähige IS-Architektur sollte daher sowohl neuen Anforderungen als auch bestehenden Gegebenheiten Rechnung tragen. Vom technologischen Wandel und dem Unternehmens-Umfeld getriebene, neue ISA-Konzepte sind daher relativ vielfältig. Durch Fokussierung der Arbeit auf die Zukunftsfähigkeit kann das Forschungsgebiet jedoch sinnvoll eingegrenzt werden. Die Zukunftsfähigkeit lässt sich hierbei auf Basis der Prinzipien der Wandlungsfähigkeit und Nachhaltigkeit im Sinne von Beständigkeit sicher-

stellen. Hierbei belegt eine umfassende Literatur-Analyse von Ullrich & Weber [UW16], dass die Prinzipien der Wandlungsfähigkeit als Teilgebiet der Zukunftsfähigkeit verhältnismässig gut erforscht sind – beispielsweise durch Andresen et al. [AGS05]) – und ein weitgehender Konsens über deren wichtigste Treiber besteht. So wird als Hilfestellung zur Definition der Zukunftsfähigkeit und ihrer zugrundeliegenden Prinzipien das Bausteinmodell der Wandlungsfähigkeit von Spath et al. [SBBD01] beigezogen und im Rahmen der Arbeit um die Prinzipien (Bausteine) und weiteren Quellen ergänzt. Dies geschieht im Kontext der Nachhaltigkeit unter Berücksichtigung der ISA-Umwelteinflüssen nach Andresen et al. [AGS05], Porter [Por85], Hall [Hal88].

Die Nachhaltigkeit als zweite Perspektive der Zukunftsfähigkeit ist von den Prinzipien her gesehen bisher in der Literatur wenig untersucht worden , lässt sich aber anhand der Prinzipien-unterstützenden Nachhaltigkeits-Paradigmen wie sie etwa durch das reaktive Manifest von Bonér [BFKT14] beschrieben werden, ausarbeiten. Einen wichtigen Beitrag als Rahmen zur Umsetzung der beiden Prinzipien-Perspektiven der Zukunftsfähigkeit bietet das Modell Designing Digital Organizations von Ross et al. [RSBJ17], bei welchem sich die «Business Modularity» Ansätze von Ross et al. [RWR06] nahtlos integrieren lassen. Engels et al. [EJH+08] bietet zusätzlich eine Hilfestellung zur Gliederung der verschiedenen Ebenen und Elementen des zu gestaltenden ISA-Artefaktes. Spichiger & Noser [SN] definiert den ISA-Kontext aus Sicht der Geschäftsfähigkeiten-basierenden Unternehmensarchitektur und liefert mit Arthur Reisen eine szenarienbasierte, integrierte Einzelfallstudie zur Validierung des ISA-Artefaktes. Weitere ISA-Patterns, welche die Paradigmen und Prinzipien der zukunftsfähigen IS-Architektur unterstützen sind AIOS mit der BII Schnittstelle nach Ziemann [Zie10] und die Big Data Lambda-Architektur nach Marz [Mar11].

Eine konsistente Bewertung der Patterns hinsichtlich der Zukunftsfähigkeit und eine Zusammenführung zu einem Artefakt fehlt aber und muss entsprechend im Rahmen dieser Arbeit ausgearbeitet werden.

Da diese Arbeit und die dabei entstehenden Artefakte auf einer breit gefassten Kombination von verschiedenen Architekturpatterns aus der Wissenschaft und aus der Praxis beruht, werden diese Literaturquellen und Grundlagenwerke zusätzlich in einem konzeptionellen Interdependenzenmodell in Form einer Landkarte (vgl. Abbildung 5 auf Seite 11) für das bessere Verständnis der Zusammenhänge und für eine bessere Nachvollziehbarkeit visualisiert. Dieses Modell kann so bei Bedarf in den nachfolgenden Kapiteln als Orientierungshilfe beigezogen werden, um die spezifischen Patterns, Literaturquellen und auszuarbeitenden Teil-Artefakte in einem übergeordneten, zusammenhängenden Kontext einordnen zu können.



Wichtige Literatur der Thesis:

1. Ross et al. - «Enterprise Achitecture as a Strategy» (2006)
2. Ross et al. - «Designing Digital Organizations» (2017)
3. Spath et al. - «Change management im Wandel» (2001)
4. Porter - «Competitive advantage: creating and sustaining superior performance» (1985)
5. Andresen et al. - «Ableitung von IT-Strategien durch Bestimmung der notwendigen Wandlungsfähigkeit von Informationssystemarchitekturen» (2005)
6. Horx – «Zukunft machen» (2007)
7. Engels et al. - «Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten» (2008)
8. Spichiger & Noser - «Geschäftsfähigkeitenbasierte Unternehmensarchitektur» (in Arbeit)

Wichtige Artefakte der Thesis:

1. ISA-Paradigmen unterstützende Patterns und Lösungstechnologien
2. ISA-Prinzipien-Paradigmen-Verknüpfung
3. Prinzipien-Pyramide der Zukunftsfähigkeit
4. ISA-Einflussfaktoren und -Zeithorizonte
5. ISA-Referenzarchitektur

Abbildung 5: Konzeptionelles Interdependenzenmodell zur Herleitung der zukunftsfähigen ISA-Referenzarchitektur

2 Geschäftsfähigkeiten-basierte Unternehmensarchitektur (GFbUA)

In diesem Kapitel wird zuerst das Framework der Geschäftsfähigkeiten-basierten Unternehmensarchitektur (GFbUA) nach Spichiger & Noser [SN] erläutert, daraus wird dann unter Abschnitt 2.1 das Resource Independent Model (RIM) hergeleitet und zuletzt unter Abschnitt 2.2 das Resource Specific Model (RSM) mit der eingegliederten IS-Architektur hergeleitet.

Die Geschäftsfähigkeiten-basierte Unternehmensarchitektur hat den Anspruch Organisationen prinzipiengeleitet zu beschreiben, damit die Veränderbarkeit des Geschäfts berücksichtigt werden können. Die Beschreibung des Geschäfts orientiert sich dabei an Geschäftsmodellen. Die verschiedenen Elemente eines Geschäftsmodells werden im GFbUA mit Hilfe der Geschäftsfähigkeiten beschrieben. Geschäftsfähigkeiten können hierbei Unternehmen, ein Unternehmens-Teilbereich oder auch eine organisationsübergreifende Struktur wie eine Wertschöpfungskette sein. Die Beschreibung der Geschäftsfähigkeiten selbst erfolgt anhand von Geschäftsobjekten und -prozessen und mittels Ressourcen wie Organisation, Betriebsmittel, Sachmittel und Informatik (vgl. Abbildung 6). Die Geschäftsfähigkeiten mit den funktionalen Beschreibungen der Geschäftsobjekte und -prozesse sind ein ressourcenunabhängiges Modell (Resource Independent Model, RIM). Das ressourcenspezifische Modell (Resource Specific Model, RSM) stellen dar, wie Ressourcen eingesetzt werden, um die Geschäftsfähigkeiten möglichst optimal zu unterstützen.

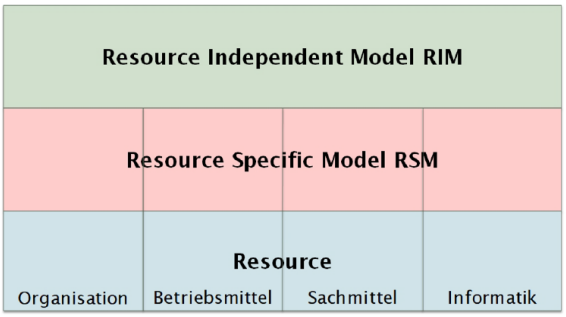


Abbildung 6: Ordnungsraster für Teilmodelle der Unternehmensarchitektur; Quelle: Spichiger & Noser [SN, S.5]

2.1 Ressource Independent Model (RIM)

Spichiger & Noser [SN] orientiert sich zur Beschreibung des Geschäfts an Geschäftsmodellen. Die verschiedenen Elemente eines Geschäftsmodells werden hierbei mit Hilfe der Geschäftsfähigkeiten beschrieben. Geschäftsfähigkeiten weisen nach Engels et al. [EJH+08] folgende Charakteristiken auf:

- Stabilität: Die Geschäftsfähigkeiten bilden eine persistente, kohäsive und logische Gruppierung von Funktionen.
- Abstraktion: Die Geschäftsfähigkeiten abstrahieren von expliziten Ressourcen, Geschäftsprozessen oder Technologien.
- Horizontale Struktur: Die Geschäftsfähigkeiten bilden eine horizontale, komplette und nicht überlappende Unterteilung der gesamten Unternehmung. Nicht überlappend bedeutet insbesondere, dass die Geschäftsfähigkeiten redundanzfrei sind.
- Vertikale Hierarchie: Eine Geschäftsfähigkeit kann in weitere granulare Geschäftsfähigkeiten zerlegt werden.

Geschäftsfähigkeiten können demnach als eine hierarchische Baumstruktur abgebildet werden (vgl. Abbildung 7), da sie selber in weitere, granularere Geschäftsfähigkeiten zerlegt werden können (Vertikale Hierarchie). Eine Geschäftsfähigkeit kann zudem Geschäftsobjekte und Geschäftsprozesse enthalten, welche wiederum durch Ressourcen unterstützt werden (Abstraktion und Stabilität). Spichiger & Noser [SN, S.3] definiert weiter, dass Geschäftsprozesse lesend und verändernd auf Geschäftsobjekte zugreifen können und es aber den Geschäftsprozessen nicht erlaubt ist, auf andere Geschäftsprozesse zuzugreifen.

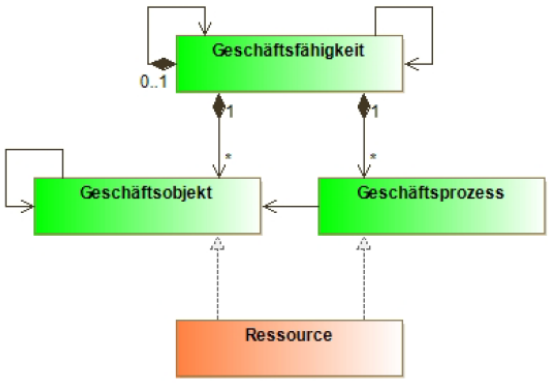


Abbildung 7: Metamodell der Geschäftsfähigkeit; Quelle: Spichiger & Noser [SN, S.4]

2.2 Ressource Specific Model (RSM)

Damit Geschäftsfähigkeiten auch tatsächlich beherrscht werden, ist es nach Spichiger & Noser [SN, S.47] von zentraler Bedeutung, dass diese auch durch die entsprechenden Ressourcen wie Personen, Betriebsmittel, Sachmittel und IKT-Mittel (vgl. Abbildung 8 auf Seite 14) unterstützt werden und diese Resourcen auch gut zusammenarbeiten.

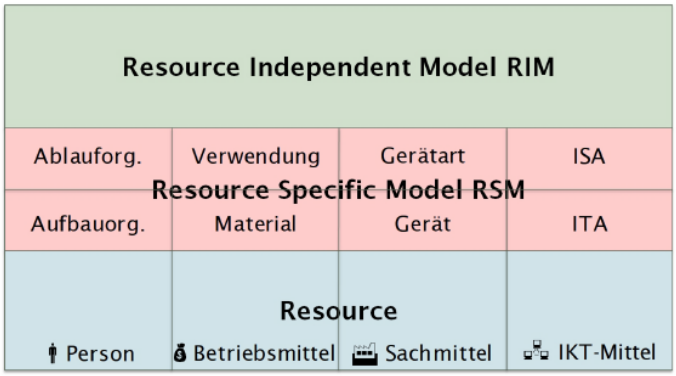


Abbildung 8: Eingliederung von RIM, RSM und Ressourcen nach GFbUA; Quelle: Spichiger & Noser [SN, S.5]

Ein wichtiger Bestandteil dieser Arbeit bezieht sich darauf, wie die IS-Architektur das Resource Independent Model (RIM) und die Ressourcen optimal unterstützen, damit die von GFbUA geforderten Geschäftsfähigkeiten erzielt werden können.

Engels et al. [EJH+08, S.79] bietet hierbei wie unter Abbildung 9 auf Seite 15 ersichtlich eine Hilfestellung, wie sich die Anforderungen an eine IS-Architektur von den Geschäftsstrategie mit den geforderten Geschäftsfähigkeiten top-down ableiten lassen.

Die Geschäftsstrategie definiert dabei den kontextuellen Rahmen, wie die Geschäftsarchitektur aufgestellt sein soll. So wird nach Engels et al. [EJH+08, S.78] die Geschäftsarchitektur mit Geschäftsprozessen, Geschäftsservices, Geschäftsobjekten und der Organisation eines Unternehmens anhand der Geschäftsstrategie und der Wertschöpfungskette gestaltet. Die IS-Architektur[2] wird von der Geschäftsarchitektur und der – mit der Geschäftsstrategie abgeglichenen – IT-Strategie abgeleitet. Die IS-Architektur strukturiert hierbei die Anwendungslandschaft aus fachlicher Sicht, während die technische Infrastruktur-Architektur[3] insbesondere die verwendeten technischen Plattformen und Systemsoftwarekomponenten beschreibt.

[2] Nach Engels et al. unter Abbildung 9 auf Seite 15 als «IS» bezeichnet

[3] Nach Engels et al. unter Abbildung 9 auf Seite 15 als «TI» bezeichnet; nach Spichiger & Noser unter Abbildung 8 «ITA» bezeichnet

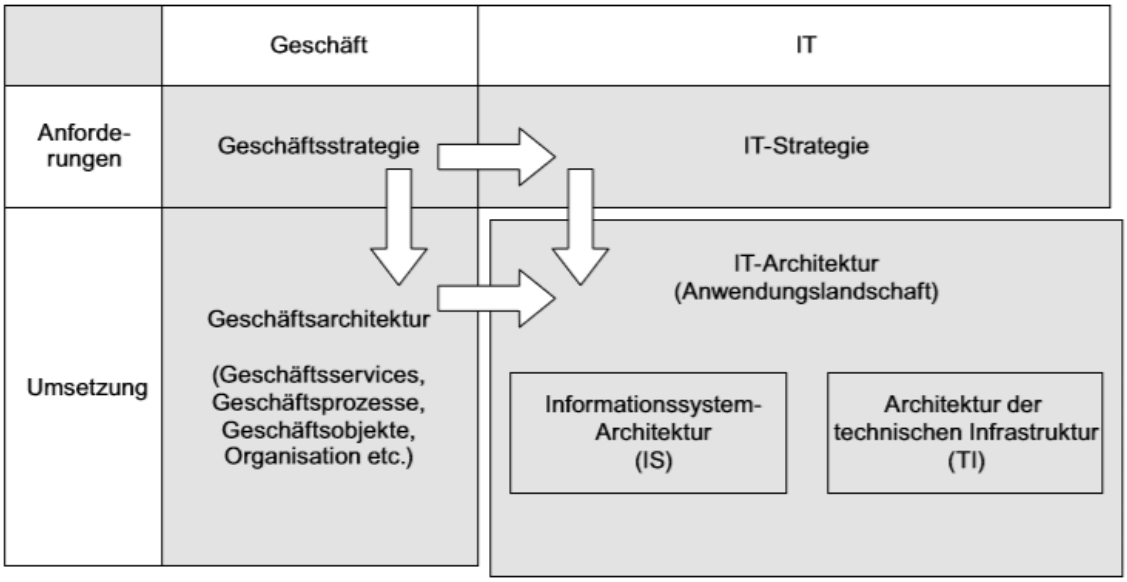


Abbildung 9: Von der Geschäftsstrategie zur IS-Architektur; Quelle: Engels et al. [EJH+08, S.79]

3 Informationssystem-Architektur (ISA)

In diesem Kapitel wird unter Abschnitt 3.1 der Begriff der Informationssystem-Architektur (ISA) definiert und anschliessend unter 3.2 der Idealzustand beschrieben. Im nachfolgenden Abschnitt 3.3 wird die Komplexitätsbeherrschung und die Verknüpfung zum RSM nach GFbUA von Spichiger & Noser [SN] aufgezeigt. Hiernach werden unter Abschnitt 3.4 IS-Architektur relevante Prinzipien respektive unter Abschnitt 3.5 unterstützende Paradigmen der Zukunftsfähigkeit hergeleitet. Im Anschluss werden zu den identifizierten ISA-Paradigmen unter 3.6 relevante ISA-Patterns erläutert und bewertet. Zum Abschluss werden die gewonnenen Erkenntnisse unter 3.7 in einem RSM-ISA-Referenzarchitektur-Artefakt hergeleitet und zusammengefügt.

Der gewählte Ansatz dieser Arbeit sieht ein stufenweises Vorgehen vor (vgl. Abbildung 10), wobei erst unter der Prämisse der Zukunftsfähigkeit die relevanten ISA-Einflussfaktoren nach Forschungsfrage FF1 identifiziert werden. Daraus werden nachfolgend die ISA-Prinzipien abgeleitet und unterstützende ISA-Paradigmen mit möglichen Lösungstechnologien untersucht. Diese Paradigmen und ihre Lösungstechnologien müssen aus verschiedenen Perspektiven und Architekturebenen in einem sinnvollen Verbund harmonieren, was in dieser Arbeit in der Form von ISA-Patterns im nachfolgenden Kapitel 3.6 aufgegriffen wird. Auf dieser Basis wird gemäss Forschungsfrage FF2 das RSM-ISA-Referenzarchitektur-Artefakt unter Abschnitt 3.7 gebildet und nach Forschungsfrage FF3 anhand von verschiedenen Fallstudien-Szenarien im darauf folgenden Kapitel 4 validiert.

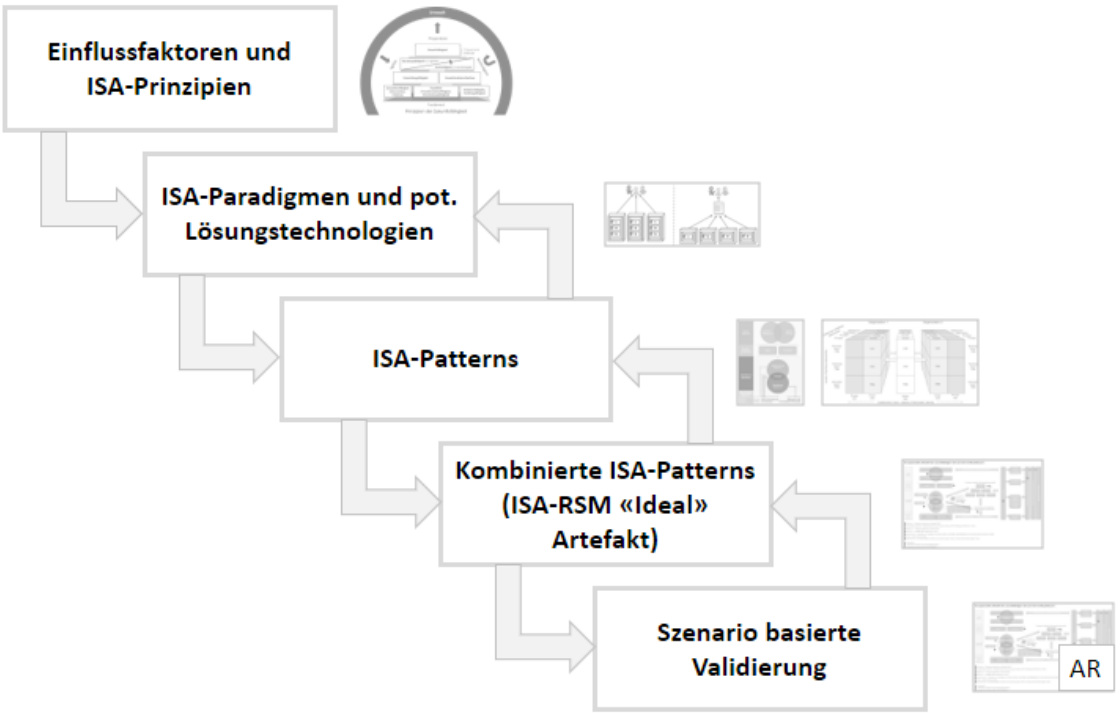


Abbildung 10: Deduktives Ableiten und iteratives Validieren

3.1 ISA-Definition

Wie Krcmar [Krc90, S.396] feststellt, sind die beiden Komponenten des Wortes *Informationssystem-Architektur* erklärungsbedürftig. Der Teilbegriff *Architektur* wird im Informationstechnologie Bereich nach IEEE als die grundsätzliche Organisation eines Systems, seiner Komponenten, deren Interdependenzen zueinander und zur System-Umwelt, sowie deren Design- und Weiterentwicklungsprinzipien definiert [Soc00]. Nach Bass et al. [BCK03, S.28ff] kann Architektur weiter als eine abstrakte, ganzheitliche Betrachtung von Strukturen und Mustern mit Planungscharakter aufgefasst werden. Architekturen sind nach Andresen et al. [AGS05, S.66] in der Regel das Ergebnis eines Planungsprozesses und stellen nach ihrer Definition selbst einen Masterplan bzw. Bauplan für ganzheitliche Realisierungen zukünftiger Massnahmen dar.

Der Teilbegriff *Informationssystem* (IS) umfasst als solches nach Davis [Dav00, S.61] Systeme zur Bereitstellung von Informations- und Kommunikationsdiensten in einer Organisation als auch die Tätigkeiten und das Management der Informationssystem-Funktion im Rahmen der Planung, Gestaltung, Entwicklung, Implementierung und dem Betrieb der Systeme sowie bei der Erbringung von Dienstleistungen. Laudon & Laudon [LL12, S.15] beschreiben hierbei zudem, dass Informationssysteme Daten, Informationen und Wissen erfassen, speichern, verarbeiten und kommunizieren.

Der kombinierte Begriff *Informationssystem-Architektur* lässt sich demnach nach Krcmar [Krc90, S.398] als IS-Generalbebauungsplan bezeichnen, wobei nach Scheer [Sch91, S.12] eine IS-Architektur sowohl deskriptiv (statisch) eine grundlegende Anordnung und Verknüpfung der IS-Elemente aus verschiedenen Sichten definiert, als nach Sinz [Sin02, S.1055] auch konstruktiv (dynamisch) deren Weiterentwicklungs- und Nutzungsprinzipien im Einklang mit der Organisation vorgibt, um nach Laudon & Laudon [LL12, S.15] sowohl kurzfristig als auch nachhaltig bestimmte Unternehmensziele zu erreichen. Nach Zachmann [Zac87] hilft es hierbei eine Architektur aus verschiedene Sichten zu betrachten um komplexe Systeme komplementär zu beschreiben.

3.2 ISA-Zielsetzung - Der Idealzustand oder «der Weg ist das Ziel»

Ein wesentliches Ziel dieser Arbeit ist die Beantwortung der Fragestellung was eine zukunftsfähige «Ideal» IS-Architektur leisten können muss. Nebst der in dieser Arbeit zu klärenden Anforderung der Zukunftsfähigkeit wird ein solides Fundament benötigt, worauf eine zukunftsfähige IS-Architektur und die durch sie realisierte Anwendungslandschaft (AL) aufbauen kann.

Um mit Anforderungen – wie sie beispielsweise aus neu angestrebten Geschäftsfähigkeiten entstehen können – umgehen zu können, muss jede IS-Architektur – respektive ihre Anwendungslandschaft – auf kurze oder lange Sicht beherrschbar sein. Damit eine Anwendungslandschaft beherrscht werden kann, muss klar sein, wie das Ideal-Zielbild

aussieht («Wo geht die Reise hin?») und welche internen und externen Anforderungen an sie gestellt werden können. Dies kann beispielsweise anhand von *Predictive-* («Was wird geschehen?») oder *Prescriptive-* («Wie können wir es geschehen lassen?») Analyse-Methoden geprüft und vorbereitet werden. Das von der IS-Architektur angestrebte Ideal-Zielbild dient als eine Art Leuchtturm für die Vorhaben der nächsten Jahre. Es legt nach Engels et al. [EJH+08, S.8] die «Leitplanken für die langfristige Weiterentwicklung der Anwendungslandschaft fest», wobei diese Leitplanken zur Überprüfung dienen, ob die Evolution noch auf dem richtigen Weg ist.

Ein Unternehmen ist immer als Teil eines grösseren Ökosystems zu betrachten, welches beispielsweise in Form von Trendeinflüssen (z.B. Konsumenten-, Technologie- und Megatrends), dem Markt, dem Gesetzgeber und nicht zuletzt den Shareholdern auf das Unternehmen und das ISA-Ideal-Zielbild einwirkt (vgl. Horx [HHSW07], Naisbitt [Nai83]). Andererseits ist ein Unternehmen selbst auch ein lernfähiges System zu betrachten (vgl. «System Dynamics» nach Bärtschi [Bä17]), was die Ideal-Position des «Ideal-Leuchtturms» laufend beeinflusst. Der «Ideal-Leuchtturm» ist folglich nicht «fest zementiert» auf einer «Insel» zu betrachten, sondern stellt eher einen im Meer fahrenden «Supertanker» dar, welcher von den externen Strömungen des Ökosystems und dem internen Unternehmensantrieb vorangetrieben wird. Aufgrund dieser Veränderlichkeit des Idealzustandes ist es wichtig, dass die angestrebten Veränderungen der Anwendungslandschaft anhand erreichbarer Soll-Zwischenschritte in Form einer gesteuerten Evolution (Managed Evolution) in Richtung des Idealzustandes getrieben werden (vgl. Jackson [Jac82]). Hierbei gibt es normalerweise ein Spannungsfeld zwischen den Anforderungen eines Ideal-Zustandes und den zu Verfügung stehenden Ressourcen (vgl. Abbildung 11 auf Seite 19).

Zusammenfassend gilt es folglich die bestehende IS-Architektur kontinuierlich nach einer sich möglicherweise verändernden idealen IS-Architektur weiter zu entwickeln und die Anwendungslandschaft so Schritt für Schritt – unter Berücksichtigung der zur Verfügung stehenden Ressourcen – nach diesem Zielbild auszurichten.

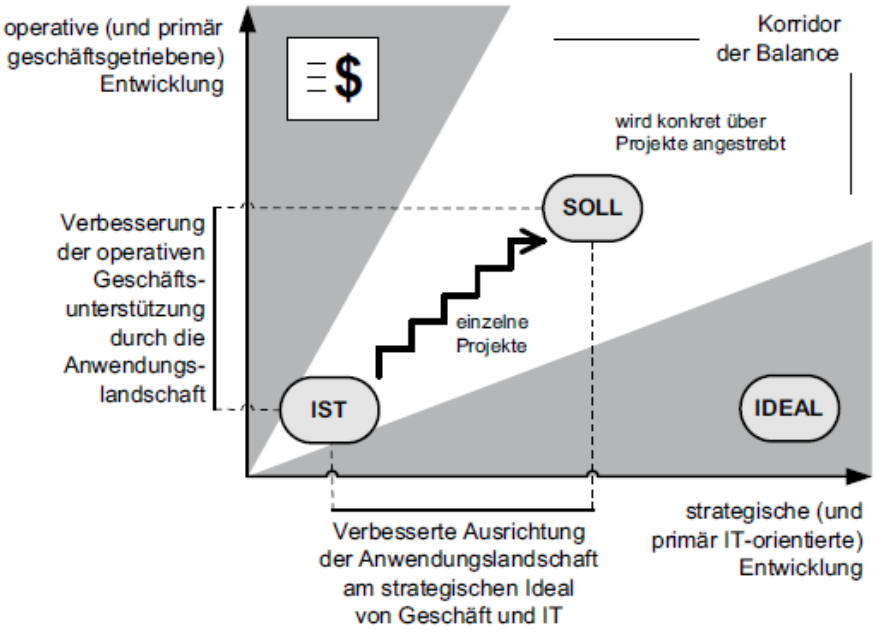


Abbildung 11: Ist, Soll und Ideal im Rahmen einer gesteuerten Evolution; Quelle: Engels et al. [EJH+08, S.8]

3.3 ISA-Strukturierung

In diesem Abschnitt werden konkrete Ansätze zur Komplexitätsbeherrschung aufgezeigt und anschliessend Anknüpfungspunkte zwischen dem Resouce Specific Model (RSM) nach Spichiger & Noser [SN] und den Methoden zur Komplexitätsbeherrschung nach Engels et al. [EJH+08] aufgezeigt.

3.3.1 ISA-Komplexitätsbeherrschung

IS-Architekturen und ihre Anwendungslandschaften (AL) sind komplex. Scheer empfiehlt daher eine Abstrahierung und Unterteilung in Sichten [Sch91, S.12]. In dieser Arbeit wird hierfür der Ansatz nach Engels et al. [EJH+08, S.144] angewendet, wobei eine Aufteilung über Abstraktionsebenen und verschiedene Artefakte zur besseren Komplexitätsbeherrschung der IS-Architektur vorgenommen wird (vgl. Abbildung 12 auf Seite 20).

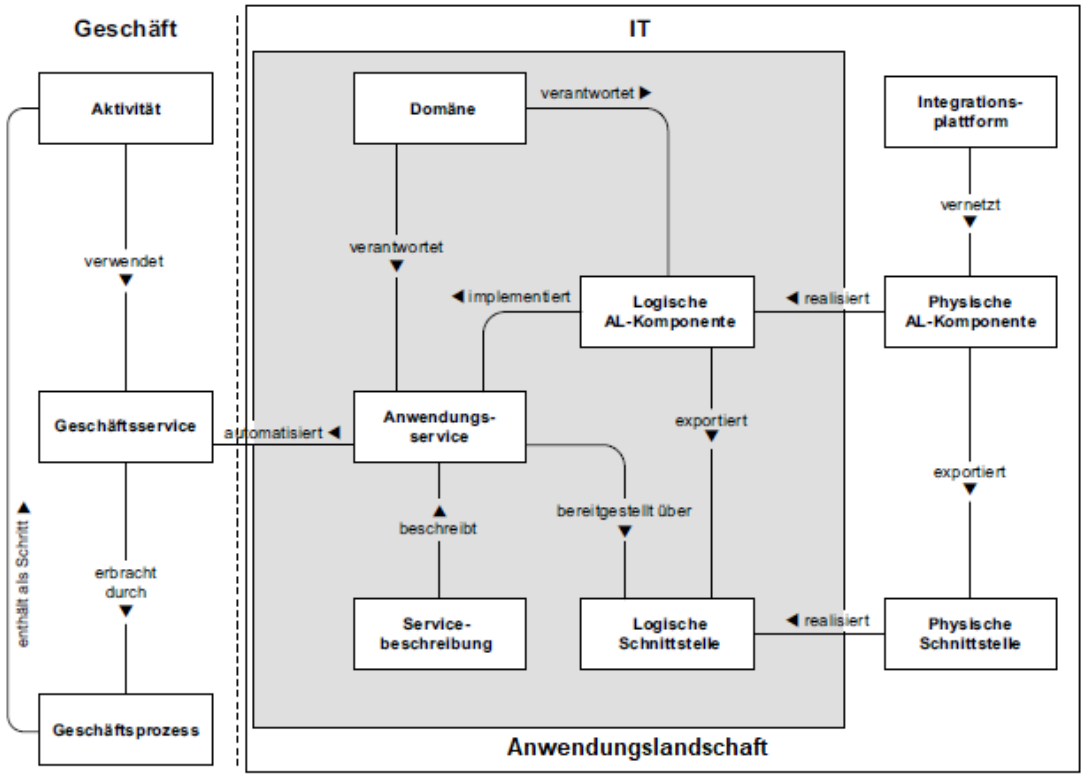


Abbildung 12: Begriffe und Zusammenhänge von Anwendungslandschaften in der Übersicht; in Anlehnung an Engels et al. [EJH+08, S.102]

Nach der IS-Architektur ausgerichtete Anwendungslandschaften definieren hierbei den Rahmen, unter welchem die Anwendungsservices die Geschäftsservices bedienen. Unter der Anwendungslandschaft werden (hierarchisch) logische Domänen als strukturierendes Konstrukt verwendet, welche die einzelnen Anwendungsservices verantworten. Die nach fachlichen Gesichtspunkten unterteilten Domänen verantworten die logischen Komponenten einer Anwendungslandschaft, was eine klare Zuteilung eines Verantwortungsbereichs ermöglicht und so der verbesserten Kommunikation zwischen Fachbereich und IT dienen kann. Die logischen Komponenten implementieren die Anwendungsservices und definieren die logischen Import- und Export-Schnittstellen.

Wesentliche IS-Architekturebenen sind daher (vgl. Abbildung 13 auf Seite 21):

- Ebene 0 - Anwendungslandschaft als Rahmen
- Ebene 1..n - Konzeptionell - Domänen ihre verantworteten Anwendungsservices
- Ebene n+1 - Logisch - Logische AL-Komponenten mit ihren Schnittstellen und Operationen
- Ebene n+1 - Physisch - Physische AL-Komponenten mit ihren Schnittstellen und Operationen

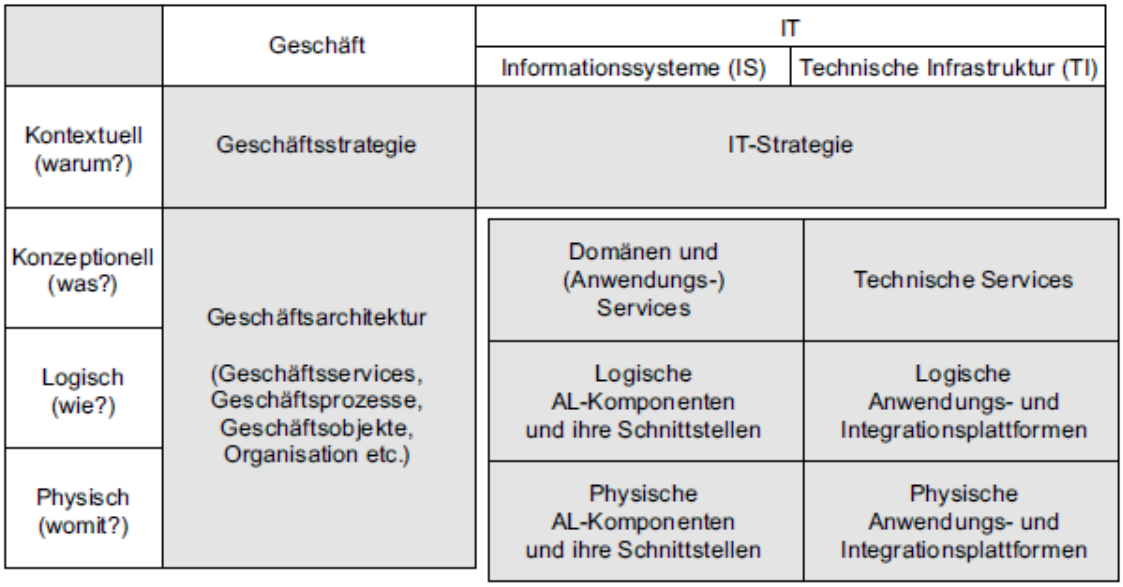


Abbildung 13: ISA-Architekturebenen; Quelle; Engels et al. [EJH+08, S.2]

Vereinfacht betrachtet können die beschriebenen ISA-Elemente wie unter Abbildung 14 ersichtlich dargestellt werden.

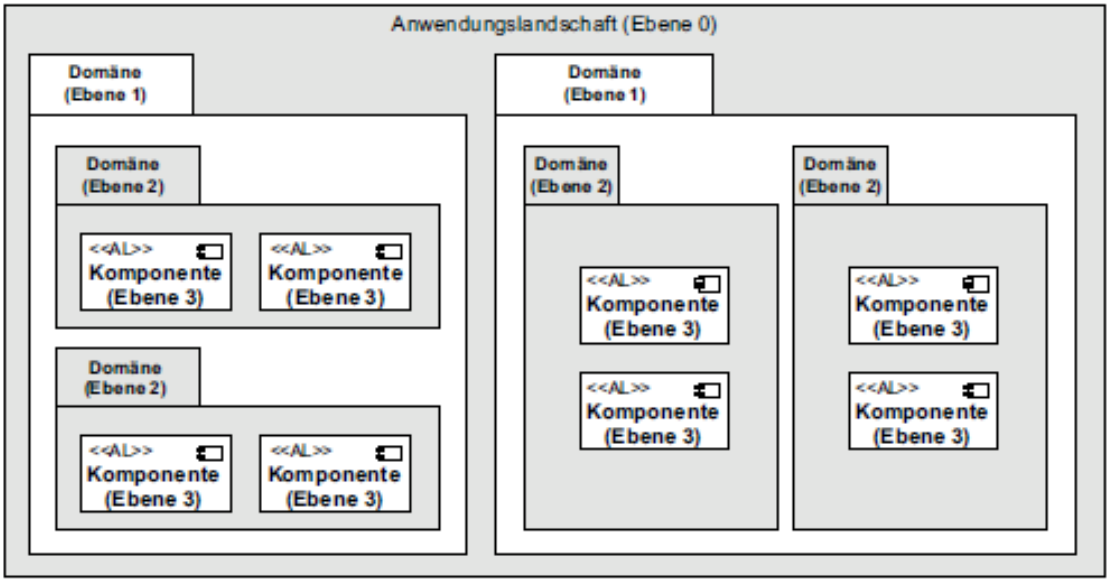


Abbildung 14: Logische ISA-Gliederung; Quelle; Engels et al. [EJH+08, S.2]

3.3.2 ISA-RSM-Verknüpfung

Spichiger & Noser [SN, S.52] verfolgt im ISA-Kontext eine Verknüpfung des Resource Independent Model (RIM) mit dem Resource Specific Model (RSM) nach Engels et al. [EJH+08, S.161ff], wobei Geschäftsfähigkeiten mit Hilfe von logischen Domänen der Anwendungslandschaft (AL) abgebildet werden.

Als weiteres Konzept kann die Kategorisierung von Komponenten angewendet werden. Hiermit kann nach Engels et al. [EJH+08, S.161] «eine für die Gestaltung von Anwendungslandschaften wesentliche Spezialisierung des allgemeinen Prinzips der Trennung der Belange» erzielt werden. Eine solche Kategorisierung der AL-Komponenten hilft zur Reduktion der Komplexität, da AL-Komponenten einer unterschiedlichen Kategorie auch in anderen Bereichen der Anwendungslandschaft angesiedelt werden, die nach verschiedenen Konstruktionsprinzipien aufgebaut sein können und sich in der Regel auch nach unterschiedlichen Lebenszyklen orientieren.

Während sich die RIM-Geschäfts*objekte* so im RSM-Kontext anhand von Operationen der AL-Komponente «Bestand», «Funktion» und «Interaktion» kategorisieren lassen, werden die RIM-Geschäfts*prozesse* als RSM-Operationen der AL-Komponente entsprechend als «Interaktion», «Prozess» und «Funktion» kategorisiert. Diese Operationen sind jeweils von einer AL-Domäne respektive einer RIM-Geschäftsfähigkeit abhängig (vgl. Abbildung 15).

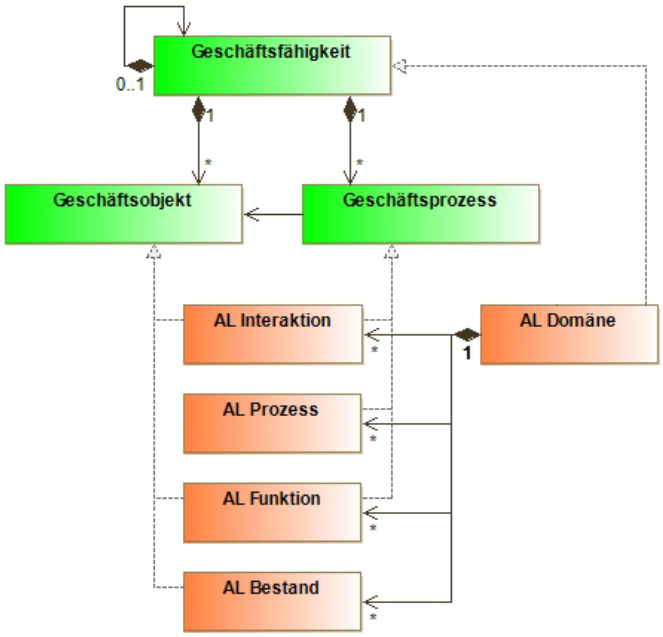


Abbildung 15: ISA-RSM-Verknüpfung, in Anlehnung an Spichiger & Noser [SN, S.52]

Die zuvor genannten Kategorien von Operationen, welche in ihrer Summe die AL-Komponente bilden, die dann wiederum die AL-Services implementieren, lassen sich nach Engels et al. [EJH+08, S.162] wie folgt beschreiben:

- Bestandskomponenten (Verwaltung von Datenbeständen und Zugriff auf diese): Bestandskomponenten haben die Hoheit über jeweils einen Ausschnitt der Geschäftsobjekte eines Unternehmens, d.h., Datenzugriffe erfolgen ausschliesslich über ihre Schnittstellen. Ihre Schnittstellen umfassen CRUD-Operationen (Create, Read,

Update, Delete) sowie darauf aufbauend komplexe Pflegeoperationen und lesende Sichten auf die gespeicherten Geschäftsobjekte. Bestandskomponenten kennen fachliche Konsistenzbedingungen und überwachen diese bei der Datenpflege. Sie implementieren elementare, auf die Daten bezogene fachliche Logik wie beispielsweise die Buchung eines Umsatzes oder die Historienführung für Daten. Darüber hinausgehendes fachliches Wissen besitzen sie nicht.
Ein Beispiel für eine Bestandeskomponente ist das Kundenmanagement, es enthält eine Bestandesoperation *«findeKunde(int kundenNr)»*

- Funktionskomponenten (IT-unterstützte Geschäftsservices mit algorithmischem Charakter):
  Funktionskomponenten implementieren fachliche Verfahren – oft mittels komplexer Algorithmen. Beispiele sind die Einplanung von Aufträgen, die Bonitätsprüfung, die Klassifikation von Kunden sowie die Erstellung von Abrechnungen. Viele dieser Operationen sind für sich allein sinnvoll, andere lassen sich nur im Kontext und unter der Kontrolle eines Prozesses ausführen. Funktionskomponenten nutzen ausschliesslich Schnittstellen von Bestandskomponenten und ggf. die anderer Funktionskomponenten.
  Ein Beispiel für eine Funktionskomponente ist die Pauschalpreisberechnung; es enthält die Funktionsoperation *«berechnePauschalpreis(int auftragsNr)»*

- Prozesskomponenten (IT-unterstützte Geschäftsprozesse):
  Jede Prozesskomponente unterstützt einen oder mehrere durch IT zu unterstützende Geschäftsprozesse. Ihre Aufgabe ist die Steuerung von Abläufen – meist über verschiedene Funktions- und Bestandskomponenten hinweg. Darunter fallen sowohl vollständig automatisierte Abläufe als auch Abläufe mit Benutzerinteraktion. Prozesse sind meist lang laufend und können zwischen einzelnen Verarbeitungsschritten pausieren. Die Operationen von Funktionskomponenten können im Gegensatz dazu nicht unterbrochen werden und müssen vollständig durchlaufen werden. Prozesskomponenten nutzen Operationen anderer Prozesskomponenten sowie die Operationen von Funktions- und Bestandskomponenten.

- Interaktionskomponenten (Interaktion mit einer Anwendungslandschaft durch Anwender oder andere Anwendungslandschaften):
  Zahlreiche Anwender verschiedener Gruppen nutzen die Services einer Anwendungslandschaft. Dabei werden häufig gleiche Informationen und Funktionen über verschiedene Kanäle angeboten. Beispiele sind Intranet-Portale für Innendienstmitarbeiter, Anwendungen auf mobilen Endgeräten für Aussendienstmitarbeiter sowie Internet-Portale für Kunden. Interaktionskomponenten ermöglichen Anwendern den Zugang zu den Services einer Anwendungslandschaft. Nach aussen bieten

sie eine einheitliche, kanalspezifische Sicht auf unterschiedliche Komponenten und Anwendungen, die in einem gemeinsamen Rahmen integriert werden. Idealerweise ist die Grenze zwischen einzelnen Anwendungen in einem Kanal für den Anwender nicht mehr sichtbar.

Die IS-Architektur relevanten Elemente lassen sich im GFbUA-Kontext von RSM und RIM wie in Abbildung 16 dargestellt kombinieren.

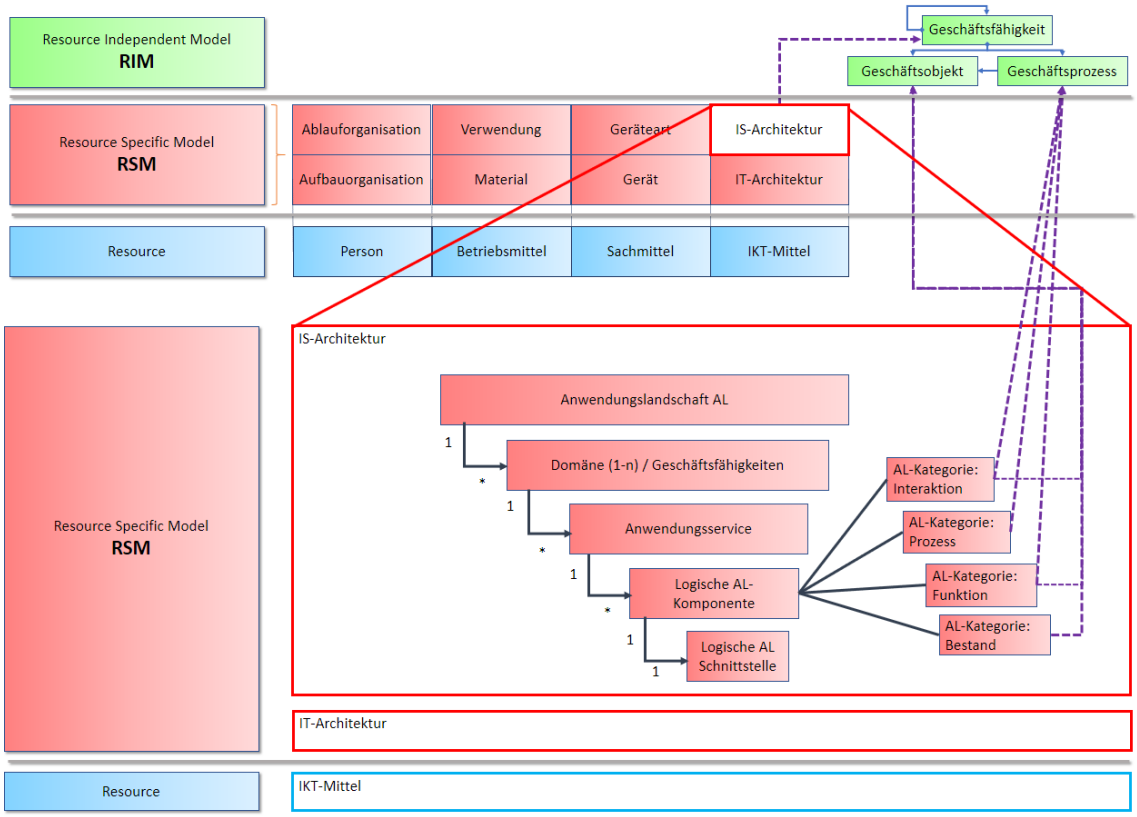


Abbildung 16: IS-Architektur im GFbUA-Kontext von RIM und RSM

3.4 ISA-Einflussfaktoren und -Prinzipien

In den vorgehenden Abschnitten wurde das Verständnis von IS-Architekturen erläutert und mit dem RSM nach Spichiger & Noser [SN] verknüpft. Um dieses Modell nun hinsichtlich der Fragestellung FF1 der ISA-Zukunftsfähigkeit zu ergründen, werden nun die in diesem Kontext relevanten Einflussfaktoren und Prinzipien definiert.

In einer sich dynamisch verändernden Umwelt ist der Unternehmenswandel zu einem Dauerzustand avanciert [Kru98, S.228 ff]. Der Erfolg und die dauerhafte Existenz eines Unternehmens hängen daher in zunehmendem Masse von seiner Fähigkeit ab, sich an Veränderungen seines Umfelds anpassen zu können [Pfa97, S.1]. In einem turbulenten Umfeld wird die Wandlungsfähigkeit eines Unternehmens somit immer mehr zum entscheidenden Wettbewerbsfaktor [AG04]. Insbesondere die eingesetzten Informationssys-

teme müssen als integrale Bestandteile des Unternehmens einen essenziellen Beitrag zur Wandlungsfähigkeit liefern [LH98]. Diese Wandlungsfähigkeit - im Sinne einer schnellen und effizienten Anpassung an neue Anforderungen - kann durch eine möglichst zukunftsfähige IS-Architektur unterstützt werden, um etwa Innovationen, neue Technologien, immer kürzere Produktlebenszyklen als auch organisatorische und rechtliche Veränderungen absorbieren und integrieren zu können.

Eine vor allem kostengetriebene IT-Anwendungslandschaft kann aber auf Dauer keine stetigen Anpassungen der IS-Architektur und einer damit einhergehenden Veränderung der Anwendungslandschaft verkraften. Um dem durch die dynamische Umwelt entstehenden Wandlungsdruck begegnen zu können, sollte eine zukunftsfähig aufgestellte IS-Architektur zudem sicherstellen, dass sich ihre Komponenten vielseitig einsetzen lassen. Nebst der Wandlungsfähigkeit sollte auch ein gewisses Mass an Nachhaltigkeit im Sinne von Beständigkeit und Robustheit gewährleistet sein. Unter Nachhaltigkeit im Sinne von Beständigkeit und Robustheit wird verstanden, dass veränderten Anforderungen durch Wiederverwendbarkeit von Komponenten entsprochen werden kann. Um auf die unvorhersehbaren Umweltveränderungen angemessen reagieren zu können, wird eine entsprechende Reaktionsfähigkeit und Handlungsfähigkeit vorausgesetzt [SBBD01].

Damit ein Unternehmen in einer von kontinuierlichem Wandel geprägten Umwelt bestehen kann und somit zukunftsfähig ist, sollte dessen IS-Architektur im Kern flexibel aufgestellt sein und sowohl die Prinzipien der Wandlungsfähigkeit als auch der Nachhaltigkeit bedienen können. Um die Anforderungen an die Prinzipien der Wandlungsfähigkeit und Nachhaltigkeit zu verstehen, müssen zuerst die potentiellen externen und internen Einflussfaktoren identifiziert werden, da diese Faktoren die Gewichtung der Prinzipien einer zukunftsfähigen IS-Architektur beeinflussen.

Potentielle externe Einflussfaktoren auf ein Unternehmen und seine IS-Architektur lassen sich nach Andresen [AGS05, S.69f] in zwei Klassen unterteilen. Einerseits das direkte Wettbewerbsumfeld nach Porter [Por85] und andererseits die globale Unternehmens-Umfeld nach Hall [Hal88], Fahey & Narayanan [FN86] und Müller-Stewens & Lechner [MSL16]:

Externe Einflussfaktoren (EFE) aus dem direkten Wettbewerbsumfeld:

EFE 1: Rivalität unter den bestehenden Anbietern (Wettbewerbsfaktor)

EFE 2: Potentielle Neuanbieter (Konkurrenzfaktor)

EFE 3: Substitutionsprodukte (Ersatzfaktor)

EFE 4: Abnehmer (Verhandlungsfaktor)

EFE 5: Lieferanten (Verhandlungsfaktor)

Externe Einflussfaktoren (EFE) aus dem globalen Unternehmensumfeld:

EFE 6: Technologische Umwelt

EFE 7: Politisch-rechtliche Umwelt

EFE 8: Arbeitsmarkt

EFE 9: Geld- und Kapitalmarkt

EFE 10: Ökologische Umwelt

Diese Faktoren sollten durch ein Unternehmen jeweils analysiert und bewertet werden. Die Filterung der systemkritischen Umwelteinflüsse erfolgt branchen- und/oder unternehmensspezifisch über die unterschiedliche Gewichtung der Einflussfaktoren. Je nach Bewertung dieser Faktoren werden die Anforderungen und somit die Schwerpunkte einer zukunftsfähigen IS-Architektur im Spannungsfeld von Wandlungsfähigkeit und Nachhaltigkeit gesetzt. Ein entsprechendes Verfahren, bei welchem diese Einflussfaktoren dem vorhandenen Potential mit Fokus auf eine Wandlungsfähigkeit nach Spath et al. [SBBD01, S.9] gegenübergestellt wird, hat Andresen et al. ausgearbeitet [AGS05, S.70ff]. Hieraus lassen sich Handlungsempfehlungen für die IS-Architektur ableiten.

Nebst den externen gilt es aber auch interne Einflussfaktoren wie etwa Governance-Richtlinien oder Geschäfts- und IT-Strategien hinsichtlich der Prinzipienausgestaltung zu berücksichtigen. Gerade im unternehmensinternen Kontext stellt sich hierbei die Forderung, dass die unterschiedlichen Zeithorizonte dieser Einflussfaktoren berücksichtigt werden sollten. So wird eine IS-Architektur in der Regel beispielsweise länger bestehen als eine IT-Strategie.

Um den verschiedenen Zeithorizonten und Veränderungsintensitäten der Einflüsse aus einem turbulenten Umfeld gerecht zu werden, müssen nach Spath et al. so neben reaktiven auch proaktive Fähigkeiten etabliert werden. Die Entwicklungsfähigkeit bezeichnet dabei die proaktive Fähigkeit des Unternehmens zur nachhaltigen Gestaltung der Unternehmensstrukturen bei längerfristig prognostizierbaren, wechselnden Anforderungen [SBBD01, S9ff]. Diese Entwicklungsfähigkeit basiert wiederum auf einer Innovationsfähigkeit, welche ein proaktives Handeln ermöglichen soll.
Die internen Einflussfaktoren stellen sich hierbei wie folgt zusammen:

Interne Einflussfaktoren (EFI):

EFI 1: (IT-) Governance Richtlinien (z.B. Sourcing)

EFI 2: Unternehmensstrategie

EFI 3: (Digital- und) IT-Strategie

EFI 4: Personal Strategie (z.B. Know-How Sicherstellung)

Die Einflussfaktoren mit ihren Zusammenhängen und beispielhaften Lifecycle-Zeithorizonten (grün für Beständigkeit, gelb für Veränderungsanforderung) werden unter Abbildung 17 visualisiert.

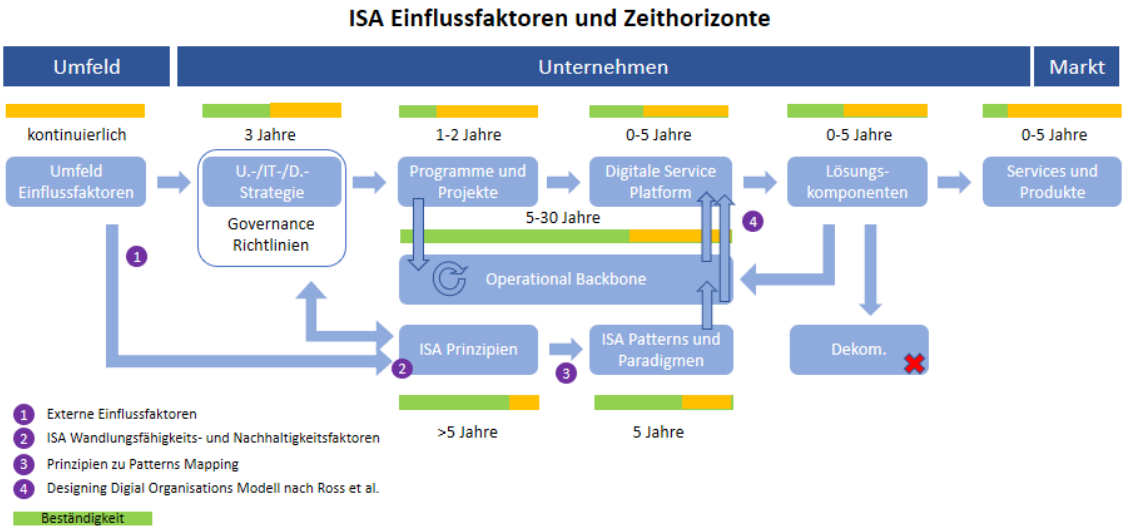


Abbildung 17: Abhängigkeiten von externen und internen Einflussfaktoren mit ihren exemplarischen Zeithorizonten

Der Bereich «Umfeld» umfasst hierbei die erläuterten Einflussfaktoren EFE 1-10, wobei sich der Bereich «Markt» spezifisch auf die Services und Produkte bezieht, welche die Abnehmer (EFE 4) betreffen. Die Einflussfaktoren EFE 1-10 sowie EFI 1-3 beeinflussen direkt die ISA-Prinzipien, wobei an dieser Stelle aber auch eine Rückkopplung der langfristig ausgelegten ISA-Prinzipien zu EFI 1-3 stattfindet. Die für die ISA-Umsetzung (Erneuerung und Neugestaltung) relevanten Programme und Projekte werden indirekt über die Anforderungen der ISA-Patterns und -Paradigmen abgeglichen, welche nach den ISA-Prinzipien ausgerichtet sind.

Der Bereich der Anwendungslandschaft teilt sich gemäss einer logischen Unterteilung nach Ross et al. [RSBJ17] in einen Core-Teil «Operational Backbone» und einen agil getriebenen Bereich «Digital Service Platform», welche beide im nachfolgenden Abschnitt 3.6 im Detail erläutert werden. «Lösungskomponenten» einer Anwendungslandschaft, welche die für EFE 4 relevanten Services und Produkte realisieren, werden in der Regel im Kontext der «Digital Service Platform» implementiert, wobei diese im Rahmen der an sie gestellten Anforderungen in den nachhaltigen «Operational Backbone» überführt werden oder am Ende ihres Lifecycles «dekommissioniert» werden.

Die in diesem Abschnitt ermittelten Prinzipien, nach welchen eine zukunftsfähige IS-Architektur ausgerichtet sein sollte, umfassen folglich:

Prinzipien (P):

P 1: Zukunftsfähigkeit

P 2: Wandlungsfähigkeit

P 3: Nachhaltigkeit

P 4: Entwicklungsfähigkeit

P 5: Robustheit und Resilienz

P 6: Innovationsfähigkeit

P 7: Flexibilität

P 8: Reaktions- und Handlungsfähigkeit

Diese Prinzipien lassen sich – inspiriert von Spath et al. [SBBD01, S.9] – anhand eines Prinzipien-Baustein-Modells zu einer Pyramide der Zukunftsfähigkeit zusammenfügen (vgl. Abbildung 18 auf Seite 29). Das Bild zeigt hierbei symbolisch das allgegenwärtige Umfeld[4], welches anhand von Einflussfaktoren Anforderungen an die Zukunftsfähigkeit stellt. Das Unternehmen und seine IS-Architektur begegnen diesen Anforderungen hierbei im Zentrum anhand der beiden Perspektiven «Wandlungsfähigkeit» und «Nachhaltigkeit», welche durch «Agilitäts-» und «Beständigkeits-Charakteristiken» die «Zukunftsfähigkeit» ermöglichen. Das Fundament mit dem «Nährboden» für die Zukunftsfähigkeit bildet die «Flexibilität» mit ihren Bausteinen «Reaktions- und Handlungsfähigkeit» auf der einen Seite und der «Innovationsfähigkeit» auf der anderen Seite.

Anhand einer Bewertung und Gewichtung der Einflussfaktoren ermittelt das Unternehmen nun eine Balance an den Berührungspunkten zwischen «Wandlungsfähigkeit» und «Nachhaltigkeit» (visualisiert durch den dazwischenliegenden Pfeil).

Die beiden Bausteine «Wandlungsfähigkeit» und «Nachhaltigkeit» bilden zusammen den obersten und entscheidenden Baustein für die Zukunftsfähigkeit des Unternehmens, da die Wandlungsfähigkeit grundsätzlich durch Prinzipien der Nachhaltigkeit gestützt sein sollten aber erst in Kombination ein stabiles Fundament für die Zukunftsfähigkeit und eine prosperierende, gesamtheitliche Lösung bilden.

[4] Die zuvor der (externen) «Umwelt» zugeordneten Einflusskatoren werden hiernach als «Umfeld» benannt, da dieser Begriff sowohl die externen, als auch die internen Einflussfaktoren umfasst

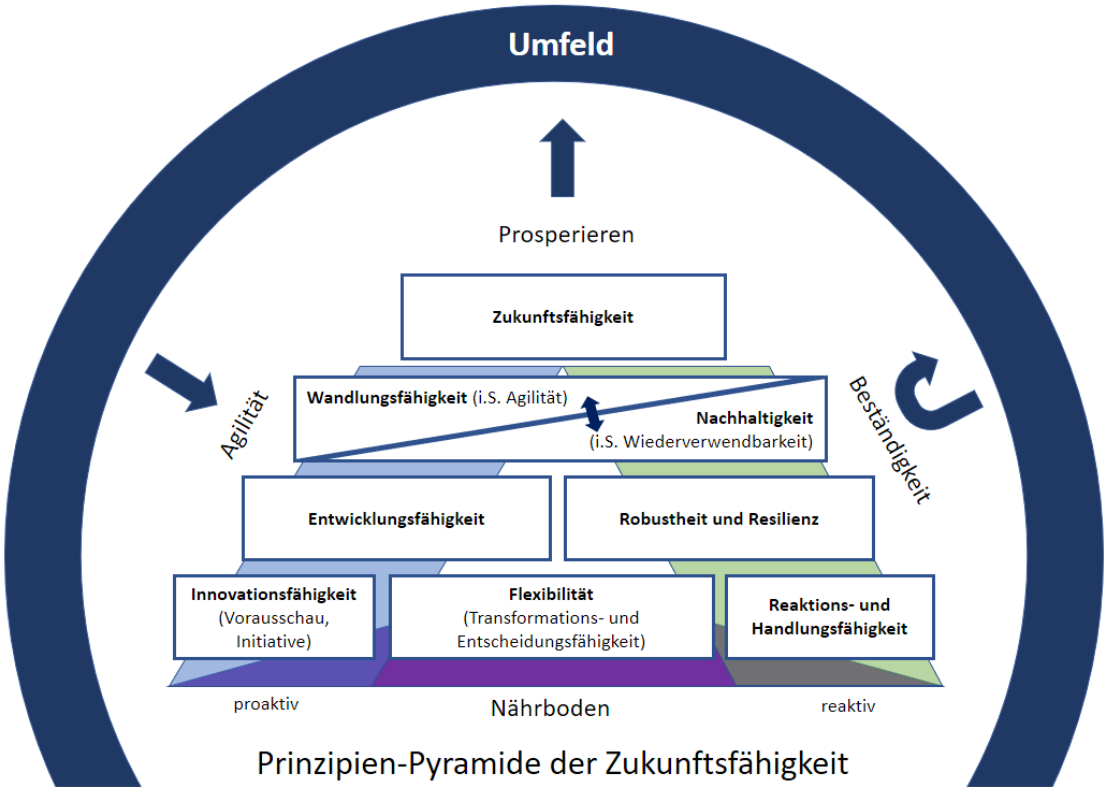


Abbildung 18: Prinzipien-Pyramide der Zukunftsfähigkeit; in Anlehnung an Spath et al. [SBBD01, S.9]

3.5 ISA-Paradigmen

Nach Ermittlung der Einflussfaktoren aus dem Unternehmensumfeld und den ISA-Zukunftsfähigkeits-Prinzipien im vorhergehenden Abschnitt werden nachfolgend die relevanten ISA-Paradigmen für die beiden ISA-Prinzipien-Perspektiven «Wandlungsfähigkeit» (Prinzip P2,P4) und «Nachhaltigkeit» (Prinzip P3,P5) und dem Fundament (Prinzip P6-8) untersucht (vgl. Abbildung 18).

Feess et al. versteht unter dem Begriff «Paradigma» im Kontext der Informatik ein «Fundamentales Konzept, das [...] eine Vorgehensweise zur Problemlösung entscheidend prägt und auch als Muster bezeichnet werden kann» [FLS18]. Auf die vorliegende Problemstellung bezogen hat ein ISA-Paradigma den Anspruch, eine wünschenswerte Eigenschaft eines Bestandteiles einer IS-Architektur bzw. des Zusammenwirkens der Elemente aufzuzeigen, welche die Prinzipien der Zukunftsfähigkeit unterstützen. Hierfür muss zunächst eine passende Auswahl von ISA-Paradigmen bestimmt werden. Während eine systematische Übersichtsarbeit von Ullrich & Weber eine Vielzahl an untersuchten und potentiell relevanten Paradigmen im Bezug auf Wandlungsfähigkeit definiert [UW16, S.792ff], liefert das «Product Quality Model» nach ISO 25010:2011 [Sta11][5] die potenti-

[5] siehe auch Anhang A

ellen Paradigmen im Kontext der Nachhaltigkeit, wobei diese Paradigmen je nach Sichtweise und Argumentation auch äquivalent aus der anderen Perspektive verwendet werden könnten. Zusätzlich werden ebenfalls noch die Prinzipien von Tallinn [EU17] sowie Ergebnisse aus den Studien von Gaughan [Gau18] sowie Ross et al. [RSBJ17] berücksichtigt, wobei sich die beiden Letzteren auf die Herausforderungen einer multimodalen IS-Applikationslandschaft aus der Fundament-Perspektive der Zukunftsfähigkeit beziehen.

Auf Basis der im Rahmen dieser Arbeit ermittelten Zukunftsfähigkeits-Prinzipien werden diejenigen Paradigmen mit dem grössten erwarteten Unterstützungspotential ausgewählt und in einer Verknüpfungstabelle (vgl. Tabelle 1) in Beziehung zu den zukunftsfähigen ISA-Prinzipien gestellt.

ISA-Prinzipien zu -Paradigmen Verknüpfung		
Prinzipien-Perspektive	Prinzipien-Bausteine	Paradigmen
Wandlungs-fähigkeit	Wandlungsfähigkeit (P2) Entwicklungsfähigkeit (P4)	Modularität (WP1)
	Wandlungsfähigkeit (P2) Entwicklungsfähigkeit (P4)	Interoperabilität (WP2)
	Wandlungsfähigkeit (P2) Entwicklungsfähigkeit (P4)	Mobilität (WP3)
Nachhaltigkeit	Robustheit und Resilienz (P5)	Verfügbarkeit (NP1)
	Robustheit und Resilienz (P5)	Reaktivität (NP2)
	Robustheit und Resilienz (P5)	Zustandslosigkeit (NP3)
	Wiederverwendbarkeit (P3) Robustheit und Resilienz (P5)	Selbstorganisation und Selbstähnlichkeit (NP4)
Fundament	Wandlungsfähigkeit (P2) Nachhaltigkeit (P3) Entwicklungsfähigkeit (P4) Flexibilität und Transformationsfähigkeit (P7) Reaktions- und Handlungsfähigkeit (P8)	Skalierbarkeit (FP1)
	Innovationsfähigkeit (P6) Flexibilität und Entscheidungsfähigkeit (P7) Reaktions- und Handlungsfähigkeit (P8)	Wissen über IS-Architektur (FP2)
	Innovationsfähigkeit (P6) Flexibilität und Transformationsfähigkeit (P7) Reaktions- und Handlungsfähigkeit (P8)	Multimodale IS-Architektur (FP3)

Tabelle 1: ISA-Prinzipien zu -Paradigmen Verknüpfung

Nachfolgend werden die identifizierten und verknüpften ISA-Paradigmen, gegliedert nach den Perspektiven Wandlungsfähigkeit (WP), Nachhaltigkeit (NP) und Fundament (FP) erläutert und zusätzlich durch potentielle Lösungstechnologien breiter abgestützt.

3.5.1 WP1 - Modularität

Aier & Dogan beschreiben Flexibilität als «eine Voraussetzung für nachhaltige Architekturen, da sich nur flexible Architekturen dauerhaft an die Anforderungen ihrer Umwelt anpassen können» [AD05, S.616].

Ein Mittel für diese Flexibilisierung und Komplexitätsreduktion innerhalb von IT- und Organisationsarchitekturen ist die Modularisierung innerhalb der Architekturen. Modularisierung bedeutet allgemein die Strukturierung eines Systems in kleine, teilautonome und überschaubare Subsysteme. Diese Subsysteme oder Module bestehen aus einer definierten Schnittstelle und einem Modulrumpf, der die spezifizierte Leistung und die Moduleigenschaften über die Schnittstelle implementiert. Modularisierte Komponenten lassen sich so entlang der Geschäftsprozesse effizient kombinieren, wiederverwenden und anpassen.

Gemäss dem *Law of Conway* [Con68] wurde die Gesetzmässigkeit feststellt und später auch nochmals durch MacCormack et al. bestätigt [MRB11], dass die Kopplung der etablierten Organisationen mit den Modulstrukturen der entwickelten Produkte korreliert, was zu vielen Doppelspurigkeiten führen kann. Conway empfiehlt aus diesem Grund, dass die IT- und Organisationsarchitekturen anhand von funktionalen Modulen entlang dem Geschäftsprozess ausgerichtet werden. Dieses Gestaltungsprinzip nach einer fachlich motivierten Ausrichtung von IT- und Organisationsarchitekturen wird in der Praxis beispielsweise anhand des Lösungstechnologie-Ansatzes der Service Orientierten Architektur (SOA) erreicht [EJH+08, S.VIII].

Der Begriff SOA ist nach Fowler [Fow05] heute jedoch breit gefasst und mehrdeutig belegt. Präzisierend wird daher im Rahmen dieser Arbeit gemäss der Definition von The Open Group [Gro16] von einer Microservice-Architektur (MSA) als ein spezieller Teilbereich der SOA gesprochen. MSA verfolgt nach Fowler [Fow14] die ISA-Paradigmen der Unabhängigkeit, Parallelisierung und Skalierung, wobei der Fokus der Organisationsarchitektur – zur Überwindung des Law of Conway – auf funktionsübergreifenden Teams liegt (vgl. Abbildung 19 auf Seite 32), welche über ein Projekt hinaus die Funktion respektive den Service betreuen.

MSA gilt als ein ISA-Paradigma, bei welchem komplexe Dienste aus unabhängigen Prozessen komponiert werden, die untereinander über genau definierte Schnittstellen kommunizieren. Der Inhalt des Microservices wird dabei stets als Blackbox betrachtet. Diese Dienste sind hochgradig entkoppelt, autonom und erledigen eine exakt definierte Aufgabe gemäss der Unix-Philosophie: «Do one thing, and do it well». Hierdurch wird ein modularer Aufbau eines Anwendungsservices ermöglicht.

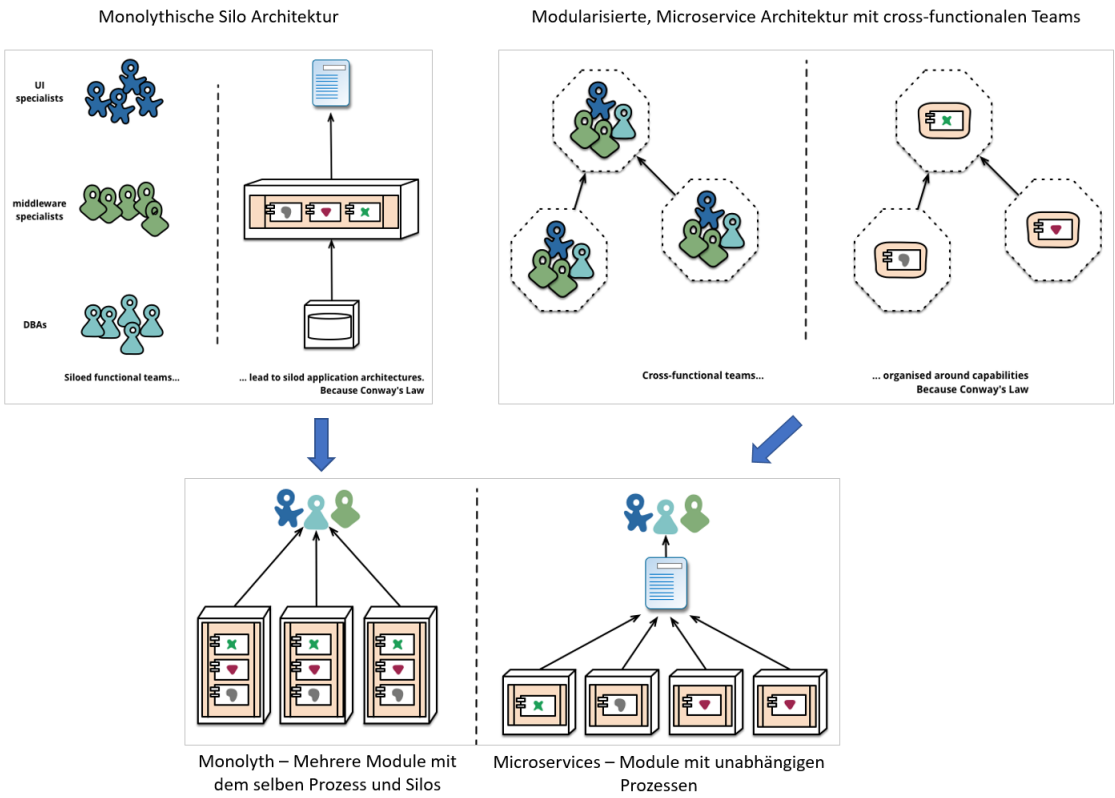


Abbildung 19: Monolythische Architektur und Microservice-Architektur im Vergleich; in Anlehnung an Fowler [Fow14]

Durch die modularisierten und entkoppelten MSA-Services stellt sich die Frage der Prozessflusssteuerung, damit die Geschäftsprozesse automatisiert gesteuert und kontrolliert ablaufen können. Die Prozessflüsse können hierbei entweder orchestriert, choreographiert oder komponiert werden. Nach Ziemann [Zie10, S.63f] gilt ein orchestrierter Geschäftsprozess, als ein Geschäftsprozess, der für die Ausführung durch eine einzelne Prozess-Engine ausgelegt ist. Funktionen, die in diesem Geschäftsprozess enthalten sind, werden als Services, gesteuert durch diese Engine aufgerufen. Ein choreographierter Geschäftsprozess ist hingegen zur wechselwirkenden Ausführung durch verschiedene Engines gedacht, wobei jede Engine für sich für die Ausführung von spezifischen Teilen des Prozesses verantwortlich ist. Von Komposition kann nach Röwekamp [Rö17] gesprochen werden, wenn die Ablaufsteuerung der Anwendung nicht explizit durch eine Engine gesteuert wird, sondern sich implizit durch die ausgelösten Events und die zugehörigen Event-Listener, also die für das Event registrierten Funktionen, ergibt. Es findet demnach eine dynamische, selbstorganisierte Komposition und keine statische Orchestrierung oder Choreographie statt.

Der Vorteil einer solchen eventgetriebenen Architektur liegt darin, dass sich durch die starke Entkopplung der einzelnen fachlichen Komponenten neue Business-Logik einfach hinzufügen und bestehende Business-Logik problemlos ändern lässt. Solange diese stabil oder abwärtskompatibel bleibt, lassen sich die einzelnen Funktionen unabhängig

voneinander ändern. Da im Kontext einer zukunftsfähigen IS-Architektur eine potentielle Integration von externen Partner-Services als zielführend erachtet wird und somit verschiedene Prozess-Engines im Spiel sind, werden an dieser Stelle Choreographie- oder Kompositions-Lösungsarchitekturen empfohlen.

Durch die mit MSA erzielte Unabhängigkeit und Parallelisierung – folglich auch die Wiederverwendbarkeit von Modulen – wird die Basis für eine Anwendungslandschaft geschaffen, welche eine sogenannte Business Modularity-Architektur[6] unterstützt. Eine Business Modularity-Architektur hat zum Ziel, agilen Strategieanforderungen gerecht zu werden, welche auf massgeschneiderten, lose gekoppelten[7], funktionalen Geschäftsprozessmodulen beruht. Sie gilt nach Ross et al. [RWR06, S.77ff] als ein kontinuierlich anzustrebendes Architektur-Paradigma des obersten Reifegrades, welches für Unternehmen wiederverwendbare IT-gestützte Geschäftsprozesskomponenten ermöglicht, um globale Standards zu erhalten und gleichzeitig lokale Unterschiede zu ermöglichen. Individuelle Prozessmodule einer Business Modularity-Architektur erweitern den Operational Backbone[8] mit seinen effizienten, integrierten Kernprozessen und ermöglichen nach Ross et al. [RWR06, S.78ff] anhand von standardisierten Schnittstellen die nahtlose Verbindung zwischen internen, externen, lokalen und unternehmensweiten Geschäftsprozessen.

Durch die Sicherstellung der Vorhersagbarkeit von Kernprozessen anhand von Schnittstellenspezifikationen können modulare Architekturen eine Plattform für Innovationen bieten. Die modulare Architektur ermöglicht durch Wiederverwendung von Modulen und einem stabilen Core mit zuverlässigen Schnittstellenvereinbarungen das Durchführen von schnellen, lokalen Experimenten, wobei die erfolgreichsten Module wieder in den Operational Backbone überführt und zur Wiederverwendung dem gesamten Unternehmen zur Verfügung gestellt werden können. Die so schnell entwickelten und massgeschneiderten Module ermöglichen es Unternehmen rasch auf veränderte Marktbedingungen zu reagieren.

Potentielle Architektur-Patterns und Lösungstechnologien sind:

- Modularisierung (Wandlungsfähigkeit, Entwicklungsfähigkeit)
- MSA (Lösungstechnologie)
- Business Modularity-Architektur nach Ross et al. [RWR06, S.64ff] (Modularisierungs-Pattern)

[6] Business Modularity-Architektur: Architektur mit Modularen Business Komponenten

[7] Lose Kopplung bedeutet, dass die Kommunikation asynchron über einen Vermittler erfolgt

[8] Operational Backbone: Der für Zuverlässigkeit augelegte, tendentiell träge und oft monolithische Bereich einer IT-Architektur

3.5.2 WP2 - Interoperabilität

Geschäftsprozesse werden nach Ziemann durch eine Sequenz von Geschäftsfunktionen gebildet, die darauf abzielen einen Output für interne und externe Kunden zu erschaffen [Zie10, S.17]. Kollaborative Geschäftsprozesse stellen hierbei Geschäftsprozesse dar, welche durch zwei oder mehr autonome Organisationen ausgeführt werden [Zie10, S.24]. Damit Organisationen kooperieren können, braucht es Interoperabilität. Interoperabilität lässt sich als Fähigkeit zur Zusammenarbeit von autonomen Systemen definieren und grenzt sich durch eine lose Kopplung und Autonomie von einem integrierten Systemen ab [Zie10, S.25ff].

Aus Geschäftssicht lässt sich Interoperabilität als Fähigkeit von Organisationen, zur Ausführung von kollaborativen Geschäftsprozessen zwischen ihnen definieren [Zie10, S.26][9]. Aus technischer Sicht gewährleistet Interoperabilität einen einfachen Zugang zu verschiedenen Daten- und Verarbeitungsressourcen innerhalb eines Arbeitsablaufs und ermöglicht die einfache Verknüpfung unterschiedlicher Informationssysteme [AGS05]. Durch die Erfordernis der Kooperation mehrerer Anwendungssysteme zur Abbildung der Geschäftsprozesse kommt der Interoperabilität zwischen den Informationssystemen eine zentrale Rolle zu. Für die Kooperation zwischen Unternehmen bedeutet die Interoperabilität eine erhöhte Kommunikations- und Kooperationsfähigkeit unter Berücksichtigung von Standards. Während sich Standardisierung auf Datenebene (Syntaktik) insbesondere auf allgemeinverständliche Datenformate wie XML bezieht, gilt es für die Standardisierung auf Objektebene (Semantik) die Kommunikation zwischen Informationssystemen zu unterstützen [AGS05, S.74]. Interoperabilität lässt sich hierbei in verschiedene Interoperabilitätsebenen unterteilen [Ins16]:

- Strukturelle Interoperabilität: Das Ziel dieser Interoperabilitätsebene besteht darin, einen Datenstrom zwischen zwei Systemen austauschen zu können. Auf dieser Ebene finden sich Bus-Systeme, serielle und parallele Anschlüsse ebenso wie Protokolle.

- Syntaktische Interoperabilität: Das Ziel dieser Interoperabilitätsebene besteht darin, die Informationseinheiten im ausgetauschten Datenstrom zu identifizieren. Standards und Formate wie XML, CSV und JSON können syntaktische Interoperabilität gewährleisten.

- Semantische Interoperabilität: Das Ziel dieser Interoperabilitätsebene besteht darin, ein gemeinsames Verständnis der Informationseinheiten bei den beteiligten Systemen herzustellen. Dazu setzt man meist auf Ordnungssysteme wie Nomenklaturen oder Taxonomien und Klassifikationssysteme.

[9] Sog. Business Interoperability

- Organisatorische Interoperabilität: Das Ziel dieser geschäftsorientierten Interoperabilitätsebene besteht darin, gemeinsame Workflows sowie Rollen- und Berechtigungskonzepte zu etablieren.

Eine Interoperabilitätsschnittstelle die alle Ebenen umfasst kann nach Ziemann anhand eines Business Interoperability Interfaces (BII) realisiert werden [Zie10, S.37]. Ein BII einer Organisation repräsentiert hierbei die Summe aller Modelle, welche die Organisation ihren Kollaborationspartnern zur Verfügung stellen muss, um einen kollaborativen Geschäftsprozess zu ermöglichen. Es beschreibt diejenigen Elemente eines kollaborativen Geschäftsprozesses, welche durch die Organisation zur Verfügung gestellt werden, aber auch diejenigen, welche von den Kollaborationspartnern erwartet werden [Zie10, S.37].

Der von Ross et al. [RWR06, S.64ff] empfohlene Einsatz von Business-Modulen, welche sowohl intern als auch extern - basierend auf Standard Schnittstellen - angesprochen und wiederverwendet werden können, lässt sich so optimal anhand eines BII realisieren.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Business Interoperability Interface (BII) nach Ziemann [Zie10, S.37] (Interoperabilitäts-Pattern)
- Business Modularity-Architektur nach Ross et al. [RWR06, S.64ff] (Interoperabilitäts-Pattern)
- Kommunikations- und Kooperationsfähigkeit von Business-Modulen (Wandlungsfähigkeit, Entwicklungsfähigkeit)
- Syntaktische Interoperabilität durch offene Standard-Formate (Wandlungsfähigkeit, Entwicklungsfähigkeit)
- Semantische Interoperabilität durch definierte Objekt-Bezeichnungen (Wandlungsfähigkeit, Entwicklungsfähigkeit)
- Organisatorische Interoperabilität durch definierte Rollen- und Berechtigungskonzepte sowie Workflows (Wandlungsfähigkeit, Entwicklungsfähigkeit)

3.5.3 WP3 - Mobilität

Mobilität bezieht sich in diesem Zusammenhang auf Ansätze der Entkopplung im Sinne einer Austauschbarkeit, Ungebundenheit und offener Standards. Einerseits soll hierbei eine Service-Entkopplung helfen, heterogene IT-Systeme über ein einheitliche Schnittstelle ansprechbar zu machen, was nachfolgend unter der Begriffserläuterung «Entkopplung durch Abstrahierung» aufgegriffen wird, andererseits soll eine physische Ressourcen-Entkopplung der Services helfen, eine von der Infrastruktur ungebundene Lösung zu ermöglichen, was unter Sektion «Physische Ressourcen Entkopplung» aufgegriffen wird.

Entkopplung durch Abstrahierung

Typischerweise sind IT-Landschaften sehr heterogenen. Eine nach Ross et al. [RSBJ17] auf offenen Standards basierte Schnittstelle, kann so als Abstraktionselement benutzt werden. Hiermit lässt sich eine Entkopplung der heterogenen Applikationslandschaft realisieren und hinter der Schnittstelle liegende Lösungskomponenten lassen sich im Sinne der Mobilität einfacher austauschen, da die Service-Konsumenten keinerlei Anpassungen vornehmen müssen. Die so erzielte Mobilität der Implementierung unterstützt hiermit die Entwicklungsfähigkeit und fördert die Wandlungsfähigkeit. Eine solche Abstrahierung kann beispielsweise durch eine MSA realisiert werden.

Physische Ressourcen Entkopplung

Wandlungsfähige IS-Architekturen und ihre Anwendungslandschaften sollten nach Andresen et al. möglichst plattformunabhängig sein [AG04]. Hierfür sollten sie möglichst entkoppelt – sprich ungebunden – von den zugrundeliegenden Service-Komponenten wie Hardware, Betriebssysteme oder Runtime-Umgebung laufen. Eine Entkopplung verhindert so allfällige Ressoucenlimitierungen, fördert die Parallelisierung von Aktivitäten durch Service-Replizierung und ermöglicht im Optimalfall eine dynamische Ressourcenallokation zur Laufzeit (Elastizität). Solche ISA-Patterns sind somit flexibel einsetzbar, entwicklungsfähig und fördern hiermit eine Wandlungsfähigkeit.

Eine Möglichkeit zur Entkopplung bieten beispielsweise sogenannte gekapselte Systeme (Container). Ein Container ist ein leichtgewichtiges, eigenständiges, ausführbares Software-Paket, das alles enthält, was zum Ausführen benötigt wird: Code, Laufzeit, Systemwerkzeuge, -bibliotheken und -einstellungen [Doc17]. Ein Container läuft unabhängig von der Umgebung immer gleich und lässt sich durch diese Entkopplung agil zwischen On-Premise Landschaften als auch der Cloud (Container as a Service; CaaS) verschieben. Mehrere Container können schnell gestartet werden und parallel auf derselben Maschine als Prozess laufen. Somit sind sie sehr einfach skalierbar und hochverfügbar. Da Container zustandslos sind, dienen sie lediglich der Erfüllung einer spezifizierten Funktionalität und halten aber selber keine Daten persistent.

Ein weitergehender Ansatz bieten auch Serverless Functions[10]. Hierbei lassen sich leichtgewichtige Funktionen wie etwa eine Benutzerauthentisierung als Funktionsaufruf ohne einen (sichtbaren) Server realisieren, und basierend auf einem Container auch beliebig horizontal skalieren. Serverless Functions stellen somit im Prinzip eine minimale Art eines Microservices dar. Im Allgemeinen sind die verschiedenen Cloud as-a-Service Lösungen einer Entkopplung zweckdienlich, indem sich der Kunde selbst nicht mehr um die darunterliegenden Komponenten kümmern muss. Eine Übersicht zu den verschiedenen «as-a-Service» Ausbaustufen wurde Abbildung 20 auf Seite 37 dargestellt.

[10] Im Cloud-Kontext als Function as a Service (FaaS) bekannt

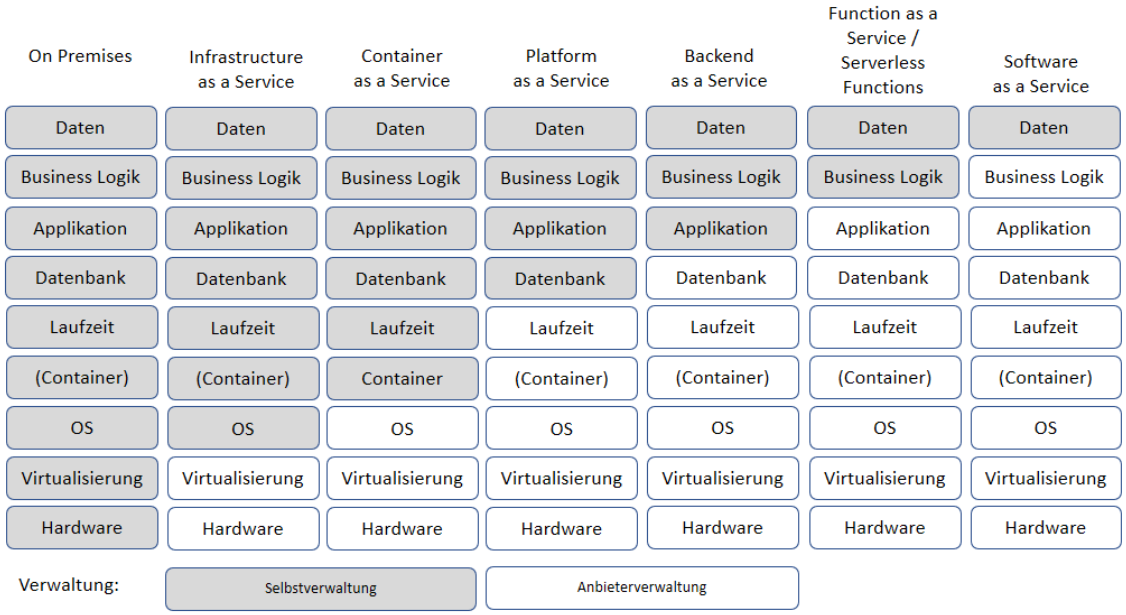


Abbildung 20: Ausbaustufen der verschiedenen «as-a-Service»-Angebote

Potentielle Architektur-Patterns und Lösungstechnologien:

- Container (Lösungstechnologie)
- MSA (Lösungstechnologie)
- FaaS / Serverless Functions (Lösungstechnologie)
- Diverse abstrahierende as-a-Service Angebote aus der Cloud (Lösungstechnologie)
- Abstraktion der heterogenen Applikationslandschaft durch offene Standards nach Ross et al. [RSBJ17] (Wandlungsfähigkeit, Entwicklungsfähigkeit)

3.5.4 NP1 - Verfügbarkeit

Um einen räumlich und zeitlich unbegrenzten Zugriff auf Anwendungen, Daten und Funktionen einer IS-Architektur gewährleisten zu können, müssen wichtige Services einer Anwendungslandschaft (hoch-)verfügbar ausgestaltet sein. Diesem Paradigma kann unter anderem mittels Redundanz begegnet werden. So können Services selbst redundant gebaut sein (z.B. Container, Serverless Functions oder MSA) oder auch die Daten der Services zur besseren Verfügbarkeit verteilt gehalten werden. Für eine parallele Verarbeitung von sehr grossen Datenmengen wie etwa im Rahmen von NoSQL-Datenbanksystemen wird dies auch aus Gründen der Antwortzeiten so gemacht. Redundanz fördert so unmittelbar die Robustheit und Resilienz von IS-Architekturen, offenbart aber gleichzeitig auch die Probleme dieses Gestaltungsparadigmas: Redundanz erzeugt Dezentralität, welche verwaltet werden muss. Neuere Lösungstechnologien wie DFS[11] oder Stream Proces-

[11] DFS: Distributed File System; Ein hochverfügbares Dateisystem zur Speicherung und parallelem Abfragen sehr grosser Datenmengen auf den Dateisystemen mehrerer Knoten.

sing Platformen (Event Store) können damit aber bereits sehr gut umgehen und beinhalten hierfür schon ausgereifte Mechanismen, so dass eigentlich nur noch das Kostenargument effektiv ins Gewicht fällt. Wenn in einem hochverfügbaren, verteilten System aber eine effektive transaktionale Konsistenz gefordert ist, stellt sich die Problemstellung des CAP-Theorems.

CAP-Theorem

Das CAP-Theorem besagt nach Fox & Brewer [FB99], dass in einem verteilten System zu einem gegebenen Zeitpunkt nur zwei der drei Eigenschaften Konsistenz (C), Verfügbarkeit (A) und Partitionstoleranz (P) gleichzeitig implementierbar sind. Würde nach Von Brauck & Neudert [VBN14] beispielsweise eine Datenbank nur auf einer einzelnen Partition betrieben, wäre Konsistenz und Verfügbarkeit gegeben, aber es würde keine alternative Partition (Ausfallsicherheit) existieren. Wäre die Datenbank hingegen auf zwei Partitionen verteilt, wäre durch notwendige Replikationen der Knoten entweder die Verfügbarkeit der Daten oder die Konsistenz der Daten eingeschränkt. Der Architekt muss so entsprechend den Anforderungen an die Transaktionalität der verarbeiteten Daten die richtige Auswahl treffen. Daten, mit strikten Anforderungen an Transaktionalität richten sich hierbei nach den ACID-Prinzipien (atomar, konsistent, isoliert und haltbar), welche sich durch relationale Datenbanksysteme realisieren lassen. Sind die Anforderungen jedoch weicher hinsichtlich der notwendigen Transaktionalität, können die Daten nach den BASE-Prinzipien (Basically Available, Soft State, Eventually Consistent) angeboten werden, was eine bessere Verfügbarkeit erlaubt.

Erweitert um die NoSQL-Dimension von Big Data findet Fowler [Fow12], dass es in diesem Kontext in der Regel immer darum geht, Entscheidungen zu treffen, die sich zuerst auf die Verfügbarkeit und dann auf die Konsistenz konzentrieren; Datenbanken, die sich an die ACID-Eigenschaften halten, machen das Gegenteil. Fowler [Fow12] argumentiert weiter, dass das CAP-Theorem in seiner klassischen Interpretation die Latenz ignoriert, obwohl in der Praxis Latenz und Partitionen eng miteinander verbunden sind. Operativ findet der wesentliche CAP-Entscheid während eines Timeouts innerhalb eines Zeitraums statt, in dem das Programm eine grundlegende Entscheidung treffen muss: (1.) den Vorgang abbrechen und damit die Verfügbarkeit verringern oder (2.) mit der Operation fortfahren und damit Inkonsistenzen riskieren. Dies kann mit einem asynchronen, eventbasierten System verhindert werden.

Damit die Verfügbarkeit einer Microservice-Architektur gewährleistet ist, sollte folglich ein skalierbares Publish/Subscribe Messaging-System mit einem persistent gehaltenen Event Store verwendet werden, da es bei synchroner Kommunikation zu langen Antwortzeiten oder Abstürzen durch Locks kommen könnte. Zum Umgang mit Locks empfiehlt Fowler [Fow11] zusätzlich den Einsatz eines Command Query Responsibili-

ty Segregation (CQRS) Patterns zu prüfen. CQRS unterstützt nach Schmutz [Sch18] eine eventbasierte Architektur, da eine Unterteilung (Segregation) des Befehls (Command) und der Abfrage (Query) angestrebt wird.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Container (Service Verfügbarkeit, Robustheit und Resilienz)
- Big Data (DFS, NoSQL) (Datenverfügbarkeit, Robustheit und Resilienz)
- MSA (Lösungstechnologie)
- FaaS / Serverless Functions (Lösungstechnologie)
- Event Store (Asynchrone Publish/Subscribe Messaging) (Lösungstechnologie)
- CQRS, BASE, ACID, MVCC (Transaktionale Patterns zur Datenkonsistenz resp. Verfügbarkeit, Robustheit und Resilienz)

3.5.5 NP2 - Zustandslosigkeit

Häufig ergibt sich nach Von Brauk & Neuert [VBN14] in der Architektur das Problem, das Daten an Resourcen gebunden werden, wie Sitzungs- oder Verbindungsdaten in einem Webserver. Dadurch entsteht u.a. das Problem, dass die Sitzung oder Verbindung nur durch die verarbeitende Instanz bedient werden kann. Im Falle einer synchronen Kommunikation, in der eine offene Transaktion auf die Antwort des schreibenden Dienstes wartet, führt dies dazu, dass Resourcen wie Threads, Speicher und Netzwerkverbindungen belegt werden. Besser ist es, dass der aufrufende Dienst seine Aktion abschickt und erst nach Abschluss der Aktion benachrichtigt wird. Die Daten werden so von den Resourcen entkoppelt, was dem Nachhaltigkeitsprinzip dient. Die Dienste können so gestaltet sein, dass sie den Kontext der Aktion nicht kennen (Zustandslosigkeit), aber das Ergebnis an interessierte Dienste verteilen. Eine verbreitetes, zustandloses (stateless) Protokoll ist beispielsweise REST[12].

Jede REST-Nachricht enthält alle Informationen, die für den Server bzw. Client notwendig sind, um die Nachricht zu verstehen [Wik18]. Weder der Server noch die Anwendung soll Zustandsinformationen zwischen zwei Nachrichten speichern. Jede Anfrage eines Clients an den Server ist insofern in sich geschlossen, als sie sämtliche Informationen über den Anwendungszustand beinhaltet, die vom Server für die Verarbeitung der Anfrage benötigt werden. Beispielsweise können eingehende Anfragen im Zuge der Lastverteilung unkompliziert auf beliebige Maschinen verteilt werden. Da jede Anfrage in sich geschlossen ist und Anwendungsinformationen somit ausschliesslich auf der Seite des Clients vorgehalten werden, ist auf der Seite des Servers keine Sitzungsverwaltung

[12] REST: Representational State Transfer

erforderlich. So wird auch die Ausfallsicherheit begünstigt, weil die Zustandslosigkeit fordert, dass transaktionale Datenübertragung in einem einzigen Seitenaufruf erfolgt.

Zustandslosigkeit lässt sich nach Von Brauk & Neuert [VBN14] aus zwei Richtungen fördern: Bottom-up durch die idempotente Gestaltung von Diensten und Top-down durch Terminierung zustandsbehafteter Kommunikation an den Grenzen des Systems. Idempotente Dienste liefern selbst bei wiederholter Ausführung einer Aktion dasselbe Ergebnis. Alle Leseoperationen sind idempotent; bei Schreiboperation erfolgt beispielsweise das Hinzufügen zu einer Liste so, dass kein Wert doppelt vorhanden ist (der alte Wert wird überschrieben) und Zahlenwerte nur gesetzt, aber nicht inkrementell erhöht werden können. Idempotente Dienste können Schreiboperationen sehr stabil horizontal skalieren. Sie verringern in Verbindung mit einem Messaging-System die Komplexität, da sie bei auftretender Nachrichtenduplikation stets das gleiche Ergebnis liefern (at-least-once Delivery). Zustandslose Dienste reagieren vorhersehbar und können in Kombination mit Caches die Leseoperationen signifikant beschleunigen.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Zustandslosigkeit fördern (z.B. REST Protokoll; Nachhaltigkeit, Robustheit)
- Idempotenz sicherstellen (Nachhaltigkeit, Robustheit)
- Asynchrone Kommunikation verwenden (Nachhaltigkeit, Robustheit)
- Event Store (Asynchrone Publish/Subscribe Messaging) (Lösungstechnologie)

3.5.6 NP3 - Reaktivität

Gemäss dem *Reaktiven Manifest* nach Bonér et al. [BFKT14] bestanden grosse Anwendungen vor wenigen Jahren noch aus Dutzenden von Servern, die Antwortzeiten im Sekundenbereich lieferten, regelmässig für Stunden gewartet wurden und Daten in der Grössenordnung von Gigabytes verarbeiteten. Der durch die digitale Transformation bedingte Wandel, bedeutet für moderne Systeme, dass sie Tausende von Vielkernprozessoren umfassen, während Benutzer Antwortzeiten im Bereich von Millisekunden sowie eine ständige Verfügbarkeit erwarten und sich die Datenmenge dabei eher in Petabytes bemisst. Die Anforderungen, die heute an Anwendungslandschaften gestellt werden, sind hierbei nur durch die gleichzeitige Ausrichtung nach vier Qualitäten zu erfüllen: Systeme müssen stets antwortbereit, widerstandsfähig, elastisch und nachrichtenorientiert sein (vgl. Abbildung 21 auf Seite 42), damit IS-Architekturen robuste und resiliente Prinzipien unterstützen. Bonér et al. [BFKT14] definiert diese vier Qualitäten hierbei wie folgt:

- **Antwortbereit**
  Das System antwortet unter allen Umständen zeitgerecht, solange dies überhaupt möglich ist. Antwortbereitschaft ist die Grundlage für Funktion und Benutzbarkeit

eines Systems, aber noch wichtiger ist, dass Fehler in verteilten Systemen nur durch die Abwesenheit einer Antwort sicher festgestellt werden können. Ohne vereinbarte Antwortzeitgrenzen ist die Erkennung und Behandlung von Fehlern nicht möglich. Konsistente Antwortzeiten stiften hierbei Vertrauen und fördern so weitere Interaktion.

- **Widerstandsfähig**
  Das System bleibt selbst bei Ausfällen von Hard- oder Software antwortbereit. Dies bezieht sich nicht nur auf hochkritische Anwendungen: jedes System, welches nicht widerstandsfähig ist, verliert durch einen Ausfall seine Antwortbereitschaft und damit seine Funktion. Widerstandsfähigkeit ist nur erreichbar durch Replizieren der Funktionalität, Eindämmung von Fehlern, Isolation von Komponenten sowie Delegieren von Verantwortung. So bleibt der Ausfall eines Teilsystems auf dieses begrenzt, andere Teilsysteme sind geschützt und in ihrer Funktion nicht behindert. Die Wiederherstellung des Normalzustandes wird einer übergeordneten Komponente übertragen, die durch die gesteuerte Replizierung der ihr untergeordneten Komponenten die geforderte Verfügbarkeit sicherstellt. All dies bedeutet, dass Nutzer eines auf diese Weise widerstandsfähigen Systems von der Last befreit sind, sich mit dessen Ausfällen auseinandersetzen zu müssen.

- **Elastisch**
  Das System bleibt auch unter sich ändernden Lastbedingungen antwortbereit. Auch hier bildet die Verteilung und Replizierung von Funktionalität die Grundlage, auf der das System auf Veränderungen reagiert. Bei Verminderung oder Erhöhung der Last werden automatisch die Replizierungsfaktoren und damit die genutzten Ressourcen angepasst. Um dies zu ermöglichen, darf das System keine Engpässe aufweisen, die den Gesamtdurchsatz vor Erreichen der geplanten Maximalauslegung einschränken. Ideal ist eine Architektur, die keine fixen Engpässe aufweist. In diesem Fall kann das bearbeitete Aufgabengebiet in unabhängige Teile zerlegt und auf beliebig viele Ressourcen verteilt werden. Reaktive Systeme unterstützen die Erfassung ihrer Auslastung zur Laufzeit, um automatisch regelnd eingreifen zu können. Dank ihrer Elastizität können sie auf Speziallösungen verzichten und mit handelsüblichen Komponenten implementiert werden.

- **Nachrichtenorientiert**
  Das System verwendet asynchrone Nachrichtenübermittlung zwischen seinen Komponenten zur Sicherstellung von deren Entkopplung und Isolation sowie zwecks Übermittlung von Fehlern an übergeordnete Komponenten. Die explizite Verwendung von Nachrichtenübermittlung führt zu einer ortsunabhängigen Formulierung des Programms und erlaubt die transparente Skalierung von Komponenten. Die

Überwachung von Nachrichtenpuffern ermöglicht einen kontinuierlichen Einblick in das Laufzeitverhalten des Systems – sowohl zur Diagnose als auch zur automatischen Ressourcensteuerung – sowie die Priorisierung und Kontrolle der Nachrichtenflüsse. Ortsunabhängigkeit bedeutet, dass Code und Semantik des Programms nicht davon abhängen, ob dessen Teile auf demselben Computer oder verteilt über ein Netzwerk ausgeführt werden. Nicht-blockierende nachrichtenorientierte Systeme erlauben eine effiziente Verwendung von Ressourcen, da Komponenten beim Ausbleiben von Nachrichten vollständig inaktiv bleiben können.

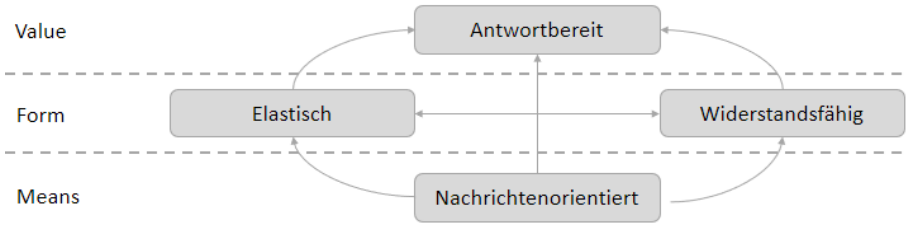


Abbildung 21: Charakteristiken eines reaktiven Systems; in Anlehnung an Bonér et al. [BFKT14]

Diese Eigenschaften werden unter dem Begriff «reaktiv» zusammengefasst. Computersysteme, die nach diesen Anforderungen entwickelt werden, erweisen sich nach Bonér et al. [BFKT14] als anpassungsfähiger, mit weniger starr gekoppelten Komponenten und in jeder Hinsicht skalierbarer, währenddem sie einfacher weiterzuentwickeln und zu verändern sind. Reaktive Systeme reagieren zuverlässiger und eleganter auf Fehler (Resilienz und Fehlertoleranz). Sie vermeiden so Ausfälle und können den bestehenden Anforderungen an Robustheit und Anpassungsfähigkeit moderner Systeme gerecht werden. Entsprechend sind auf Softwareebene asynchrone, eventgetriebene, reaktive Architekturmuster wie Events, Messaging und Publish/Subscribe zu fördern.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Antwortbereite Service-Architektur mit Antwortzeitgrenzen (Verfügbarkeit, Robustheit und Resilienz)
- Widerstandsfähigkeit mit Verteilung und Replizierung von Funktionalität (Robustheit und Resilienz)
- Elastizität und Parallelisierung mit Verteilung und Replizierung von Funktionalität (Robustheit und Resilienz, Wiederverwendbarkeit)
- Nachrichtenorientierte Service-Architektur mit den Eigenschaften asynchron, entkoppelt, isoliert und nicht blockierend (Robustheit und Resilienz)
- Event Store (Asynchrone Publish/Subscribe Messaging) (Lösungstechnologie)

3.5.7 NP4 - Selbstorganisation und Selbstähnlichkeit

Selbstorganisation (Autopoiesis) bezeichnet nach Maturala & Varela [MV87] die Fähigkeit eines Systems durch selbstregulierende und -lenkende Mechanismen die Systemstruktur aus den Prozessen ihrer eigenen Leistung heraus zu bestimmen, um den langfristigen Systembestand zu gewährleisten. Die aus dem System resultierende Struktur bzw. Ordnung weist nach Gabler [Gab06, S.17] durch das Prinzip der Selbstorganisation eine hohe Leistungsfähigkeit auf. Die Anforderungen der Selbstorganisation werden nach Andresen et al. [AGS05, S.74] von IS-Architekturen erfüllt, wenn sie in der Lage sind, ihre innere Struktur bzw. Architektur ganz oder teilweise selbst zu bestimmen. Ein Beispiel für ein autopoietisches System kann etwa ein Service sein, welcher bei zunehmender Last autonom und dynamisch Ressourcen hinzufügt, wie etwa ein FaaS-Service in der Cloud oder ein containerbasiertes System, welches sich automatisch replizieren und so horizontal skalieren lässt.

Selbstähnlichkeit ist nach Andresen et al. [AGS05, S.74] eine Eigenschaft, welche durch Zusammenlegen oder Aufteilen im Wesentlichen immer wieder gleiche Muster auf einer anderen Grössenskala ermöglicht. Selbstähnliche und selbstorganisierende Elemente führen nach Gnonau [Gro06, S.203] zu einem autopoietischen Systemverhalten, das sich positiv auf die Wandlungsfähigkeit von IS-Architekturen auswirkt. Als Beispiel für die Vorteile von Selbstähnlichkeit kann auch die leichtere Erlernbarkeit der Bedienung von Anwendungssystemen genannt werden, die auf unterschiedlichen Ebenen und Plattformen basieren, aber dennoch eine immer wiederkehrende Bedienphilosophie aufweisen, was Ross et al. [RSBJ17] etwa durch das Sicherstellen des allgemeinen Verständnisses und das Anwenden der architektonischen Prinzipien empfiehlt.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Selbstorganisation (Skalierbarkeit, Robustheit und Resilienz)
- Selbstähnlichkeit (Wissen über IS-Architektur (Fundament), Wiederverwendbarkeit)
- MSA (Lösungstechnologie)
- FaaS / Serverless Function (Lösungstechnologie)
- Container (Lösungstechnologie)

3.5.8 FP1 - Skalierbarkeit

Skalierbarkeit beschreibt nach Von Brauk & Neudert [VBN14] die Fähigkeit einer IT-Architekturkomponente, unter einer verändernden (meist steigenden) Last anpassbar zu sein und stellt einen Schlüsselfaktor für einen reibungslosen, nachhaltigen Systembetrieb

dar. Hierbei gilt es einerseits mit verändertem Datenvolumen (Durchsatz) und schwankenden Zugriffszahlen (Latenzzeit) elastisch umgehen zu können, wie es beispielsweise bei einem Buchungssystem der Fall sein kann.

Im Kontext dieser Arbeit wird Skalierbarkeit daher als wichtiges Wandlungsfähigkeits- und Nachhaltigkeitsparadigma definiert und entsprechend als fundamentales Prinzip der Zukunftsfähigkeit im Rahmen des Flexibilitätsbausteins kategorisiert. Eine Skalierung lässt sich grundsätzlich durch eine geeignete Software- und Hardwarearchitektur oder eine Kombination davon vertikal und horizontal durchführen.

Scale-Up (Vertikale Skalierung)

In vielen Szenarien lassen sich durch phyisches Hinzufügen oder virtuelles Freischalten von leistungsfähigeren Ressourcen eine vertikale Skalierung realisieren, wie es heute oft im Rahmen von Cloud-Lösungen angeboten wird. Dies hat keinen direkten Einfluss auf die Anwendungsarchitektur als solche und wird daher oft im Zusammenhang mit Legacy=Lösungen angewendet, welche nicht darauf optimiert sind, verteilt oder parallel betrieben zu werden.

Scale-Out (Horizontale Skalierung)

Die alternative Skalierungsstrategie stellt die horizontale Skalierung dar, welche durch hinzufügen von zusätzlichen Knoten (Parallelisierung) oder zusätzlichen Funktionskomponenten (Cluster) erreicht werden kann. Hierbei muss auf Applikationsarchitekturebene eine Parallelisierung unterstützt sein und es stellen sich Herausforderungen hinsichtlich Datenkonsistenz, Verfügbarkeit und Redundanz. Dies kann erfordern, dass ein zusätzlicher Abstraktions-Layer für die konsumierenden Applikationen und Dienste in Form eines Adapters dazwischen geschaltet werden muss, welcher beispielsweise die referentielle Integrität sicherstellt.

Ein klassisches Verfahren, um horizontale Skalierbarkeit zu gewährleisten, ist nach Von Brauk & Neudert [VBN14] die horizontale Schichtenbildung durch eine hierarchische Zerlegung der Geschäftsprozesse. Schichten können so unabhängig voneinander entwickelt, verteilt betrieben und als Ganzes separiert von den darüber liegenden Schichten skaliert werden.

Ein weitergehender Lösungsansatz ist das Architektur-Pattern einer serviceorientierten Architektur (SOA) respektive einer Microservice-Architektur (MSA) oder Serverless Functions. In einer MSA ist durch die Grundannahme, dass es sich um ein verteiltes System von Diensten handelt, eine klarere Trennung von Geschäftsprozessen und Ressourcen gewährleistet. Im Unterschied zur Schichtenarchitektur wird ein Geschäftsprozess in einer MSA bereits zu Beginn als Summe autonomer, funktional kohärenter Dienste gestaltet, die über Vermittler (Broker) lose miteinander gekoppelt sind und asynchron kommuni-

zieren [VBN14]. Jeder Dienst hat hierbei eine klare fachliche Zuständigkeit und besitzt die Datenhoheit über diese Domäne. Der Zugriff erfolgt über eine Schnittstelle, wobei die interne Struktur den anderen Diensten unbekannt ist. Kann ein Dienst seine festgelegten Antwortzeiten nicht einhalten, ist kein aufrufender Dienst direkt betroffen, sondern eingehende Anfragen können vom Vermittler zwischengespeichert und später verarbeitet werden (Store and Forward). Autonom, asynchron kommunizierende Dienste erlauben zudem die parallele Ausführung einer Aktion, was idealerweise die gesamte Ausführungsdauer eines Geschäftsprozesses reduziert.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Vertikale Skalierung (Entwicklungsfähigkeit, Flexibilität und Transformationsfähigkeit, Reaktions- und Handlungsfähigkeit, Wandlungsfähigkeit, Nachhaltigkeit)
- Cloud (Lösungstechnologie)
- Horizontale Skalierung (Entwicklungsfähigkeit, Flexibilität und Transformationsfähigkeit, Reaktions- und Handlungsfähigkeit, Wandlungsfähigkeit, Nachhaltigkeit)
- Schichtenbildung (Entwicklungsfähigkeit, Flexibilität und Transformationsfähigkeit, Reaktions- und Handlungsfähigkeit, Wandlungsfähigkeit, Nachhaltigkeit)
- Verteiltes System, Parallelisierung (Flexibilität und Transformationsfähigkeit, Reaktions- und Handlungsfähigkeit, Wandlungsfähigkeit, Nachhaltigkeit)
- MSA (Lösungstechnologie)
- FaaS / Serverless Function (Lösungstechnologie)
- Container (Lösungstechnologie)

3.5.9 FP2 - Wissen über die IS-Architektur

Das Wissen über die IS-Architektur bezeichnet nach Andresen et al. [AGS05, S75] sowohl das personengebundene, stillschweigende Wissen als auch das explizite Wissen. Letzteres liegt beispielsweise in Form von schriftlich fixierten Prozessen, Regeln, Richtlinien, Verantwortlichkeiten, Kommunikationskanälen, Modellen usw. vor. Stillschweigendes Wissen bezieht sich insbesondere auf die Fähigkeiten der Mitarbeiter, die mit der Gestaltung oder Nutzung der IS-Architektur befasst sind. Ihre Erfahrungen umfassen beispielsweise das Potenzial, neue Anwendungen zu integrieren, neue Prozesse abzubilden oder neue Wertschöpfungspartner in die Leistungserstellung einzubeziehen.

Im Kontext von Ross et al. [RWR06] propagierten Business Modularity-Architektur, welche auch eine Integration von externen Dienstleistern in den Geschäftsprozess erlaubt, ist es aber ebenso wichtig, dass diese Dienstleister Informationen über die für sie relevanten Schnittstellen, Prozesse, Informationsobjekte und Rollen zur Verfügung haben.

Dies kann beispielsweise anhand eines Business Interoperability Interfaces (BII) nach Ziemann [Zie10] sichergestellt werden.

Hinsichtlich der Zukunftsfähigkeit ermöglicht das Wissen über eine IS-Architektur folglich eine Bewertung des Innovationspotentials einer neuen Lösung und bedient somit eine Innovationsfähigkeit. Es gewährleistet aber auch eine Entscheidungsfähigkeit, sowie eine Reaktions- und Handlungsfähigkeit. Aus diesen Gründen wird Wissen als Paradigma kategorisiert, dass sich auf die fundamentale Prinzipien-Perspektive der Zukunftsfähigkeit bezieht.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Implizites und explizites Wissen über die IS-Architektur sicherstellen (Innovationsfähigkeit, Flexibilität und Entscheidungsfähigkeit, Reaktions- und Handlsungsfähigkeit)
- BII Interface nach Ziemann [Zie10, S.37] (Lösungstechnologie)

3.5.10 FP3 - Multimodale IS-Architektur

IS-Architekturen, die nach multimodalen Ansätzen gestaltet sind, sollen helfen die unterschiedlichen Anforderungen an die Veränderungsgeschwindigkeit einer Applikationslandschaft zu strukturieren und durch domänenspezifische Massnahmen besser beherrschbar zu machen. Ross et al. [RSBJ17] empfiehlt hierfür das «Digital Organization Design», welches die Anwendungslandschaft in die Domänen «Digital Services Platform» und «Operational Backbone» unterteilt, welche über die «Digital Linkages» verbunden werden (vgl. Abbildung 22)[13].

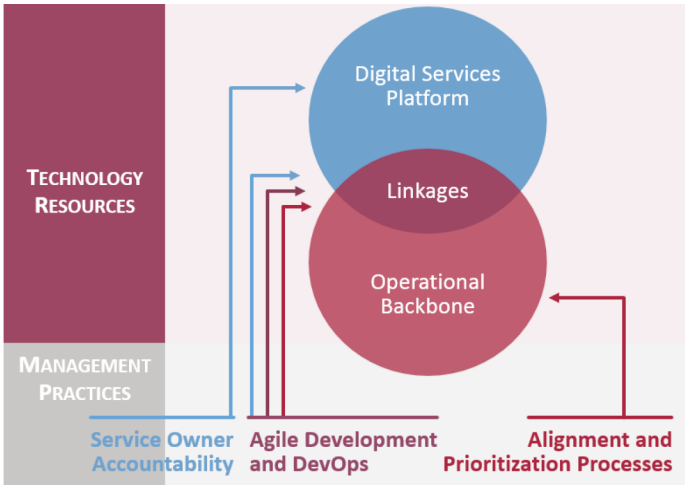


Abbildung 22: Ausschnitt aus dem Gestaltungsmodell digitaler Organisationen; Quelle: Ross et al. [RSBJ17]

[13] Siehe auch Kapitel 3.6 auf Seite 50 mit dem ISA-Pattern «Digital Organization Design».

Der Operational Backbone enthält hierbei die klassischen ISA-Elemente, welche etwa Zuverlässigkeits- und Integritäts-Anforderungen gerecht werden. Die Digital Services Platform enthält Komponenten, die agile und innovative Lösungen, eine schnelle Time-to-market sowie eine Integration von Partner Services ermöglichen, wobei die Digital Linkages diese beiden Bereiche koppeln. Dies kann nach Ross et al. [RSBJ17, S.11] anhand von spezifischen Managementmethoden für jeden der drei Bereiche unterstützt werden:

- Operational Backbone:
  Traditionelle Alignment-Methoden mit Roadmaps, Priorisierungen, funktionsübergreifenden Architektur-Reviews und etablierten Architektur-Richtlinien.

- Digial Service Platform:
  Service Design und -Beurteilung, sicherstellen einer Service-Ownership zur Gewährleistung von Kosten und Qualität, Evaluation von Ökosystem-Partnern.

- Digital Linkages:
  Agile und iterative Methodologien in Kombination mit funktionsübergreifenden Teams und einem Outside-in Design Konzept (Kundensicht) und Minimum Viable Product Vorgehensweisen.

Einen in die gleiche Richtung zielenden Ansatz für eine multimodal ausgestaltete IS-Architektur zeigt Gaughan [Gau18, S.11] anhand eines Pace Layer-Modells auf. Das Pace Layer-Modell soll nach Gaughan helfen, die oftmals vorherrschende «One Size Fits All»-Mentalität zu durchbrechen. Die Ebenenunterteilung des Pace Layer-Modells wird aufgrund des erforderlichen Anwendungsportfolio-Veränderungstempos segmentiert, welches wiederum die Anforderungen der Geschäftsfähigkeiten geprägt ist.

Das Pace Layer-Modell dient hierbei als Strategie-Rahmen und als Kommunikationsinstrument zur Unterstützung der IT bei der Bewältigung der Herausforderungen im Zusammenhang mit dem Wandel der Unternehmenskultur, kurz- und langfristiger Priorisierungen sowie der modernisierung von monolythischen Legacy-Umgebungen [Gau18, S.3].

Das Pace Layer-Konzept wird in drei Ebenen unterteilt (vgl. Abbildung 23 auf Seite 48), wobei alle Anwendungen der Anwendungslandschaft aufgrund ihrer Änderungsanforderungen und Charakteristik zu einer entsprechenden Pace Layer-Ebene zugewiesen werden[14]:

- Systems of Record-Ebene:
  Der Fokus liegt auf Betriebseffizienz, Standardisierung der Prozesse, Senkung der Komplexität und gut geplanten, konsistenten Änderungszyklen.

[14] Ein Beispiel hierfür im Anhang B ersichtlich

- Systems of Differentiation-Ebene:
  Der Fokus liegt auf einer Outside-in Sicht (Kundensicht), kundenorientiertem Denken, einer beschleunigten Veränderungsrate und einzigartigen Ansätzen zur nachhaltigen Differenzierung.

- Systems of Innovation-Ebene:
  Experimentieren mit einer ausfallsicheren Mentalität, aber auch der Erkenntnis, dass Erfolg erhöhte Investitionen bedingt.

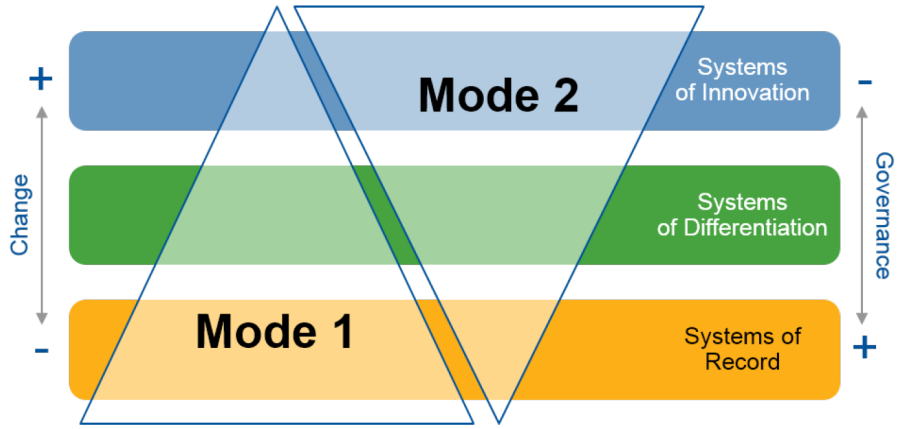


Abbildung 23: Zusammenhang zwischen multimodalen Ansprüchen und den verschiedenen Funktionsprinzipien; Quelle: Gaughan [Gau18]

Im Allgemeinen eignen sich die zugewiesenen «Systems of Record» Anwendungen gut für ein «traditionelles» Modus 1-Paradigma, bei welchem Organisationen ein sehr zielgerichtetes Veränderungsmanagement haben und die Planungshorizonte und -anforderungen für Veränderung gut verstanden werden. Anwendungen, welche der Differenzierungs- und Innovations-Ebene zugewiesen wurden, erfordern normalerweise eine beschleunigte Veränderungsrate und haben oft Änderungsanforderungen, die nicht klar definiert sind. Dies sind oft diejenigen Anwendungen, welche angesichts der Unsicherheit, die sich in Folge von geänderten Geschäftsbedingungen ergeben, am besten geeignet sind, um in einem Arbeitsstil nach Modus 2 verwaltet zu werden.

Die Modus-Zuordnung von Anwendungen einer Ebene ist im Allgemeinen konsistent, aber wie unter Abbildung 23 ersichtlich, ist nicht immer alles schwarz auf weiss, da Geschäftsbedingungen vorherrschen können, die für ein «System of Record» einen Arbeitsstil nach Modus 2 vorsehen. Modus 2 eignet sich hierbei grundsätzlich für Vorhaben mit Erkundungs- oder experimentellem Charakter [Gau18, S.10]:

- Erkundung:
  Ein Erforschungs-Vorgehen wird verfolgt, wenn die Organisation weiss, was sie erreichen will und wenn sie die Notwendigkeit versteht, dass eine Anwendung schnell

weiterentwickelt werden muss. Dies ist oft der Fall, wenn Anwendungen differenzierende Fähigkeiten ermöglichen sollen oder die Notwendigkeit besteht, dass bestehende Wettbewerbsvorteile erhalten bleiben.

- Experimentell:
  Ein experimentelles Vorgehen wird angewendet, wenn eine Organisation auf der Suche nach neuen Ideen ist (Innovation), ohne sie vollständig zu verstehen und ohne zu wissen, ob es sich lohnt oder nicht. Dieses Vorgehen ist geprägt von geringen Investitionen, kurzen Zykluszeiten und einer kontinuierlichen Ergebnis-Analyse, ob weitere Investitionen gerechtfertigt sind.

Unterschiedliche Veränderungsraten der Geschäftsfähigkeiten erfordern es möglicherweise, dass eine Anwendungsarchitektur modularer gestaltet wird. Dies führt in der Regel zu mehreren essentiellen Fragen [Gau18, S.10]:

- Wie lassen sich monolithische Anwendungen, die auf allen Pace Layers relevant sind, integrieren respektive in Module aufteilen, um die geforderten Anpassungsraten zu unterstützen?
  Allenfalls müssen wichtige Services von monolithische Anwendungen ausgelagert werden.

- Was sind die kritischen Services, die im «System of Record» existieren und von differenzierenden und innovativen Anwendungen konsumiert werden?
  Allenfalls müssen Services sowohl für den internen als auch für den externen Gebrauch über APIs bereitgestellt werden.

- Gibt es Möglichkeiten zur Konsolidierung?
  Oft gibt es mehrere Anwendungen, die die gleichen grundlegenden Fähigkeiten unterstützen. Bei dieser Gelegenheit könnte die Anzahl der redundanten Anwendungen und damit die Komplexität reduziert werden.

- Stehen die Anwendungsinvestitionen im Einklang mit den Unternehmenszielen?
  Organisationen können im Rahmen dieser Fragestellung oft feststellen, dass sie verhältnismässig viel mehr in «Systems of Record» investieren als sie für angemessen erachten. Eine Roadmap zur Reduzierung der Kosten und der Komplexität der «Systems of Record» kann hierbei helfen, eine Verlagerung der Ausgaben in Richtung Differenzierung und Innovation stattfinden zu lassen.

Diese Fragestellungen zu klären, kann nach Gaughan [Gau18, S.10] dazu beitragen, die richtigen Prioritäten für die Anwendungsorganisation zu ermitteln, um sicherzustellen, dass sich die Investitionen auf die Geschäftsstrategie auswirken. So kann das Pace Layer-Modell helfen, einige der Herausforderungen bei der Einführung einer multimodalen Architektur zu bewältigen, da der Pace Layer dazu verwendet werden kann, eine Verbindung

von der Geschäftsstrategie über die Geschäftsfähigkeiten hin zu den Anwendungen herzustellen. Dabei lässt sich möglicherweise die Priorität einiger grundlegender Anpassungen erhöhen, die zur Unterstützung des digitalen Geschäfts notwendig sind.

Potentielle Architektur-Patterns und Lösungstechnologien:

- Pace Layer-Modell nach Gaughan [Gau18] (Innovationsfähigkeit, Flexibilität und Entscheidungsfähigkeit, Reaktions- und Handlsungsfähigkeit)
- Digital Organisation Design nach Ross et al. [RSBJ17] (Innovationsfähigkeit, Flexibilität und Entscheidungsfähigkeit, Reaktions- und Handlsungsfähigkeit)

3.6 ISA-Patterns

Im vorhergehenden Abschnitt wurden verschiedene potentielle Lösungen, respektive Muster und Schemata (sog. Patterns) identifiziert. In diesem Abschnitt gilt es nun diese Lösungs-Patterns, als mögliche Teil-Lösung zur Problemstellung der zukunftsfähigen IS-Architektur zu beschreiben und zu bewerten. Diese Patterns beleuchten hierbei das im Anschluss herzuleitende Artefakt aus verschiedenen Perspektiven. Patterns haben aber auch immer Stärken und Schwächen. Zur Kompensation der Schwächen sollten sie daher möglichst mit anderen Patterns kombinierbar sein. Durch eine geeignete Kombination soll so im nächsten Abschnitt ein optimales ISA-Zielbild in Form eines Referenzarchitektur-Artefaktes hergeleitet werden.

Da sich ISA-Patterns in der Regel nach verschiedenen Ebenen, respektive Granularitätsstufen ausrichten, wurde im Rahmen der nachfolgenden Bewertungen jeweils ein Indikator-Symbol beigefügt, welches eine subjektive Kategorisierung über die «Flughöhe» des Lösungs-Patterns gibt. Der Indikator (vgl. Abbildung 24) lässt sich aufgrund der Charakteristiken der Patterns grob in folgende Ebenen gliedern: Hoch (Konzeptionelle Architekturübersicht); Mittel (Konkretere Skizzierung von Lösungs-Ansätzen); Tief (Umschreiben von spezifischen Funktionen und Outputs o.ä.).

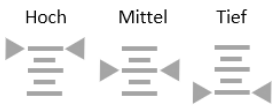


Abbildung 24: Architekturebenen-Indikator

3.6.1 Pattern A - Digital Organisation Design

Personalisierte Kundenbindung und sogenannte integrierte digitalisierte Lösungen stehen nach Ross et al. [RSBJ17] direkt in Zusammenhang mit der Marktperformance einer Unternehmung. Je enger die Kundenbindung eines Unternehmens und je integrierter dessen digitalisierte Lösungen sind, desto höher sind Profitabilität, Effizienz und Kundenzufriedenheit im Vergleich zu den Branchen-Peers.

In einem Markt, in dem zunehmend personalisierte Produkte sowie kontextspezifische und zeitkritische Werbung im Vordergrund stehen, werden digitalisierte Lösungen nach Zealley [Zea18] als technologischer Enabler benötigt. Immer wichtiger hierfür werden angereicherte Daten, um tiefere Kundenkenntnisse für bessere Kundenangebote zu erlangen. Hierdurch lassen sich differenzierte, personalisierte und wettbewerbsfähige Kundenerlebnisse über alle Kanäle schaffen. Eine Integration von Partner-Lösungen erlaubt es so eigene Dienstleistungen und Produkte schneller an den Markt zu bringen.

Wichtige Faktoren nach Ross et al. [RSBJ17] sind demnach:

1. Modulare Lösungskonzepte
2. Geschwindigkeit und Agilität
3. Daten und Prozess-Integration (Bsp. Single Source of Truth)

Um eine agile und innovative Unternehmung zu fördern, werden nach Ross et al. [RSBJ17, S.3f] drei sogenannte Schlüssel-Technologieressourcen gebraucht; Erstens ein starkes betriebliches Rückgrat (Operational Backbone), das automatisierte Transaktionsverarbeitung anbietet und Einsicht in Master- und Transaktionsdaten ermöglicht. Zweitens wird eine Digital Service Platform, welche wiederverwendbare Business-, Technologie- und Datenkomponenten ermöglicht, empfohlen. Drittens wird durch einen Kopplungs-Layer (Digital Linkage) der Brückenschlag zwischen den neuen digitalen Services und den Daten- und Infrastruktur-Services, welche in die Operational Backbone eingebettet sind, unterstützt (vgl. Abbildung 25 auf Seite 52).

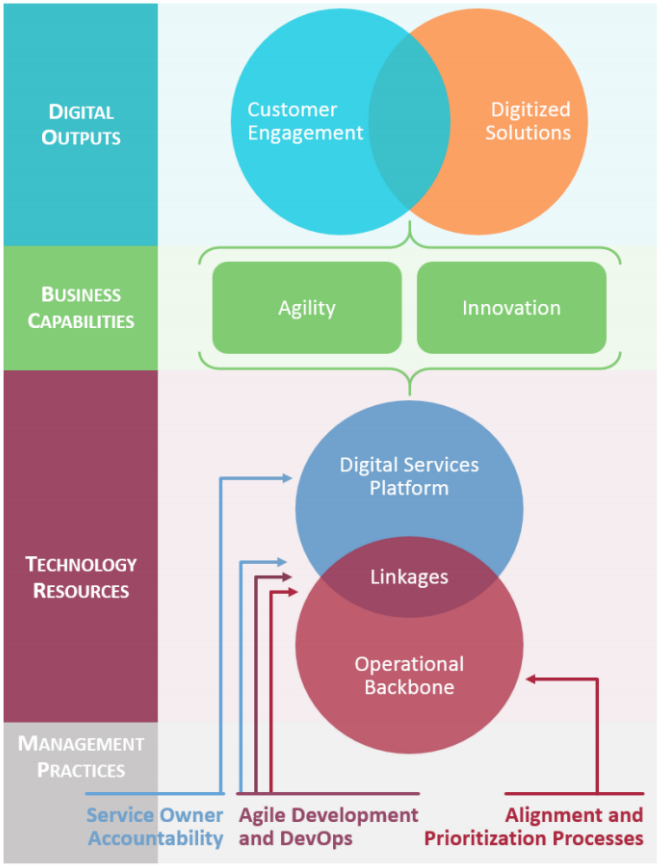


Abbildung 25: Gestaltungsmodell digitaler Organisationen; Quelle: Ross et al. [RSBJ17, S.4]

Als optimales Umfeld für die Schlüssel-Technologieressourcen sollen Unternehmen anhand ihrer Managementmethoden sicherstellen, dass Governance-Prozesse die Einhaltung architektonischer Prinzipien sicherstellen. Ebenso sollten organisatorische Management-Richtlinien definiert werden, damit Service Owner ihre Verantwortung – analog zu Geschäftsfunktions- oder Produktlinien-Owner – wahrnehmen. Weiter sollten Anpassungen und Weiterentwicklungen durch funktionsübergreifende, agile Herangehensweisen wie etwa agile Entwicklungsmethoden und DevOps begegnet werden. Durch gesteigerte Agilität und Innovationsfähigkeit lassen sich so nach Ross et al. [RSBJ17] integrierte, digitalisierte Lösungen und eine personalisierte Kundenbindung erzielen.

Nachfolgend werden die verschiedenen IS-Architektur relevanten Bereiche einer «Digital Organization» nach Ross et al. [RSBJ17, S.11ff] erläutert und gleichzeitig aufgezeigt, anhand von welchen Massnahmen diese Bereiche einen multimodalen Ansatz unterstützen können, um so die Agilität und Innovationsfähigkeit einer Organisation zu fördern:

Operational Backbone

Der Operational Backbone ist das zentrale Rückgrat der Anwendungslandschaft einer Organisation. Er unterstützt die von Unternehmen benötige Operational Excellence durch Sicherstellung einer durchgängigen Verarbeitung von Transaktions- und Stammdaten, durch Zugriffsverwaltung auf Daten, durch Skalierung sowie durch Sicherstellung der Verfügbarkeit. Der Operational Backbone gewährleistet somit, dass Organisationen ihr aktuelles Geschäft zuverlässig betreiben können.

Empfohlene Ausprägungen:

- Automatisiert repetitive Geschäftsprozesse
- Gewährleistet und benutzt die Single Source of Truth
- Unterstützt eine durchgängige Datenverarbeitung
- Ermöglicht die Sicht auf Daten
- Ermöglicht einen zuverlässigen, stabilen, sicheren Betrieb
- Ist modular aufgebaut

Digital Services Platform

Die Digital Services Platform nutzt die Vorteile von neuen Technologien und Partnerschaften, um neue Funktionalitäten rasch und einfach einführen zu können. Sie dient folglich als architektonisches Fundament für die geforderte Flexibilität und das Lernen und Weiterentwickeln als Organisation. Zusammen mit dem Operational Backbone ermöglicht die Digital Services Platform also Agilität und Innovation in einem zuverlässigen, sicheren und skalierbaren Umfeld.

Empfohlene Ausprägungen:

- Ermöglicht Zugriff auf ein Repository für wiederverwendbare Geschäfts- und Technologie-Services
- Unterstützt die Anbindung von Partner Services
- Bietet ein API für interne und externe Partner
- Ermöglicht Zugriff auf ein Repository für die Analyse von Sensordaten und Social Media-Daten
- Nutzt Cloud-Plattformen (PaaS)
- Nutzt Open Source Software
- Ermöglicht A/B Experimente

Digital Linkages

Digitale Verknüpfungen (Digital Linkages) stellen den Datentransfer zwischen den beiden Domänen «Digital Service Platform» und «Operational Backbone» sicher.

Empfohlene Ausprägungen:

- Erlaubt Digital Services den Zugriff auf Kunden- und Produkt-Stammdaten
- Verknüpft Digital Services mit Transaktionsverarbeitungssystemen

Kombination mit dem Phase Layer-Modell

An dieser Stelle macht es Sinn, die Erkenntnisse bezüglich dem Phase Layer-Modell von Gaughan [Gau18] aus dem vorhergehenden Abschnitt aufzugreifen. Gaughan sieht vor, dass hinsichtlich den multimodalen Anforderungen eine dreistufige Unterteilung der Anwendungsservices vorgenommen wird, um den unterschiedlichen Anforderungen bestmöglich gerecht zu werden. Während sich die Phase Layer-Domäne der «Systems of Record» nach Gaughan mit dem Operational Backbone-Domäne nach Ross et al. [RSBJ17] weitgehend deckt, gibt es zwischen den «Systems of Differentiation» und den «Systems of Innovation» eine zusätzliche Unterscheidung. Diese lässt sich aus Charakteristika-Sicht aber gut mit dem Digital Organizations-Modell von Ross et al. im Bereich der Digital Service Platform ergänzen. Die Ergänzungen sind hierbei eher auf den methodischen Vorgehensmustern (Finanzierung, Deployment-Anforderungen, sowie Entwicklungs- und Testing-Methodiken) zu finden als im Bereich der IS-Architektur selbst.

Ein grundsätzlich wichtiger Aspekt ist zudem, dass Vorkehrungen getroffen werden, um Module der Digital Service Platform bei Bedarf in den Operational Backbone überführen zu können (vgl. Ross [RWR06]). Dies für den Fall, dass über die Zeit erhöhte Anforderungen an den Service gestellt werden.

Bewertung des Patterns

Dieses Pattern definiert einen sinnvollen Rahmen für eine kontextuelle Zuordnung der ISA-Komponenten, wie etwa Digital Service Platform (Agilitätsanspruch) oder Operational Backbone (Nahhaltigkeitsanspruch). Das Pattern bedindet sich hinsichtlich des Architektur-Levels eher auf einer übergeordneten, konzeptionellen Architekturübersichtsebene und wird diesbezüglich daher als «hoch» eingestuft. In der nachfolgenden Tabelle 2 auf Seite 55 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern A - Digital Organization Design		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Im Digital Service Platform-Bereich und im Backbone-Bereich wird eine Modularität (MSA) angestrebt. Interoperabilität zwischen internen und externen Partnern gilt als zentrales Kredo. Insgesamt ein vielversprechender, multimodaler Lösungsansatz, der auch traditionelle Systeme berücksichtigt und so das Beste aus unterschiedlichen Welten vereint.	Modularität Interoperabilität Multimodale IS-Architektur
Opportunities	Highlevel-Architektur, welche die verschiedenen Anforderungsbereiche übersichtlich gliedert und auch verständlich kommuniziert werden kann. Skalierbarkeit wird gefördert durch einen angestrebten PaaS Cloud-Ansatz. Durch eine Überführung in den Operational Backbone wird die Verfügbarkeit unterschiedlichen Ansprüchen gerecht. Das Pattern erlaubt Anknüpfungspunkte zu weiteren Architekturpatterns.	Verfügbarkeit Skalierbarkeit Wissen über IS-Architektur
Weaknesses	Die Mobilität respektive Entkopplung wird nur indirekt via MSA begünstigt. Das Pattern bietet wenige Anknüpfungspunkte bezüglich Reaktivität, Zustandslosigkeit, Selbstorganisation und Selbstähnlichkeit (allenfalls via MSA).	Mobilität Reaktivität Zustandslosigkeit Selbstorg. und -ähnlichkeit
Threats	Allgemein: Das Pattern muss mit weiteren Patterns kombiniert werden, um die Lücken bezüglich tieferliegenden Ebenen zu schliessen.	-

Tabelle 2: Bewertung Pattern A - Digital Organization Design

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstellen (vgl. Abbildung 26 auf Seite 56) und vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

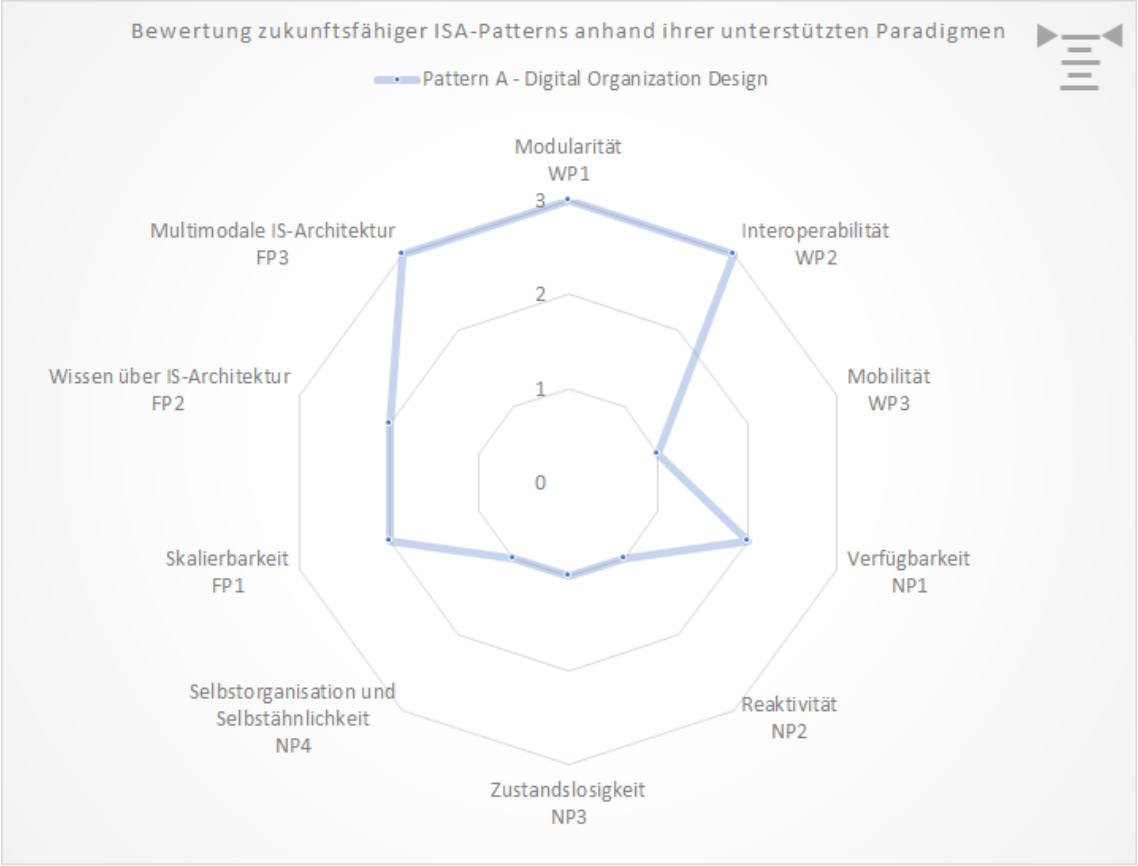


Abbildung 26: Netzdiagramm zur Bewertung von Pattern A - Digital Organization Design

3.6.2 Pattern B - Architecture of Interoperable Information Systems

Die Architecture of Interoperable Information Systems (AIOS) nach Ziemann [Zie10] ist eine Referenzarchitektur für die Entwicklung interoperabler Informationssysteme (IS). Wenn eine Organisation automatisierte Geschäftsprozesse mit anderen Organisationen betreiben will, müssen ihre IT-Systeme zusammenarbeiten können, d.h. interoperabel sein [Zie10, S.25ff]. Das AIOS stellt einen allgemeinen Bauplan für diese Organisationen dar, um durch systematische Anpassung und Erweiterung ihre IS interoperabel zu machen. Es beschreibt generisch die verschiedenen Ebenen, Ansichten, Beziehungen und technischen Mittel, die für den effizienten Aufbau interoperabler IS erforderlich sind. Dazu werden Konzepte aus den Bereichen SOA, organisationsübergreifende Geschäfte und Geschäftsprozessmodellierung kombiniert.

Definition

Ähnlich wie die Automatisierung von Prozessen innerhalb von Organisationen ist die Automatisierung von organisationsübergreifenden Geschäftsprozessen ein wichtiger Trend. Dabei streben kollaborierende Organisationen eher eine lose Kopplung ihrer IS an als eine enge Integration: Die kollaborierenden IS sollen zusammenarbeiten können, aber so

viel Unabhängigkeit wie möglich behalten [Zie10, S.27]. Dieses Merkmal wird auch als Interoperabilität oder im Kontext kollaborierender Organisationen als Business Interoperabilität bezeichnet, d.h. die Fähigkeit autonomer Organisationen, einen kollaborativen Geschäftsprozess zwischen ihnen durchzuführen [Zie10, S.26]. Eine Architektur interoperabler IS kann daher als systematischer, modellbasierter Bauplan eines organisationsübergreifenden IS definiert werden, der es Organisationen ermöglicht, einen kollaborativen Geschäftsprozess zwischen ihnen durchzuführen [Zie10, S.7].

Hintergrund und Anwendung

Die Hauptelemente des AIOS umfassen nach Ziemann [Zie10]:

1. Eine Beschreibung der verschiedenen Datentypen im interoperablen IS sowie deren Beziehungen. Dies wird auch der *statische Teil* oder die Struktur der Architektur genannt. Hier wird festgehalten, welche Informationselemente (z.B. Beschreibungen von Nachrichten, Austauschsequenzen, Rollen und Services) den Kooperationspartnern zur Verfügung gestellt werden müssen und wie diese optimal mit internen Elementen korrelieren können.
2. Eine Beschreibung verschiedener Vorgehensweisen zur Implementierung oder Anpassung interoperabler IS. Dies wird auch der *dynamische Teil* der Architektur genannt. Hier wird festgehalten, wie die oben genannten Informationselemente iterativ entwickelt werden können.
3. Ein Konzept für die technischen Komponenten zur Umsetzung der Architektur, z.B. Design-Tools, interne und extern sichtbare Repositories. Ein Element hierfür ist ein «BII-Repository», in welchem jede Organisation die Inhalte der Business Interoperability Interfaces (BII) für Kooperationspartner veröffentlicht. Da es externe Sichten auf IS-Elemente umfasst, bietet es Veröffentlichungs- und Entdeckungsfunktionalitäten, wie sie in einer serviceorientierten Architektur benötigt werden: Im BII werden die extern relevanten Prozesse, Services, Organisationsstrukturen etc. auf verschiedenen Ebenen der technischen Granularität beschrieben, so dass andere Organisationen auch nach Elementen auf Geschäftsebene und nicht nur nach technischen Artefakten suchen können. Hierbei werden, anders als beim traditionellen SOA-Ansatz, statt eines zentralen Service-Verzeichnisses verschiedene partnerspezifische Repositories implementiert.

Struktur

Der statische Teil der Architektur baut nach Ziemann [Zie10] auf drei orthogonalen Achsen auf (vgl. Abbildung 27 auf Seite 59):

Kollaborationssicht

Ähnlich wie bei privaten, öffentlichen und globalen Sichten, wie sie aus der Geschäftsprozess- und Workflow-Modellierung bekannt sind, werden im AIOS entsprechende private, öffentliche und globale Sichten auf IS-Elemente bereitgestellt.

1. Die private Sicht umfasst die nur intern sichtbaren Elemente des IS.
2. Die öffentliche Sicht fungiert als Schnittstelle zu den internen, privaten Systemelementen; sie schützt interne Systeme und ermöglicht Interoperabilität ohne wesentliche Änderungen an den internen Systemen. Diese öffentliche Sicht beschreibt die Grenzen des IS einer Organisation zu ihren Kooperationspartnern und verbindet interne und externe IS, wodurch auch die Inhalte des BII einer Organisation bereitgestellt werden.
3. Die globale Sicht kann verwendet werden, um die öffentlichen Ansichten verschiedener Systeme zu korrelieren und zu verbinden.

Unternehmensdimensionen

Um Geschäftsprozesse umfassend zu beschreiben, bietet diese Achse klare Sichten auf Prozesse, Funktionen, Daten und Organisationselemente.

1. In der organisatorischen Dimension werden Rollen, Einheiten und andere für die Zusammenarbeit relevante Organisationselemente beschrieben und mit internen Elementen verknüpft. So wird z.B. sichergestellt, dass die Kooperationspartner ein gemeinsames Verständnis der interagierenden Rollen haben.
2. In der Datendimension werden die in der Kollaborationssicht verwendeten Dokumentarten definiert und mit den intern verwendeten Dokumentarten verknüpft.
3. In der Funktionsdimension werden die in der Kollaborationssicht angebotenen Business Functions und Services beschrieben.
4. In der Prozessdimension werden die Prozesse beschrieben, die jede Organisation anbietet, sowie die Beziehung dieser öffentlichen Prozesse zu den angrenzenden Prozessen der Partnerorganisationen.

So werden in Kombination mit der Achse Kollaborationssicht private, öffentliche und globale Sichten auf Prozesse, Funktionen, Daten und organisatorische Rollen bereitgestellt.

Ebenen der technischen Granularität

Die Beschreibung von Systemelementen auf verschiedenen Ebenen der technischen Granularität unterstützt eine systematische Entwicklung von kollaborativen IS, beginnend bei der Definition der Geschäftsanforderungen bis hin zur Code-Ebene. Neben dem Konstruktionsaspekt wird dabei auch eine mehrdimensionale Interoperabilitätsbeschreibung

bereitgestellt, die die Synchronisation kollaborierender Systeme auf jeder Ebene erleichtert. Hierbei werden drei Ebenen verwendet:

1. Geschäftsebene: Hier werden die zu automatisierenden Prozesse von einer technikunabhängigen Ebene aus beschrieben. Im GFbUA nach Spichiger & Noser [SN] wird diese Ebene als RIM-Ebene bezeichnet, hier wird der Begriff CIM[15] verwendet.
2. Technische Ebene: Hier wird das IT-Konzept beschrieben. So werden z.B. die Modelle der ersten Ebene technisch angereichert, statt betriebswirtschaftlicher Funktionen werden nun Komponenten beschrieben, jedoch noch auf einer grobkörnigen, konzeptionellen Ebene. Da die Modelle auf der zweiten Ebene die Grundlage für eine automatisierte Generierung von ausführbarem Code darstellen, müssen sie eventuell weiter an die Einschränkungen der Implementierungsebene angepasst werden. Im GFbUA nach Spichiger & Noser [SN] wird diese Ebene als RSM-Ebene bezeichnet, hier wird der Begriff PIM[16] verwendet.
3. Ausführungsebene: Hier sind die Modelle maschineninterpretierbar und können zur Laufzeit bei der Ausführung von Prozessen verwendet werden. Im GFbUA nach Spichiger & Noser [SN] wird diese Ebene als Ressourcen-Ebene bezeichnet, hier wird der Begriff PSM[17] verwendet.

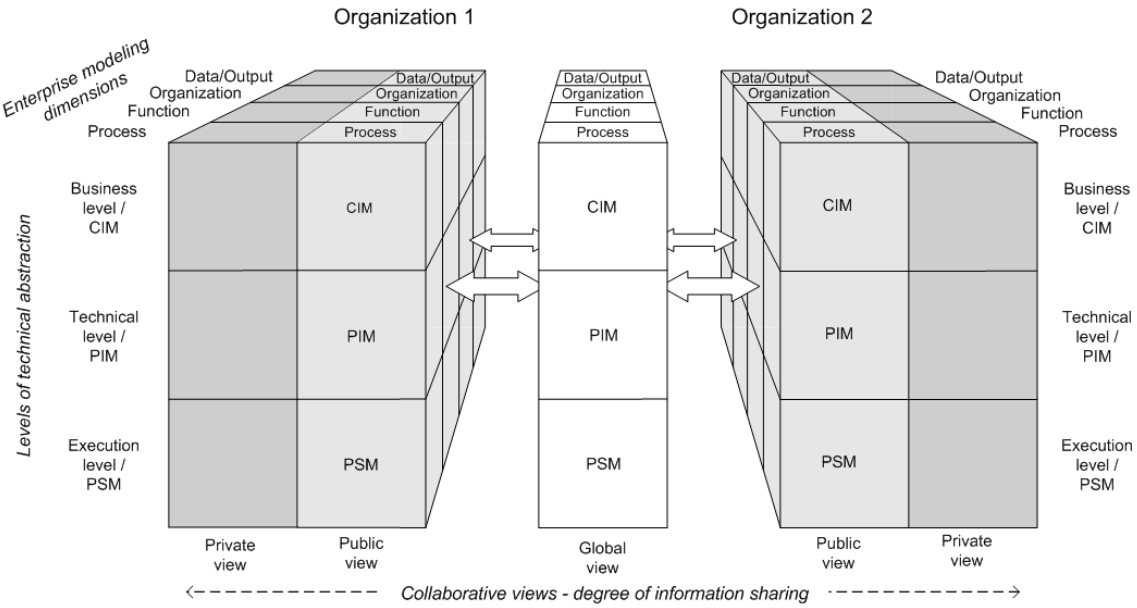


Abbildung 27: Architecture of Interoperable Information Systems; Quelle: Ziemann et al. [Zie10]

[15] CIM: Computation Independent Model, nach der Model Driven Architecture (MDA) [Tru06]
[16] PIM: Platform Independent Model, nach der Model Driven Architecture (MDA) [Tru06]
[17] PSM: Platform Specific Model, nach der Model Driven Architecture (MDA) [Tru06]

Bewertung des Patterns

Das vorliegende Pattern erweitert eine ISA optimal im Rahmen der Interoperabilität zwischen verschiedenen Unternehmen, kann aber bei Bedarf durchaus auch innerhalb (grösserer) Organisationen sinnvoll eingesetzt werden. Es deckt hinsichtlich dem Architektur-Level ein breites Spektrum ab und wird diesbezüglich daher als «mittel» eingestuft. In der nachfolgenden Tabelle 2 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern A - Digital Organization Design		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Das Pattern bedient die Interoperabilitätsproblematik optimal. Durch klar dokumentierte Interfaces, sowohl aus technischer Sicht als auch aus Geschäftssicht, wird das Wissen über die IS-Architektur zudem verbessert.	Interoperabilität Wissen über IS-Architektur
Opportunities	Anhand des SOA-Bezuges der Referenzarchitektur wird eine Modularisierung direkt gefördert. Die Stabilität respektive Verfügbarkeit der Services wird durch klar dokumentierte Schnittstellen verbessert.	Modularität Verfügbarkeit
Weaknesses	Sehr viele der Zukunftsfähigkeits-Paradigmen wie etwa Entkopplung, Reaktivität, Zustandslosigkeit, Selbstorganisation und Selbstähnlichkeit, multimodale IS-Architektur werden nur indirekt über die SOA-Anlehnung berücksichtigt.	Mobilität Reaktivität Zustandslosigkeit Selbstorg. und -ähnlichkeit Skalierbarkeit Multimodale IS-Architektur
Threats	Allgemein: Das Pattern ist sehr spezifisch auf die Lösung der Interoperabilitätsproblematiken ausgelegt und berücksichtigt die anderen Zukunftsfähigkeits-Paradigmen meistens nur indirekt. Es muss daher unbedingt mit weiteren Patterns kombiniert werden, um die Lücken zu schliessen.	-

Tabelle 3: Bewertung Pattern B - Architecture of Interoperable Information Systems

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstel-

len (vgl. Abbildung 28 auf Seite 61) und vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

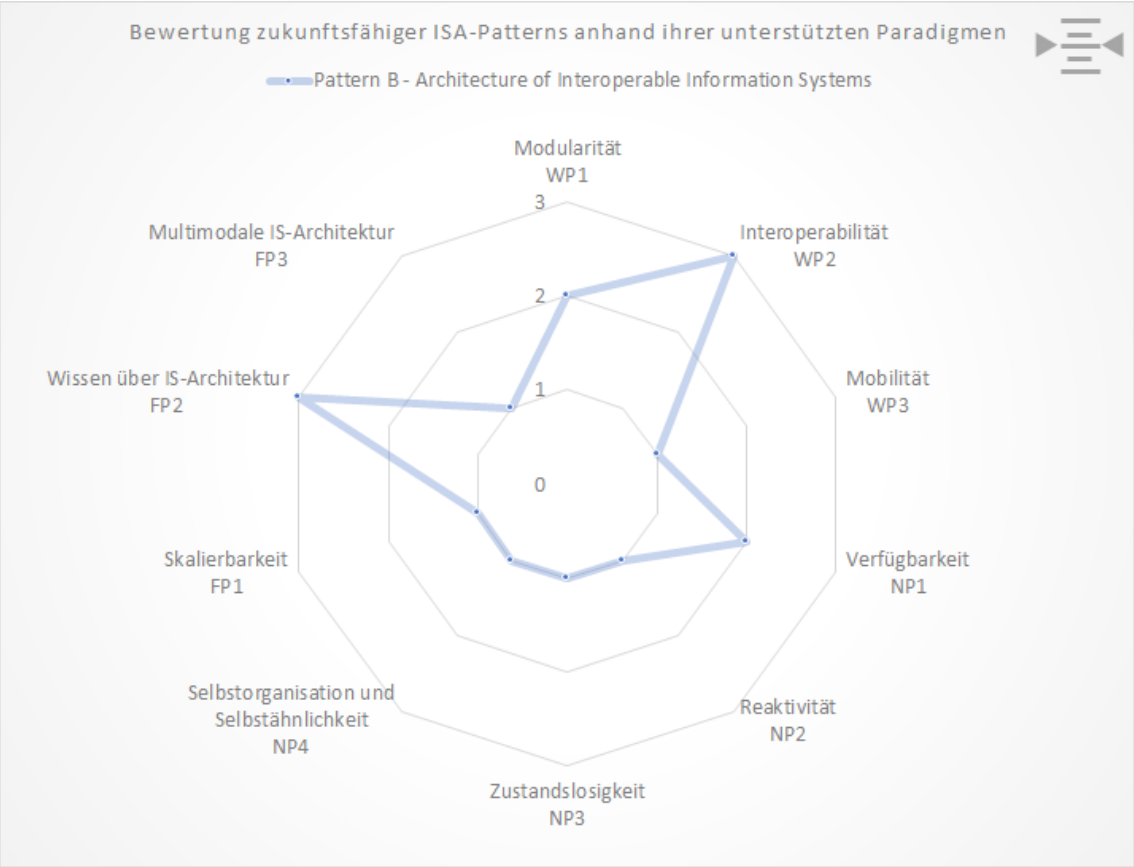


Abbildung 28: Netzdiagramm zur Bewertung von Pattern B - Architecture of Interoperable Information Systems

3.6.3 Pattern C - Microservice-Architektur

Microservices sind nach Hert [Her16] ein Architekturstil für die Erstellung von skalierbaren und flexiblen Applikationen. Hierbei wird der Ansatz verfolgt, komplexe Applikationen mit Hilfe von feingranularen Bausteinen, den sogenannten Microservices, aufzubauen. Die Funktionalität der einzelnen Microservices werden via APIs oder Message/Event-Bus für andere Microservices und Clients zugänglich gemacht. Jeder Microservice funktioniert für sich autonom und implementiert eine sehr spezifische Funktion. Unter Abbildung 29 auf Seite 62 ist eine Microservice-Architektur (MSA) schematisch abgebildet. Sie zeigt unter anderem die Microservice-Landschaft, welche über einen Message/Event-Bus kommuniziert und über ein API-Gateway als Reverse Proxy-System von den Clients angesprochen werden kann.

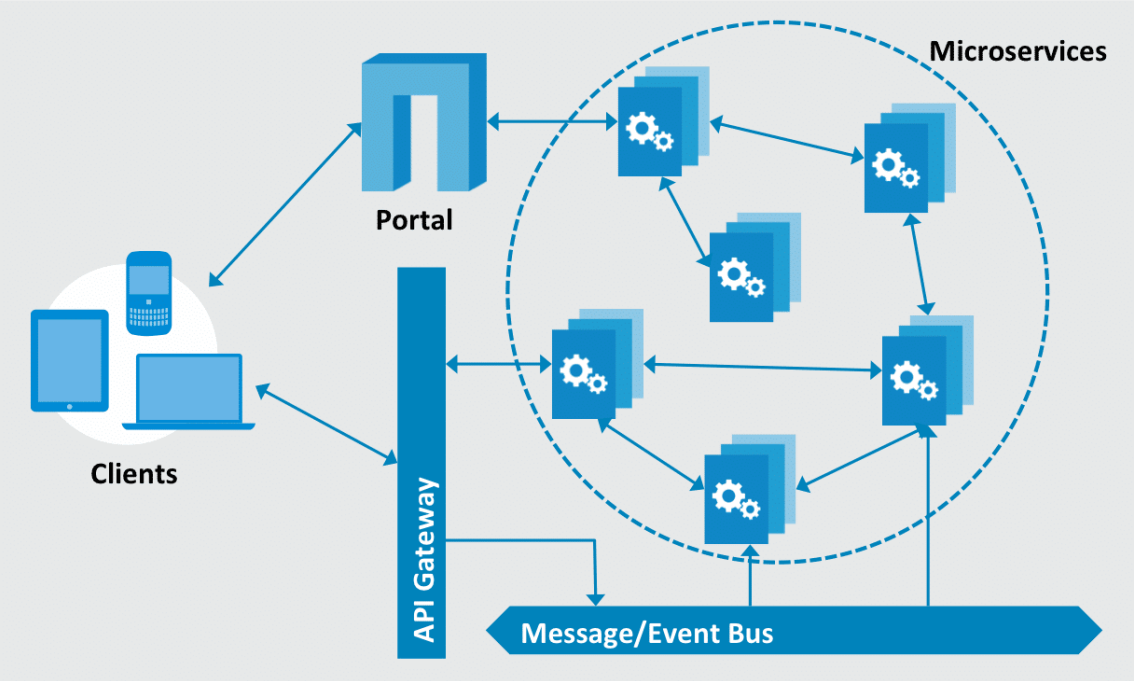


Abbildung 29: Schematische Microservice-Architektur; Quelle: Hert [Her16]

Eigenschaften einer Microservice-Architektur

Hert [Her16] definiert die Eingenschaften einer MSA wie folgt:

- Autonom: Microservices funktionieren für sich unabhängig. Dies bezieht sich sowohl auf die Laufzeit, wie auch auf die Entwicklung und das Deployment. Jeder Microservice kann eigenständig weiterentwickelt und ausgerollt werden.
- Entkoppelt: Microservices sind untereinander stark entkoppelt. Sie kommunizieren nur über klar definierte und sprachunabhängige Schnittstellen. Für synchrone Aufrufe sind das APIs und für die asynchrone Kommunikation kommt ein Message/Event-Bus zum Einsatz.
- Skalierbarkeit: Microservices können unabhängig und nach Bedarf (elastisch und horizontal) skaliert werden.
- Resilienz: Microservices sind ausfallsicher und fehlertolerant. Fehler wirken nur lokal und breiten sich nicht im ganzen System aus, was nach Hert eine Verfügbarkeit des Systems von 100% ermöglichen kann. Ein resilientes System hat nach Tresh [Tre18] die Fähigkeit zur Selbstheilung. Um ein möglichst verfügbares System zu erlangen wird der Ansatz verfolgt, anhand von Service-Isolation und einer entkoppelten Kommunikation die Mean Time to Repair (MTTR) zu minimieren. Zusätzlich sollte ein Monitoring in Kombination mit einem Error Handler (vgl. Abbildung 30 auf Seite 63) implementiert werden.

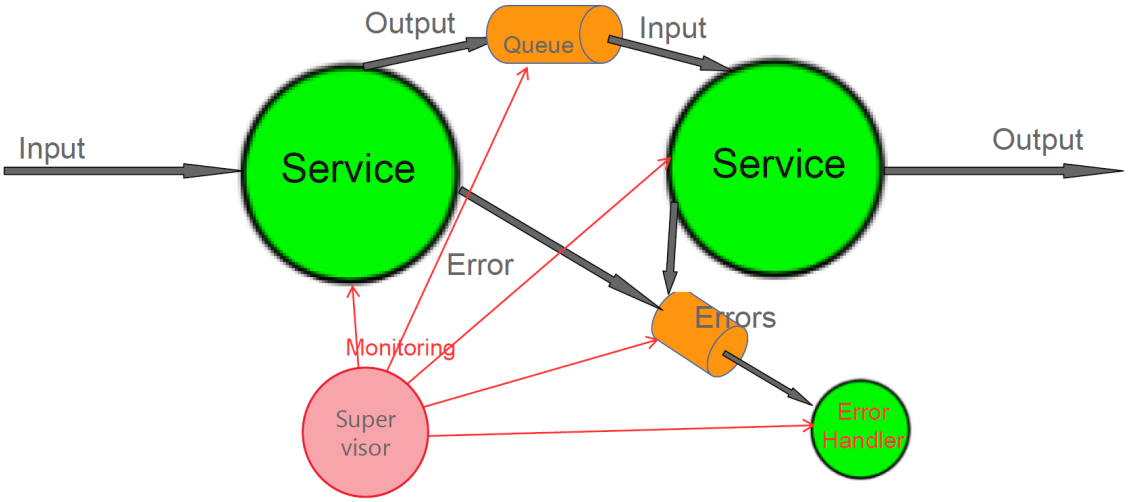


Abbildung 30: Resiliente Microservice-Architektur; Quelle: Tresh [Tre18]

Grundkonzepte einer funktionierenden Microservice-Architektur

Nach Hert [Her16] basiert eine funktionierende MSA auf folgenden Grundkonzepten, welche sich weitgehend auch mit den Erkenntnissen von Ross et al. [RSBJ17] decken:

- Continuous Delivery: Hoher Grad an Automatisierung für Provisionierung, Skalierung, Deployment und Regressionstest. Ermöglicht es, einzelne Microservices unabhängig und sicher voneinander auszurollen.
- DevOps: Schnelle Weiterentwicklung und sicherer Betrieb gelingen nur, wenn Entwicklung und Betrieb zusammenarbeiten. Die Organisationsstruktur eines Unternehmens findet sich auch im Design ihrer Systeme wieder (vgl. Conway's Law auf Seite 31).
- Zentrale Basisdienste: Gemeinsame Funktionen wie Logging, Monitoring und Konfiguration sollen zentral für alle Microservices bereitgestellt werden. Dies vereinfacht den Betrieb und beschleunigt die Entwicklung.
- API-Gateway: Entkoppelt den Client von den individuellen Microservices. Ermöglicht die Komposition der feingranularen Microservices in grobgranulare, clientorientierte Services. Security ist ein weiterer Aspekt, der zentral über ein API-Gateway gelöst werden kann.
- Domain Driven Design: Die passende Aufteilung der fachlichen Domäne in einzelne Microservices ist von zentraler Bedeutung für eine funktionierende MSA. Domain Driven Design stellt dazu die nötigen Werkzeuge bereit.

Wann ist der Einsatz einer Microservices-Architektur sinnvoll?

Nach Hert [Her16] eignen sich MSA primär für den Bau von Applikationen über die sich ein Unternehmen differenziert (was dem Digital Service Platform Teil nach Ross et al. [RSBJ17] entspricht). Diese Applikationen unterliegen oft stetigem Wandel und müssen aufgrund von Marktveränderungen schnell angepasst werden. Konkrete Beispiele wären der Online-Shop eines Retailers, das Kundenportal einer Versicherung oder das e-Banking einer Bank. Für eine erfolgreiche Umsetzung von MSA sind nebst den technischen Herausforderungen aber auch organisatorische Anpassungen nötig. So sollten funktionsübergreifende Teams, DevOps-Konzepte und Continuous Delivery-Methodiken angewendet werden.

Bewertung des Patterns

Das MSA-Pattern ist aus technischer Sicht eher einer IT-Architektur als einer IS-Architektur zuzuordnen. Im Zusammenhang mit MSA werden jedoch viele IS-Architekturrelevante Aspekte der Zukunftsfähigkeit wie etwa Skalierbarkeit oder Verfügbarkeit tangiert, weshalb die Diskussion und Bewertung des Patterns im ISA-Kontext im Sinne einer MSA-Domäne angebracht ist. Die Verwendung eines MSA-Patterns macht speziell im Bereich der Digital Servie Platform nach Ross et al. [RSBJ17] Sinn. MSA ermöglicht eine auf Funktionalität fokussierte Umsetzung einer Anwendungslandschaft. Der Architektur-Level wird daher diesbezüglich als «tief» eingestuft. In der nachfolgenden Tabelle 4 auf Seite 65 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern C - Microservice-Architektur		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Ziel einer MSA ist die Modularität, ebenso können die besten Skaliervorteile einer MSA in einer Cloud-Lösung erzielt werden. Durch eine entsprechene Architektur wird bis zu 100% Verfügbarkeit ermöglicht. Reaktivität wird durch Nachrichtenorientiertheit, Resilienz und Elastizität ermöglicht. Durch die Segregation von Bestand und Funktion wird eine Zustandslosigkeit angestrebt. Ein Ziel von MSA ist Skalierbarkeit und Elastizität. Durch die Aufteilung in verschiedene Funktionen respektive Services wird ebenfalls eine Multimodalität begünstigt.	Modularität Mobilität Verfügbarkeit Reaktivität Zustandslosigkeit Skalierbarkeit Multimodale IS-Architektur
Opportunities	Interoperabilität wird durch eine Trimmung der Services auf eine Funktionalität begünstigt, ist aber nicht explizit ein Thema von MSA. Eine Selbstorganisation und -Ähnlichkeit wird durch genau vorgegebene Muster begünstigt, das Innere von Microservices ist aber als Blackbox zu sehen.	Interoperabilität Selbstorg. und -ähnlichkeit
Weaknesses	Die Komplexität nimmt gesamtheitlich durch MSA zu, das Wissen über die IS-Architektur daher tendenziell eher ab.	Wissen über IS-Architektur
Threats	Allgemein: Vorbedingung für eine erfolgreiche MSA sind auch organisatorische Voraussetzungen wie DevOps, Continuous Delivery Konzepte und funktionsübergreifende Entwicklungs-Teams.	-

Tabelle 4: Bewertung Pattern C - Microservice-Architektur

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstellen (vgl. Abbildung 31 auf Seite 66) und vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

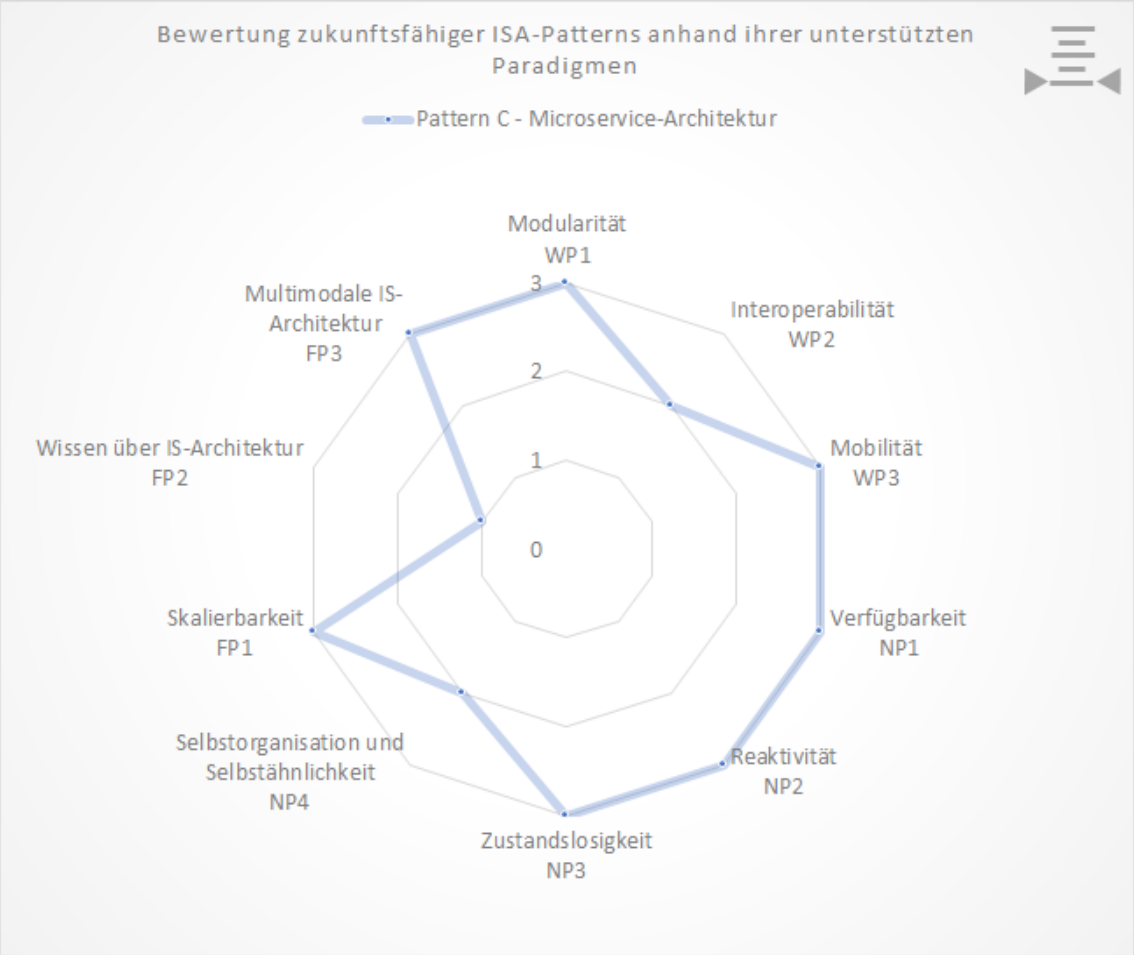


Abbildung 31: Netzdiagramm zur Bewertung von Pattern C - Microservice-Architektur

3.6.4 Pattern D - Quasar IS-Architektur

Das hergeleitete Pattern von Spichiger & Noser unter Abschnitt 2.2 auf Seite 13 richtet sich nach der sog. Quasar-IS-Architektur von Engels et al. [EJH+08]. Im Wesentlichen werden hierbei die unter Abbildung 32 auf Seite 67 dargestellten Schritte verfolgt. Zu beachten ist hierbei, dass Datenbanken, Application Server oder ein Enterprise Service Bus (ESB) zur technischen Infrastruktur gehören. Sie implementieren keine Geschäftslogik in Form von Anwendungsservices, sind also keine AL-Komponenten der IS-Architektur [EJH+08, S.159].

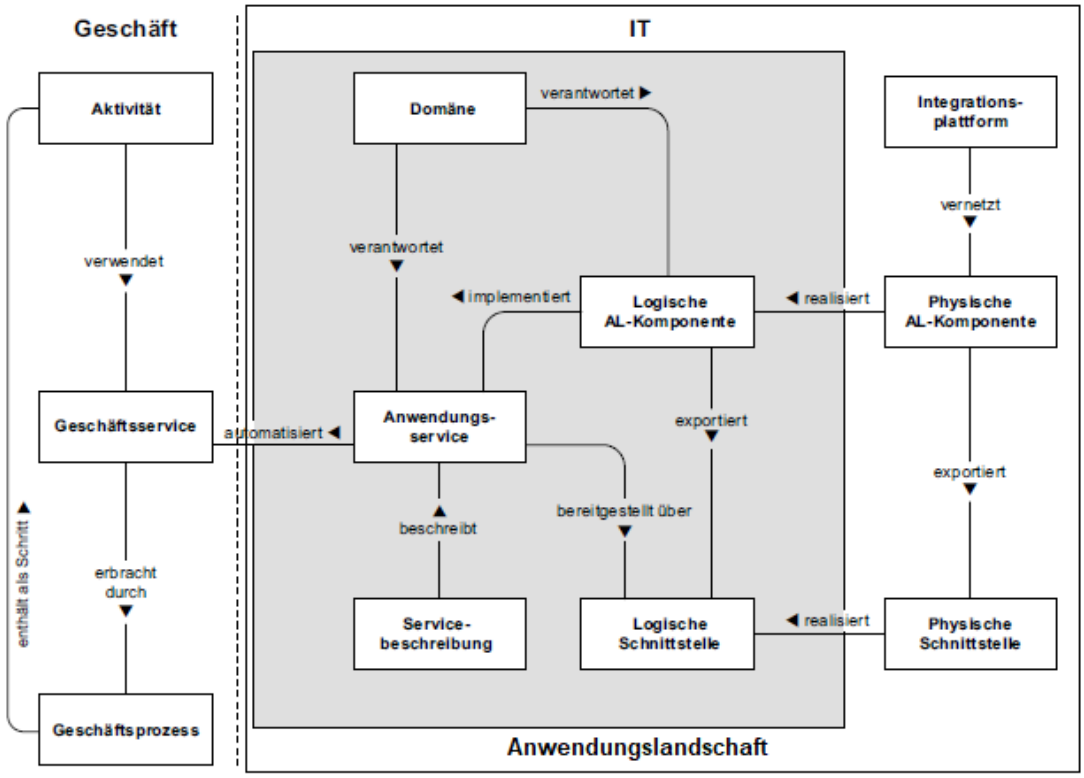


Abbildung 32: Begriffe und Zusammenhänge von Anwendungslandschaften in der Übersicht; Quelle: Engels et al. [EJH+08, S.79]

Bewertung des Patterns

Das vorliegende Pattern erlaubt eine detaillierte Darstellung vieler wichtiger Elemente einer ISA und ihrer Kopplungen. Es beantwortet aber eher den statischen Teil einer IS-Architektur und liefert keine Ansätze zum Ausbau und den Management-Praktiken. Es kann daher aber auch schnell unübersichtlich werden und sollte aus diesem Grund szenariobasiert angewendet werden. Das Pattern geht hinsichtlich dem Architektur-Level sehr ins Detail. Der Architektur-Level wird diesbezüglich daher als «tief» eingestuft. In der nachfolgenden Tabelle 5 auf Seite 68 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern D - Quasar IS-Architektur		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Das Pattern eignet sich aufgrund der detaillierten Unterteilungen der Komponenten gut zur Modularisierung. Es ist zudem sehr detailliert, was das Wissen über die IS-Architektur begünstigt.	Modularität Wissen über IS-Architektur
Opportunities	Selbstorganisation und Selbstähnlichkeit wird durch ein stringentes, konsistentes Anwenden der Vorgaben begünstigt.	Selbstorg. und -ähnlichkeit
Weaknesses	Interoperabilität und Mobilität respektive Entkopplung wird nur indirekt via SOA ermöglicht, ist aber kein expliziter Bestandteil des Patterns. Ebenso Reaktivität und Zustandslosigkeit.	Interoperabilität Mobilität Reaktivität Zustandslosigkeit
Threats	Das Pattern beinhaltet keine Angaben zu den Themen Verfügbarkeit, Skalierbarkeit und multimodale IS-Architektur.	Verfügbarkeit Skalierbarkeit Multimodale IS-Architektur

Tabelle 5: Bewertung Pattern D - Quasar IS-Architektur

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstellen (vgl. Abbildung 33 auf Seite 69) und vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

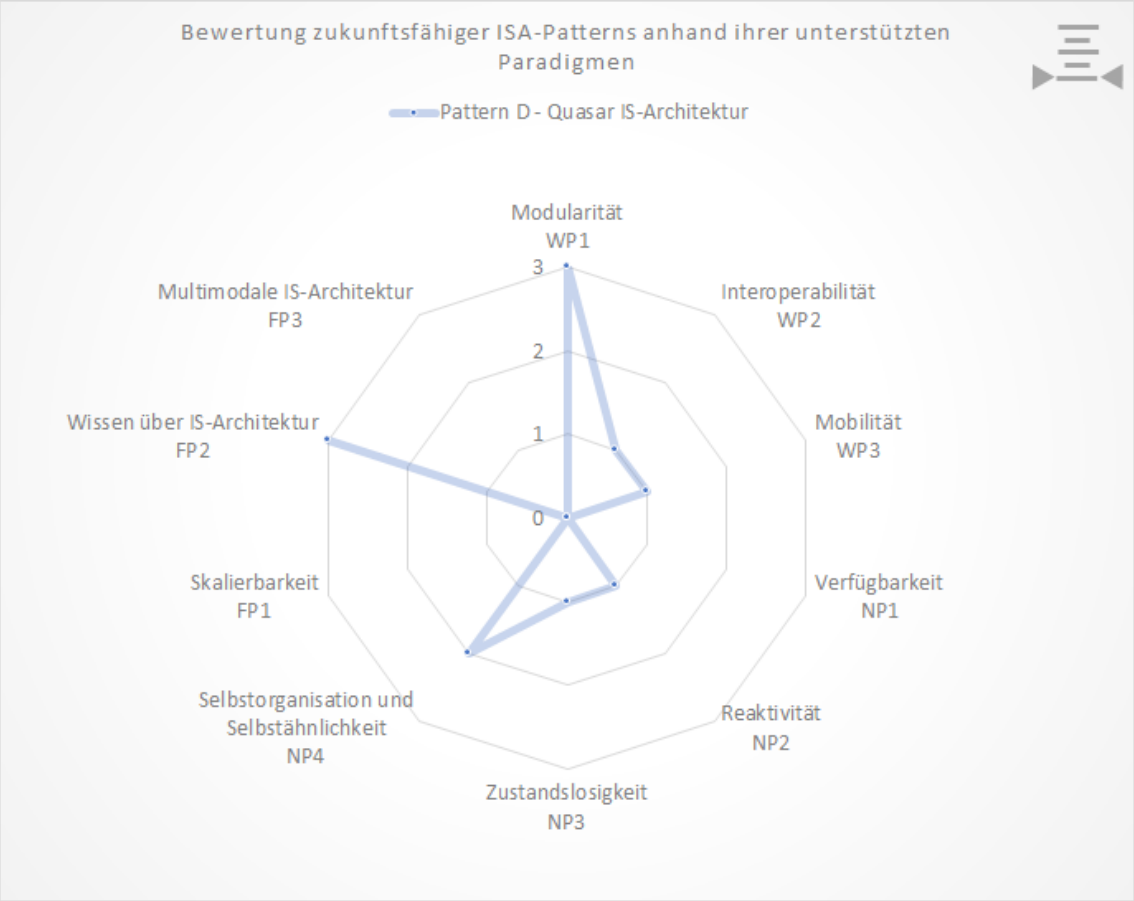


Abbildung 33: Netzdiagramm zur Bewertung von Pattern D - Quasar-IS-Architektur

3.6.5 Pattern E - Serverless Functions-Architektur

Grosse Enterprise-Projekte bringen eine entsprechende Komplexität mit sich. Ziel einer jeden Anwendung ist es, Fachlichkeit beziehungsweise Use Cases zu realisieren. Nicht selten liegt der Fokus der Entwicklung gemäss Röwekamp [Rö17] mehr auf dem Aufsetzen und Betreiben der Infrastruktur und Backend-Services als auf der Umsetzung der Geschäftslogik.

Abhilfe schafft eine Verlagerung einzelner Komponenten auf einen standardisierten Service-Stack (z.B. in der Cloud), da so mit der Implementierung der Geschäftslogik direkt auf einer Infrastruktur abstrahierenden Ebene gestartet werden kann. Einen sehr weitgehenden Abstrahierungsansatz bieten hierfür Serverless Functions, im Cloud Kontext auch als FaaS bekannt (vgl. Abbildung 20 auf Seite 37).

Bei Serverless Functions wird alles unterhalb der Geschäftslogik als gegeben betrachtet, was für den Anwendungsentwickler bedeutet, dass er sich um keinerlei Infrastruktur mehr kümmern muss[18]. Dazu gehören Server, Netzwerk, Storage, ggf. Virtualisierungsebenen, Betriebssystem, Laufzeitumgebung, Datenbanken und die Anwendung selbst.

[18] Dies gilt speziell bei FaaS in der Cloud, On-Premises muss eine solche Platform natürlich erst aufgebaut und entsprechend betrieben werden.

So lassen sich einzelne Funktionen als Services ohne sichtbare Infrastruktur betreiben und die Geschäftslogik, aufgeteilt in kleine Funktionsbausteine, abbilden. Mit Serverless Functions implementierte Anwendungen sind nach Fowler [Fow16] somit zustandslos, was aus Datensicht bedeutet, dass sie in einer externen Datenquelle persistiert werden müssen.

Angesprochen werden die Funktionen von aussen über ein API-Gateway oder intern als asynchroner Event einer anderen Komponente. Die Ablaufsteuerung der Anwendung ergibt sich implizit durch die Komposition der Events. Serverless Functions ermöglichen nach Jacobs [Jac18] eine besonders einfache Skalierbarkeit, da die zustandslosen Funktionen trivial horizontal skaliert werden können. Wird eine Funktion häufiger aufgerufen, werden automatisch entsprechend viele Runtimes gestartet und nach Abarbeitung der Funktion wieder beendet. Die Nutzung von Serverless Functions eignet sich nach Röwekamp [Rö17] somit insbesondere für Szenarien, in denen das Lastverhalten einzelner Business-Methoden extrem schwanken kann.

Durch eine Bezahlung pro Funktionsaufruf kann im Cloud-Kontext gerade bei stark schwankender Last auf grosse Kapazitäten zurückgegriffen werden, die nur bei der tatsächlichen Verwendung zu bezahlen sind. Ohne Funktionsaufruf entstehen somit auch keine Kosten.

Bewertung des Patterns

Das vorliegende Pattern unterstützt die «Digital Service Platform» Domäne nach Ross et al. [RSBJ17] sowohl On-Premises als auch in der Cloud durch eine rasche Implementierung von Funktionen, unabhängig von der darunterliegenden Infrastruktur. Zudem lässt es sich optimal in eine eventorientierte Architektur integrieren und ist sehr einfach skalierbar. Das Pattern ist einfach verständlich und lässt sich für einen Teilbereich der Architektur implementieren. Hinsichtlich dem Architektur-Level ist es einer granularen Stufe zuzuordnen und wird daher diesbezüglich als «tief» eingestuft. In der nachfolgenden Tabelle 6 auf Seite 71 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern E - Serverless Functions-Architektur		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Durch die Ausrichtung auf Geschäftslogik respektive -funktionalität wird eine Modularität ermöglicht. Der Mobilität sind durch die Abstrahierung und Portierbarkeit des Funktionscodes fast keine Grenzen gesetzt. Es wird eine Hochverfügbarkeit und elastische Skalierung ermöglicht. Das Pattern basiert auf einem reaktiven, zustandslosen Ansatz. Durch das einfache Konzept und einen elastischen Skalierungsansatz wird eine Selbstorganisation und Selbstähnlichkeit ermöglicht. Das Wissen über die IS-Architektur lässt sich durch die Abstrahierung und Fokussierung auf die Funktionalität sehr einfach dokumentieren. Allgemein: Einfaches und effizientes Konzept.	Modularität Mobilität Verfügbarkeit Reaktivität Zustandslosigkeit Selbstorg. und -ähnlichkeit Skalierbarkeit Wissen über IS-Architektur
Opportunities	Interoperabilität wird durch Eventorientierung und ein API Gateway unterstützt. Eine multimodale IS-Architektur wird speziell im Bereich «Digital Service Platform» ermöglicht.	Interoperabilität Multimodale IS-Architektur
Weaknesses	Allgemein: Das Pattern eignet sich hauptsächlich für den Teilbereich «Digital Service Platform» der Architektur.	-
Threats	Allgemein: Im Cloud Bereich muss Vendor Lock-In kritisch betrachtet werden. Hier lässt sich aber gegenüber einer On-Premises-Architektur eine bessere Effektivität hinsichtlich Setup- und Betriebskosten erzielen.	-

Tabelle 6: Bewertung Pattern E - Serverless Functions-Architektur

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstellen (vgl. Abbildung 34 auf Seite 72) und vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

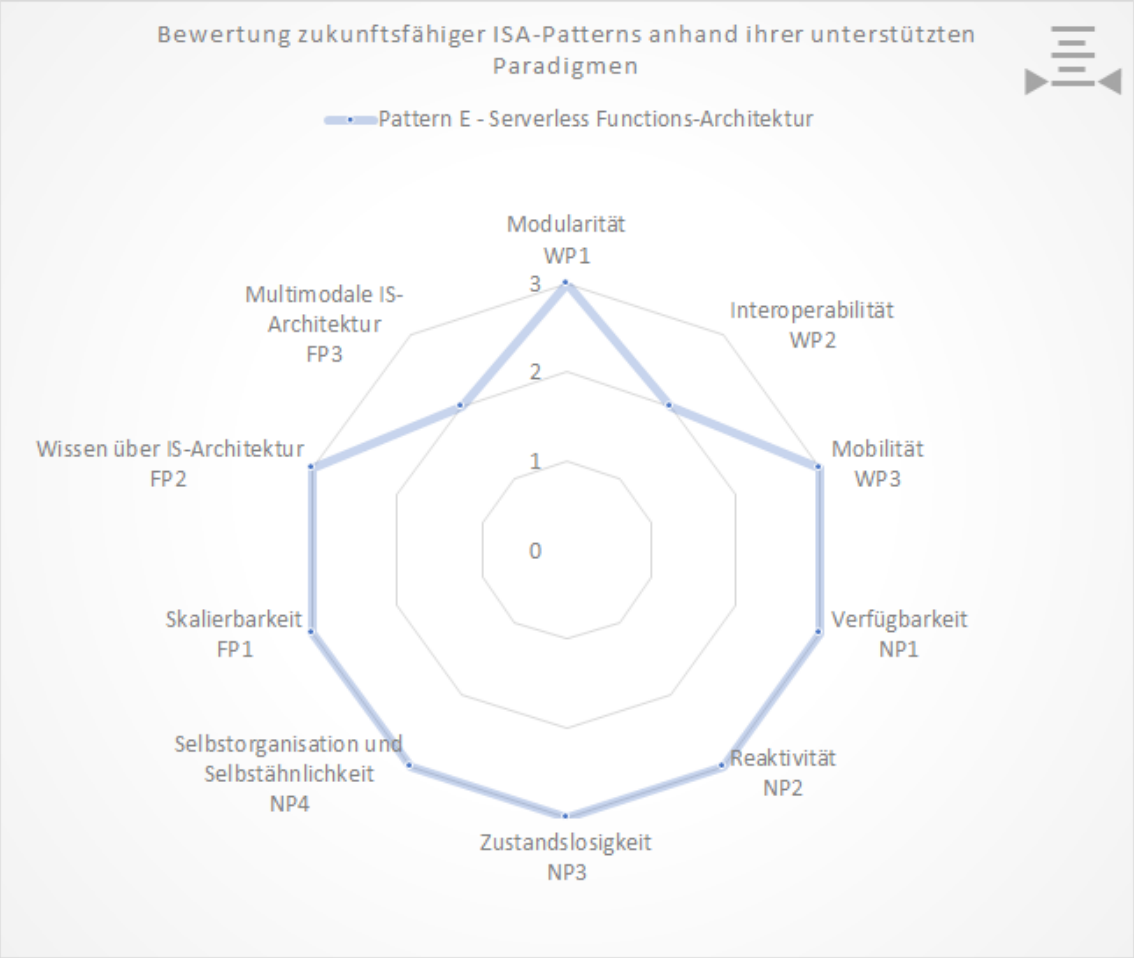


Abbildung 34: Netzdiagramm zur Bewertung von Pattern E - Serverless Functions-Architektur

3.6.6 Pattern F - Big Data-Architektur

Der Begriff «Big Data» beschreibt nach Gartner [Pet11] ein Datenwachstum in drei Dimensionen (3-V-Modell). Diese 3-V-Modell-Dimensionen beziehen sich nach Klein [KT-GH18] «auf ein ansteigendes Volumen (Volume) der Daten, auf eine ansteigende Geschwindigkeit (Velocity), mit der Daten erzeugt und verarbeitet werden und auf eine steigende Vielfalt (Variety) der erzeugten Daten». Big Data-Architektur-Konzepte sind hinsichtlich einer zukunftsfähigen IS-Architektur sehr vielversprechend, da sie nebst der Lösung der «3-V» Problematik auch komplementär als Intermediär (Digital Linkage) zwischen einem Operational Backbone und einer Digital Services Platform eingesetzt werden können und eine Basis für die nach dem Council of EU [EU17] geforderte Single Source of Thruth Anforderung ermöglichen.

Lambda (λ) Big Data-Referenzarchitektur

Der Begriff «Lambda Architektur» umfasst nach Marz et al. eine generische, skalierbare und fehlertolerante Datenverarbeitungsarchitektur [MHB17]. Die Lambda-Architektur zielt darauf ab, die Anforderungen an ein robustes System zu erfüllen, das sowohl gegen Hardwareausfälle als auch gegen menschliche Fehler tolerant ist. Zudem soll es in der Lage sein, ein breites Spektrum von Anwendungsfällen, eventorientiert und asynchron bedienen zu können, wobei (near-)realtime Verarbeitung und Anzeige unterstützt werden. Das resultierende System ist hierbei nahezu ohne Limitierungen linear horizontal und vertikal skalierbar.

Unter Abbildung 35 ist eine Big Data Lambda-Architektur abgebildet. Der Name «Lambda» (λ) entsteht durch eine um 90 Grad gedrehte Analogie des Symbols, wenn der erste Teil der Datenflüsse dargestellt wird. Ausgehend vom Collection Layer werden die Datenflüsse Lambda-förmig in zwei Streams ausgegeben und so einerseits in einer «Batch Data Processing Area» (Batch-Layer) und andererseits einer «Real-Time Data Processing Arena» (Speed-Layer) weiterverarbeitet. Der Batch-Layer hat zwei Funktionen: erstens die Verwaltung des umfassenden, kontinuierlich anwachsenden, unveränderlichen[19] Rohdatensatzes, basierend auf einem DFS[20] -Speichersystem und zweitens eine Vorberechnung der Batch-Views mittels MapReduce[21] Verfahren. Der Speed-Layer hingegen ermöglicht realtime Streaming und kann so die Verzögerungen des Batch-Layers kompensieren. Der hinter Batch-Layer und Speed-Layer liegende Service-Layer optimiert die Batch- und Speed-Layer Daten für Abfragen beispielsweise anhand von Indexierungen und Key-Value Datenbanken in einem Result-Store und bietet diese in Views an. Hiermit lassen sich beliebige Abfragen in kürzester Zeit durch aggregierte Sichten von Ergebnissen aus Batch-Layer Views und Speed-Layer Views beantworten. Die Result-Store-Datenbanken des Service-Layers werden nicht als unveränderliche Speicher verwendet: Sie können jederzeit gelöscht und aus dem Batch-Layer Rohdatenspeicher neu generiert werden. Fast jede Datenbank, ob in-memory oder persistent, kann im Service-Layer verwendet werden. Dazu gehören auch spezielle Datenbanken, wie eine Datenbank für Volltextsuche oder eine Graphen-Datenbank.

[19] Unveränderlich im Sinne eines «Create» und «Read» ohne «Update» und «Delete»

[20] DFS: Distributed File System; Ein hochverfügbares Dateisystem zur Speicherung sehr grosser Datenmengen auf den Dateisystemen mehrerer Knoten.

[21] MapReduce: Beim MapReduce-Verfahren werden die Daten in drei Phasen verarbeitet (Map, Shuffle, Reduce), wodurch sich Berechnungen parallelisieren und auf mehrere Rechner verteilen lassen.

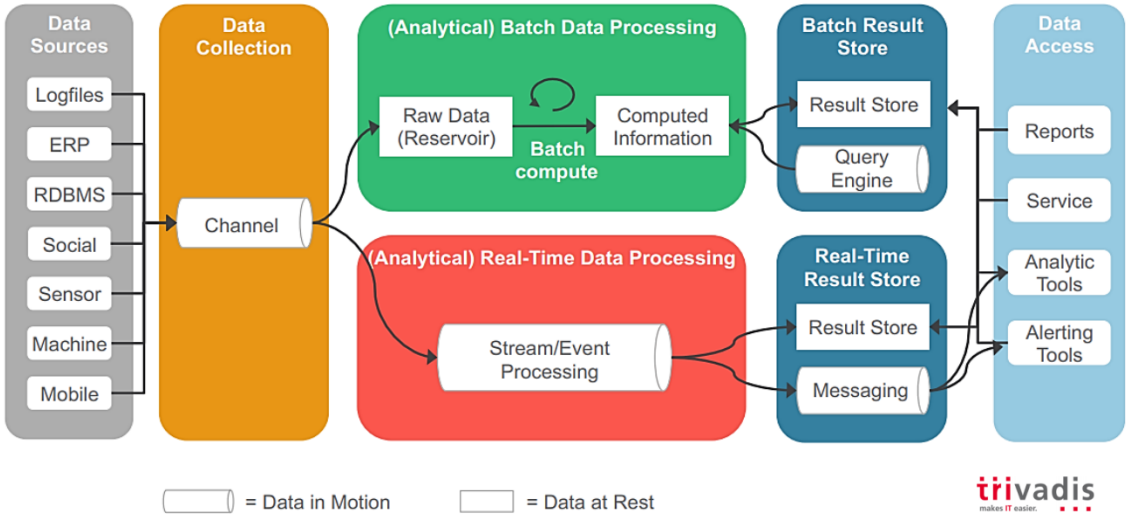


Abbildung 35: Lambda-Architektur für Big Data; Quelle: Schmutz [Sch17a]

Kappa (χ) Big Data-Referenzarchitektur

Die «Kappa Architektur» ist eine Spezialisierung der Lambda Architektur nach Kreps [Kre14]. Hierbei wird ein vollständiges «Data-in-Motion» Paradigma, basierend auf einem Real-time Streaming-Pattern angestrebt. So werden prinzipiell alle Daten durch den Streaming-Layer gespiesen. Ein Kappa-Architektur ist demnach wie eine vereinfachte Lambda-Architektur aufgebaut, bei welcher ein (möglicher) Batch-Layer nicht direkt durch den Service-Layer, sondern über den Streaming-Layer angesprochen wird (vgl. Abbildung 36).

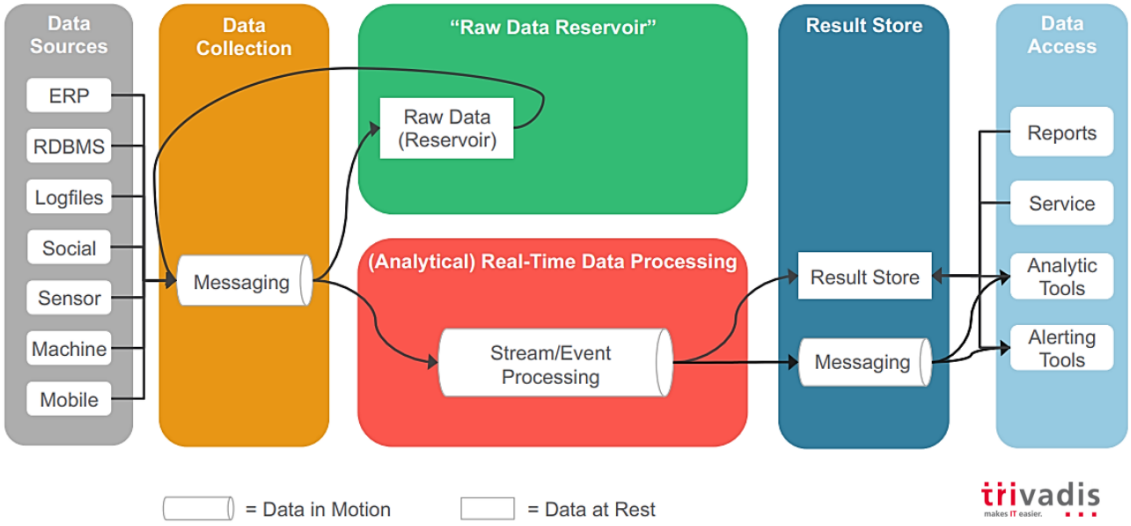


Abbildung 36: Kappa-Architektur für Big Data; Quelle: Schmutz [Sch17a]

Unified ($\lambda/\varkappa$) Big Data-Referenzarchitektur

Nach Schmutz [Sch17a] hat sich eine Kombination der beiden Ansätze in der Praxis bewährt, wobei etwa analytische Modelle aus dem Batch-Result-Store als Quelle für die eingesetzten Prediction-Models im Speed-Layer verwendet werden und der Rohdatenspeicher aus dem Batch-Layer selektiv integriert wird. Auf jeden Fall müssen diese Architekturen jeweils an die jeweiligen Anforderungen angepasst werden, so kann es etwa sein, dass ein Speed-Layer ohne Batch-Layer oder umgekehrt zweckmässig ist.

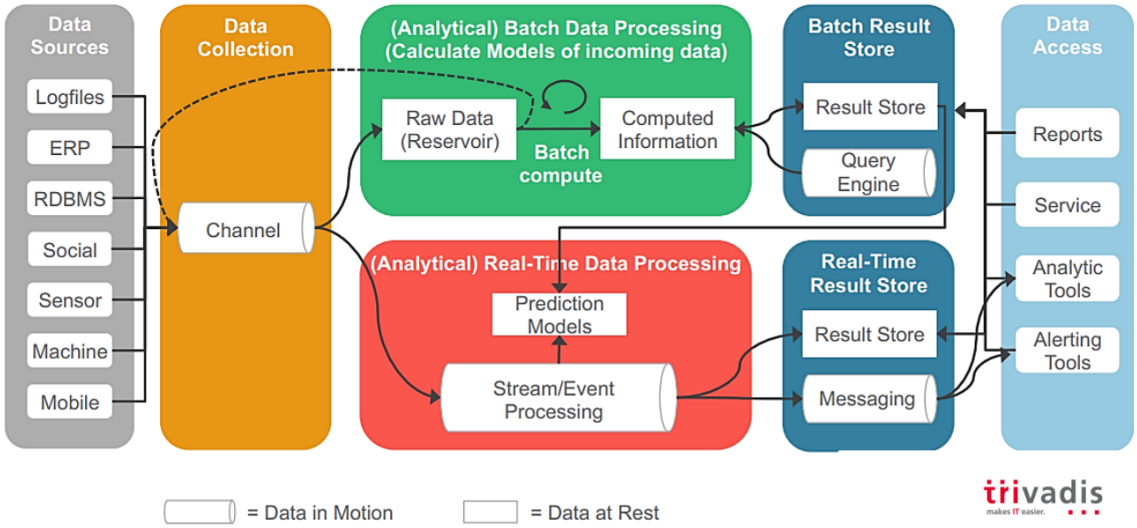


Abbildung 37: Unified-Architektur für Big Data; Quelle: Schmutz [Sch17a]

Bewertung des Patterns

Das Big Data-Architektur-Pattern ermöglicht eine sehr skalierbare und breit einsetzbare Anwendung und kann als mächtiger «Enabler» der Zukunftsfähigkeit, nicht nur im analytischen Kontext, sondern auch hinsichtlich eines zentralen Bindeglieds einer IS-Architektur gesehen werden. So lassen sich etwa anhand des Patterns analytische Anforderungen bedienen, eine Brücke zwischen den ISA-Domänen «Operational Backbone» und «Digital Service Platform» realisieren oder «Once Only» respektive «Single Source of Truth» Konzepte ermöglichen. Der persistente, nicht mutierbare, hochverfügbare Rohdatenspeicher schafft durch konsequentes Einspeisen aller Daten sogar eine auditfähige Datenhaltung und kann von Microservices über OLTP-Systeme bis hin zu logischen Data Warehouses (LDW) alle möglichen Abnehmer bedienen. Bezüglich des Architektur-Levels wird das Pattern als «tief» eingestuft, da es verschiedene Charakteristiken und technische Kopplungen einzelner Komponenten beschreibt. In der nachfolgenden Tabelle 7 auf Seite 76 werden die Charakteristiken des Lösungsansatzes anhand einer SWOT-Analyse bewertet.

Bewertung Pattern F - Big Data-Architektur		
SWOT	Argumentation	ISA-Paradigmen
Strengths	Das Pattern begünstigt Hochverfügbarkeit und Reaktivität, sprich Antwortbereitheit, Widerstandfähigkeit, Elastizität (Skalierbarkeit) und Nachrichtenorientiertheit. Das Pattern lässt sich komplett oder selektiv in der Cloud betreiben, die Fragestellung der Datensicherheit muss geklärt werden.	Mobilität Verfügbarkeit Reaktivität Skalierbarkeit
Opportunities	Eine zentrale Ansammlung von Wissen über IS-Architektur wird ermöglicht (Vorallem aus Datenflusssicht) und eine multimodale IS-Architektur gefördert. Allgemein: Die Lösung ist sehr zukunftsoffen und vielseitig Einsetzbar.	Wissen über IS-Architektur Multimodale IS-Architektur
Weaknesses	Das Pattern kann mit Modularität umgehen, liefert aber direkt keine Ansätze hierfür. Allgemein: Kann durch einen «Zoo» von Big Data-Anwendungen und -Technologien komplex werden. Erfordert Know-how und Ressourcen (Hardware).	Modularität
Threats	Das Pattern liefert keine Anknüpfungspunkte bezüglich Interoperabilität, Zustandslosigkeit und Selbstorganisation und Selbstähnlichkeit. Allgemein: «Single Source of Truth» aber potentiell auch «Single Point of Failure» als zentrales Nervensystem im Informationsfluss.	Interoperabilität Zustandslosigkeit Selbstorg. und -ähnlichkeit

Tabelle 7: Bewertung Pattern F - Big Data-Architektur

Zusammenfassend lassen sich die Erkenntnisse in Form eines Netzdiagrammes darstellen (vgl. Abbildung 38) auf Seite 77 vergleichbar machen. Die Skala wird von Wert 0 (Keine Angaben; Threats) über Wert 1 (Tiefe Ausprägung; Weaknesses) und Wert 2 (Mittlere Ausprägung; Opportunities) hin zu Wert 3 (Starke Ausprägung; Strengths) definiert.

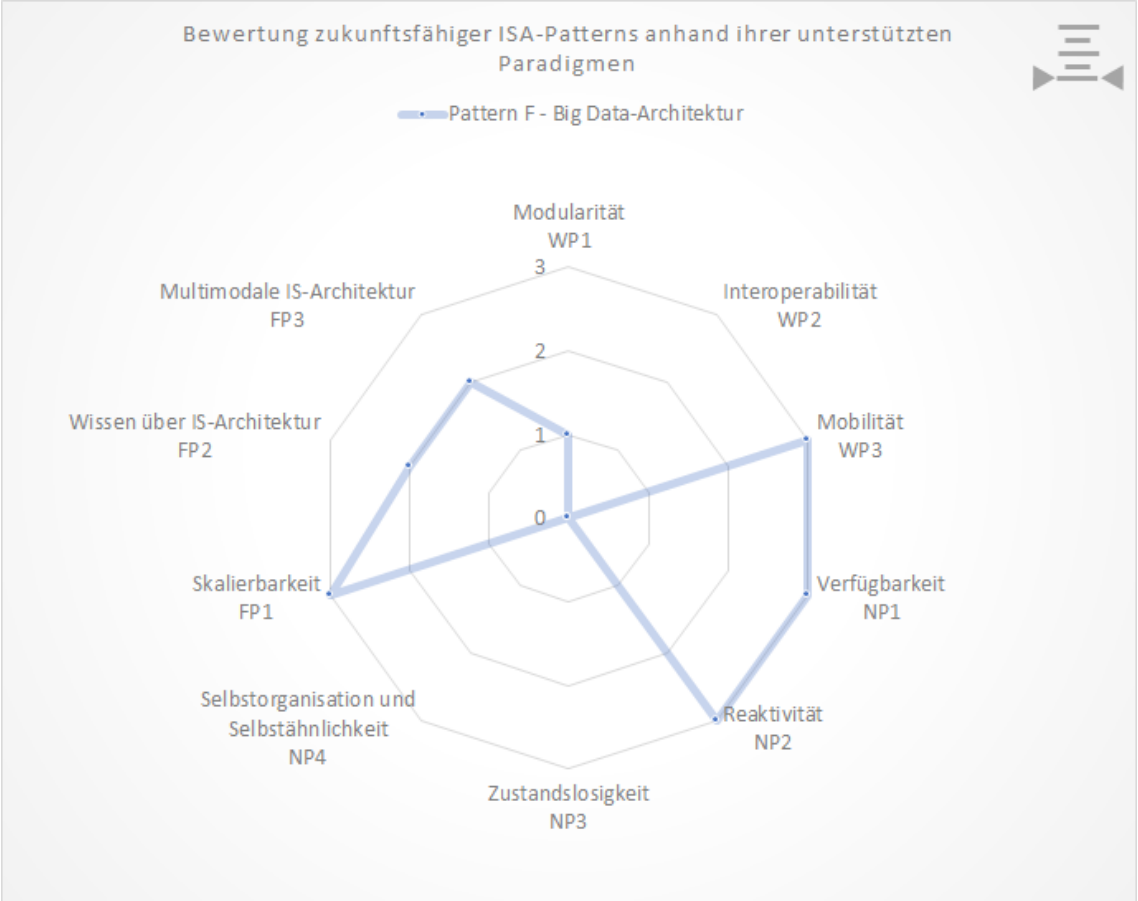


Abbildung 38: Netzdiagramm zur Bewertung von Pattern F - Big Data-Architektur

3.6.7 Zusammenfassung und Vergleich der ISA-Patterns A-F

Nach den individuellen Bewertungen der ISA-Patterns hinsichtlich der Unterstützung der Zukunftsfähigkeits-Paradigmen werden diese unter Abbildung 39 respektive 40 auf Seite 78 verdichtet dargestellt, wobei sich sowohl Lücken als auch Schwerpunkte respektive Überschneidungen der analysierten Patterns zeigen. Die Skala wird wiederum von Wert 0 (Keine Angaben) über Wert 1 (Tiefe Ausprägung) und Wert 2 (Mittlere Ausprägung) hin zu Wert 3 (Starke Ausprägung) definiert.

Als erstes kann festgestellt werden, dass nur durch eine Kombination der Patterns eine umfassende Referenzarchitektur erstellt werden kann, die das Potential sämtlicher Zukunftsfähigkeits-Dimensionen vollständig ausschöpft. Hinsichtlich der Bewertungen lässt sich zudem feststellen, dass viele Patterns zudem auf den unterschiedlichen Architektur-Ebenen von hoch bis tief nur einen Teilaspekt der Architektur abdecken.

Im Kern zeigt sich aber durchgehend, dass eine eventorientierte Architektur angestrebt werden sollte, welche eine asynchrone, lose gekoppelte Kommunikation ermöglicht. Zudem macht eine möglichst granulare Fokussierung auf Funktionalitäten – getreu dem Motto «do one thing and do it well» – Sinn, da so eine Modularität, eine verbesserte Verfügbarkeit und Reaktivität ermöglicht wird. Anhand einer engen Fokussierung

auf eine Funktion je Service kann zudem die Interoperabilität und Mobilität einfacher realisiert werden und das Wissen über die Architektur und ihre Zusammenhänge lässt sich besser dokumentieren und vermitteln. Die Zustandslosigkeit ermöglicht durch die damit realisierte Austauschbarkeit direkt ein einfacheres horizontales, elastisches Skalieren und sollte daher gefördert werden. Eine solche Austauschbarkeit und dynamische Skalierung dient ebenfalls einem selbstähnlichen Design der Komponenten, was wiederum eine Selbstorganisation hinsichtlich einer elastischen Skalierung begünstigt. Ein multimodale Architektur kann durch geeignete Management-Praktiken und isolierte Services unterstützt werden, da so verschiedene Release-Geschwindigkeiten ermöglicht werden.

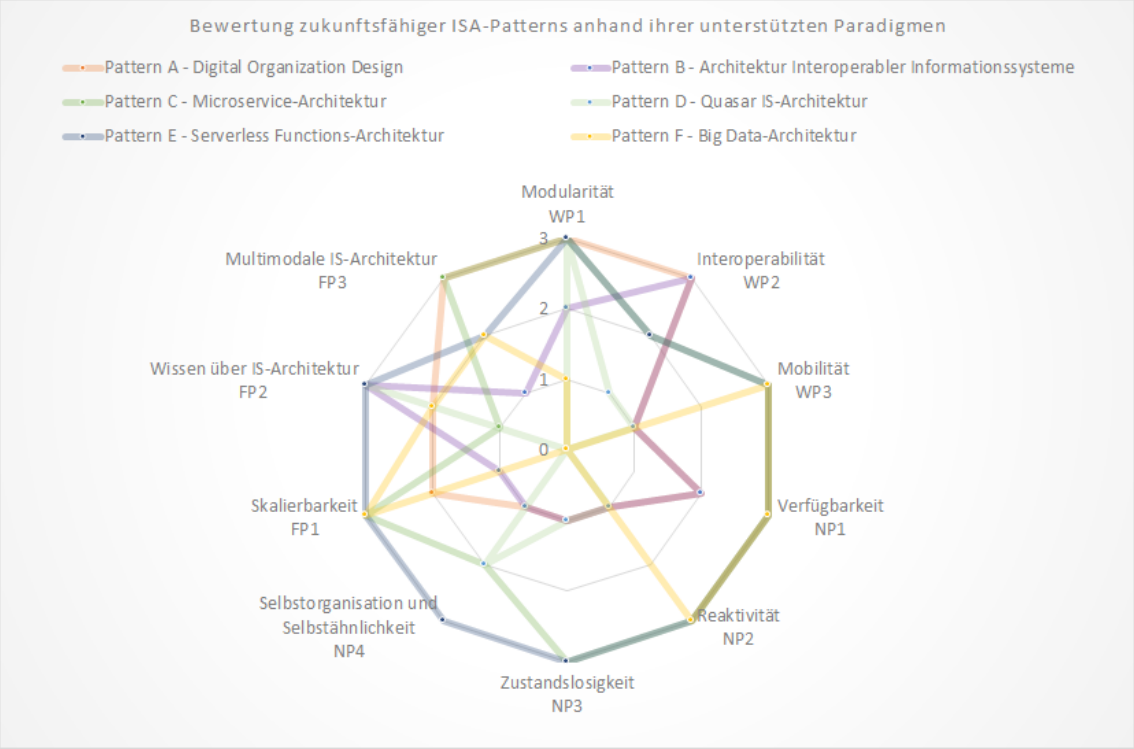


Abbildung 39: ISA-Patterns Netzdiagramm

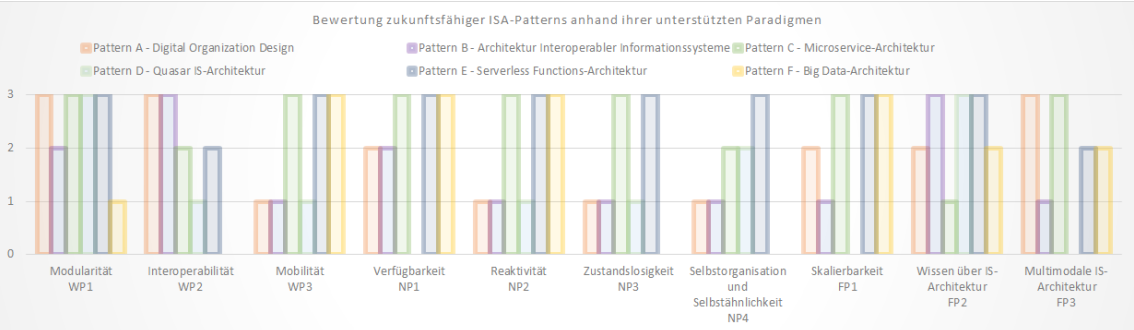


Abbildung 40: ISA-Patterns Balkendiagramm

3.7 Kombination der ISA-Patterns zu einer RSM-ISA-Referenzarchitektur

In diesem Abschnitt wird das ressourcenspezifische ISA-Artefakt im Sinne einer Referenzarchitektur auf Basis der bewerteten ISA-Patterns gebildet.

Eine Referenzarchitektur gilt in der Informatik als ein Referenzmodell für eine Klasse von Architekturen. Die Referenzarchitektur kann als Modellmuster – also ein idealtypisches Modell – für die Klasse der zu modellierenden Architekturen betrachtet werden. Im vorliegenden Fall wird daher das Idealbild einer zukunftsfähigen, ressourcenspezifischen Architektur als Referenzarchitektur-Artefakt erstellt. Die folgenden Ansprüche, die eine Architektur nach Vitruv[22] zu erfüllen hat (vgl. Blaser [Bla86]), gelten besonders für eine Referenzarchitektur:

- nützlich (utilitas)
- robust (firmitas) – somit dauerhaft, zeitlich weitgehend stabil
- anmutig oder schön (venustas)

Die Perspektiven «Wandlungsfähigkeit» und «Nachhaltigkeit» der Prinzipien-Pyramide der Zukunftsfähigkeit[23] decken sich hierbei genau mit den Definitionen «utilitas» und «firmitas» von Vitruv, was eine gute Ausgangsbasis für das Referenzarchitektur-Artefakt gewährleistet.

Das generisch aufgebaute RSM-ISA-Artefakt (vgl. Abbildung 41 auf Seite 80) wird nach dem RSM-Modell[24] gebildet. Zusätzlich werden die im letzten Abschnitt 3.6 bewerteten zukunftsfähigkeitsunterstützenden Patterns A-F integriert. Im Modell sind hierbei folgende Patterns integriert:

- Pattern A - Digital Organisation Design
  Die multimodale IS-Architektur ist als Domäne Digital Service Platform, Digital Linkage Layer und Operational Backbone im Modell ersichtlich. Zudem wurden noch die Pace Layer-Ausprägungen integriert, welche Systems of Record, Systems of Differentiation und Systems of Innovation unterscheiden.
- Pattern B - Architecture of Interoperable Information Systems (AIOS)
  Von AIOS wird das Business Interoperability Interface (BII) für die Partnerintegration ins Modell integriert.
- Pattern C - Microservice-Architektur (MSA)
  Das MSA-Pattern ist als Domäne integriert und kann in den Domänen Cloud und On-Premises verwendet werden.

[22] Vitruv (Marcus Vitruvius Pollio) war ein römischer Architekt, Ingenieur und Architekturtheoretiker. Er lebte im 1. Jahrhundert v. Chr.
[23] vgl. Abbildung 17 auf Seite 27
[24] vgl. Abbildung 16 auf Seite 24

- Pattern D - Quasar IS-Architektur
  Die Elemente aus der Quasar IS-Architektur decken sich weitgehend mit den Elementen aus GFbUA von Spichiger & Noser [SN]. Sie umfassen so alle Elemente von Anwendungslandschaft bis zu den Komponententypen und Schnittstellen. Die übergeordneten Geschäftsfähigkeiten, -prozesse und -objekte aus GFbUA sind im Artefakt entsprechend verknüpft mit den Quasar ISA-Elementen.
- Pattern E - Serverless Functions-Architektur
  Serverless Functions und FaaS sind als Domänen, unterteilt in die Bereiche Cloud und On-Premises integriert.
- Pattern F - Big Data-Architektur
  Die Big Data-Architektur wird zusammengefasst als Analytics Domäne (Big Data Service Layer) innerhalb des Digital Linkage Layers.

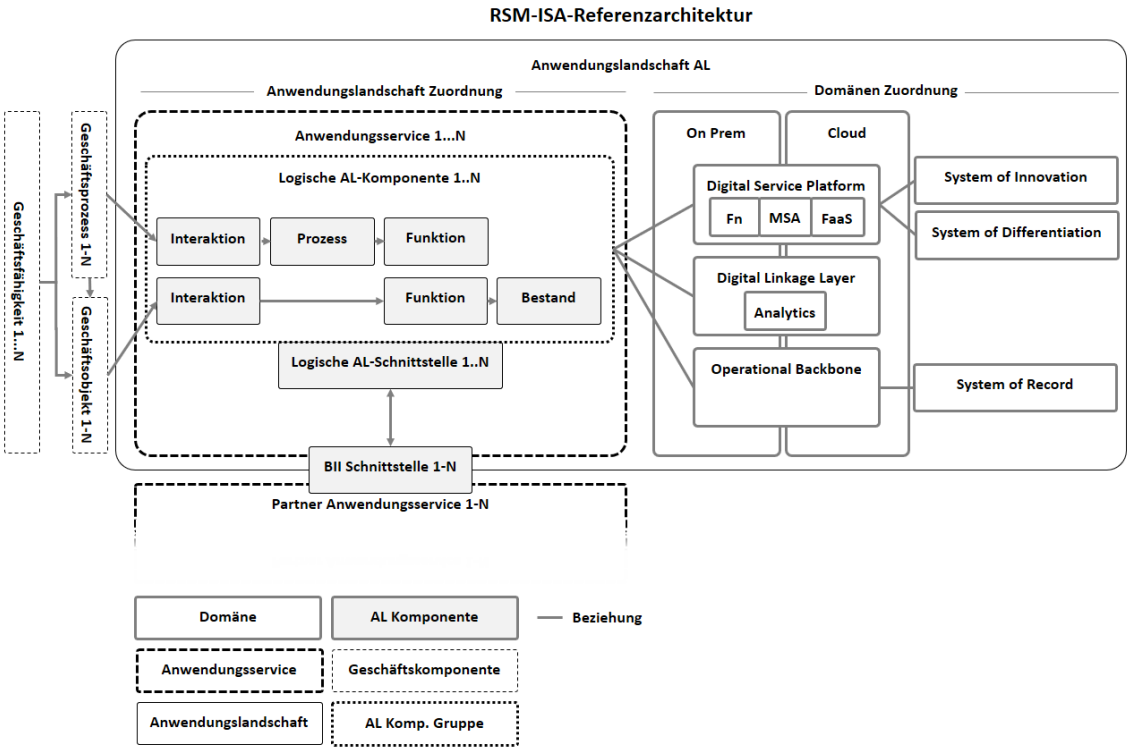


Abbildung 41: RSM-ISA-Referenzarchitektur

Wie in der Abbildung ersichtlich müssen die einzelnen AL-Komponenten jeweils verschiedenen Domänen zugeordnet werden. Das bedeutet, dass z.B. eine Bestandskomponente jeweils einer der Domänen Digital Service Platform, Digital Linkage Layer oder Operational Backbone zugeordnet wird. Zudem muss entschieden werden, ob diese Komponente in der Cloud- oder On-Premise-Domäne positioniert wird, da hier z.B. unterschiedliche Management Methoden, Deploymentprozesse, Datenschutzrichtlinien gelten können. Eine mögliche zusätzliche Präzisierung der Methodiken erlauben hier auch die Pace Layer-Domänen.

4 Anwendung des RSM-ISA-Referenzarchitektur-Artefakts

In diesem Kapitel wird das Artefakt gemäss Forschungs-Guideline 3 (Design Evaluation) analytisch in Form eines Architektur-«Fit» und deskriptiv anhand einer integrierten Einzelfallstudie geprüft und bewertet. Hierfür werden verschiedene Anwendungsfälle anhand von abgeleiteten Leistungsanforderungen der Fallstudie Arthur Reisen (AR) nach GFbUA von Spichiger & Noser [SN] erläutert, wonach unter 4.1 das Artefakt RSM-ISA angewendet und validiert wird. Abschliessend werden unter 4.2 die Resultate kritisch betrachtet und Artefakt-Verbesserungsmassnahmen diskutiert.

4.1 Fallstudie «Arthur Reisen» (AR)

Arthur Reisen (AR) wird in GFbUA nach Spichiger & Noser [SN] als Weiterentwicklung des Fallbeispiels Christoph Kolumbus Reisen (CKR) von Engels et al. [EJH+08] instanziiert und auf eine zukunftsfähige Strategie für die digitale Transformation und ein entsprechendes Geschäftsmodell ausgerichtet. Sämtliche digitalisierten Geschäftsfähigkeiten müssen hierbei bestmöglich durch die IS-Architektur bedient werden. In der Abbildung 42 auf Seite 82 sind hierzu die Elemente des Fallbeispiels CKR von Engels et al. [EJH+08] in einem Business Model Canvas nach Osterwalder et al. [OPW11] abgebildet und rot um die neuen Elemente von AR nach Atilgan und Hurst [AH16] ergänzt.

Abbildung 42: Business Model Canvas nach Osterwalder [OPW11] zum Geschäftsmodell Arthur Reisen; Quelle: Atilgang & Hurst [AH16].

4.1.1 Ableitung der ISA-Leistungsanforderungen

Die Anforderungen aus Sicht der AR Geschäftsfähigkeiten, -objekte und -prozesse werden im RIM nach Spichiger & Noser [SN, S.23ff] anhand eines umfassenden Beispieles hergeleitet. Da sich IS-Architekturen als Bindeglied zwischen den fachlichen Anforderungen aus Geschäftssicht und technischer Infrastruktur aus IT-Sicht sieht, werden in diesem Abschnitt nachfolgend die technischen Leistungsanforderungen von AR hergeleitet, um ein möglichst umfassendes Zielbild von AR zu erhalten.

Ausgehend vom Ableitungsmodell nach Engels et al. [EJH+08, S.79] (vgl. Abbildung 43 auf Seite 83) werden in diesem Abschnitt die ISA-Leistungsanforderungen aus technischer Sicht nach dem Fallbeispiel AR von Spichiger & Noser [SN] hergeleitet. Diese ISA-Leistungsanforderungen beziehen sich in der genannten Abbildung auf den horizontalen Pfeil von der Geschäftsarchitektur – mit den abstrahierten RIM-Komponenten Geschäftsfähigkeit, Geschäftsprozesse und Geschäftsobjekte – zur IT-Architektur und der darin

enthaltenen IS-Architektur. Eine konkrete IT-Strategie, die den vertikalen Pfeil von der IT-Strategie zu den IT-Architektur und IS-Architektur abbilden würde, liegt im Fallbeispiel von Spichiger & Noser nicht vor. Sie lässt sich jedoch weitgehend aus dem Kontext der beschriebenen AR Business Model Building Blocks[25] ableiten und entsprechend in den Anforderungen berücksichtigen. Nachfolgend werden die einzelnen Business Model Building Blocks zu AR erläutert und daraus die spezifischen ISA-Leistungsanforderungen für AR (LAR 1..n) abgeleitet, welche durch das vollständige RSM-ISA-Artefakt abgedeckt werden sollen. Im Anschluss an die Analyse werden die Leistungsanforderungen verdichtet, um ein konsistentes Bild zu erhalten.

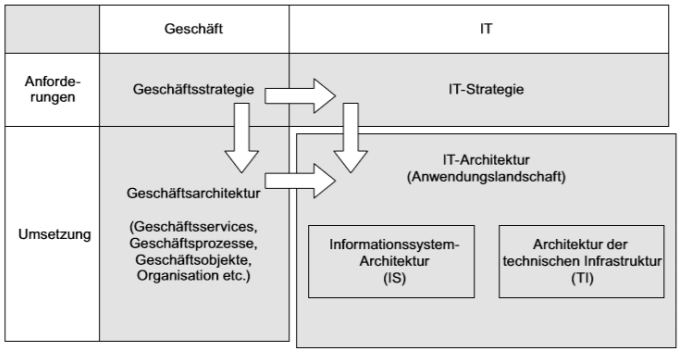


Abbildung 43: Zusammenhang zwischen Strategie und Achitektur; Quelle: Engels et al. [EJH+08, S.79]

Schlüsselpartnerschaften nach AR

Spichiger & Noser [SN] beschreibt im Bereich der Schlüsselpartnerschaften von AR, dass für die Bereitstellung der Resourcen soweit möglich die Kooperation mit bestehenden Anbietern gesucht wird. Dabei steht es allen Lieferanten frei, ihre Angebot direkt über AR zu platzieren oder ihre bisherige Intermediäre weiter zu bedienen. Für die Umsetzung des Geschäftsmodells kritisch ist, dass Lieferanten auch deklarieren, welche Mengen sie in der Lage sind bereitzustellen, damit auf dieser Basis Standardangebote erstellt werden können.

LAR 1: Kooperationsfähigkeit;
Einfache Anbindung und Integration von externen, digitalisierten Partnerprozessen über eine Schnittstelle.

Schlüsselaktivitäten nach AR

Spichiger & Noser [SN] beschreibt unter den Schlüsselaktivitäten von AR, dass sämtliche Angebote der bisherigen Christoph Kolumbus Reisen (CKR) mandantenfähig bereitge-

[25] Business Model Building Blocks: Elemente aus dem Business Model Canvas, nach Osterwalder et al. [OPW11]

stellt werden. Neben den bisherigen Reise-, Übernachtungs- und Eventangeboten werden neu zu den Reisen passende Medien angeboten. Sämtliche Leistungen können zu neuen Reiseangeboten kombiniert werden. Reiseanbieter können bei AR eigene Leistungen erstellen und diese, wenn gewünscht, als Lieferant auch weiteren Reiseanbietern anbieten. Sämtliche Vertragsbeziehungen können dynamisch erstellt werden. Die Verrechnung ist über eine breite Zahl von Zahlungsanbietern möglich. Reiseanbietern wird ermöglicht, Offertanfragen an Lieferanten zu platzieren, damit die Reiseanbieter für sie passende Angebote finden können. Eine grundsätzliche Herausforderung besteht in der Standarisierung der verschiedensten Geschäftsbedingungen auf für alle akzeptable Standards.

LAR 2: Mandantenfähigkeit;
Mandantenfähige, digitalisierte, dynamisch skalierbare Systemlandschaft.

LAR 3: Semantische Interoperabilität;
Standardisierte Modul-Schnittstellen für die Anbindung interner und externer Services (Reiseangebot, Zahlungsanbieter) entlang der Wertschöpfungskette, die den ganzen Customer Lifecycle (von der Angebots-Platzierung über die Offertenerstellung bis hin zu After Sales Analysen) bedienen kann.

LAR 4: Unstrukturierte Medien;
Zur Unterstützung von neuen Medien sollen auch unstrukturierte Datenquellen in die Informationslandschaft integriert werden können (Stichwort *Variety* der 3-V's von Big Data).

Schlüsselressourcen nach AR

Spichiger & Noser [SN] beschreibt, dass AR-Schlüsselressourcen primär die Mitarbeiter sind. Sie benötigen eine sehr hohe Kompetenz bzgl. der Produktgestaltung in der Reisebranche sowie eine hohe Kompetenz, diese Produktgestaltung ins Geschäftsmodell der AR, die Geschäftsarchitektur sowie die IS umzusetzen. Die so realisierte IS-Architektur und die entsprechenden IS bilden nach Ross et al. [RWR06] den Kern der Ausführungsbasis des Unternehmens.

LAR 5: Erweiterungsfähigkeit;
Anpassungsfähige, modularisierte IS-Architektur zur einfachen Anpassung und Weiterentwicklung von Services.

LAR 6: Portierbarkeit;
Cloudbasierte Scale-Out-Architekturen zur dynamischen Integration von Partnern ohne Ressourcenlimitierungen.

LAR 7: Real-Time Analytics;
Real-Time Analytics-Fähigkeiten zur Begleitung von Kunden im Rahmen eines Premium Services (Stichwort *Velocity* der 3-V's von Big Data).

Wertangebote nach AR

Nach Spichiger & Noser [SN] kommt AR in die Lage, die bisherigen Angebote für Standard- und Premiumkunden auch für Dritte (Reiseanbieter) mandantenfähig anzubieten. Zudem wird Reiseanbietern ermöglicht, eigene Standardangebote auf der Basis der bestehenden Leistungsangebote für sich als Mandant bereitzustellen. Für strategische Partner bei den Reiseanbietern wird auch das Premiumangebot von AR mandantenfähig bereitgestellt. Als Teil des Angebots können zu den Reiseangeboten auch Medien für die Vorbereitung der Reise oder für unterwegs mit eingebunden werden. Reisende erhalten zudem die Möglichkeit, nach der Reise weitere Medien (inklusive Urheberrechtsentschädigungen) zu beziehen bzw. diese in Reisedokumentationen zu integrieren.

LAR 8: «Pay as you go»;
«Pay as you go» Dienste für vertragsungebundene Leistungsabrechnungen erfordern eine zeitnahe, transparente und nachvollziehbare Service-Benutzung entlang der Wertschöpfungskette, welche durch die IS-Architektur unterstützt werden muss.

LAR 9: Mandantenfähigkeit (vgl. LAR 2);
Eine IS-Architektur mit mandantenfähigen Services und Schnittstellen für Partner.

LAR 10: Mobilität und Verfügbarkeit;
Mobilität und Verfügbarkeit (24x7) der Services für den Dienstleistungskonsum rund um die Uhr der Reisenden.

LAR 11: Unstrukturierte Medien (vgl. LAR 4);
IS-Architektur zur Unterstützung unterschiedlicher Medien (Stichwort *Variety* der 3-V's von Big Data).

Kundenbeziehungen nach AR

Spichiger & Noser [SN] beschreibt, dass neben den bisherigen Standard- und Premiumkunden auch Reiseanbieter die Hauptkundschaft von AR sind. Im Sinne der Maturität «Dynamic Venturing» [RWR06] besteht so die Möglichkeit, sämtliche Verträge für Reiseanbieter dynamisch zu erstellen. Reiseanbieter können gegenüber anderen Reiseanbietern rekursiv wiederum als Lieferanten von Teilleistungen auftreten.

LAR 12: Dynamic Venturing;
Crossorganisationale Wertschöpfungskette.

LAR 13: Portalintegration;
Web-Portal für Communities und Co-Creation Partnerinteraktionen.

Kanäle nach AR

Spichiger & Noser [SN] spezifiert, dass Angebote für Endkunden im Namen der Mandanten über ein Webportal bereitgestellt werden können. Weiter stehen alle Angebote auch über einen Marktplatz zur Verfügung, wobei der Mandant auswählen kann, ob einige (oder alle) seiner Angebote nicht im Marktplatz erscheinen sollen und damit nur über sein mandantenspezifisches Portal zugänglich sein sollen. Für sich selber und für Mandanten erstellt AR zudem Reisekataloge. Alle Angebote für Reiseanbieter sind zudem als Web-Services verfügbar. Dank guter Beratung durch seine kompetenten Reisebüros wird AR weiterempfohlen. Um mit den Kunden in engem Kontakt zu bleiben, werden Email, Post, aber auch ein Call-Center verwendet.

LAR 14: Dynamic Venturing (vgl. LAR 12);
Digitalisierte Prozesse entlang der Wertschöpfungskette (Angebot) und des Customer Lifecycles (CRM, Omni Channel) mit Schnittstellen für die Anbindung und Integration von externen Partnerprozessen.

LAR 15: Mandantenfähigkeit (vgl. LAR 2);
Mandantenfähige Webportal Plattform (Marktplatz) mit Kundenidentifikation und Schnittstellen (Webservices) für interne und externe Partner mit der Möglichkeit zur Angebotserstellung und –abwicklung.

Kundensegmente nach AR

Spichiger & Noser [SN] spezifiziert für den Bereich der Kundensegmente von AR, dass die bisherigen Kundensegmente Standardkunde und Premiumkunde weiterhin bedient werden. Die Möglichkeiten des Premiumangebots werden aber stark ausgebaut, um AR als Marktplatz im Reisemarkt zu positionieren. Entsprechend will AR als neues Kundensegment vor allem andere Reiseanbieter gewinnen, die die Plattformen von AR zur Angebotserstellung und –abwicklung nutzen.

LAR 16: Modularität;
Ausbaufähigkeit der Services (Premiumangebot).

LAR 17: Marktplatz/Portal (vgl. LAR 13);
Marktplatz in Form einer mandantenfähigen Plattform für Reiseanbieter mit der Möglichkeit zur Angebotserstellung und –abwicklung.

LAR 18: Technische Interoperabilität (vgl. LAR 14);
Digitalisierte Prozesse entlang der Wertschöpfungskette (Angebot) und des Customer Lifecycles (CRM, Omni Channel) mit Schnittstellen für die Anbindung und Integration von externen Partnerprozessen.

Kostenstruktur nach AR

Die Bereitstellung der entsprechenden Services erfordert nach Spichiger & Noser [SN] insbesondere grosse Vorausinvestitionen in die IS-Landschaft. Um die Infrastrukturkosten elastisch zu halten, werden die Services in der Cloud betrieben. Ein kompetentes Produktmanagement tritt als Anforderungssteller gegenüber der Anwendungsentwicklung auf.

LAR 19: Portierbarkeit (vgl. LAR 6);
Infrastruktur basierend auf Cloud Services.

LAR 20: Kostentransparenz (vgl. LAR 2); Kostentransparenz anhand von Ressourcenverbrauch (z.B. pro Mandant und Transaktion).

Einnahmequellen nach AR

Nach Spichiger & Noser [SN] bestehen die AR-Einnahmequellen aus einer Kommission für den Verkauf von Leistungen von Drittanbietern über die eigene Plattform. Damit setzt AR auf eine konsequente Umsetzung des «Pay as you go»-Prinzips, wie dies für vertragsungebundene Leistungen sonst üblich ist und im Zusammenhang mit Cloud-basierten Diensten auch immer üblicher wird. Strategische Partner beteiligen sich mit Investitionen an der Weiterentwicklung der Plattformen.

LAR 21: «Pay as you go» (vgl. LAR 8);
Vertragsungebundene Leistungsabrechnung für Nutzung an Partner.

LAR 22: Modularität (vgl. LAR 16), «Pay as you go» (vgl. LAR 8);
Nachvollziehbarkeit und damit einhergehende kostentransparente Service-Berechnungen für die Abrechnung.

LAR 23: Portierbarkeit (vgl. LAR 6);
Skalierbare Services (Cloud Paradigma)

Übersicht über die ISA-Leistungsanforderungen

In diesem Abschnitt werden die ermittelten ISA-Leistungsanforderungen des Anwendungsfalles AR in der Tabelle 8 auf Seite 88 verdichtet und in einem (RSM-)IS-Architekturmodell visualisiert.

Geschäftsmodellbasierte Leistungsanforderungen von Arthur Reisen (AR) aus ISA-Sicht		
ID	Verdichtete Leistungsanforderung-Beschreibung	LAR
1	Zur Unterstützung von neuen Medien sollen auch unstrukturierte Datenquellen in die Informationslandschaft integriert werden können (Stichwort *Variety* der 3-V's von Big Data).	Unstrukturierte Medien (LAR 4)
2	Cloudbasierte, dynamisch skalierbare, anpassungsfähige, modulare Architekturen zur einfachen Weiterentwicklung und dynamischen Integration von Partnern (Mandantenfähigkeit) ohne Ressourcenlimitierungen.	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) Erweiterungsfähigkeit (LAR 5) Portierbarkeit (LAR 6) Dynamic Venturing (LAR 12)
3	Real-time Analytics-Fähigkeiten zur Begleitung von Kunden im Rahmen eines Premium Services (Stichwort *Velocity* der 3-V's von Big Data).	Real-time Analytics (LAR 7)
4	«Pay as you go» Dienste für vertragsungebundene Leistungsabrechnungen erfordern eine zeitnahe, transparente und nachvollziehbare Service-Benutzung entlang der Wertschöpfungskette.	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) «Pay as you Go» (LAR 8) Real-time Analytics (LAR 7)
5	Mobilität und Verfügbarkeit (24x7) der Services für den Dienstleistungskonsum rund um die Uhr der Reisenden.	Portierbarkeit (LAR 6) Mobilität und Verfügbarkeit (LAR 10)
6	Digitalisierte Prozesse entlang der Wertschöpfungskette (Angebot) und des Customer Lifecycles (CRM, Omni Channel, von der Angebots-Platzierung über die Offertenerstellung bis zur After Sales-Analytik). Standardisierte Modul-Schnittstellen für die Anbindung und Integration von externen Partnerprozessen (Bsp: Reiseangebot, Zahlungsanbieter) (Stichwort: Crossorganisationale Wertschöpfungskette, Dynamic Venturing)	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) Dynamic Venturing (LAR 12) Portal Integration (LAR 13)
7	Mandantenfähige Webportal Plattform(en) (Marktplatz, Communities, Co-Creation) mit Benutzer-Identifikation und Schnittstellen (Webservices) für externe Partner mit der Möglichkeit zur Angebotserstellung und –abwicklung.	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) «Pay as you go» (LAR 8) Dynamic Venturing (LAR 12) Portal Integration (LAR 13)

Tabelle 8: ISA-Leistungsanforderungen abgeleitet vom Geschäftsmodell AR

Aufgrund der identifizierten Leistungsanforderungen von AR wurde ein Architekturmodell erstellt, welches sich an die RSM-ISA-Referenzarchitektur (vgl. Abbildung 41 auf Seite 80) anlehnt und die wichtigsten Domänen, Schlüsselelemente und deren Beziehungen aus Sicht der Leistungsanforderungen aufzeigt (vgl. Abbildung 44). In diesem Modell sind aus Gründen der Übersichtlichkeit noch keine RSM-Elemente nach Spichiger & Noser [SN] respektive Engels et al. [EJH+08] enthalten. Ebenso ist noch keine Domäne für die Zuordnung Cloud oder On-Premises integriert. Das Architekturmodell wird aus einer Datenflusssicht betrachtet, was im Sinne eines Actio=Reactio[26] Ablaufmusters einfach nachvollzogen werden kann.

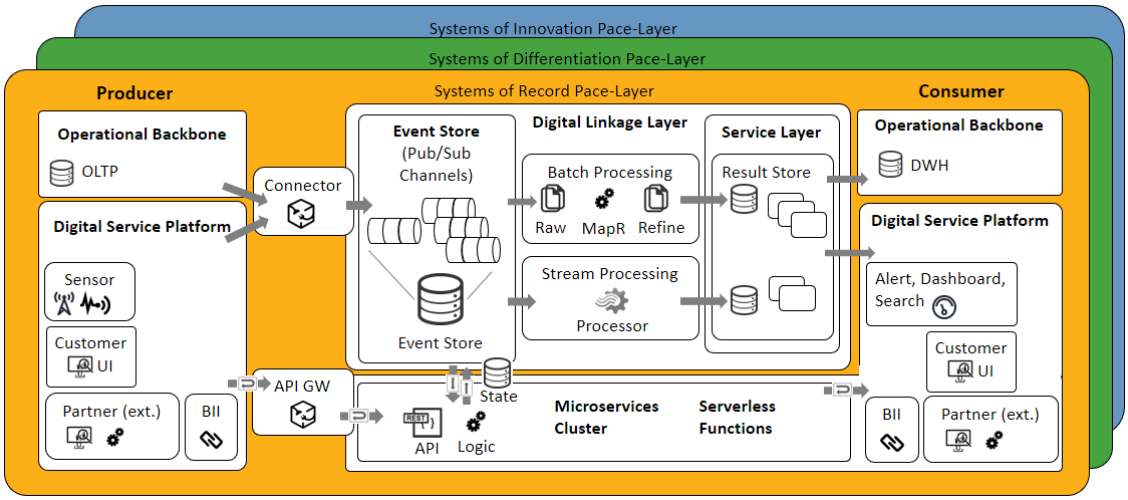


Abbildung 44: (RSM-)IS-Architektur aus Sicht der Leistungsanforderungen AR

4.1.2 RSM-ISA-Validierungsszenarios

IS-Architekturen sind komplex und ihre Darstellung wird schnell unübersichtlich. Aus diesem Grund wird für die szenariobasierte Validierung des Artefakts nicht nur eine Unterteilung nach Ebenen und Domänen vorgenommen, wie unter Kapitel 3.3 auf Seite 19 empfohlen, sondern es wird auch eine szenariobasierte Fokussierung der abzubildenden IS-Architektur auf eine Geschäftsfähigkeit, respektive einen Geschäftsprozess und die tangierten Geschäftsobjekte vorgenommen. So werden anhand dieses spezifischen Szenarios alle involvierten Anwendungsservices der Anwendungslandschaft mit ihren Interdependenzen und ihren logischen Komponenten – bestehend aus den Kategorien Bestand, Funktion, Prozess, Interaktion – aufgezeigt. Eine Liste der Geschäftsfähigkeiten, -objekte und -prozesse von Arthur Reisen ist unter Tabelle 9 auf Seite 90 abgebildet[27].

[26] Actio≙Reactio; Gemäss dem dritten Newton'schen Gesetz, auch Lex Tertia, oder Reaktionsprinzip genannt.

[27] Ein alternativer Ansatz für Validierungsszenarios nach den Wertangeboten aus dem Business Model Canvas von AR ist im Anhang C ersichtlich.

Geschäftsfähigkeiten, -prozesse und -objekte von Arthur Reisen			
ID	Geschäftsfähigkeit	Geschäftsprozesse	Geschäftsobjekte
1	Beschaffung	Leistung evaluieren	Partner Leistung
2	Gestaltung	Saison planen Pauschalreise vertreiben	Saisonplan Pauschalreise Individualreise
3	Beratung	Kunde beraten	Kunde
4	Vereinbarung	Reise buchen Leistung einkaufen Virtuelles Lager bewirtschaften	Reisende Kundenreise
5	Erfüllung	Leistung beziehen	
6	Nachsorge	Kunde pflegen	

Tabelle 9: Geschäftsfähigkeiten, -prozesse und -objekte von Arthur Reisen; Quelle: Spichiger & Noser [SN, S.24]

4.1.3 Validierung anhand Szenario «Geschäftsprozess Reise buchen»

Die Validierung des RSM-ISA-Referenzarchitektur-Artefaktes wird anhand der Geschäftsfähigkeit «Vereinbarung» mit dem Geschäftsprozess «Reise buchen» und dem Geschäftsobjekt «Kundenreise» durchgeführt. Diese Angaben beruhen auf den Definitionen aus GFbUA nach Spichiger & Noser [SN] (vgl. Anhang D). Beigezogen werden die ISA-Leistungsanforderungen von AR in Form eines Webportals, einer BII-Partnerschnittstelle für die Bankangelegenheiten und einem Bot[28], welcher alternativ zur Buchung auf der Website verwendet werden kann.

Aus RSM-ISA-Architektursicht werden die Domänen «Operational Backbone», «Digital Linkage Layer» sowie «Digital Service Platform» integriert. Zusätzlich werden die RSM-Komponentenkategorien «Interaktion», «Prozess», «Funktion» und «Bestand», sowie die RSM-Objekte «Anwendungsservice», «Komponente» und «Interface» verwendet. Als auslösende Trigger dienen «Event»-Elemente.

Abbildung 44 auf Seite 89 zeigt hierbei das RSM-ISA-Artefakt für den Geschäftsprozess «Reise buchen» von Arthur Reisen auf[29].

[28] Zur Prüfung der einfachen Erweiterbarkeit des Artefakts

[29] Für die Modellierung wurde Archimate 3.0 verwendet. ArchiMate gilt als technischer Standard von The Open Group und basiert auf Konzepten nach dem IEEE 1471:2000 Standard [Gro00].

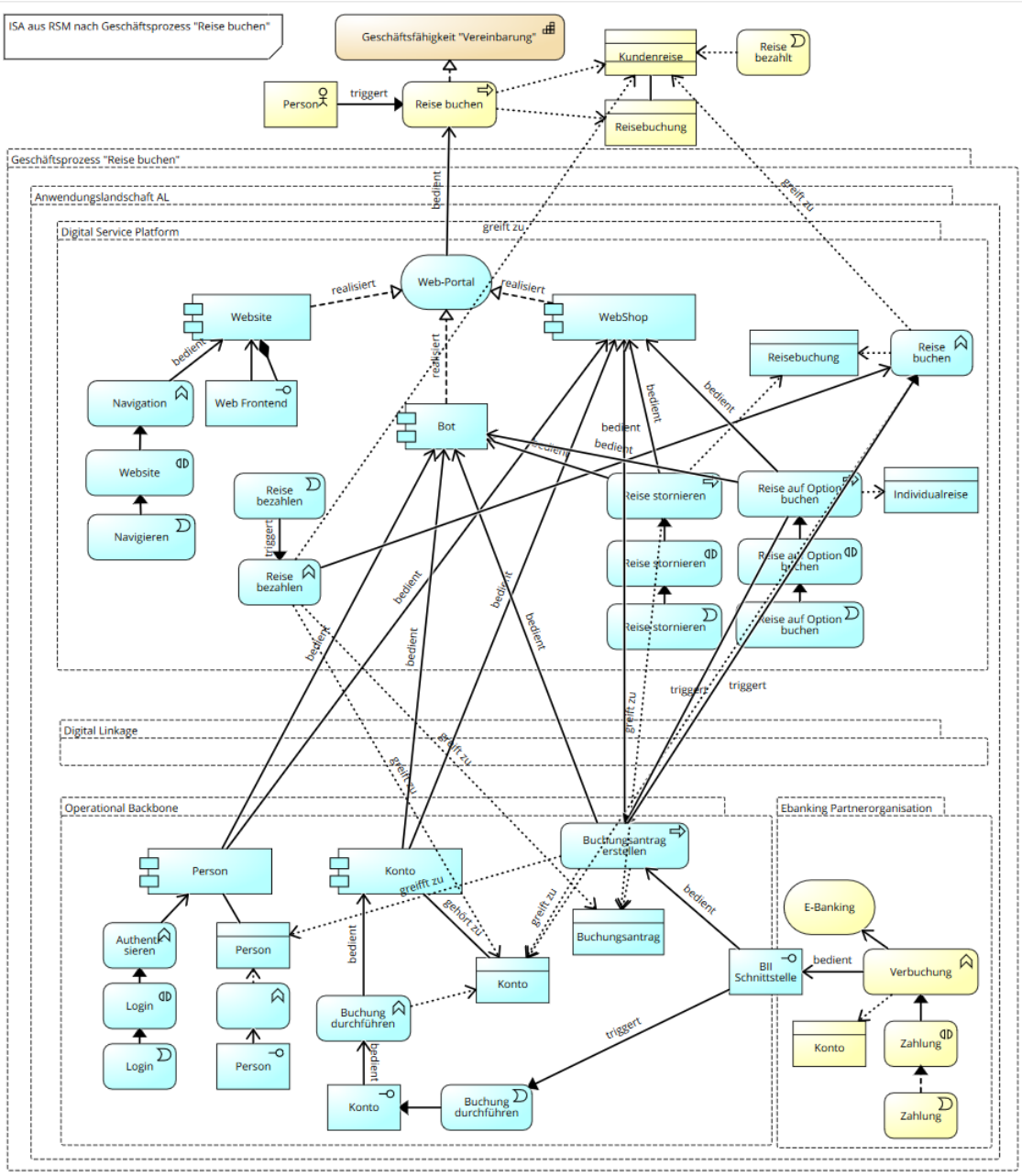


Abbildung 45: RSM-ISA-Artefakt für den Geschäftsprozess «Reise buchen» von AR

4.2 Beurteilung und Verbesserungsmassnahmen

Die Beurteilung des Design Science-Artefakts erfolgt gemäss Forschungs-Guideline 3 (Design Evaluation) respektive Forschungs-Guideline 6 (Design as a Search Process).

4.2.1 Forschungs-Guideline 3 (Design Evaluation)

Aus Sicht der analytischen Design-Bewertung des Artefakts kann festgestellt werden, dass das überprüfte Szenario RSM-IS-Architektur des Geschäftsprozesses «Reise buchen» von AR für den Aufbau nach den Vorgaben der RSM-ISA-Referenzarchitektur für

die umgesetzten Elemente geeignet ist. Der Architektur-Fit kann durch die Kombination der verschiedenen Patterns somit bestätigt werden.

Aus Sicht der deskriptiven, szenariobasierten Design-Bewertung lässt sich feststellen, dass die RSM-Granularität und die Vielschichtigkeit der Domänen eine über das Szenario ausgehende, umfassende Abbildung schnell unübersichtlich lassen werden und daher eine geschäftsfallspezifische ISA-Abbildung, wie sie angewendet wurde, sicher zweckmässig ist.

4.2.2 Forschungs-Guideline 6 (Design as a Search Process)

Die Abstützung auf die zukunftsfähige Patterns gewährleistet eine nachhaltige und agile Architektur. Die Anwendung nach den Vorgaben des RSM-ISA-Referenzartefaktes hat aber aufgezeigt, dass eine Umsetzung auf dem von RSM geforderten Granularitätslevel durch die Aufteilung in verschiedene Dimensionen schnell unübersichtlich wird.

Ein Verbesserungsansatz hierfür wäre eine kontextbasierte Unterteilung nach den einzelnen Komponenten des Geschäftsprozesses. So könnten auch die zusätzlichen Domänen wie Cloud und On-Premises, respektive die Pace Layers eingebunden werden, welche aus Übersichtlichkeitsgründen nicht im aktuellen Artefakt von AR integriert sind. Alternativ könnte eine vertikale Aufteilung nach verschiedenen Granularitätsstufen (Multitier-Architektur-Sicht) vorgenommen werden. Hierbei könnte beispielsweise eine Aggregation auf Komponentenstufe gemacht werden, ohne eine Kategorisierung nach Interaktion, Prozess, Funktion und Bestand vorzunehmen. Auf eine detaillierte IS-Architektur sollte aber nicht verzichtet werden, denn diese ist gerade bei einer fein nach Funktionalität unterteilten Microservice-Architektur wichtig, da hier eine Kategorisierung nach allen Domänen der Referenzarchitektur erforderlich ist.

5 Ergebnisse

In diesem Kapitel wird die Beantwortung der Forschungsfragen auf Basis dieser Arbeit unter Abschnitt 5.1 aufgegriffen und im nachfolgenden Abschnitt 5.2 kritisch diskutiert. Zum Abschluss werden unter Abschnitt 5.3 Ansätze für weiterführende Forschungen aufgeführt und die Arbeit abgerundet.

5.1 Beantwortung der Forschungsfragen

Die der Arbeit zugrundeliegenden Forschungsfragen konnten vollumfänglich beantwortet werden. Nachfolgend werden sie nochmals zusammenfassend aufgearbeitet.

FF 1: Was müssen zukunftsfähige «Ideal» IS-Architekturen im Rahmen der an sie gestellten Anforderungen leisten?

In Kapitel 3 wurde der Begriff «IS-Architektur» definiert, der Begriff «Ideal» diskutiert und das Ziel einer zukunftsfähigen IS-Architektur beschrieben. Anschliessend wurde im Abschnitt 3.4 anhand der Prinzipien-Pyramide der Zukunftsfähigkeit (vgl. Abbildung 18 auf Seite 29) mittels verschiedenen Perspektiven aufgezeigt, dass zukunftsfähige IS-Architekturen verschiedene Prinzipien befriedigen müssen, um proaktiv und reaktiv auf Einflüsse aus dem internen und externen Umfeld reagieren zu können. Im Abschnitt 3.5 wurden zehn unterstützende Paradigmen für diese Prinzipien hergeleitet, erläutert und zusätzlich anhand von konkreten Lösungstechnologien und Patterns gestützt.

FF 2: Wie lassen sich zukunftsfähige ISA-Patterns aufgrund ihrer Charakteristiken optimal zu einem ISA-Referenzarchitektur-Artefakt kombinieren?

Diese Fragestellung wurde anhand des Abschnittes 3.6 durch eine Analyse und Bewertung von ISA-Patterns beantwortet. Hierbei wurden verschiedene, im vorhergehenden Abschnitt 3.5 im Rahmen der Herleitung der ISA-Paradigmen, identifizierte ISA-Patterns A-F aufgegriffen, erläutert und ihre Charakteristiken anhand einer SWOT-Analyse bewertet. Im Anschluss wurden die gewonnenen Erkenntnisse nochmals zusammengefasst und diskutiert. Unter Abschnitt 3.7 wurden diese bewerteten Patterns dann zu einem RSM-ISA-Referenzarchitektur-Artefakt (vgl. Abbildung 41 auf Seite 80) kombiniert.

FF 3: Wie lässt sich anhand der Geschäftsfähigkeiten aus der Fallstudie «Arthur Reisen» nach Spichiger & Noser [SN] das «Ideal» RSM-ISA-Artefakt für Arthur Reisen gestalten?

Unter Kapitel 4 wurde das Ideal RSM-ISA-Artefakt anhand der szenariobasierten Fallstudie «Arthur Reisen» validiert. Hierfür wurden zuerst unter Abschnitt 4.1.1

die IT getriebenen Leistungsanforderungen aus einer Geschäftsmodellsicht verdichtet und anschliessend unter Abschnitt 4.1.2 die Geschäftsfähigkeiten-basierten Validierungsszenarios ausgearbeitet. Im Anschluss wurde unter Abschnitt 4.1.3 das RSM-ISA-Artefakt für AR auf Basis des Szenarios des AR Geschäftsprozesses «Reise buchen» gebildet (vgl. Abbildung 45 auf Seite 91). Die Beurteilung unter Abschnitt 4.2 hat hiernach gezeigt, dass die RSM-Granularität und die Vielschichtigkeit der Domänen eine weitergehende, umfassendere Abbildung schnell unübersichtlich werden lassen und eine geschäftsfallspezifische ISA-Abbildung zweckmässig ist. Allenfalls sollte aus Übersichtlichkeitsgründen sogar eine zusätzliche kontextbasierte Aufteilung vorgenommen werden.

5.2 Diskussion

Die vorliegende Arbeit orientiert sich an einem deduktiven Vorgehen zur Ergründung von zukunftsfähigkeitsfördernden Prinzipien, Paradigmen und Patterns von IS-Architekturen. Hierbei wird hauptsächlich das Ideal-Zielbild geschärft und durch konkrete Lösungstechnologien und bewertete Patterns gestützt. Ein Produkt dieser Arbeit sind nebst der Kombination verschiedener bewerteter Patterns auch die daraus hergeleiteten Artefakte selbst, welche sich gerade in Bezug auf die RSM-ISA-Referenzarchitektur vielfältig einsetzen lassen. Im Zusammenhang mit der Ist- und Soll-ISA-Landschaft werden nur die Umfeldeinflüsse berücksichtigt, aber keine spezifischen Hilfsartefakte ausgearbeitet, da diese sehr individuell ausfallen können. Ein essenzieller Aspekt für die wissenschaftliche Stringenz der Arbeit stellt die Einhaltung der Forschungs-Guidelines von Hevner et al. [HMR04] dar, welche im folgenden Abschnitt nochmals diskutiert und reflektiert werden.

5.2.1 Beurteilung zur Einhaltung der Forschungs-Guidelines

Die verwendete Design-Science Methodologie nach Hevner et al. [HMR04] war für diese Arbeit sehr zweckmässig. Die der Methodologie zugrundeliegenden Forschungs-Guidelines konnten hierbei eine wertvolle Hilfestellung zur Gewährleistung der Wissenschaftlichkeit liefern.

Anhand von Guideline 1 (Design as an Artefact) wurde so sichergestellt, dass die Artefakterzeugung (RSM-ISA-Referenzarchitektur) angestrebt wird.

Durch Guideline 2 (Problem Relevance) wurde sichergestellt, dass es sich um eine effektive Forschungslücke handelt, nämlich der aktuellen Herausforderung von Unternehmen, die zukunftsfähige Informationssystem-Architekturen anstreben müssen, kombiniert mit der Ausarbeitung des ressourcenspezifischen Informationssystem-Architekturmodells der Geschäftsfähigkeiten-basierten Unternehmensarchitektur nach Spichiger & Noser.

Zur Ausrichtung nach Guideline 3 (Design Evaluation) wurde sichergestellt, dass die Design-Evaluationsmethoden erstens analytisch und zweitens deskriptiv eingehalten wurden. Dies bedeutet, dass eine Architektur-Analyse kombinatorisch und aus verschiedenen Perspektiven gemacht wurde und die Anwendbarkeit deskriptiv anhand verschiedener fallstudienbasierter Leistungsanforderungen und Szenarien bewertet wurde. Hier kann ein möglicher Kritikpunkt sein, dass noch weitere Szenarien hätten getestet werden müssen, um durch eine umfassendere Validierung der Referenzarchitektur die Belastbarkeit des Artefaktes sicherzustellen und es gegebenenfalls noch weiter zu verbessern.

Durch die deduktive, nachvollziehbare Herleitung und Erzeugung der beiden Artefakte «RSM-ISA-Referenzarchitektur» und «RSM-ISA-AR» konnte sichergestellt werden, dass Hevner's Guideline 4 (Research Contributions) bedient wird, womit der Beitrag zur Forschung gewährleistet ist.

Anhand der Abstützung auf wissenschaftlich hergeleitete Modelle[30], kombiniert mit verbreiteten Best-Practice Lösungen[31] und der stringenten Ausrichtung nach den Methodologien[32], konnte die Einhaltung der Guideline 5 (Research Rigor) sichergestellt werden.

Anhand der Berücksichtigung und Kombination verschiedener Architektur-Patterns auf verschiedenen Architektur-Ebenen konnte Guideline 6 (Design as a Search Process) sichergestellt werden. Ein möglicher Kritikpunkt hierbei ist die Validierung anhand nur eines Szenarios aus der Fallstudie. Das Artefakt sollte für eine breitere Abstützung sinnvollerweise durch zusätzliche Szenarien und Experten-Interviews validiert werden, um einer möglichen Subjektivität entgegenzuwirken. Die praktische (Teil-)Umsetzung des Artefaktes zur Validierung wäre zusätzlich erstrebenswert.

Die Kommunikation der Forschung wird gemäss Guideline 7 (Communication of Research) durch die Integration des RSM-ISA-AR-Artefakts in das entstehende Buch «Geschäftsfähigkeiten-basierte Unternehmensarchitektur» und das zur Verfügung stellen der Arbeit in der Bibliothek der Berner Fachhochschule und der Bibliothek der Züricher Hochschule für Angewandte Wissenschaften sichergestellt.

5.2.2 Fazit

Die vorliegende Arbeit beantwortet alle gestellten Forschungsfragen. Jedoch könnten Forschungslücken bestehen, welche aus einer nicht auszuschliessenden Subjektivitätsbehaftung der Arbeit – wie beispielsweise durch nicht berücksichtigte Paradigmen oder Patterns – bestehen können. Die aus dieser Arbeit resultierten Ergebnisse könnten durch Experteninterviews überprüft, gestützt oder ergänzt werden. Durch die thematisch breit abgestützte, nachvollziehbare, zukunftsorientierte Umsetzung dieser Arbeit ermöglicht diese – nur

[30] vgl. Ross et al. [RSBJ17], Ziemann [Zie10]
[31] vgl. Engels et al. [EJH+08] und Schmutz [Sch14]
[32] vgl. Hevner et al. [HMR04], Simon [Sim96] und Yin [Yin09]

schon bei einer Teilumsetzung – die Gestaltung einer nachhaltig ausbau- und zukunftsfähigen Informationssystem-Architektur und fördert die Sensibilisierung der Unternehmen für dieses wichtige Thema.

5.3 Ausblick und weiterführende Forschungen

«Prognosen sind eine schwierige Sache. Vor allem, wenn sie die Zukunft betreffen.»[33]. Wie dieses Zitat andeutet, ist das Thema Zukunftsfähigkeit – gerade in der heutigen schnelllebigen Zeit – mit einem Unsicherheitsfaktor versehen. Um dieser Unsicherheit entgegenwirken zu können, sollte eines iterativer Suchprozesses angewendet werden. Die Bewertung der ISA-Patterns und die Ausgestaltung des zukunftsfähigen RSM-ISA-Referenzarchitektur-Artefakts basieren auf einer Momentaufnahme und sollten daher fortlaufend überprüft werden. Die zugrundeliegenden Rahmenbedingungen in Form von Umfeldeinflüssen und Prinzipien sollten aufgrund ihrer generischen Art aber relativ stabil bleiben. Wichtiger für eine breit abgestützte Referenzarchitektur ist es daher, bestehende Paradigmen, Lösungstechnologien und Patterns regelmässig zu überprüfen und das Artefakt gegebenenfalls durch Ergänzungen zu erweitern.

Aus wissenschaftlicher Sicht sollte, wie auch im vorhergehenden Abschnitt 5.2 erläutert, eine weiterführende Forschung zum Entkräften der Subjektivität anhand von Experteninterviews durchgeführt werden. Zudem wäre eine (Teil-)Umsetzung der Architektur zur Validierung des Artefaktes sinnvoll. Diese Vorhaben bedürfen jedoch eines Zeithorizonts, der über den Rahmen dieser wissenschaftlichen Arbeit hinaus geht.

Zusammenfassend erscheinen folgende, ergänzenden Forschungen zweckmässig:

- Überprüfen der Paradigmen (Experteninterview)
- Überprüfen der Patterns (Experteninterview)
- Überprüfen der Lösungstechnologien (Experteninterview)
- Szenariobasiertes ausarbeiten und Testen des Artefaktes
- Validieren des Artefaktes anhand einer (Teil-)Umsetzung der Architektur

Durch das zusätzliche qualitative Verifizieren der Artefakte lässt sich ein hohes Mass an praktischer und wissenschaftlicher Relevanz für ein Thema festigen, welches Unternehmen stärker den je beschäftigen wird.

[33] Zitat von Samuel Langhorne Clemens, besser bekannt unter dem Namen Mark Twain, US-amerikanischer Erzähler und Satiriker (1835 - 1910)

Literatur

[AD05] AIER, Stephan ; DOGAN, Turgut: Indikatoren zur Bewertung der Nachhaltigkeit von Unternehmensarchitekturen. Version: 2005. http://link.springer.com/10.1007/3-7908-1624-8_32. In: FERSTL, Otto K. (Hrsg.) ; SINZ, Elmar J. (Hrsg.) ; ECKERT, Sven (Hrsg.) ; ISSELHORST, Tilman (Hrsg.): *Wirtschaftsinformatik 2005*. Heidelberg : Physica-Verlag HD, 2005. – ISBN 978–3–7908–1574–0 978–3–7908–1624–2, 607–626

[AG04] ANDRESEN, Katja ; GRONAU, Norbert: Der Faktor Wandlungsfähigkeit bei der Planung neuer Fabriken - Ein Marktüberblick von Unternehmensberatungen im Bereich Fabrikplanung. Version: 2004. https://publishup.uni-potsdam.de/opus4-ubp/frontdoor/deliver/index/docId/607/file/WI_2005_19.pdf. 2004 (4). – Forschungsbericht. – S. 34–40

[AGS05] ANDRESEN, Katja ; GRONAU, Norbert ; SCHMID, Simone: Ableitung von IT-Strategien durch Bestimmung der notwendigen Wandlungsfähigkeit von Informationssystemarchitekturen. Version: 2005. http://link.springer.com/10.1007/3-7908-1624-8_4. In: FERSTL, Otto K. (Hrsg.) ; SINZ, Elmar J. (Hrsg.) ; ISSELHORST, Tilman (Hrsg.): *Wirtschaftsinformatik 2005*. Heidelberg : Physica-Verlag HD, 2005. – ISBN 978–3–7908–1574–0 978–3–7908–1624–2, 63–82

[AH16] ATILGAN, M. ; HURST, M.: Geschäftsarchitektur Arthur Reisen / Berner Fachhochschule Fachbereich Wirtschaft. Bern, 2016. – Forschungsbericht

[Avr16] AVRAM, Abel: *FaaS, PaaS, and the Benefits of the Serverless Architecture*. https://www.infoq.com/news/2016/06/faas-serverless-architecture. Version: Juni 2016

[BCK03] BASS, Len ; CLEMENTS, Paul ; KAZMAN, Rick: *Software architecture in practice*. 2nd ed. Boston : Addison-Wesley, 2003 (SEI series in software engineering). – ISBN 978–0–321–15495–8

[BFKT14] BONÉR, Jonas ; FARLEY, Dave ; KUHN, Roland ; THOMPSON, Martin: *Das Reaktive Manifest*. https://www.reactivemanifesto.org/de. Version: September 2014

[BJPS06] BECKER, Jörg ; JANIESCH, Christian ; PFEIFFER, Daniel ; SEIDEL, Stefan: Evolutionary method engineering: towards a method for the analysis and conception of management information systems. In: *AMCIS 2006 Proceedings* (2006), S. 470

[Bla86] BLASER, Werner: Architekturprinzipien nach Vitruv - eine Gegenwartsdeutung. (1986). http://dx.doi.org/10.5169/seals-76285. – DOI 10.5169/seals–76285

[Bä17] BÄRTSCHI, Michael: Analyse und Simulation des Geschäftsmodells von Arthur Reisen mittels System Dynamics. In: *Berner Fachhochschule* (2017), S. 65

[Bü14] BÜTTNER, Von Jean-Martin: «Die Daten über uns sind das Erdöl der Neuzeit». In: *Der Bund* (2014), Juni. //www.derbund.ch/digital/daten/Die-Daten-ueber-uns-sind-das-Erdoel-der-Neuzeit/story/30037334. – ISSN 0774–6156

[CKKK06] CHOI, Duke H. ; KIM, Chul M. ; KIM, Sang-Il ; KIM, Soung H.: Customer loyalty and disloyalty in internet retail stores: its antecedents and its effect on customer price sensitivity. In: *International Journal of Management* 23 (2006), Nr. 4, S. 925

[Con68] CONWAY, Melvin: Home Page of Mel Conway's Site. In: *F. D. Thompson Publications, Inc.* Datamation (1968), April. http://www.melconway.com/research/committees.html

[Dav00] DAVIS, Gordon B.: Information Systems Conceptual Foundations: Looking Backward and Forward. Version: 2000. http://link.springer.com/10.1007/978-0-387-35505-4_5. In: STAGE, Jan (Hrsg.) ; DEGROSS, Janice I. (Hrsg.) ; BASKERVILLE, Richard (Hrsg.): *Organizational and Social Perspectives on Information Technology*. Boston, MA : Springer US, 2000. – ISBN 978–1–4757–6107–8 978–0–387–35505–4, 61–82

[Doc17] DOCKER: *What is a Container*. https://www.docker.com/what-container. Version: Januar 2017

[EFNV16] ESTERMANN, Beat ; FRAEFEL, Marianne ; NEURONI, Alessia ; VOGEL, Jürgen: Conceptualizing a National Data Infrastructure for Switzerland. In: *EGPA, Utrecht, the Netherlands* (2016)

[EJH+08] ENGELS, Gregor ; JUWIG, Oliver ; HESS, Andreas ; HUMM, Bernhard ; LOHMANN, Marc: *Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten*. dpunkt. verlag, 2008

[EU17] EU, Council of t.: *Tallinn Declaration on eGovernment*. https://www.eu2017.ee/sites/default/files/2017-10/Tallinn_eGov_declaration.pdf. Version: Oktober 2017

[FB99] FOX, Armando ; BREWER, Eric A.: Harvest, yield, and scalable tolerant systems. In: *Hot Topics in Operating Systems, 1999. Proceedings of the Seventh Workshop on*, IEEE, 1999, S. 174–178

[FLS18] FEESS, Eberhard ; LACKES, Richard ; SIEPERMANN, Markus: *Definition: Paradigma*. https://wirtschaftslexikon.gabler.de/definition/paradigma-42740/version-266083. Version: Februar 2018

[FN86] FAHEY, Liam 1951-(viaf)109068782 ; NARAYANAN, V. K.: *Macroenvironmental analysis for strategic management*. Saint Paul (Minn.) : West publishing Co., 1986 http://lib.ugent.be/catalog/rug01:000163633. – ISBN 0–314–85233–6

[Fow05] FOWLER, Martin: *Service Oriented Ambiguity*. https://martinfowler.com/bliki/ServiceOrientedAmbiguity.html. Version: Juli 2005

[Fow11] FOWLER, Martin: *CQRS*. https://martinfowler.com/bliki/CQRS.html. Version: Juli 2011

[Fow12] FOWLER, Martin: *CAP Twelve Years Later: How the "Rules"Have Changed*. https://www.infoq.com/articles/cap-twelve-years-later-how-the-rules-have-changed. Version: Mai 2012

[Fow14] FOWLER, Martin: *Microservices*. https://martinfowler.com/articles/microservices.html. Version: März 2014

[Fow16] FOWLER, Martin: *Serverless Architectures*. https://martinfowler.com/articles/serverless.html. Version: August 2016

[Gab06] GABLER, Springer: Grundlagen moderner Organisationsgestaltung. (2006), S. 5

[Gau18] GAUGHAN, Dennis: Use Bimodal and Pace-Layered IT Together to Deliver Digital Business Transformation. In: *Gartner* (2018), April, S. 15

[GL] GILBERT, Seth ; LYNCH, Nancy: Brewer's Conjeture and the Feasibility of Consistent, Available,Partition-Tolerant Web Servies. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.20.1495&rep=rep1&type=pdf. – Forschungsbericht

[Gro00] GROUP, The O.: *IEEE 1471-2000 - IEEE Recommended Practice for Architectural Description for Software-Intensive Systems*. https://standards.ieee.org/findstds/standard/1471-2000.html. Version: 2000

[Gro06] GRONAU, N.: *Wandlungsfähige Informationssystemarchitekturen - Nachhaltigkeit bei organisatorischem Wandel (2. Auflage)*. Gito-Verlag, 2006 (Reihe Wirtschaftsinformatik). https://books.google.ch/books?id=r7KwU5NZ7wcC. – ISBN 978–3–936771–60–2

[Gro16] GROUP, The O.: *Microservices Architecture – SOA and MSA*. http://www.opengroup.org/soa/source-book/msawp/p3.htm. Version: 2016

[Hal88] HALL, R.H.: Organizations: Structures, Processes and Outcomes. In: *Organization Studies* 9 (1988), Nr. 2, 282–282. http://dx.doi.org/10.1177/017084068800900217. – DOI 10.1177/017084068800900217

[Her16] HERT, Matthias: *Microservices-Architektur – Hype oder neue Realität?* https://ipt.ch/microservices-architektur-hype-oder-neue-realitaet/. Version: 2016

[HHSW07] HORX, Matthias ; HUBER, Jeanette ; STEINLE, Andreas ; WENZEL, Eike: *Zukunft machen*. Campus Verlag, 2007 http://www.horx.com/Zukunftsforschung/Docs/02-M-03-Trend-Definitionen.pdf

[HM16] HERZWURM, Georg ; MIKUSZ, Martin: *Qualitätsmerkmale von Software*. http://www.enzyklopaedie-der-wirtschaftsinformatik.de/lexikon/is-management/Systementwicklung/Management-der-Systementwicklung/Software-Qualitatsmanagement/Qualitatsmerkmale-von-Software/index.html. Version: Dezember 2016

[HMR04] HEVNER, Alan R. ; MARCH, Salvatore T. ; RAM, Sudha: Design Science in Information Systems Research. In: *MIS Quarterly* 28 (2004), Nr. 1, S. 75–105

[Inn17] INNERN, Bundesministerium des: *Daten als Rohstoff der Zukunft*. http://www.bmi.bund.de/SharedDocs/pressemitteilungen/DE/2017/01/open-data-gesetz.html. Version: Januar 2017

[Ins16] INSTITUT, Johner: *Interoperabilität: Zusammenarbeiten der IT-Systeme sicherstellen*. https://www.johner-institut.de/blog/tag/interoperabilitat/. Version: 2016

[IWB17] IWB: *Energieanlagen jederzeit im Blick*. https://www.iwb.ch/Ueber-uns/Kraftwerke/IoT_Betriebsfuehrung.html. Version: 2017

[Jac82] JACKSON, Michael A.: A system development method. In: *Tools and notions for program construction: An advanced course* (1982), S. 1–25

[Jac18] JACOBS, Steffen: *Function as a Service*. https://blog.oio.de/2018/02/14/function-as-a-service/. Version: Februar 2018

[Kau15] KAUFMANN, Timothy: Datenzentrierte Geschäftsmodelle. Version: 2015. http://link.springer.com/10.1007/978-3-658-10272-2_2. In: *Geschäftsmodelle in Industrie 4.0 und dem Internet der Dinge*. Wiesbaden : Springer Fachmedien Wiesbaden, 2015. – ISBN 978–3–658–10271–5 978–3–658–10272–2, 11–30

[Krc90] KRCMAR, Helmut: Bedeutung und Ziele von Informationssystem-Architekturen. (1990), S. 9

[Kre14] KREPS, Jay: *Questioning the Lambda Architecture*. https://www.oreilly.com/ideas/questioning-the-lambda-architecture. Version: Juli 2014

[Kru98] KRUEGER, Wilfried: Management permanenten Wandels. In: *Organisation im Wandel der Märkte*. Wiesbaden, 1998

[KTGH18] KLEIN, Dominik ; TRAN-GIA, Phuoc ; HARTMANN, Matthias: *Big Data*. https://gi.de/informatiklexikon/big-data/. Version: Mai 2018

[LH98] LEWIN, A ; HUNTER, S: Information Technology & Organizational Design: A Longitudinal Study of Information Technology Implementations in the U.S. Retailing Industry, 1980-1996. In: *Organisation im Wandel der Märkte*. Wiesbaden, 1998

[LL12] LAUDON, Kenneth C. ; LAUDON, Jane P.: *Management Information Systems*. 12. New Jersey : Pearson Education, Inc., 2012. – ISBN 978–0–13–214285–4

[Mar11] MARZ, Nathan: *How to beat the CAP theorem - thoughts from the red planet - thoughts from the red planet*. http://nathanmarz.com/blog/how-to-beat-the-cap-theorem.html. Version: Oktober 2011

[MHB17] MARZ, Nathan ; HAUSENBLAS, Michael ; BIJNENS, Nathan: *Lambda Architecture*. http://lambda-architecture.net/. Version: 2017

[MRB11] MACCORMACK, Alan ; RUSNAK, John ; BALDWIN, Carliss Y.: Exploring the Duality between Product and Organizational Architectures: A Test of the Mirroring Hypothesis. In: *SSRN Electronic Journal* (2011). http://dx.

doi.org/10.2139/ssrn.1104745. – DOI 10.2139/ssrn.1104745. – ISSN 1556–5068

[MSL16] MÜLLER-STEWENS, Günter ; LECHNER, Christoph: *Strategisches Management: wie strategische Initiativen zum Wandel führen: der Strategic Management Navigator*. 5., überarbeitete Auflage. Stuttgart : Schäffer-Poeschel Verlag, 2016. – ISBN 978–3–7910–3439–3 978–3–7992–6982–7. – OCLC: 913962079

[MV87] MATURANA, Humberto ; VARELA, Francisco: *The Tree of Knowledge: The Biological Roots of Human Understanding*. http://www.cybertech-engineering.ch/research/references/Maturana1988/maturana-h-1987-tree-of-knowledge-bkmrk.pdf. Version: 1987

[Nai83] NAISBITT, John: *Megatrends: Ten New Directions Transforming Our Lives*. 1983 https://dacemirror.sci-hub.tw/journal-article/d7a7685b25d2d67ce6c9488640398d12/10.1016@0007-68138390036-8.pdf. – ISBN 0–446–51251–6

[OPW11] OSTERWALDER, A. ; PIGNEUR, Y. ; WEGBERG, J.T.A.: *Business Model Generation: Ein Handbuch für Visionäre, Spielveränderer und Herausforderer*. Campus Verlag, 2011. – ISBN 978–3–593–39474–9

[Pet11] PETTEY, Christy: Gartner Says Solving 'Big Data' Challenge Involves More Than Just Managing Volumes of Data / Gartner. Version: Juni 2011. https://www.gartner.com/newsroom/id/1731916. 2011. – Forschungsbericht

[Pfa97] PFAU, Wolfgang: Flexibilitätspotentiale zur Verbesserung der Migrationsfähigkeit der Informationsinfrastruktur. Version: 1997. http://www.springerlink.com/index/10.1007/978-3-663-08281-1_1. In: *Betriebliches Informationsmanagement*. Wiesbaden : Deutscher Universitätsverlag, 1997. – ISBN 978–3–8244–6613–9 978–3–663–08281–1, 1–3

[Por85] PORTER, Michael E.: *Competitive advantage: creating and sustaining superior performance*. New York : London : Free Press ; Collier Macmillan, 1985. – ISBN 978–0–02–925090–7

[RSBJ17] ROSS, Jeanne W. ; SEBASTIAN, Ina M. ; BEATH, Cynthia M. ; JHA, Lipsa: Designing Digital Organizations - Summary of Survey Findings / MIT CISR. Version: Februar 2017. https://media-publications.bcg.com/MIT-CISR-Designing-Digital-Survey.PDF. 2017 (415). – Forschungsbericht. – 21 S.

[RWR06] ROSS, Jeanne W. ; WEILL, Peter ; ROBERTSON, David: *Enterprise architecture as strategy: creating a foundation for business execution*. Boston, Mass : Harvard Business School Press, 2006. – ISBN 978–1–59139–839–4. – OCLC: ocm66463473

[Rö17] RÖWEKAMP, Lars: *Serverless Computing, Teil 1: Theorie und Praxis*. https://www.heise.de/developer/artikel/Serverless-Computing-Teil-1-Theorie-und-Praxis-3756877.html. Version: April 2017

[SBBD01] SPATH, D ; BAUMEISTER, M ; BARRHO, T ; DILL, C: Change management im Wandel. In: *Industriemanagement* 17 (2001), Nr. 4, S. 9–13

[Sch91] SCHEER, August-Wilhelm: *Architektur integrierter Informationssysteme: Grundlagen der Unternehmensmodellierung*. 1991 http://link.springer.com/openurl?genre=book&isbn=978-3-642-97334-5. – ISBN 978–3–642–97333–8 978–3–642–97334–5 978–3–642–97403–8 978–3–642–97404–5. – OCLC: 913802861

[Sch14] SCHMUTZ, Guido: Big Data und Fast Data - Lambda Architektur und deren Umsetzung. In: *DOAG Konferenz 2014* (2014), November, S. 49

[Sch17a] SCHMUTZ, Guido: *Best Practice Big Data Architekturen*. 2017

[Sch17b] SCHWARTZ, Alexander: Microservices: Mehr als nur ein Hype? In: *Informatik-Spektrum* 40 (2017), Dezember, Nr. 6, 590–594. http://dx.doi.org/10.1007/s00287-017-1078-6. – DOI 10.1007/s00287–017–1078–6. – ISSN 0170–6012, 1432–122X

[Sch18] SCHMUTZ, Guido: *Building event-driven Microservices with Kafka Ecosystem*. März 2018

[Sim96] SIMON, H. A.: The Sciences of the Artificial. In: *MIT Press, Cambridge, MA* (1996), Nr. 3rd Edition

[Sin02] SINZ, Elmar J.: Architektur von Informationssystemen. In: RECHENBERG, Peter (Hrsg.) ; POMBERGER, Gustav (Hrsg.): *Informatik-Handbuch*. 3. Auflage. München : Hanser, 2002, S. S. 1055 – 1068

[SN] SPICHIGER, Andreas ; NOSER, Philipp: *Geschäftsfähigkeitenbasierte Unternehmensarchitektur (in Arbeit)*. Bern,

[Soc00] SOCIETY, IEEE C.: *IEEE Recommended Practice for Architectural Description of Software-Intensive Systems*. Place of publication not identified

: publisher not identified, 2000 http://ieeexplore.ieee.org/servlet/opac?punumber=7040. – ISBN 978–0–7381–2518–3. – OCLC: 812593980

[SP14] SCHNEIDER-PUNGS, Tobias: *Predictive Maintenance: Two Sides Of A Coin*. https://www.digitalistmag.com/industries/manufacturing-industries/2014/10/27/predictive-maintenance-two-sides-coin-01632217. Version: Oktober 2014

[Sta11] STANDARDIZATION, International O.: *ISO 25010:2011(en), Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models*. https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en. Version: 2011

[Ter17] TEREKHOVA, Maria: *Lemonade shakes up the insurance industry*. http://www.businessinsider.com/lemonade-shakes-up-the-insurance-industry-2017-9. Version: September 2017

[Tre18] TRESH, Anatole: *The Big Five - IT Achitektur Heute*. März 2018

[Tru06] TRUYEN, Frank: The Fast Guide to Model Driven Architecture. (2006), Januar, S. 16

[UW16] ULLRICH, André ; WEBER, Edzard: Einsatz stilisierter Fakten zur Bewertung wandlungsfähiger Unternehmensarchitekturen. (2016), S. 15

[VBN14] VON BRAUK, Stefan ; NEUDERT, Christian: *Prinzipien skalierbarer Architektur*. https://jaxenter.de/prinzipien-skalierbarer-architektur-848. Version: Juli 2014

[WC12] WELLER, Bart ; CALCOTT, Lori: *The definitive guide to Google AdWords: create versatile and powerful marketing and advertising campaigns*. New York, NY : Apress, Springer, 2012 (The expert's voice in web development). – ISBN 978–1–4302–4014–3 978–1–4302–4015–0. – OCLC: 795395712

[Wik18] WIKIPEDIA: *Representational State Transfer*. https://de.wikipedia.org/w/index.php?title=Representational_State_Transfer&oldid=176563228. Version: April 2018. – Page Version ID: 176563228

[Yin09] YIN, R.K.: *Case Study Research: Design and Methods*. SAGE Publications, 2009 (Applied Social Research Methods). https://books.google.ch/books?id=FzawIAdilHkC. – ISBN 978–1–4129–6099–1

[Zac87] ZACHMAN, J.A.: A Framework for Information Systems Architecture. In: *IBM Systems Journal* Volume 26 (1987), Nr. Number 3. https://www.zachman.com/images/ZI_PIcs/ibmsj2603e.pdf

[Zac02] ZACHMAN, John: The zachman framework for enterprise architecture. In: *Zachman International* 79 (2002)

[Zea18] ZEALLEY, John: Marketers Need to Stop Focusing on Loyalty and Start Thinking About Relevance. In: *Harward Business Review* (2018), März

[Zie10] ZIEMANN, Jörg: *Architecture of Interoperable Information Systems: An Enterprise Model-Based Approach for Describing and Enacting Collaborative Business Processes*. Bd. 23. Logos Verlag Berlin GmbH, 2010

Anhang

A Qualitätsmodell für externe und interne Qualität nach ISO 25010

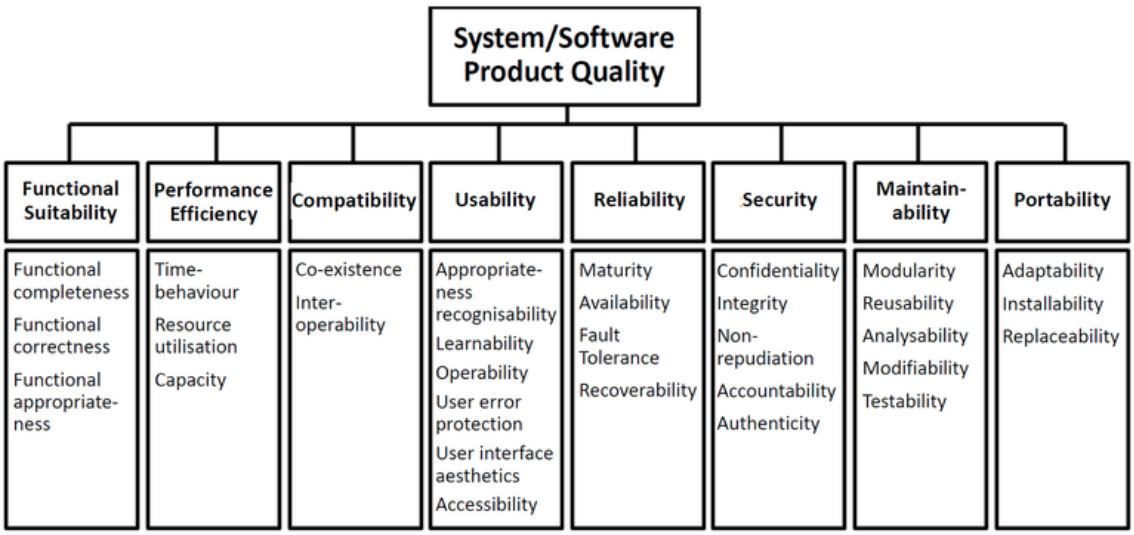


Abbildung 46: Qualitätsmodell für externe und interne Qualität nach ISO 25010; Quelle: Herzwurm & Mikusz [HM16]

B Beispiel einer Pace Layered-Architektur

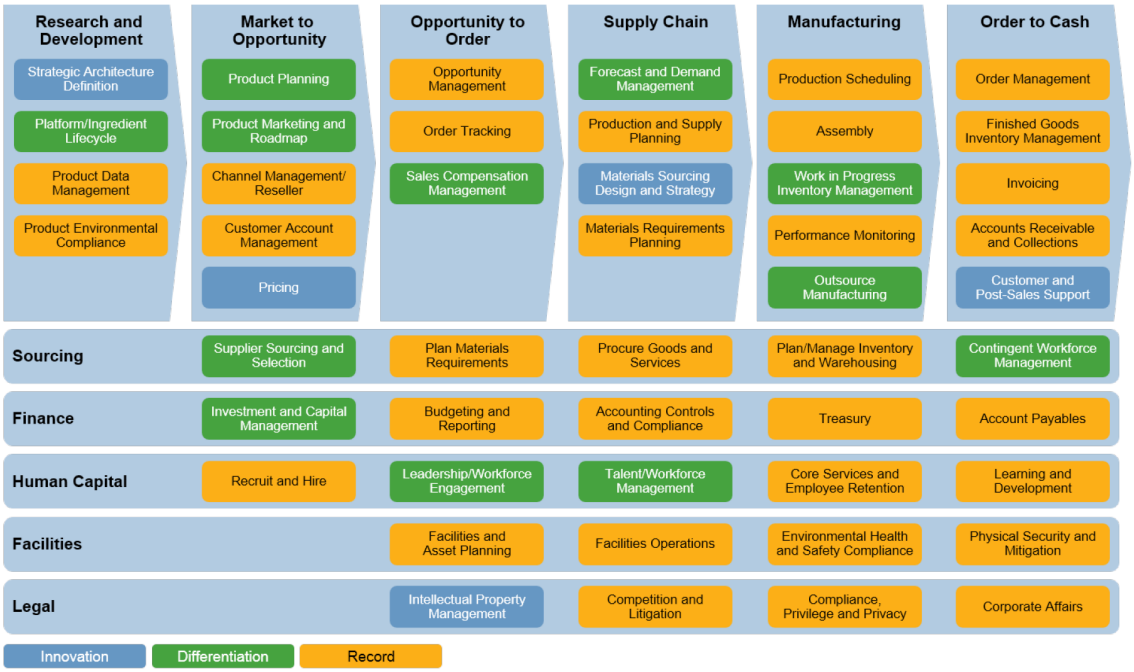


Abbildung 47: Beispiel einer Pace Layered-Architektur; Quelle: Gaughan [Gau18]

C Alternative ISA-RSM-Validierungsszenarios nach Wertangebot AR

Abgeleitete ISA-RSM-Validierungsszenarios nach Wertangebot von Arthur Reisen		
ID	AR Wertangebot	AR Leistungsanforderungen Mapping
1	Mandantenfähige Reiseangebote und rekursive Integration	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) Portierbarkeit (LAR 6) «Pay as you Go» (LAR 8) Mobilität und Verfügbarkeit (LAR 10) Dynamic Venturing (LAR 12) Portal Integration (LAR 13)
2	Standardangebot (Pauschalreise)	Unstrukturierte Medien (LAR 4) Erweiterungsfähigkeit (LAR 5) Mobilität und Verfügbarkeit (LAR 10) Portal Integration (LAR 13)
3	Individualangebot (Individualreise)	Unstrukturierte Medien (LAR 4) Erweiterungsfähigkeit (LAR 5) Mobilität und Verfügbarkeit (LAR 10) Portal Integration (LAR 13) «Pay as you Go» Nutzungsverrechnung für Kunden (LAR 8)
4	«Pay as you go» Nutzungsverrechnung (für Kunde und Partner)	Kooperationsfähigkeit (LAR 1) Mandantenfähigkeit (LAR 2) Interoperabilität (LAR 3/18) Portierbarkeit (LAR 6) Real-time Analytics (LAR 7) «Pay as you Go» (LAR 8) Mobilität und Verfügbarkeit (LAR 10) Dynamic Venturing (LAR 12) Portal Integration (LAR 13)

Tabelle 10: Abgeleitete ISA-RSM-Validierungsszenarios nach Wertangebot von Arthur Reisen

D Geschäftsprozess «Reise buchen» nach GFbUA

Nachfolgend sind die verwendeten Diagramme aus dem Prozess «Reise buchen» nach Spichiger & Noser [SN] dargestellt, nach welchen die szenariobasierte Validierung der RSM-ISA-Referenzarchitektur durchgeführt und zu einem RSM-ISA für AR gebildet wurde.

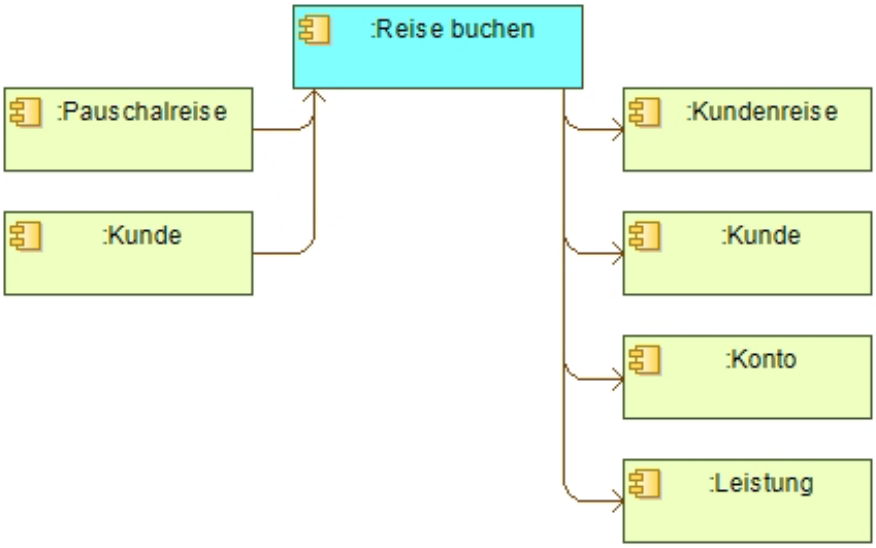


Abbildung 48: Überblick über den Geschäftsprozess «Reise buchen»; Quelle: Spichiger & Noser [SN, S.31]

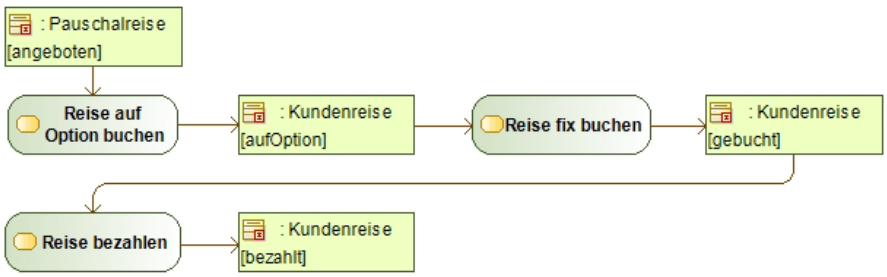


Abbildung 49: Geschäftstransaktionen des Geschäftsprozesses «Reise buchen»; Quelle: Spichiger & Noser [SN, S.32]

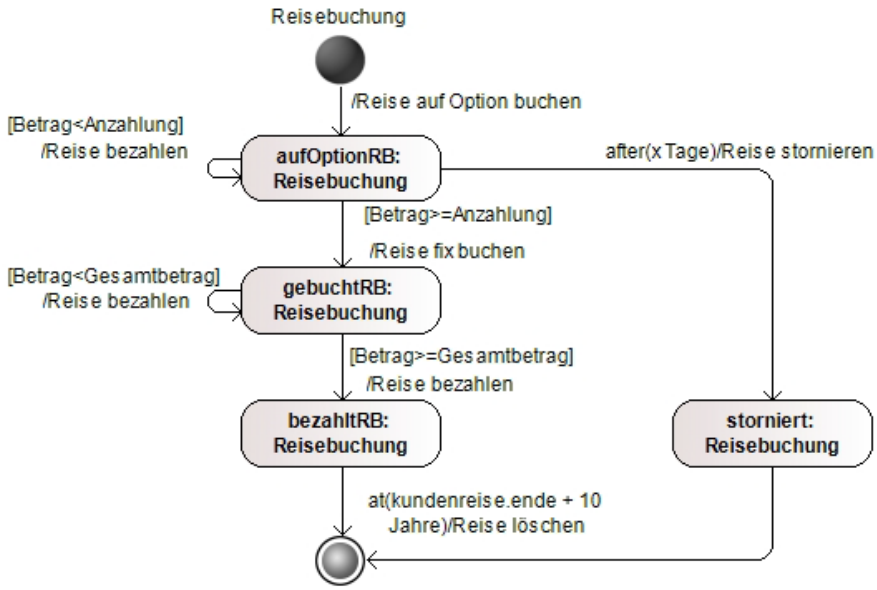


Abbildung 50: Zustandsdiagramm von «Reisebuchung»; Quelle: Spichiger & Noser [SN, S.32]

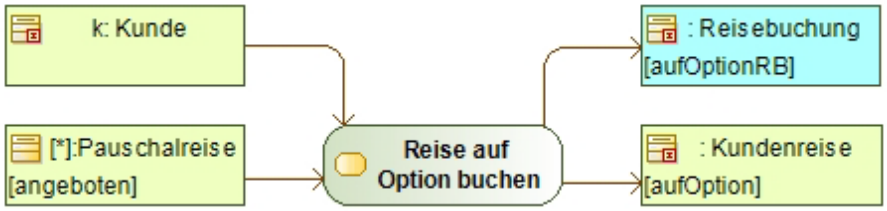


Abbildung 51: Geschäftstransaktion «Reise auf Option buchen»; Quelle: Spichiger & Noser [SN, S.32]

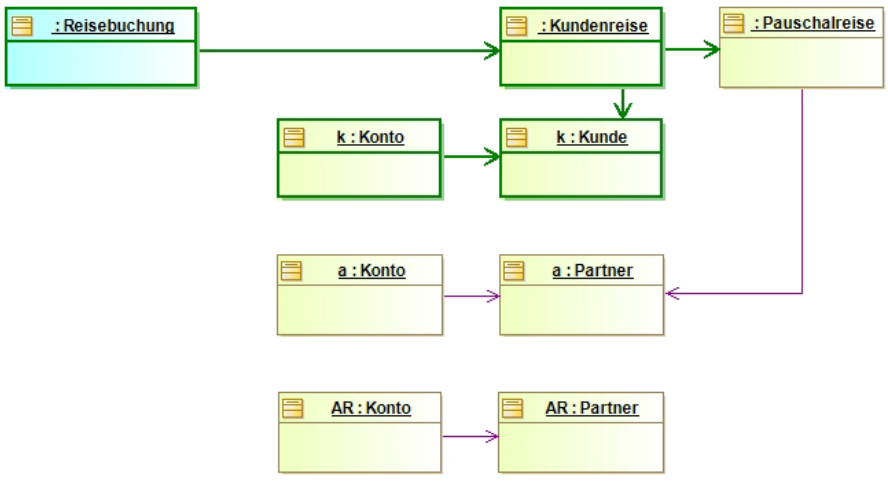


Abbildung 52: Objektdiagramm nach der Geschäftstransaktion «Reise auf Option buchen»; Quelle: Spichiger & Noser [SN, S.33]

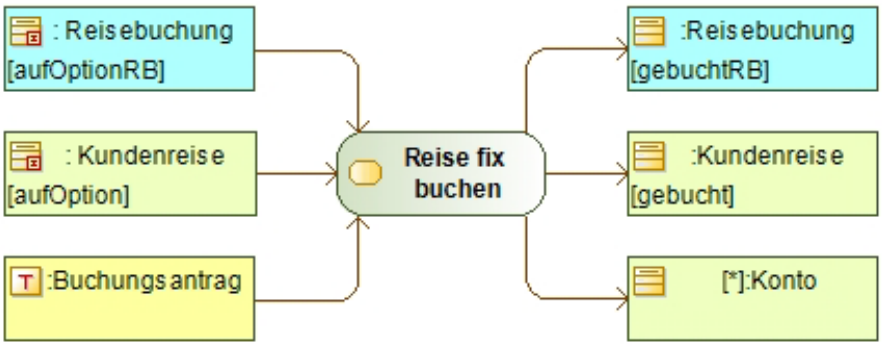


Abbildung 53: Geschäftstransaktion «Reise fix buchen»; Quelle: Spichiger & Noser [SN, S.33]

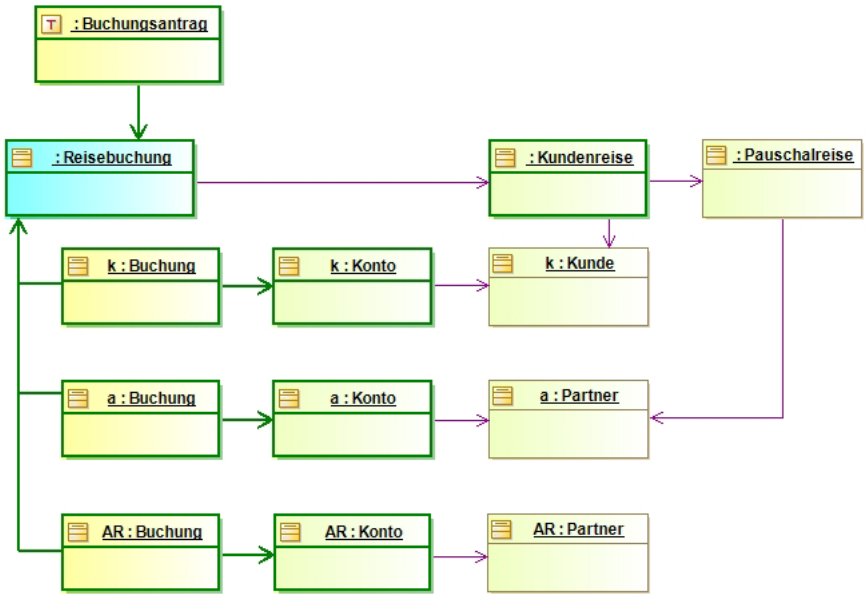


Abbildung 54: Objektdiagramm nach der Geschäftstransaktion «Reise fix buchen»; Quelle: Spichiger & Noser [SN, S.34]

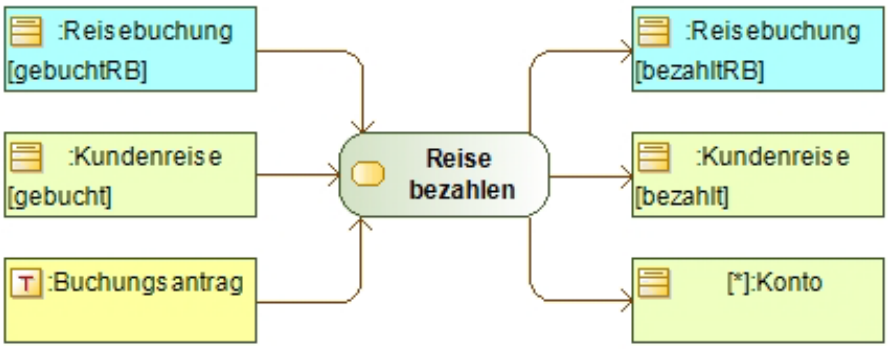


Abbildung 55: Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.34]

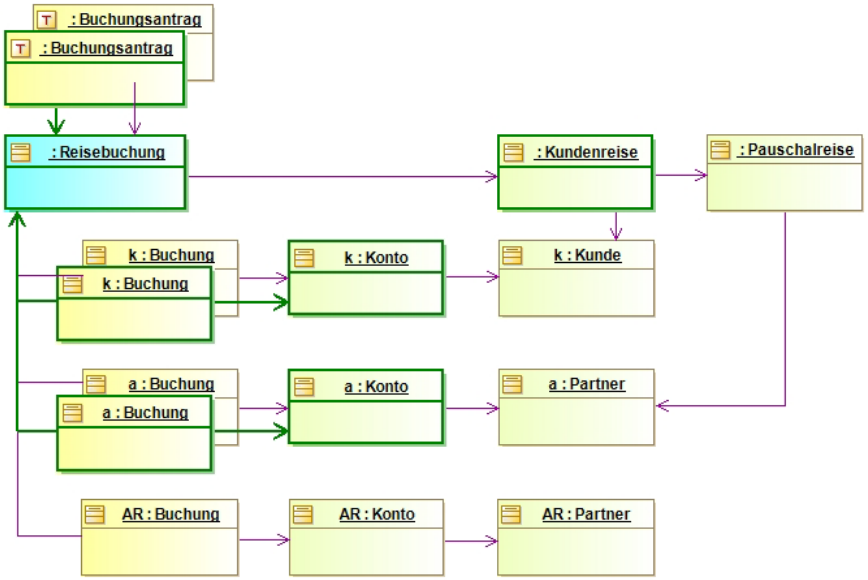


Abbildung 56: Objektdiagramm nach der Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.35]

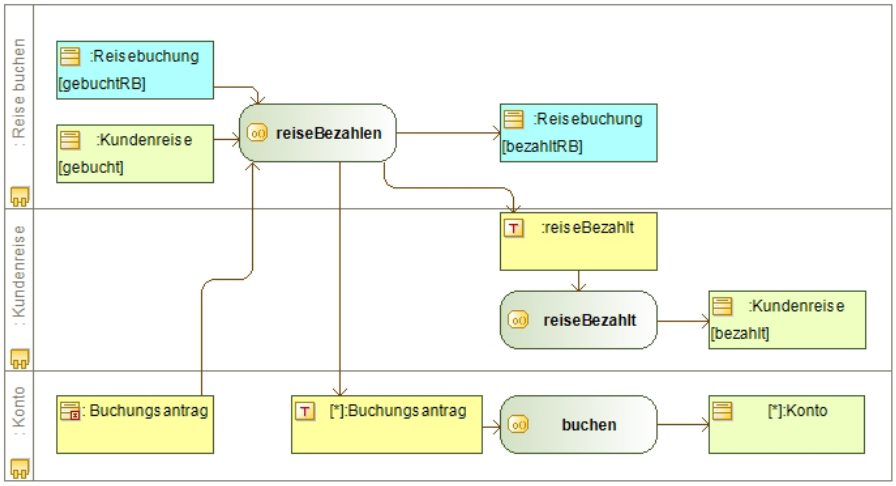


Abbildung 57: Umsetzung der Geschäftstransaktion «Reise bezahlen»; Quelle: Spichiger & Noser [SN, S.35]