Messung der Komplexität von IT-Landschaften auf der Basis von Architektur-Metamodellen: Ein generischer Ansatz und dessen Anwendung im Rahmen der Architektur-Transformation

Christian Schmidt

Scape Consulting GmbH
Westhafen Tower
Westhafenplatz 1
60327 Frankfurt
christian.schmidt@scape-consulting.de

Thomas Widjaja, Alexander Schütz

Fachgebiet Wirtschaftsinformatik | Software Business & Information Management
Technische Universität Darmstadt
Hochschulstraße 1
64289 Darmstadt
widjaja@is.tu-darmstadt.de
schuetz@is.tu-darmstadt.de

Abstract: Das Management von Komplexität entwickelt sich mehr und mehr zu einer Schlüsselfähigkeit innerhalb des IT-Managements. Bis heute existieren jedoch kaum wissenschaftlich fundierte Verfahren zur Messung und Bewertung von Komplexität auf Unternehmensebene. Ausgehend von bestehenden Forschungsarbeiten wird in diesem Beitrag ein generischer Ansatz zur Komplexitätsmessung von IT-Landschaften auf der Basis von Architektur-Metamodellen vorgestellt und operationalisiert. Der Ansatz ist durch eine universelle Anwendung des Entropie-Maßes gekennzeichnet und unabhängig vom eingesetzten Architektur-Framework nutzbar. Der praktische Einsatz der Methodik wird anhand eines Fallbeispiels über die Architektur-Transformation einer Versicherung demonstriert und erprobt. Aus den Einsatzerfahrungen werden zudem erste Empfehlungen zur Ausgestaltung eines übergreifenden Komplexitätsmanagements abgeleitet.

1 Einleitung

Seit der Einführung von Informationstechnologie (IT) ist die Komplexität der Unternehmens-IT kontinuierlich gewachsen. Inzwischen gehören die IT-Landschaften großer Unternehmen zu den komplexesten vom Menschen geschaffenen Systemen überhaupt [Za97]. In der Praxis schlägt sich die Komplexität in einer Vielzahl von Problemen nieder (vgl. [MLH11] [Wi04] [Na05]). So sind einerseits die Betriebs- und Ausfallrisiken erheblich angestiegen. Darüber hinaus sind die durch Wartung und Betrieb der IT verur-

Aus: Horbach, M. (Hrsg.): INFORMATIK 2013 – Proceedings 43. Jahrestagung der Gesellschaft für Informatik (GI), Koblenz, September 2013. Edition Lecture Notes in Informatics (LNI), Köllen Verlag, Bonn. S. 1261-1275.

sachten Kosten deutlich angewachsen (vgl. [Na05] [Wi04] [MLH11]). Gleichzeitig hat die erhöhte Komplexität auch zu einer zunehmenden ”Verkrustung” der IT und einem Verlust an Flexibilität gegenüber Veränderungen geführt. Dies spiegelt sich vor allem in hohen Entwicklungskosten und einer gestiegenen Produkteinführungszeit (”Time-to-Market”) wider.

Die Ursachen des Komplexitätswachstums sind vielfältig. So kann ein Anstieg der IT-Komplexität einerseits auf eine gewachsene Geschäfts-Komplexität und die damit verbundenen IT-Anforderungen zurückzuführen sein. Beispielsweise erfordern die Einführung neuer Produkte oder die Erfüllung regulatorischer Anforderungen häufig zusätzliche Funktionalitäten. Zum Zweiten kann IT-Komplexität ihre Ursache auch in der Art und Weise der IT-Implementierung haben. Theoretische Erklärungsansätze hierfür bilden die Stakeholder-Theorie sowie die Theorie der Software-Evolution. So unterliegen IT-Landschaften in der Regel einem multipolaren, d. h. von unterschiedlichen Anspruchsgruppen (Stakeholdern) mit unterschiedlicher (meist partikularer) Zielsetzung vorangetriebenen, Evolutionsprozess [SB11]. Die fortwärende Veränderung führt dabei nach dem ”Zweiten Gesetz der Software-Evolution” zu einem permanenten Anstieg der System-Komplexität [Le97]. Drittens kann eine gewachsene IT-Komplexität auch das Ergebnis bewusster Entscheidungen zur Adressierung nicht-funktionaler Aspekte wie IT-Sicherheit oder Hersteller-Unabhängigkeit (z. B. Dual-Vendor-Strategie) darstellen.

Aufgrund der vielfältigen Wirkungen muss Komplexität als eine der wesentlichen Steuerungsgrößen des (IT-) Managements angesehen werden. Vor diesem Hintergrund sind viele Unternehmen dazu übergegangen, die Komplexität ihrer IT übergreifend zu managen. Entsprechende Aufgaben werden dabei in der Regel durch das IT-Architekturmanagement wahrgenommen [Sc09]. Bis heute existieren jedoch kaum wissenschaftlich fundierte Verfahren zur Messung von Komplexität auf der Ebene von IT-Landschaften bzw. Unternehmensarchitekturen. Sofern in der Praxis überhaupt Messungen durchgeführt werden, überwiegt der Einsatz spezifischer Indikatoren, welche jeweils unterschiedliche Komplexitäts-Facetten abdecken. Hierbei handelt es sich häufig um einfache Zählgrößen (z. B. Anzahl Anwendungen) oder Verhältnismaße (z. B. Anzahl Anwendungen pro Funktion), woraus sich eine Reihe von Schwierigkeiten ergeben. So erlauben insbesondere Durchschnittswerte häufig keine differenzierte Betrachtung. Andererseits besteht das Problem der Vergleichbarkeit und Integration der verschiedenen Indikatoren, da diese meist auf unterschiedlichen Berechnungsmethoden und Skalen basieren.

Vor dem Hintergrund der genannten Defizite wird in diesem Beitrag ein generischer Ansatz zur Messung der Komplexität von IT-Landschaften entwickelt und im Hinblick auf die bestehenden Anwendungsmöglichkeiten untersucht. Als theoretische Grundlage dient dabei die Systemtheorie (Abschnitt 2). Im Hinblick auf die verwendeten Komplexitätsmaße werden bestehende Forschungsergebnisse aus dem Bereich der Wettbewerbskontrolle adaptiert (Abschnitt 3). Darauf aufbauend wird ein mathematisches Modell formuliert, welches die Kennzahlenberechnung auf der Grundlage beliebiger Metamodelle beschreibt (Abschnitt 4). Der Einsatz des Modells wird anschließend anhand eines Fallbeispiels demonstriert und erprobt (Abschnitt 5). Aus den Einsatzerfahrungen werden schließlich erste Hypothesen für die Ausgestaltung eines übergreifenden Komplexitätsmanagements abgeleitet (Abschnitt 6).

2 Komplexität von IT-Landschaften

2.1 IT-Landschaften und Unternehmensarchitekturen

Hinsichtlich der Begriffe Unternehmensarchitektur bzw. Enterprise Architecture existiert bis heute kein einheitliches Verständnis. Im vorliegenden Beitrag erfolgt eine Anlehnung an das Begriffssystem nach [Sc09] [FS01] [De11]. Ausgangspunkt ist dabei ein Verständnis eines Unternehmens als eines komplexen, offenen und dynamischen Systems. Der mit der Informationsverarbeitung befasste Teil dieses Systems wird entsprechend als Unternehmensinformationssystem (UIS) (auch IT-Landschaft) bezeichnet.

Die Betrachtung von Unternehmenssystem und UIS kann je nach Zielsetzung auf unterschiedlichen Abstraktionsebenen und aus unterschiedlichen Blickwinkeln erfolgen. Sie impliziert damit immer zugleich eine Modellbildung [Ro79]. Unter einem (System-) Modell wird dabei gemäß der Modelltheorie eine verkürzende und zweckorientierte Abbildung des betrachteten Gegenstandsbereiches verstanden (unabhängig von der Form der Repräsentation, z. B. als Graph oder Repository-Datensatz) [St73].

Aus Sicht eines ganzheitlichen (Komplexitäts-) Managements ist in erster Linie die grundlegende Organisation von Unternehmenssystemen und IT-Landschaften (bestehend aus deren Hauptkomponenten und den zwischen diesen bestehenden Beziehungen) relevant. In Übereinstimmung mit [Ie00] wird hierfür der Begriff Unternehmens- bzw. IT-Architektur (Enterprise bzw. Enterprise IT Architecture) verwendet. Entsprechende Modelle werden auch als Architektur-Modelle bezeichnet.

2.2 Komplexität von IT-Landschaften und existierende Methoden zur Messung

Der Begriff Komplexität wird heute in den verschiedensten Forschungsdomänen verwendet und meist unterschiedlich definiert. Im Bereich der Systemtheorie wird Komplexität allgemein charakterisiert durch die Anzahl von Komponenten [Kl01] [FC93] [As56] und Beziehungen [FC93] sowie deren Heterogenität [Kl01] [As56] [Si62].

Innerhalb der (Wirtschafts-) Informatik setzen sich bisher nur wenige Arbeiten mit Komplexität auf der Ebene von IT-Landschaften bzw. Unternehmensarchitekturen auseinander. So definieren [SM03] den Begriff ”Computing Complexity” als Anzahl und Varietät der Komponenten und Beziehungen innerhalb einer IT-Landschaft sowie deren Änderungsrate. Eine ähnliche Sicht findet sich bei [De11], welcher zudem ein Kennzahlensystem für die Ermittlung der Gesamtkomplexität einer IT-Landschaft vorschlägt. Dabei werden die Anzahl, Vielfalt und Veränderungsdynamik von Informationstransformationsaufgaben, IT-Systemen, Informationsbeziehungen und IT-Basisinfrastruktur betrachtet. Auch von [En08] werden verschiedene quantitative Bewertungskriterien für Anwendungslandschaften vorgeschlagen. Diese umfassen die Aspekte ”funktionale Redundanz”, ”fachliche Strukturierung”, ”Domänenreinheit”, ”Kategorienreinheit” sowie ”nichtzyklische Abhängigkeiten”. Schließlich finden sich auch in [Mat12] eine Reihe von Komplexitäts-bezogenen Kennzahlen wie z. B. ”Business applications compliant with IT architecture and technology standards” (gemessen als Anzahl der konformen

Applikationen pro Gesamtzahl Applikationen) oder die ”IT component category standardization” (gemessen als Anteil der standardisierten Komponenten innerhalb einer Kategorie).

Den vorgenannten Ansätzen zur Messung von IT-Komplexität ist gemeinsam, dass sie jeweils von spezifischen Komplexitäts-Aspekten ausgehen und damit eine bestimmte Modellbildung implizieren. Für eine Anwendung müssen daher auch Daten in entsprechender Form zur Verfügung stehen. Zudem erschweren die teils unterschiedlichen Berechnungsmethoden und Skalen eine einheitliche Interpretation. Ein Ansatz, der diese Probleme vermeidet, wurde von [SWK13] vorgestellt. Ausgehend von [SM03] wird IT-Komplexität hier systemtheoretisch definiert. Die (struktuelle) Komplexität eines Systems wird dabei allgemein verstanden als Anzahl und Heterogenität der System-Elemente und deren Beziehungen. Der Ansatz ist damit potentiell mit beliebigen Architektur-Frameworks nutzbar. Allerdings existiert bisher keine Operationalisierung, welche die Anwendung im konkreten Unternehmenskontext beschreibt.

Im Rahmen des vorliegenden Beitrages wird der Ansatz nach [SWK13] aufgegriffen und weiterentwickelt. Im folgenden Abschnitt wird dazu zunächst das allgemeine Modell vorgestellt. Dieses wird in Abschnitt 4 erweitert und operationalisiert.

3 Allgemeines Modell zur Quantifizierung von Komplexität

Grundidee des vorliegenden Ansatzes ist es, Systemelemente und Beziehungen jeweils nach einem gleichartigen Verfahren zu behandeln und auf den Aspekt der Heterogenität zurückzuführen. Heterogenität wird dabei definiert als die Verschiedenartigkeit von Elementen (oder Beziehungen) innerhalb eines Systems hinsichtlich bestimmter Merkmalsausprägungen (sogenannter Charakteristiken).

3.1 Konzeptualisierung von Komplexität

Formal lassen sich Systeme als Tupel $S = (T, R)$ bestehend aus einer Menge von Systemkomponenten T (”Things”) und den zwischen diesen bestehenden Beziehungen R (”Relationships”) beschreiben [Kl01]. Dem Komplexitätsverständnis von [SWK13] folgend kann die (strukturelle) Komplexität eines Systems S dann wie folgt konzeptualisiert werden:

	Dinge (T)	**Relationen (R)**
Anzahl (N)	N_T	N_R
Heterogenität (H)	H_T	H_R

Tabelle 1: Komplexitätsdimensionen eines Systems S

Da die Bestimmung von N_T und N_R in der Regel trivial ist, kann die Quantifizierung von Komplexität somit auf die Quantifizierung von Heterogenität zurückgeführt werden.

3.2 Quantifizierung von Heterogenität

Im Folgenden wird ein Verfahren zur Quantifizierung von Heterogenität vorgestellt, welches für alle Arten von Systemelementen und Beziehungen gleichermaßen anwendbar ist. Grundidee des Ansatzes ist es, Heterogenität als statistische Eigenschaft aufzufassen und anhand von Häufigkeitsverteilungen zu quantifizieren. So lässt sich etwa die Heterogenität der in einer bestimmten IT-Landschaft eingesetzten Datenbank-Management-Systeme (DBMS) wie folgt als empirische Häufigkeitsverteilung beschreiben.

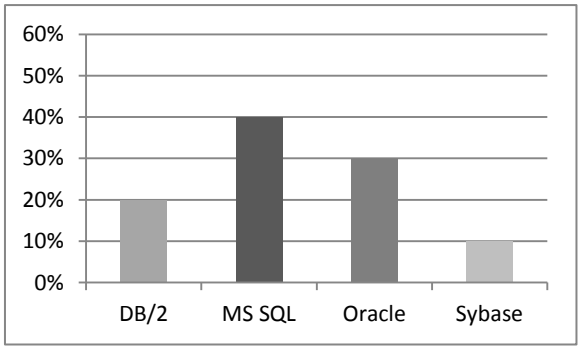

Abbildung 1: Häufigkeitsverteilung der in einer IT-Landschaft eingesetzen DBMS

Die Interpretation von Heterogenität als Häufigkeitsverteilung ermöglicht die Anwendung erprobter statistischer Maßzahlen. So können insbesondere die in verschiedenen Bereichen der Ökonomie eingesetzten und gut untersuchten Konzentrationsmaße adaptiert werden [Wi12]. Konzentrationsmaße werden z. B. von Kartellbehörden verwendet, um die Marktmacht von Unternehmen zu bestimmen [US12]. Ein weiteres Anwendungsfeld bildet die Produkt- oder Geschäftsfeld-Diversifikation [JB79].

In [SWK13] wurden die an Heterogenitätsmaße zu stellenden Anforderungen den in der Literatur häufig verwendeten Kennzahlen gegenübergestellt. Dabei zeigte sich, dass vor allem das von [Sh48] eingeführte Entropie-Maß für die Analyse von IT-Heterogenität geeignet ist. Das Entropie-Maß steigt mit der Anzahl unterschiedlicher Charakteristika und mit zunehmender Gleichverteilung der Elemente. Es reagiert darüber hinaus auch auf Charakteristiken mit vergleichsweise kleinem Anteil. Proportionale Änderungen der Häufigkeiten wirken sich hingegen nicht aus. Das Entropie-Maß ist wie folgt definiert:

$$EM = \sum_{i=1}^{n} f_i \ln\left(\frac{1}{f_i}\right)$$

Dabei bezeichnet f_i die relative Häufigkeit der Elementausprägung (Charakteristik) i. Das Entropie-Maß nimmt den minimalen Wert 0 an, wenn alle Elemente einer einzigen Charakteristik zugeordnet sind. Der maximale Wert $\ln(m)$ wird dagegen bei Gleichverteilung der Elemente auf m Charakteristiken erreicht.

4 Operationalisierung im Kontext von Unternehmensarchitekturen

Eine praktische Anwendung des in Abschnitt 3 dargestellten Ansatzes zur Quantifizierung von Komplexität erfordert eine weitergehende Operationalisierung. So ist für eine sinnvolle Betrachtung im Kontext von Unternehmensarchitekturen einerseits eine Differenzierung nach verschiedenen Element- und Beziehungstypen erforderlich (z. B. Anwendungen, Infrastruktur). Andererseits ist für die praktische Anwendung die Verfügbarkeit entsprechender Daten ausschlaggebend. Im vorliegenden Kontext bedeutet dies, dass ein integriertes System-Modell der IT-Landschaft bzw. der übergeordneten Unternehmensarchitektur benötigt wird. Dabei ist zu beachten, dass die Berechnungen je nach Wahl und Granularität dieses Modells zu unterschiedlichen Ergebnissen führen können. Dies ist insofern von Bedeutung als es sich bei IT-Landschaften und Unternehmensarchitekturen nicht um reale Gegenstände handelt. Zwar beinhalten diese letztlich auch physische Elemente (wie Recheneinheiten und auf diesen ausgeführten Binärcode). Bei den aus Management-Sicht relevanten Steuerungseinheiten wie Anwendung, Domäne oder Geschäftsfunktion handelt es sich jedoch um Abstraktionen bzw. rein logische Konstrukte, welche je nach Zielsetzung sehr unterschiedlich modelliert werden können.

4.1 Nutzung von Architektur-Frameworks und -Werkzeugen als Datenbasis

Der Einsatz von Architektur-Modellen bildet heute ein zentrales Instrument des IT-Architekturmanagements [Sc09]. Dabei hat sich im Laufe der Zeit der Einsatz von standardisierten Architektur-Frameworks etabliert. Diese bilden einen einheitlichen konzeptionellen Rahmen, nach dem Aspekte der Unternehmensarchitektur innerhalb der Organisation abgebildet werden können. Die Mehrzahl der Architekturmanagement-Funktionen ist ferner dazu übergegangen, Architektur-Daten in einem zentralen Repository zu verwalten [Se05]. Der Datenbestand solcher Architektur-Repositories ist dabei in der Regel durch ein entsprechendes konzeptionelles Schema strukturiert, welches die Elementtypen mit ihren Attributen sowie die zwischen diesen zulässigen Relationstypen spezifiziert. In der Praxis hat sich hierfür der Begriff Metamodell etabliert.

Aus Sicht der Autoren erscheint es zweckmäßig, bei der Komplexitätsanalyse auf den in den jeweiligen Unternehmen etablierten Architektur-Frameworks und Repositories aufzusetzen. Auf diese Weise kann das Management von Komplexität nahtlos in die bestehenden Prozesse integriert und der existierende Datenbestand nutzbar gemacht werden. Damit erscheint potentiell auch eine vollständige Automatisierung der Kennzahlenberechnung realisierbar. Die zunehmende Standardisierung von Metamodellen eröffnet dabei nicht zuletzt eine Perspektive für Unternehmensvergleiche und Benchmarkings.

4.2 Metamodell-Mapping

Architektur-Frameworks und Tools müssen vielen unterschiedlichen Anforderungen genügen. Vor einer Anwendung des Komplexitätsmodells ist daher eine Selektion der aus Komplexitätssicht relevanten Element- und Beziehungstypen erforderlich. Hierbei muss eine sorgfältige Prüfung der Anwendbarkeit und Semantik der betreffenden Komplexitätsmaße erfolgen. Prinzipiell können die folgenden Aspekte betrachtet werden:

(1) Komplexität von Elementtypen: Anzahl und Heterogenität der Elementinstanzen eines Typs hinsichtlich bestimmter (eigener) Merkmale
(2) Komplexität von Relationstypen: Anzahl und Heterogenität der Relationsinstanzen eines Typs (direkte Beziehungen) hinsichtlich bestimmter Merkmale der an der Beziehung beteiligten Elementtypen
(3) Komplexität von Pfadtypen: Anzahl und Heterogenität von Relationsinstanzfolgen eines Typs (indirekten Beziehungen) hinsichtlich bestimmter Merkmale der auf dem Pfad liegenden Elementtypen

Relationstypen (2) können dabei als Spezialfall von Pfadtypen (3) angesehen werden.

4.3 Mathematische Formalisierung

Im Folgenden wird eine mathematische Formalisierung vorgestellt, welche eine präzise Spezifikation der zu betrachtenden Komplexitätsaspekte in beliebigen Metamodellen erlaubt. Dazu werden sowohl das Metamodell als auch das zugehörige Instanzmodell als gerichtete Graphen im Sinne der Graphentheorie beschrieben.

Sei $MM = (ET, RT)$ das Metamodell einer Unternehmensarchitektur bestehend aus einer Menge von Elementtypen ET und einer Menge von Beziehungstypen RT mit $RT \subseteq ET \times ET$. Sei zudem für alle Elementtypen $i \in ET$ AT_i die Menge der auf i definierten Attributtypen. Sei weiter $PT = \{ j = (i_1, \dots, i_n) \mid n > 1 \wedge i_1, \dots, i_n \in ET \wedge \forall g, h \in \{1, \dots, n\} \, mit \, g \neq h : i_g \neq i_h \wedge \forall m \in \{1, \dots, n-1\} : (i_m, i_{m+1}) \in RT\}$ die Menge aller Pfade in MM.

Sei nun $M = (EI, RI)$ ein zugehöriges Instanzmodell bestehend aus einer Menge von Elementinstanzen EI sowie einer Menge von Beziehungsinstanzen RI mit $RI \subseteq EI \times EI$. Dabei beschreibe für alle Elementtypen $i \in ET$ $EI_i \subseteq EI$ die Menge der Ausprägungen vom Typ i, d. h. $EI_i = \{k \mid k \, ist \, vom \, Typ \, i\}$. Es sei weiter für alle $i \in ET, a \in AT_i$ und alle $k \in EI_i$ AI_k^a der Attributwert der Instanz k hinsichtlich des Attributtyps a sowie für alle $i \in ET$ und $a \in AT_i$ $AV_i^a = \bigcup_{k \in EI_i} AI_k^a$ der Wertebereich des Elementtyps i hinsichtlich des Attributtyps a. Sei schließlich für alle $j = (i_1, \dots i_n) \in PT$ PI_j die Menge der Pfadinstanzen vom Typ j, d. h. $PI_j = \{ (k_1, \dots, k_n) \mid n > 1 \wedge \forall m \in \{1, \dots, n-1\} : (k_m, k_{m+1}) \in RI\}$. Für alle $i \in ET$ und alle $a \in AT_i$ beschreibt dann die Funktion $f_{i,a}: AV_i^a \rightarrow [0 \dots 1]$

$$f_{i,a}(x) = \frac{|\{k \in EI_i \mid AI_k^a = x\}|}{|EI_i|}$$

die relative Häufigkeitsverteilung der Elementinstanzen vom Typ i entlang der Charakteristika des Attributtyps a. Für alle Pfade $j = (i_1, \dots i_n) \in PT$ und alle i_m mit $m \in \{1, \dots, n\}$ sowie alle $a \in AT_{i_m}$ beschreibt ferner die Funktion $f_{j,i_m,a}: AV_{i_m}^a \rightarrow [0 \dots 1]$

$$f_{j,i_m,a}(x) = \frac{\left|\left\{(k_1,\dots,k_n) \in PI_j \;\middle|\; \exists p \in \{1,\dots,n\} \, mit \, k_p \in EI_{i_m} \, und \, AI_{k_p}^a = x\right\}\right|}{|PI_j|}$$

die relative Häufigkeitsverteilung der Pfadinstanzen (incl. Beziehungsinstanzen) vom Typ j entlang der Charakteristika des Attributtyps a von Elementtyp i_m. Die Formalisierung wird anhand des folgenden Beispiels veranschaulicht.

$ET = \{Application, Function, Platform\}$
$RT = \{(Platform, Application), (Application, Function)\}$
$AT_{Application} = \{Name, Vendor\} \quad AT_{Platform} = \{Name\} \quad AT_{Function} = \{Name\}$
$PT = \{(Platform, Application), (Application, Function), (Platform, Application, Function)\}$
$EI_{Application} = \{SAP\ FI, ResQ\} \quad EI_{Platform} = \{Oracle\ DB\ PF, SAP\ PF, Windows\ PF\}$ $EI_{Function} = \{Financial\ Accounting, Reserving\}$
$RI_{(Platform,Application)} = \{(SAP\ PF, SAP\ FI), (Oracle\ DB\ PF, SAP\ FI), (Windows\ PF, ResQ)\}$ $RI_{(Application,Function)} = \{(SAP\ FI, Financial\ Accounting), (ResQ, Reserving)\}$
$AI_{SAP\ FI}^{Vendor} = SAP \quad AI_{ResQ}^{Vendor} = Towers\ Watson \quad AV_{Application}^{Vendor} = \{SAP, Towers\ Watson\}$
$PI_{(Platform,Application,Function)} = \left\{ \begin{array}{c} (SAP\ PF, SAP\ FI, Financial\ Accounting), \\ (Oracle\ DB\ PF, SAP\ FI, Financial\ Accounting), \\ (Windows\ PF, ResQ, Reserving) \end{array} \right\}$
$f_{Application,Vendor}(SAP) = \frac{\left\lvert\left\{k \in EI_{Application} \middle\vert AI_k^{Vendor} = SAP\right\}\right\rvert}{\lvert EI_{Application}\rvert} = {}^1/_2$
$f_{(Platform,Application),Platform,Name}(SAP\ PF) = \frac{\left\lvert\left\{(k_1,\ldots,k_n) \in PI_{(Platform,Application)} \middle\vert \exists\ p \in \{1,\ldots,n\}\ mit\ k_p \in EI_{Platform}\ und\ AI_{k_p}^{Name} = SAP\ PF\right\}\right\rvert}{\lvert PI_{(Platform,Application)}\rvert} = {}^1/_3$

Tabelle 2: Beispiel zur Formalisierung

Unter Rückgriff auf Abschnitt 3 können nun wie folgt die relevanten Komplexitätsaspekte spezifiziert und quantifiziert werden:

	Elementtyp i nach Attribut a $C_{i,a} = (N_i, H_{i,a})$	**Pfadtyp j nach Attribut a von Elementtyp i_m** $C_{j,i_m,a} = (N_j, H_{j,i_m,a})$
Anzahl (N)	$N_i = \lvert EI_i \rvert$	$N_j = \lvert PI_j \rvert$
Heterogenität (H)	$H_{i,a} = \sum_{x \in AV_i^a} f_{i,a}(x) \ln \frac{1}{f_{i,a}(x)}$	$H_{j,i_m,a} = \sum_{x \in AV_{i_m}^a} f_{j,i_m,a}(x) \ln \frac{1}{f_{j,i_m,a}(x)}$

Tabelle 3: Quantifizierung der Komplexitätsaspekte

5 Anwendungsbeispiel

5.1 Kontext und Methodik

Das vorgestellte Komplexitätsmodell wurde im Rahmen eines realen Projektes zur Architektur-Transformation einer Spezialversicherung angewendet. Beim betrachteten Unternehmen handelt es sich um eine internationale Tochtergesellschaft eines deutschen Versicherungskonzerns mit mehreren Standorten und zwei primären Geschäftsfeldern. Für das Tochterunternehmen wurde im Zeitraum Mai bis Dezember 2012 eine neue Ziel-Architektur entwickelt. Dabei wurde entsprechend der TOGAF ADM [To12] ein vollständiger Architektur-Entwicklungszyklus durchlaufen. Zudem wurden eine Transformations-Roadmap sowie eine mittelfristige Zwischen-Architektur definiert.

Haupttreiber für die Neubebauung waren eine hohe Redundanz der Systeme zwischen den verschiedenen Standorten, eine mangelnde Akzeptanz der operativen Kernsysteme im Technical Accounting sowie der extensive Einsatz von Office-basierten End-User-Applikationen. Weiterhin sollten die Landschaft technologisch modernisiert und die bisherigen Individualschnittstellen durch eine zentrale Integrationsplattform ersetzt werden. Hierdurch sollten nicht zuletzt auch der Automatisierungsgrad erhöht und manuelle Eingriffe reduziert werden. Hinzu kam der Wunsch der Konzernmutter, die Landschaft der Tochter auf die technologischen Standards der Gruppe auszurichten.

Die Entwicklung der Zielarchitektur erfolgte im Rahmen eines Projektes unter Einbeziehung aller wichtigen Stakeholder incl. des Top-Managements der Tochtergesellschaft. Die Auswahl der Zielarchitektur geschah anhand unterschiedlicher qualitativer Kriterien wie der Qualität und Vollständigkeit der Geschäfts-Unterstützung sowie der Konformität mit Gruppen-Standards. Ergänzend wurde auch eine quantitative Komplexitätsbetrachtung auf Basis des in diesem Beitrag vorgestellten Modells durchgeführt. Dazu wurden von den beteiligten Architekten zunächst die verschiedenen Element- und Beziehungstypen des Metamodells im Hinblick auf eine sinnvolle Interpretation der Komplexitätsmaße untersucht und die zu betrachtenden Komplexitätsaspekte definiert. Auf dieser Grundlage wurden dann die entsprechenden Berechnungen durchgeführt. Als Datenquelle konnte dabei das im Konzern eingesetzte Architektur-Repository genutzt werden.

5.2 Identifikation der relevanten Komplexitätsaspekte

Abbildung 2 zeigt das innerhalb des Konzerns eingesetzte Metamodell in einer leicht vereinfachten Darstellung. Dabei sind die in die Komplexitätsanalyse einbezogenen Modellaspekte mit durchgehenden Linien und die ausgeblendeten Aspekte gestrichelt dargestellt. Im Mittelpunkt des Modells steht das Anwendungssystem. Ein Anwendungssystem stellt bestimmte fachliche Funktionen zur Verfügung und unterstützt damit Geschäftsfelder (Lines of Businesses, LoB) sowie Organisationseinheiten. Anwendungen können aus mehreren Komponenten bestehen, die wiederum auf technischen Plattformen ausgeführt werden. Hierbei handelt es sich um logische Zusammenfassungen von für die Ausführung von Anwendungen benötigten Infrastruktursystemen. Anwendungen können weiterhin Schnittstellen bereitstellen und nutzen sowie Informationsobjekte bearbeiten.

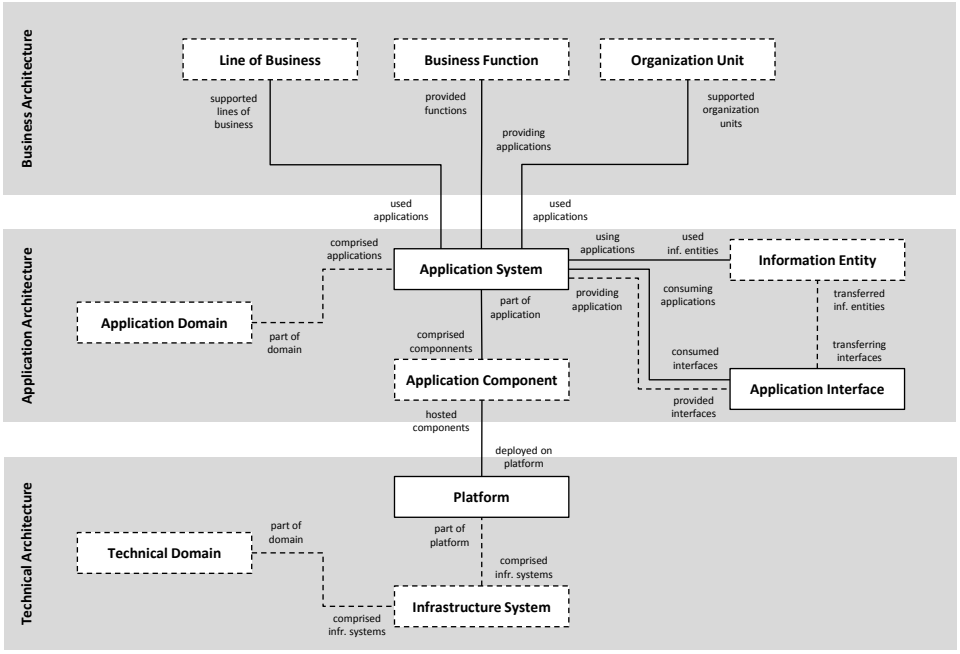

Abbildung 2: Metamodell des Anwenderunternehmens

Die identifizierten Komplexitätsaspekte werden im Detail in Tabelle 4 beschrieben. Dabei werden sowohl eine Interpretation der beiden Komplexitäts-Facetten als auch die im Rahmen des Fallbeispiels ermittelten Ist- und Ziel-Werte angegeben.

5.3 Ergebnisse der Komplexitätsberechnung

Wie aus Tabelle 4 ersichtlich ist, weist die Ziel-Architektur (mit Ausnahme der Plattform-Unterstützung) in allen untersuchten Aspekten eine geringere Komplexität auf als die Ist-Architektur. So wird mit Umsetzung der Zielarchitektur insbesondere die Anzahl der Applikationen deutlich reduziert ($N^{Ist} = 51; N^{Ziel} = 23$). Dies ist im Wesentlichen auf die Eliminierung von End-User-Anwendungen und Sonderverarbeitungen zurückzuführen. Der vergleichsweise geringe Rückgang der Hersteller-Heterogenität ($H^{Ist} = 3{,}90;\ H^{Ziel} = 3{,}08$) kann dadurch erklärt werden, dass Individualentwicklungen auch in der Ziel-Architektur eine tragende Rolle spielen.

Die größten Veränderungen zeigen sich im Bereich der Schnittstellen. Hier führt die Einführung zentraler Integrationslösungen (Enterprise Application Integration / EAI für die Realtime-Übertragung auf Einzelsatzebene und Extract-Transform-Load / ETL für die batch-orientierte Massendatenübertragung) zu einer signifikanten Reduktion der technologischen Heterogenität ($H^{Ist} = 3{,}36; H^{Ziel} = 0{,}66$). Der Übergang von individuellen (Peer-to-Peer) zu zentral verwalteten und wiederverwendbaren Schnittstellen schlägt sich darüber hinaus in einer Verringerung der Schnittstellen-Beziehungen nieder. Dieser

Komplexitätsaspekt	**Interpretation Anzahl (*N*)**	**Interpretation Heterogenität (*H*)**	**Ist / Ziel**
$C_{Application\ System,Vendor}$	Anzahl aller vorhandenen (produktiven) Applikationen. *N* sinkt allgemein mit der Außerbetriebnahme und Konsolidierung von Anwendungen.	Konzentration der Anwendungen nach Herstellern. Min-Wert: 0 (bei ausschließlicher Nutzung von Anwendungen eines einzigen Herstellers) Max-Wert: ln(Anzahl Hersteller) (bei gleichverteilten Herstellern) *H* sinkt mit Deaktivierung oder Ersetzung von Anwendungen von Nischenanbietern durch Lösungen bereits stark vertretener Hersteller.	*(51; 3,90)* → *(23; 3,08)*
$C_{Application\ Interface,Type}$	Anzahl aller vorhandenen (produktiven) Applikationsschnittstellen. *N* sinkt allgemein mit der Außerbetriebnahme und Konsolidierung von Schnittstellen.	Konzentration der Schnittstellen nach Schnittstellentechnologie Min-Wert: 0 (bei ausschließlicher Nutzung einer einzigen Schnittstellentechnologie) Max-Wert: ln(Anzahl der Schnittstellentechnologien) (bei gleichverteilten Technologien) *H* sinkt mit der Eliminierung von Schnittstellentechnologien sowie mit der Verlagerung von Schnittstellen auf Basis von Randtechnologien (bzw. Individualschnittstellen) hin zu Lösungen auf Basis von zentralen Integrationsplattformen	*(40; 3,36)* → *(39; 0,66)*
$C_{(Application\ System,Business\ Function),\ Application\ System,Name}$	Anzahl der von allen Applikationen (ggf. redundant) bereitgestellten Geschäftsfunktionen. Min-Wert (bei voller funktionaler Abdeckung): Anzahl Geschäftsfunktionen (keine funktionale Redundanz) Max-Wert: Anzahl Anwendungen x Anzahl Geschäftsfunktionen (vollständige funktionale Redundanz) *N* sinkt mit dem Abbau funktionaler Redundanz (sowohl funktionale Entflechtung als auch Konsolidierung von Anwendungen)	Konzentration der Geschäftsfunktionen nach bereitstellenden Anwendungen. Min-Wert: 0 (bei Bereitstellung aller Funktionen durch eine einzige Applikation) Max-Wert: ln(Anzahl Applikationen) (bei gleichverteilter Bereitstellung von Geschäftsfunktionen durch Applikationen) *H* sinkt mit der Deaktivierung von Anwendungen und mit der Verlagerung von Funktionen von Rand- hin zu Kernsystemen (mit integrierter Funktionalität).	*(148; 3,09)* → *(108; 2,50)*
$C_{(Application\ System,Organization\ Unit),\ Application\ System,Name}$	Anzahl der Unterstützung von Org.-Einheiten durch Applikationen. Min-Wert (bei Nutzung aller Applikationen): Anzahl Applikationen (jede Anwendung wird nur von einer Org.-Einheit genutzt) Max-Wert: Anzahl Anwendungen x Anzahl Org.-Einheiten (Nutzung aller Anwendungen durch alle Org.-Einheiten) *N* sinkt mit der Einstellung der Anwendungsbereitstellung in bestimmten Org.-Einheiten.	Konzentration der Anwendungsnutzung durch Org.-Einheiten. Min-Wert: 0 (bei Unterstützung aller Org.-Einheiten durch eine einzige Applikation) Max-Wert: ln(Anzahl Applikationen) (bei gleichverteilter Unterstützung von Org.-Einheiten durch Applikationen *H* sinkt mit Deaktivierung von Anwendung sowie mit der Verlagerung von ”lokalen” auf ”globale” Applikationen (Standortkonsolidierung).	*(65; 3,88)* → *(44; 3,12)*

Komplexitätsaspekt	**Interpretation Anzahl (N)**	**Interpretation Heterogenität (H)**	**Ist / Ziel**
$C_{(Application\ System, Line\ of\ Business),\ Application\ System, Name}$	Anzahl der Unterstützung von LoBs durch Applikationen. Min-Wert (bei Nutzung aller Applikationen): Anzahl Applikationen (jede Anwendung unterstützt eine LoB) Max-Wert: Anzahl Anwendungen x Anzahl LoBs (jede Anwendung unterstützt jede LoB) N sinkt mit der Einstellung der Anwendungsunterstützung in bestimmten LoBs	Konzentration der Anwendungsnutzung durch LoBs. Min-Wert: 0 (bei Unterstützung aller LoBs durch eine Applikation) Max-Wert: ln(Anzahl Applikationen) (bei gleichverteilter Unterstützung von LoBs durch Applikationen) H sinkt mit der Deaktivierung von Anwendungen und der Verlagerung von spartenspezifischen hin zu spartenübergreifenden Applikationen (Spartenkonsolidierung).	*(91; 3,90)* → *(42; 3,11)*
$C_{(Application\ System, Information\ Entity),\ Application\ System, Name}$	Anzahl der von allen Applikationen (ggf. mehrfach) verarbeiteten Datenobjekte. Min-Wert (bei voller Abdeckung aller Datendomänen): Anzahl Datenobjekte (jedes Objekt nur von einer Anwendung verarbeitet) Max-Wert: Anzahl Applikationen x Anzahl Datenobjekte (vollständige Datenredundanz) N sinkt mit einer Einschränkung der Datennutzung durch einzelne Applikationen (sowohl Entflechtung als auch Konsolidierung)	Konzentration der Datenobjekte nach verarbeitenden Anwendungen. Min-Wert: 0 (bei ausschließlicher Verarbeitung aller Datenobjekte durch eine einzige Applikation) Max-Wert: ln(Anzahl Applikationen) (bei gleichverteilter Verarbeitung von Datenobjekten durch Anwendungen) H sinkt mit der Deaktivierung von Anwendungen sowie der Verlagerung der Datennutzung von Rand-Systemen hin zu Kernsystemen (mit integriertem Datenbestand)	*(182; 3,33)* → *(120; 2,43)*
$C_{(Application\ System, Application\ Interface),\ Application\ Interface, Name}$	Anzahl der von allen Applikationen (ggf. mehrfach) genutzten Schnittstellen. Min-Wert (bei Nutzung aller Schnittstellen): Anzahl Schnittstellen (keine Wiederverwendung) Max-Wert: Anzahl Schnittstellen x Anzahl Anwendungen (jede Schnittstelle von jeder Anwendung genutzt) N sinkt mit einer Reduktion der Schnittstellennutzung.	Konzentration der Schnittstellennutzung durch Applikationen. Min-Wert: 0 (bei Nutzung einer einzigen Schnittstelle) Max-Wert: ln(Anzahl Schnittstellen) (bei gleichverteilter Nutzung aller Schnittstellen H sinkt mit der Verlagerung von wenig genutzten Schnittstellen hin zu viel genutzten Schnittstellen (z. B. zentralen Services)	*(40; 3,69)* → *(67; 3,45)*
$C_{(Platform, Component, Application\ System),\ Platform, Name}$	Anzahl der Unterstützung von Anwendungen durch Plattformen. Min-Wert: Anzahl Applikationen (wenn jede Applikation nur auf einer Plattform läuft) Max-Wert: Anzahl Anwendungen x Anzahl Komponenten x Anzahl Plattformen (wenn jede Komponente jeder Applikation jede Plattform benötigt) N sinkt mit der Reduktion der für einzelne Anwendungen benötigten Plattformen (Reduktion der technischen Komplexität)	Konzentration der Plattformnutzung durch Applikationen. Min-Wert: 0 (bei Deployment aller Applikationen auf einer einzigen Plattform) Max-Wert: ln(Anzahl Plattformen) (bei gleichverteiltem Deployment der Applikationen auf allen Plattformen) H sinkt mit der Eliminierung von Plattformen sowie mit der Verlagerung von Anwendungen von Rand- auf Kern-Plattformen (Plattformkonsolidierung)	*(29; 2,18)* → *(31; 1,78)*

Tabelle 4: Betrachtete Komplexitätsaspekte

Effekt wird allerdings durch die Einführung neuer Schnittstellen (insbesondere im Bereich Reporting und Masterdatenmanagement) überkompensiert ($N^{Ist} = 40; N^{Ziel} = 67$).

Eine deutliche Komplexitätsreduktion wird auch im Bereich des Business-Alignments erreicht. So nimmt im Zielbild insbesondere die Komplexität der Beziehung zwischen Anwendungen und unterstützten Geschäftsfunktionen sichtbar ab ($N^{Ist} = 148; N^{Ziel} = 108; H^{Ist} = 3{,}09; H^{Ziel} = 2{,}50$). Dies spiegelt die angestrebte funktionale Entflechtung der Anwendungen sowie die Zentralisierung der Funktionalität in wenigen Kernsystemen (Technical Accounting, Financial Accounting, Reserving) wider. Entsprechendes gilt für die Verarbeitung von Datenobjekten ($N^{Ist} = 182; N^{Ziel} = 120; H^{Ist} = 3{,}33; H^{Ziel} = 2{,}43$). Die ebenfalls deutliche Komplexitätsreduktion in der Beziehung zwischen Anwendungen und unterstützten Organisationseinheiten ($N^{Ist} = 65; N^{Ziel} = 44; H^{Ist} = 3{,}88; H^{Ziel} = 3{,}12$) kann durch die Eliminierung von lokalen Spezialanwendungen (z. B. lokale Hauptbuch-Anwendung) und die damit einhergehende Standortkonsolidierung erklärt werden. Ähnliches gilt für die Unterstützung der beiden Geschäftsfelder. Auch hier führt die Entfernung von Geschäftsfeld-spezifischen Anwendungen (z. B. für das Exposure Management) und die Schaffung von übergreifenden Lösungen zu einer Reduktion der Komplexitätswerte ($N^{Ist} = 91; N^{Ziel} = 42; H^{Ist} = 3{,}90; H^{Ziel} = 3{,}11$).

Im Bereich der technischen Architektur zeigt sich ein differenziertes Bild. So wird mit der Zielarchitektur einerseits die Anzahl der technischen Plattformtypen reduziert. Dies spiegelt sich auch in einer deutlichen Senkung der Plattform-Heterogenität wider ($H^{Ist} = 2{,}18; H^{Ziel} = 1{,}78$).[1] Andererseits nimmt die Quantität der Anwendungsunterstützung in der Zielarchitektur leicht zu ($N^{Ist} = 29; N^{Ziel} = 31$). Dies ist im Wesentlichen auf die Einführung von neuen (Server-) Anwendungen in den Bereichen CRM, Claims Management, Solvency II und Reporting zurückzuführen.

6 Diskussion, Einschränkungen und Ausblick

Der Einsatz des vorgeschlagenen Verfahrens zur Komplexitätsmessung von IT-Landschaften hat sich im Rahmen des betrachteten Praxisprojektes als zielführend und praktikabel erwiesen. So konnten die bisher rein qualitativen (und teils subjektiven) Bewertungen von Ist- und Zielarchitekturen durch die beteiligten Architekten im Wesentlichen bestätigt und quantitativ untermauert werden. Insbesondere konnte die mit Umsetzung der Zielarchitektur einhergehende Reduktion der Komplexität erstmals in einfacher und nachvollziehbarer Weise beschrieben und kommuniziert werden.

Als Vorteile gegenüber bestehenden Methoden wurden von den Projektteilnehmern vor allem der ganzheitliche Ansatz und die vergleichsweise einfache Berechnung auf Basis von Repository-Daten angesehen. So lässt sich die Methodik gleichermaßen auf alle Architektur-Ebenen und beliebige Teilmodelle anwenden. Die Ausrichtung am Metamodell unterstützt zudem eine ganzheitliche und mehrdimensionale Betrachtung, anhand der

[1] Es wurden nur Server-Plattformen in die Analyse einbezogen. Bei Berücksichtigung von Client-Plattformen kommt es aufgrund der Eliminierung von End-User-Applikationen zu einem Anstieg der Heterogenität.

sich gut die jeweiligen Problembereiche und Optimierungspotentiale aufzeigen lassen. Allerdings erfordern die Identifikation der relevanten Modellaspekte sowie die Interpretation der Daten große Sorgfalt. So zeigte sich im vorliegenden Fall, dass nicht alle Element- und Beziehungstypen gleichermaßen für die Komplexitätsanalyse geeignet sind.[2]

Bei der Interpretation der Modellergebnisse ist zu beachten, dass die einzelnen Komplexitätsmaße jeweils nur spezifische Aspekte der Gesamt-Komplexität abbilden. Eine Reduktion auf einer Ebene kann dabei mitunter zu einer Erhöhung an anderer Stelle führen. So bedingt im vorliegenden Fall zwar die weitgehende Funktions- und Datenintegration in wenigen Kernapplikationen eine Reduktion der betreffenden Komplexitätswerte auf Applikationsebene. Diese wird jedoch teilweise durch eine erhöhte Komplexität auf Subsystemebene ”erkauft”, welche im vorliegenden Fall nicht betrachtet wurde.

Wie die Projektbeteiligten betonen, bildet Komplexität letztlich auch nur eines von mehreren Bewertungskriterien für IT-Architekturen. Wichtig sind daneben insbesondere die Vollständigkeit der IT-Unterstützung sowie deren Qualität (und Akzeptanz) aus Anwendersicht. Wie auch im vorliegenden Fall am Beispiel der Schnittstellen deutlich wurde, bedingen diese Faktoren häufig einen Anstieg der IT-Komplexität. Bei Komplexitätsbetrachtungen sollte daher immer zwischen der durch Geschäfts-Anforderungen induzierten (minimalen) Komplexität und der darüber hinausgehenden überschüssigen Komplexität differenziert werden. Zu beachten ist ferner, dass Komplexität nicht nur negative Wirkungen verursacht, sondern sich auch positiv auf Aspekte wie Sicherheit und Hersteller-Unabhängigkeit auswirken kann. Es ist daher davon auszugehen, dass es sich bei Komplexitätsentscheidungen um mehrdimensionale Problemstellungen handelt, die je nach gegebenen Kontextfaktoren eine unterschiedliche Positionierung erfordern.

Die vorliegenden Ergebnisse sind mit einer Reihe von Einschränkungen verbunden. So besitzt das Fallbeispiel bisher primär illustrativen Charakter. Es kann zudem nicht als repräsentativ angesehen werden. Vor einer Verallgemeinerung sind daher weitere Analysen notwendig. Zukünftig sollte unter anderem genauer untersucht werden, welche Metamodell-Aspekte sich grundsätzlich für eine Komplexitätsanalyse eignen. Darüber hinaus sollte analysiert werden, inwieweit sich die einzelnen Komplexitätsaspekte aggregieren lassen. In diesem Kontext könnten sich auch spezielle grafische Sichten als hilfreich erweisen. So wären etwa ”Komplexitätskarten” denkbar, welche verschiedene Bereiche eines Metamodells entsprechend der jeweiligen Komplexität einfärben.

Literatur

[As56] Ashby, W. R.: An Introduction to Cybernetics. Chapman & Hall, London, 1956.

[De11] Dern, G.: Integrationsmanagement in der Unternehmens-IT: Systemtheoretisch fundierte Empfehlungen zur Gestaltung von IT-Landschaft und IT-Organisation. Vieweg + Teubner Verlag, Wiesbaden, 2011. Zugl. Diss., Universität Duisburg-Essen, 2010.

[En08] Engels, G. et al.: Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten. Dpunkt-Verlag, Heidelberg, 2008.

[2] Einen Sonderfall bilden insbesondere die rein logischen Elementtypen wie Geschäftsfunktionen oder Information Entities, welche normativen Charakter besitzen und damit per Definition frei von Heterogenität sind.

[FC93] Flood, R. L.; Carson, E. R.: Dealing with Complexity: An Introduction to the Theory and Application of Systems Science. Springer, Berlin, 1993.

[FS01] Ferstl, O. K.; Sinz, E. J.: Grundlagen der Wirtschaftsinformatik. Bd. 1. 4. Auflage. Oldenbourg, München, Wien, 2001.

[Ie00] IEEE Computer Society (Hrsg.): IEEE Recommended Practice for Architectural Description of Software-Intensive Systems – IEEE Stdd 1471-2000. IEEE Computer Society, New York, 2000.

[JB79] Jacquemin, A. P.; Berry, C. H.: Entropy Measure of Diversification and Corporate Growth. The Journal of Industrial Economics (27:4), 1979; S. 359-369.

[Kl01] Klir, G. J.: Facets of System Science. Springer, Berlin, 2001.

[Le97] Lehman, M. M.: Laws of software evolution revisited. In: (Montangero, C. Hrsg.): Software Process Technology – Proceedings 5th European Workshop on Software Process Technology (EWSPT '96), Nancy, 1996. Springer, Berlin, 1997; S. 108–124.

[Mat12] Matthes, F.; Monahov, I.; Schneider, A.; Schulz, C.: EAM KPI Catalog v. 1.0. Technical Report, Chair for Software Engineering of Business Information Systems, Technische Universität München, München, 2012.

[MLH11] Möller, D.; Legner, C.; Heck, A.: Understanding IT Tranformation – an Explorative Study. European Conference on Information Systems (ECIS). Helsinki, 2011.

[Na05] Namba, Y.: City Planning Approach for Rebuilding Enterprise Information Systems. Diss., Tokyo Institute of Technology, 2005.

[Ro79] Ropohl, G.: Eine Systemtheorie der Technik – Zur Grundlegung der Allgemeinen Technologie. Carl Hanser, München, 1979.

[SB11] Schmidt, C.; Buxmann, P.: Outcomes and Success Factors of Enterprise IT Architecture Management: Empirical Insight from the International Financial Services Industry. European Journal of Information Systems (20), 2011; S. 168-185.

[Sc09] Schmidt, C.: Management komplexer IT-Architekturen – Empirische Analyse am Beispiel der internationalen Finanzindustrie. Gabler, Wiesbaden, 2009. Zugl. Diss., Technische Universität Darmstadt, 2008.

[Se05] Sebis (Hrsg.): Enterprise Architecture Management Tool Survey 2005. Ernst-Denert-Stiftungslehrstuhl, Chair for Informatics 19, Technische Universität München, 2005.

[Sh48] Shannon, C. E.: A Mathematical Theory of Communication. Bell System Technical Journal, 27, 1948; 379-423 und 623-656.

[Si62] Simon, H. A.: The Architecture of Complexity. Proceedings of the American philosophical society 106 (6), 1962; S. 467-482.

[SM03] Schneberger, S. L.; McLean, E. R.: The Complexity Cross: Implications for Practice. Communications of the ACM, 46 (9), 2003; S. 216-225.

[St73] Stachowiak, H.: Allgemeine Modelltheorie. Springer, Wien, New York, 1973.

[SWK13] Schütz, A.; Widjaja, T.; Kaiser, J.: Complexity in Enterprise Architectures – Conceptualization and Introduction of a Measure from a System Theoretic Perspective. European Conference on Information Systems (ECIS), Utrecht, 2013.

[To12] The Open Group (Hrsg.): TOGAF (The Open Group Architecture Framework) V. 9.1. http://www.opengroup.org/togaf, 2012.

[US12] U.S. Department of Justice and the Federal Trade Commission (Hrsg.): Horizontal Merger Guidelines. http://www.justice.gov/atr/public/guidelines/hmg-2010.html, 2010.

[Wi04] Winter, R.: Architektur braucht Management. Wirtschaftsinformatik 46, Nr. 4, 2004; S. 317–319.

[Wi12] Widjaja, T.; Kaiser, J.; Tepel, D.; Buxmann, P.: Heterogeneity in IT Landscapes and Monopoly Power of Firms: A Model to Quantify Heterogeneity. International Conference on Information Systems (ICIS), Orlando, 2012.

[Za97] Zachman, J. A.: Enterprise Architecture: The Issue of the Century. Database Programming and Design Magazine, Nr. 3, 1997.