PROGRAMMIERER, ARCHITEKT UND DANN? DAS BERUFSBILD DES UNTERNEHMENSARCHITEKTEN

Die Berufsbezeichnung „Softwarearchitekt" ist heute derart in Mode, dass man kaum noch Programmierer findet. Parallel dazu hat sich in den letzten zehn Jahren das Berufsbild des „Unternehmensarchitekten" entwickelt. Da es davon weniger gibt, ist diese Bezeichnung heute noch nicht so stark verbreitet. Dieser Artikel positioniert Unternehmensarchitektur als aktuelles Produkt einer Kette immer umfangreicherer Abstraktionen und Metaphern, die man benötigt, wenn man mit einer stark wachsenden Menge von Software konfrontiert ist und versucht, diese zu managen. Er gibt einen Überblick über die Tätigkeitsfelder von Unternehmensarchitekten und zeigt, wohin sich dieses relativ junge Berufsbild aktuell entwickelt.

der autor

Wolfgang Keller
(E-Mail: wk@objectarchitects.de)
arbeitet als freiberuflicher Softwarearchitekt. Er ist unter anderem Autor des 2006 im dpunkt.verlag erschienen Buchs „IT-Unternehmensarchitektur".

Abstraktionen und Metaphern

Der Begriff „Architektur" ist seit mindestens 20 Jahren ein Thema in der Welt der Software. Architektur im Sinne von „Architektur von Gebäuden" gibt es seit tausenden von Jahren – schon vor dem Bau von Pyramiden. Im Gegensatz dazu ist Software ein sehr junges Gebiet. Noch Anfang der 50er Jahre war Software nicht das selbstständige Handelsgut, zu dem sie heute geworden ist. Software, so wie wir sie heute kennen, hat sich erst vor ca. 50 Jahren entwickelt. Davor war sie jeweils an einen Computer gebunden und wurde von den Hardwareherstellern als mehr oder weniger fester Bestandteil ihrer Computer ausgeliefert, d. h. sie war kein separat handelbares Gut. Das änderte sich erst, als sich Gemeinschaften von Anwendern bestimmter Computer bildeten, die zunächst Programme untereinander austauschten. So diente beispielsweise die 1995 gegründete und noch heute existierende Organisation SHARE (heute GUIDE/SHARE der IBM, vgl. [Hur92]) dem damals aufkommenden Austausch von Programmen.

In den folgenden Jahren hatten die größten Softwareprojekte sehr oft neue Betriebssysteme zum Gegenstand. Das Betriebssystem OS/360 von IBM war ein typischer Vertreter dieser Art von Vorhaben. Die Erfahrungen, die Fred Brooks, der Chefdesigner dieses Systems, in den 60er Jahren machte, sind unter anderem in das bekannte Buch „The Mythical Man Month" eingeflossen (vgl. [Bro75]).

Die führenden Akteure der damaligen Zeit betrieben „Programming in the Large". Mit der zunehmenden Menge an Software (siehe Abb. 1) wuchsen aber auch die Probleme. 1968 trafen man sich auf der berühmten Garmischer NATO-Konferenz, um die Mengen an Software besser in den Griff zu bekommen und um Auswege aus der damals wahrgenommenen Softwarekrise zu suchen. In OBJEKTspektrum 6/2008 wurde diese Konferenz ausführlich als Wiege des „Software-Engineering" gewürdigt.

Die *Metapher* (aus dem Griechischen Wort für „Übertragung", abgeleitet von „anderswohin tragen") ist eine rhetorische Figur, bei der ein Wort nicht in seiner wörtlichen, sondern in einer übertragenen Bedeutung gebraucht wird, und zwar so, dass zwischen der wörtlich bezeichneten Sache und der übertragen gemeinten eine Beziehung der Ähnlichkeit besteht. Beispiele für verbreitete Metaphern sind:

- Wüstenschiff – Kamel
- Das Recht mit Füßen treten – Das Recht gering schätzen, verletzen
- Warteschlange – Wartende Reihe von Personen, Fahrzeugen, Aufträgen
- Jemandem das Herz brechen – Jemandem sein Lebensglück zerstören
- Nussschale – Kleines Boot
- Baumkrone – Die Spitze eines Baumes

Kasten 1: *Definition des Begriffs „Metapher" (Quelle: Wikipedia).*

Software-Engineering ist eine Metapher (siehe Kasten 1), die versucht, die Vorgehensweisen von Ingenieurdisziplinen auf die Softwareentwicklung zu übertragen. Menschen, die die Berufsbezeichnung „Softwarearchitekten" trugen, wurden damals noch nicht gesehen.

Die Wissenschaftler, die damals technische Grundlagen – wie Datenabstraktion oder Modularisierung – erarbeiteten, forschten über Programmierung. Sie schufen höhere Abstraktionsebenen, statt der damals noch geläufigen Maschinensprachen, um mit der stetig wachsenden Menge an Software umgehen zu können. In den 60er Jahren entstanden Sprachen mit weniger mächtigen Abstraktionen (zum Beispiel COBOL 1962), Sprachen mit mächtigeren Abstraktionen (zum Beispiel ADA) entstanden allerdings erst um 1980. „Softwarearchitekt" war auch damals noch keine geläufige Berufsbezeichnung und Programmierer waren immer noch die tonangebende Fraktion.

Es sollte bis in die Mitte der 90er Jahre dauern, bis der Begriff Softwarearchitektur wirklich in der Breite verwendet wurde (vgl. [Gar96]) und sich in der Folge weiter ausbreitete. So wie beim Begriff „Software-Engineering" handelt es sich auch bei „Softwarearchitektur" um eine Metapher: Der Bau von Software wurde mit dem Bau von größeren Gebäuden verglichen und es wurde versucht, Methoden und Vorgehen der (Bau)architekten auf den „Bau" von Softwaresystemen zu übertragen. In der selben Zeit wurde mit den *Design Patterns* (vgl. [Gam95]) eine Grundlage geschaffen, um Musterlösungen der Softwarearchitek-

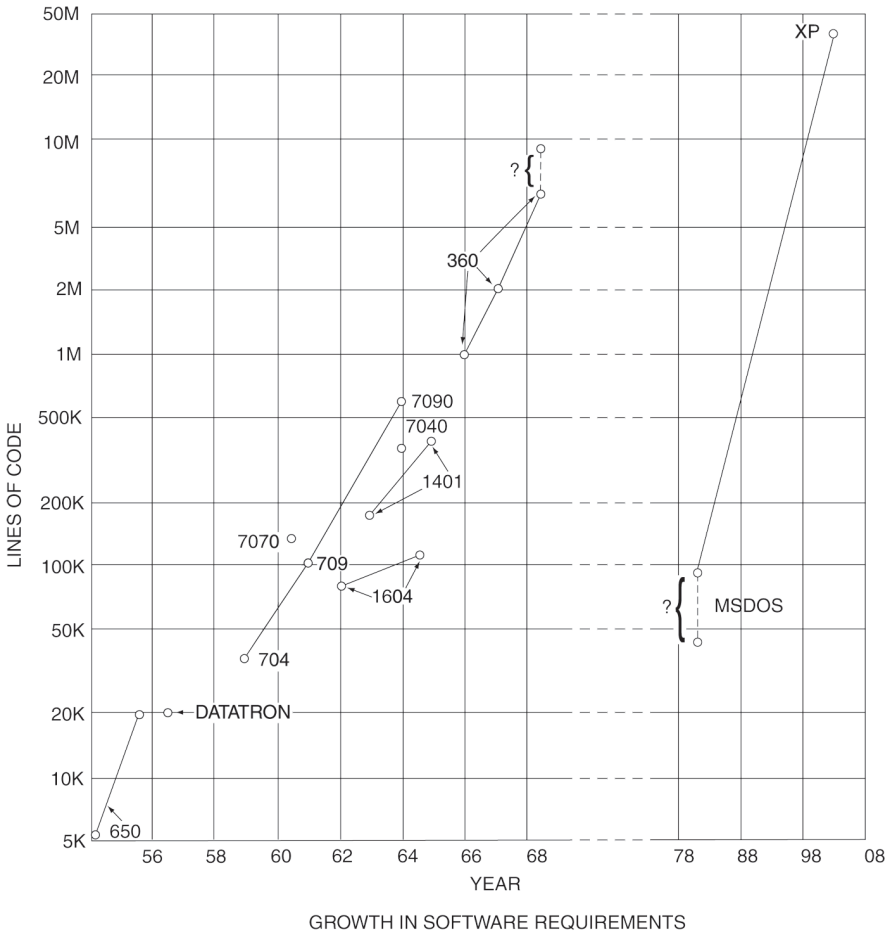

Abb. 1: Exponentielles Wachstum von Programmgrößen (aus: [McC08]).

ten beschreiben zu können. So genannte *Architecture Styles*, die heute zum Repertoire eines jeden Softwarearchitekten gehören, wie zum Beispiel *Pipes and Filters* oder *Blackboard*, findet man als formale Architekturstile bei Garlan und Shaw (vgl. [Gar96]) oder als Architekturmuster bei Buschmann et al. (vgl. [Bus96]). Softwareentwickler, die diese Abstraktionen und die dazugehörigen Beschreibungsmittel, wie zum Beispiel Architektursichten, beherrschten, nannten sich fortan Softwarearchitekten. In der Folge etablierte sich ein Berufsbild, das sich in Kursen, Vorlesungen und nicht zuletzt auch Zertifizierungen niederschlug.

Was ein Softwarearchitekt können sollte, ist also einigermaßen umrissen. Und da Softwarearchitekten in der Regel deutlich besser bezahlt wurden und werden als „bloße" Programmierer, kann man seitdem beobachten, dass sich immer mehr Softwareexperten als Architekten bezeichnen und immer weniger als Programmierer.

Unternehmensarchitekten

Das Wachstum der Systeme machte jedoch nicht an der Grenze einzelner Systeme halt. Die Metapher der *Softwarearchitektur* stößt spätestens bei der Beherrschung kompletter Softwarelandschaften an ihre Grenzen. Daher bildete sich eine weitere Metapher heraus: die der Stadtplanung (*City Planning Metapher*).

Die Systeme aus den 70er und 80er Jahren waren geprägt von *Stovepipe-Architekturen* (Säulenarchitekturen). Das bedeutet, dass die Systeme, die eine Firma für ihr Geschäft und die Erledigung ihrer Geschäftsprozesse benötigte, meist gut vertikal integriert waren – von der Datenbank bis zur Oberfläche –, aber weniger gut horizontal für die Abarbeitung von Geschäftsprozessen. Horizontal waren sie dann beispielsweise über Dateitransfer und Batch-Prozesse miteinander verbunden. Das ist lange her, mögen Sie sagen. Denkt man aber in den zeitlichen Dimensionen der Gebäudearchitektur von tausenden von Jahren, sind die rund 15 Jahre seit Mitte der 90er-Jahre, aus denen wir etwa die ersten Workflow-Systeme wie „IBM FlowMark" kennen, nur ein Wimpernschlag. Diejenigen, die mit solchen Systemen damals als erste versuchten, integrierte Geschäftsprozesse zu schaffen, konnten sich allerdings noch nicht einmal als Softwarearchitekten bezeichnen – geschweige denn als Unternehmensarchitekten.

Ähnlich wie die Softwarearchitekten, die in den 70er und 80er Jahren Softwarearchitektur praktizierten, ohne es zu wissen, praktizierten auch die ersten Unternehmensarchitekten Dinge, die teilweise heute noch nicht zu Ende beschrieben und zu Ende gedacht sind. Heute geläufige Begriffe wie „Anwendungs-Portfolio" gab es damals noch nicht.

Anhänger von John Zachman wenden jetzt vielleicht ein, dass dieser in seinem Artikel „A framework for information systems architecture" (vgl. [Zac87]) das ganze Thema Unternehmensarchitektur ja bereits grundlegend abhandelt. Wenn Sie den Artikel lesen, werden Sie sicher feststellen, dass er auf dem Gebiet der Beschreibung großer Einzelsysteme viel geleistet hat. Die heute für Unternehmensarchitekten wichtigen Begriffe wie Anwendungs-Portfolio, Bebauungsplanung oder Governance kommen dort aber noch nicht vor.

Die erwähnte Stadtplanungs-Metapher als Grundlage für Unternehmensarchitektur kam erst 10 bis 15 Jahre später auf, z. B. in einem Buch von Longepe (vgl. [Lon03]), das weit weniger bekannt ist als die weit verbreiteten Artikel von Zachman. Das maximale Abstraktionsniveau für die Behandlung von Problemen der Softwarekonstruktion hat sich in Intervallen von etwa 10 bis 15 Jahren gesteigert: von einfachen Konstrukten der Maschinensprache, über Abstraktionen der Programmierung (wie Schleifen und Sprünge), Datenabstraktionen und Objekten, bis hin zu Architekturstilen und Mustern für den Gebrauch in einzelnen Systemen. Schließlich betrachten heutige Unternehmensarchitekten Gruppen oder Portfolios von Systemen – die so genannten Anwendungs-Portfolios. Das heißt natürlich nicht, dass irgendeine der vorherigen Abstraktionsstufen und Metaphern überflüssig geworden wäre. Man benötigt heute sogar noch mehr exzellente Programmierer, wenn man

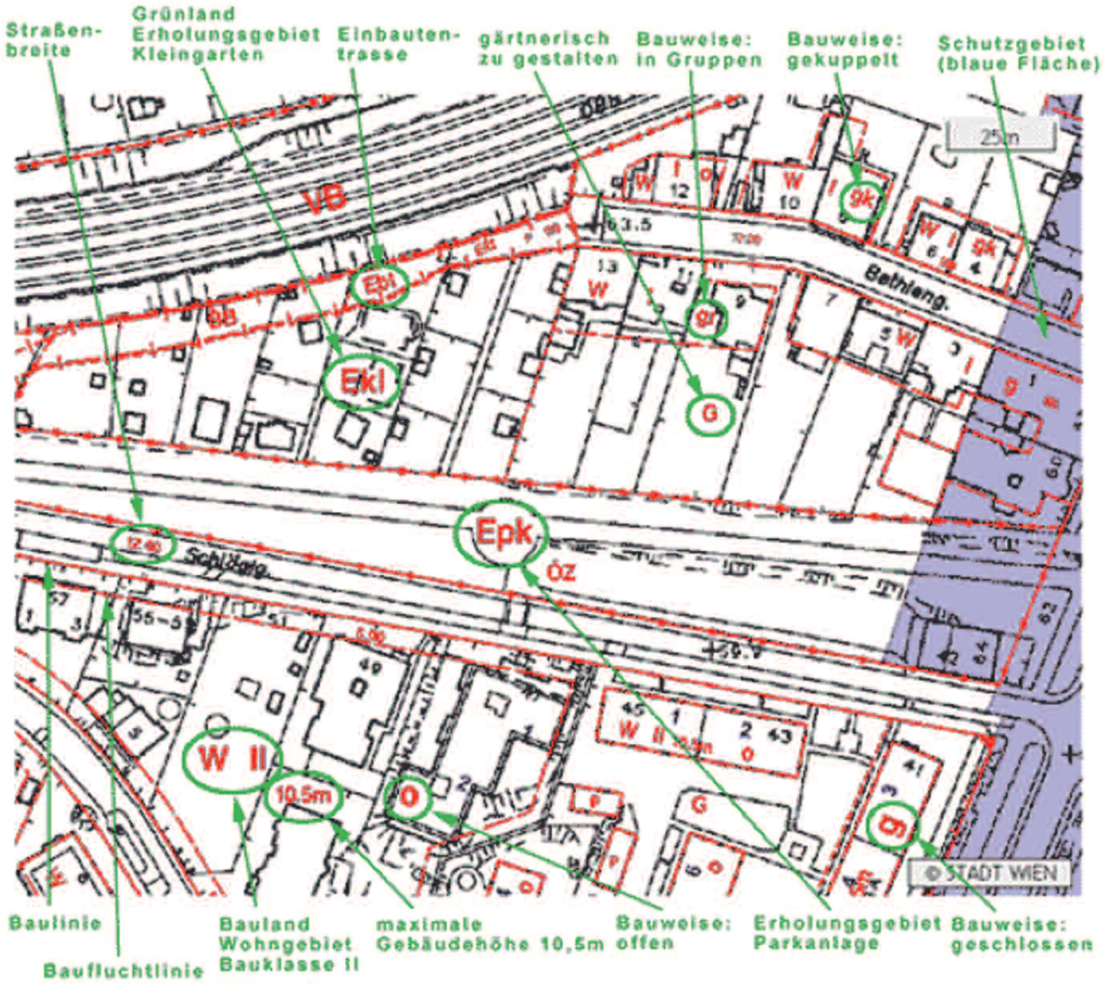

Abb. 2: Beispiel für einen Bebauungsplan (Auszug aus einem Bebauungsplan der Stadt Wien).

die höheren Abstraktionen nicht auf Sand bauen möchte.

Die Stadtplanungs-Metapher

Wenn sich ein Softwarearchitekt mit einem System auseinandersetzt – oder mit einem Cluster aus vielleicht zehn Systemen, die in einem Projekt verändert werden – muss sich ein Unternehmensarchitekt mit der kompletten Menge aller Softwaresysteme einer Firma auseinandersetzen, die diese benötigt, um die Summe ihrer Geschäftsprozesse abzuarbeiten. Das können bei einem mittelständischen Unternehmen 50 bis 100 Systeme sein. Bei einem typischen, global agierenden Automobilhersteller sind es 3.000, 4.000 oder mehr Systeme, in die mehr als eine Milliarde Euro pro Jahr fließen können.

Der Unternehmensarchitekt als „Stadtplaner des CIO" hat es hier – wie ein Stadtplaner – mit so ziemlich allen Gruppen von Stakeholdern eines Unternehmens zu tun: sowohl mit dem Management, als auch mit unterschiedlichsten Gruppen von Anwendern, mit Aufsichtsbehörden (z. B. Finanzaufsicht, Datenschutz) und schließlich mit den Gruppen, die Softwaresysteme neu einbringen oder existierende Software warten oder auch außer Betrieb setzen. Wegen der Ähnlichkeit zu den Aufgaben eines Stadtplaners hat sich die Stadtplanungs-Metapher für das Berufsbild von Unternehmensarchitekten herausgebildet. Wenn man sich die Aufgaben und Arbeitsmittel eines Stadtplaners (z. B. den Bebauungsplan, siehe Abb. 2) ansieht, gehen die Analogien sogar noch weiter.

Ein Stadtplaner macht keine detaillierten Vorgaben für genau ein Gebäude, sondern sorgt durch seine Bebauungsplanung dafür, dass Richtlinien aufgestellt werden. Des Weiteren sorgt er durch seine Projektbegleitungsaktivitäten dafür, dass die Regeln auch eingehalten werden.

Aufgaben von Unternehmensarchitekten

Die Bebauungsplanung für die IT-Systeme eines Unternehmens ist allerdings nur eine der wesentlichen Aufgaben von Unternehmensarchitekten. Im Verantwortungsbereich von Unternehmensarchitekten liegt die Ausrichtung der IT-Strategie des Unternehmens (*IT/Business-Alignment*) – das ist eine ihrer zentralen Aufgaben. Sie müssen also mindestens dafür sorgen, dass das Budget, das in eine Anwendungslandschaft fließt, gemäß den Unternehmenszielen ausgegeben wird. Noch besser ist es, wenn Unternehmensarchitekten aufzeigen können, wie man mit IT völlig neue Chancen für ein Unternehmen aufbauen und nutzen kann.

In dem in Abbildung 3 dargestellten Prozessmodell sind die wesentlichen Aufgaben und Arbeitsgebiete für Unternehmensarchitekten zusammengefasst:

- Zunächst sind Unternehmensarchitekten meist an der *Entwicklung einer IT-Strategie* beteiligt. Ohne solche strategischen Vorgaben fällt es schwer, das Portfolio der Anwendungen sinnvoll zu steuern und ein *Alignment* herzustellen.

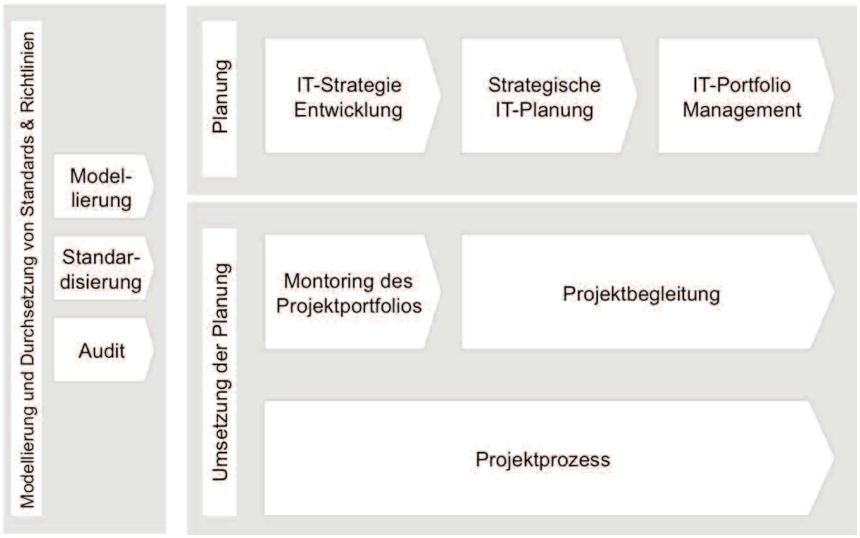

Abb. 3: Wesentliche Prozesse eines Unternehmensarchitektur-Managements (aus: [Der08]).

- Das *IT-Portfolio-Management* umfasst mehr als das Management nur der Anwendungen. Neben dem Anwendungs-Portfolio werden hier auch das Infrastruktur-Portfolio und das Projekt-Portfolio betrachtet. Sinnvoller ist es aber zum Beispiel, sich das Projekt-Portfolio des gesamten Unternehmens anzuschauen, weil es heute kaum noch Projekte ohne nennenswerten IT-Anteil gibt.
- Das Thema der restlichen drei Prozesse des Blocks in Abbildung 3 rechts unten ist die so genannte *Architecture Governance*, also das Überwachen der Umsetzung der von der Unternehmensarchitektur vereinbarten Pläne und Richtlinien. Durch das *Monitoring des Projekt-Portfolios* müssen zunächst interessante, d. h. erfolgskritische, Projekte identifiziert werden. Im Rahmen der *Projektbegleitung* werden diese überwacht, beraten und, wenn nötig, unterstützt. Diese Tätigkeiten müssen in den Projektprozess und damit in den Softwareentwicklungsprozess gut integriert sein.
- Darüber hinaus haben Unternehmensarchitekten zahlreiche *Nebenprozesse* zu erledigen; z. B. müssen sie geeignete Werkzeuge für ihre Arbeit bereitstellen, Architekturrichtlinien abstimmen und verabschieden und die Portfolios auf eine angemessene Standardisierung überwachen.

Falls Sie jetzt erst ein vages Gefühl haben, was denn die genauen Tätigkeitsfelder von Unternehmensarchitekten sind, ist das nur natürlich. Mehr dazu erfahren Sie in den Artikeln von Olaf Kiese und Jens Müller (Seite 34), Oliver F. Nandico (Seite 18) sowie Holger Wolff (Seite 10) in dieser Ausgabe von OBJEKTspektrum. Darüber hinaus gibt es weiterführende Literatur, z. B. [Der09] [Kel06] und [Han08].

Die Metropolis-Metapher

Der Begriff und die Funktion „Unternehmensarchitektur" setzen sich in großen Unternehmen zunehmend durch. Die Softwarekrise, die uns nicht erst seit der Garmischer Konferenz von 1968 begleitet, geht jedoch weiter, da Software immer noch stark wächst und die Software-Schaffenden somit weiter am jeweils oberen Rand ihrer methodisch abgesicherten Kompetenz agieren müssen. Es gibt weitere Phänomene im Umgang mit großen Mengen an Software, für die die Stadtplanungs-Metapher nicht ausreicht. Über B2B, WebServices und MashUps stehen Technologien zur Verfügung, um Softwaresysteme über Unternehmensgrenzen hinweg zu etablieren und zu betreiben. Dabei unterliegt eine Lösung häufig der Kontrolle mehrerer Beteiligter. Oft weiß der Anbieter eines Dienstes nicht, wer diesen nutzt. Ein Unternehmensarchitekt hat also keine klare Unternehmensgrenze mehr, an der er sich in jedem Fall orientieren kann.

Die Forschung interessiert sich bereits für diese neuen Gebiete und Phänomene. Am Software-Engineering Institute (SEI) der Carnegie Mellon University (CMU) wird schon seit längerer Zeit Forschung zu so genannten *Ultra-Large Scale Systems* betrieben (vgl. [Nor06]). Ein solches System ist zum Beispiel die Menge aller Knotenrechner des Internet. Solche Systeme haben Eigenschaften (vgl. [Kaz08]), die sich ein Unternehmensarchitekt heute noch kaum als erwünscht vorstellen kann, denn:

- Sie entziehen sich einer einheitlichen Kontrolle.
- Ihre Anforderungen sind widersprüchlich und niemand kann sie komplett kennen.
- Sie entwickeln sich dynamisch und ohne zentrale Kontrolle.
- Sie reagieren nicht-deterministisch.
- Sie sind ausreichend korrekt, aber nicht beweisbar korrekt.
- Ihr Verhalten entwickelt sich dynamisch.

Meistens beinhalten solche Megasysteme Elemente von Massenkollaboration und wachsen exponentiell. Ähnlich wie Megastädte (*Megacities*) mit Einwohnerzahlen im zweistelligen Millionenbereich, sind sie kaum jemals zu erfassen und zu kartieren und noch weniger bis in den letzten Winkel regierbar. Dazu sind sie zu groß und verändern sich zu schnell.

Für die Gründerväter des Software-Engineerings sind nicht-deterministisches Verhalten, widersprüchliche Anforderungen und ausreichende Korrektheit eher Reizwörter. Für die Kontroll-Freaks in vielen Unternehmen wiederum ist die Abwesenheit einheitlicher Kontrollen ein rotes Tuch und schafft zudem Probleme bei der Einhaltung gesetzlicher Vorschriften (*Compliance*).

Die Metropolis-Metapher bricht mit vielen Vorstellungen der vorangegangenen Metaphern. Architekten können hier lediglich Kerne definieren und diese möglichst stabil gestalten, sodass sich auf der Basis solcher Kerne die „Megalopolis" entwickeln kann. Elemente von Megastädten mit durch Massenkollaboration entstandenen Inhalten findet man vor allem im Bereich Web 2.0 und Open-Source. Dabei entwickeln sich durch Massenkollaboration komplett neue Management-Modelle bei denen lediglich die Kerne einem relativ strikten Management unterworfen sind.

Was bringt die Zukunft?

In den ca. 50 Jahren, in denen man von Software als „kommerziellem Gut" reden kann, haben sich im Rhythmus von 10 bis 15 Jahren neue Metaphern mit einem jeweils höheren Abstraktionsniveau herausgebildet Parallel hierzu sind die Softwaresysteme, die zu entwickeln waren und sind, stark gewachsen. Die Methoden, um mit ihnen umzugehen, wurden entsprechend angepasst. Insofern konnte auch die so genannte Softwarekrise nie enden, weil die Einsatzgebiete von Software und damit auch ihre Größe noch lange nicht am Ende des Wachstums angekommen sind. Neben dem Umgang mit sehr großen Systemen und der Beherrschung der Metropolis-Metapher liegen weitere Herausforderungen in einer deutlich verbesserten Integration von Betriebswirtschaft und Informatik, um von einer IT-Unternehmensarchitektur hin zu einer wirklichen Unternehmensarchitektur zu kommen, in der von Geschäftsmodell und -strategie, bis hin zum IT-Einsatz mit durchgehenden Methoden und Abstraktionen gearbeitet werden kann. ■

Literatur & Links

[Bro75] F.P. Brooks, Jr., The Mythical Man-Month. Essays on Software Engineering, Addison-Wesley 1975

[Bus96] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, M. Stal, Pattern Oriented Software Architecture, A System of Patterns, Wiley 1996

[Der08] G. Dern, W. Keller, Vorlesung IT-Unternehmensarchitektur, Universität Potsdam, Hasso Plattner-Institut 2008

[Der09] G. Dern, Management von IT-Architekturen, Vieweg Verlag 2009

[Gam95] E. Gamma, R. Helm, R. Johnson, J. Vlissides, Design Patterns, Elements of Reusable Object-oriented Software, Addison-Wesley 1995

[Gar96] D. Garlan, M. Shaw, Software Architecture: Perspectives of an Emerging Discipline, Prentice Hall 1996

[Han08] I. Hanschke, Strategisches Management der IT-Landschaft. Ein praktischer Leitfaden für das Enterprise Architecture Management, Hanser Verlag 2008

[Hur92] H. Hurst, 30 years of expert dialogue with IBM – a history of SHARE Europe (SEAS), White Paper siehe: www.daube.ch/share/seas01.html

[Kaz08] R. Kazman, The Metropolis Model: A New Logic for System Development, Vortrag auf der Tagung Informatik 2008, beherrschbare Systeme Dank Informatik, München 2008

[Kel06] W. Keller, IT-Unternehmensarchitektur, dpunkt.verlag 2006

[Lon03] C. Longepe, The Enterprise Architecture It Project: The Urbanisation Paradigm; Kogan Page Science; 2003

[McC08] R.M. McClure, Rückblick: Garmisch 1968 und die Folgen, in: OBJEKTspektrum 6/2008

[Nor06] L. Northrop (Hrsg.), Ultra-Large-Scale Systems: The Software Challenge of the Future, Public Report der Carnegie Mellon University, Software Engineering Institute, Juni 2006

[Zac87] J.A. Zachman, A Framework for Information Systems Architecture, in: IBM Systems Journal, Vol. 26, No. 3, 1987, siehe: www.research.ibm.com/journal/sj/263/ibmsj2603E.pdf