Architect meets Business Analyst: Bei zaghaften Annäherungsversuchen sollte es nicht bleiben!

Schwelgt ein Entwicklerteam zu sehr im Coding, in Technologien oder methodischen Trends, verliert es das „Business" aus dem Blick. Auch bei agiler Methodik und mit zunehmend höherer Architekturebene schlägt oft ein Architekt die Brücke zur Fachlichkeit. Ob Softwarearchitekt, Solution oder Enterprise Architect: Ausreichendes Domänenwissen ist Teil des Jobs. Damit sich ein Architekt im Spannungsfeld zwischen Technik und Fachlichkeit nicht verliert, sollte er mit einem alten Bekannten gemeinsame Sache machen, dem Business Analyst.

Wieso Rollen? Wir sind doch agil!

Zugegeben, so alt ist der Bekannte nicht. Der Business Analyst[1]) als Rolle im Projekt oder im Unternehmen ist eine relativ neue Erscheinung. Sein Verwandter, der Requirements Engineer ist schon länger bekannt. Der Artikel beleuchtet beide Rollenbilder und stellt sie ins Verhältnis zum Architekten.

Im ersten Schritt geht der Artikel primär von der Rollenausprägung des Softwarearchitekten aus, der innerhalb eines Entwicklerteams eine oder auch mehrere Anwendungen betreut. Weitere Rollenausprägungen wie Solution und Enterprise Architect kommen im Kontext höherer Architekturebenen zur Sprache.

Warum ist es überhaupt sinnvoll, über Rollen zu diskutieren? Wir sind doch agil und haben ein interdisziplinäres Team ohne starre Rollenzuteilung! Treten wir einen Schritt zurück. Eine Rolle in einem Vorgehensmodell, einer Projekt- oder Aufbauorganisation ist ein Konzept, in dem mehrere Aufgaben zusammengefasst sind. Dahinter steckt die Idee: Bestimmte Aufgaben gehören inhaltlich zusammen, sind teilweise voneinander abhängig und lassen sich im Normalfall einer einzelnen Person übertragen.

Bei der Definition einer Rolle ist es daher sinnvoll, zunächst die zugehörigen Aufgaben zu umreißen. Bezogen auf den Softwarearchitekten schlägt [Arc42] folgende Aufgaben vor:

- Anforderungen und Randbedingungen klären,
- Strukturen entwerfen,
- querschnittliche Konzepte entwerfen,
- Architektur kommunizieren,
- Umsetzung begleiten,
- Architektur bewerten.

Jene Architekturaufgaben fallen in unterschiedlicher Ausprägung bei jeder Art von Softwareentwicklung an, egal ob sie in einem klassischen oder agilen Kontext stattfindet. Sie fallen auch unabhängig davon an, auf wie viele Schultern sie verteilt werden. Wichtig ist, die Aufgaben überhaupt zu adressieren und sie nicht von vornherein aus ideologischen oder persönlichen Gründen unter den Tisch fallen zu lassen. Andernfalls entsteht eine „zufällige Architektur", die wesentliche Qualitätsziele der Software oder gar die Geschäftsziele des Unternehmens konterkariert.

[Zör15] nennt ein Beispiel für ein häufiges ideologisches Missverständnis bezüglich der Architekturdokumentation: „Das haben wir nicht dokumentiert, wir gehen agil vor." Dokumentation als Unteraspekt der Aufgabe „Architektur kommunizieren" ist nicht obsolet, nur weil das Team agil vorgeht. Allerdings gibt es fundierte Kritik aus dem agilen Umfeld, die nicht die Architekturaufgaben an sich betrifft, sondern die explizite Vergabe der Architektenrolle.

Unter den Aufgaben nimmt der Artikel zunächst die Anforderungsklärung in den Fokus, kommt im letzten Abschnitt aber auf die agile Kritik zurück. Warum ist die Anforderungsklärung wichtig? [Arc42] vermerkt dazu: „Diese Tätigkeit kann entfallen, *wenn* gut geschriebene und qualitätsgesicherte Anforderungen vorliegen." Die Krux liegt jedoch im „Wenn". Je unvollständiger oder unklarer die Anforderungen sind, desto mehr ist ein Architekt tendenziell mit deren Klärung beschäftigt. Darüber hinaus sind die Motivation und die Ausbildung von „IT-lern" meist technischer Natur. Daher erscheint die Gefahr, dass ein Entwicklerteam die Fachlichkeit vernachlässigt, größer als die Gefahr, dass es die Verbindung zur Technik verliert. Die Auswirkungen falsch verstandener Fachlichkeit können ungleich größer sein als falsch eingesetzte Technik.

Der viel zitierte CHAOS-Report der Standish Group weist ebenfalls auf dieses Risiko hin. Seit 1994 untersucht er die Gründe für den Erfolg oder das Scheitern von Softwareprojekten. 2015 umfasste die Datenbasis ca. 50.000 kleine bis große Projekte weltweit. Auch wenn es Kritik an der Methodik des CHAOS-Reports gibt, zeichnet er einen Trend, den andere Studien im Kern bestätigen. Zu den häufigsten Gründen für scheiternde Projekte zählen: unvollständige oder unklare Anforderungen, mangelnde Einbindung der Nutzer sowie unrealistische Erwartungen.

Nicht zuletzt deshalb hat das Thema „Business/IT Alignment" in den letzten Jahren hohe Aufmerksamkeit erfahren. Neuere Gegenstimmen kritisieren vor allem eine *zu* starre Ausrichtung. Ihnen zufolge muss sich IT nicht ausschließlich am „Business" ausrichten und darf sich durchaus in eigener Geschwindigkeit bewegen. Dass IT letztlich dem Unternehmen dienen muss, steht jedoch außer Frage. Wie also steht es mit einem fundierten Interesse und Verständnis für den geschäftlichen Bedarf? Auftritt: Business Analyst!

Business Analyst oder Requirements Engineer?

Zu den Urahnen moderner Business Analysts zählen prominente Ökonomen wie Adam Smith und Frederick W. Taylor. Moderne Vertreter des Rollenbildes sehen sich

[1]) Der Einfachheit halber verwendet der Artikel die männlichen Formen für Rollenbezeichnungen. Selbstverständlich sind auch weibliche Vertreter des jeweiligen Rollenbilds gemeint. Weiterhin verwendet der Artikel bei englischen Rollenbezeichnungen keine deutschen deklinierten Formen, da das Englische keine Kasusendungen kennt.

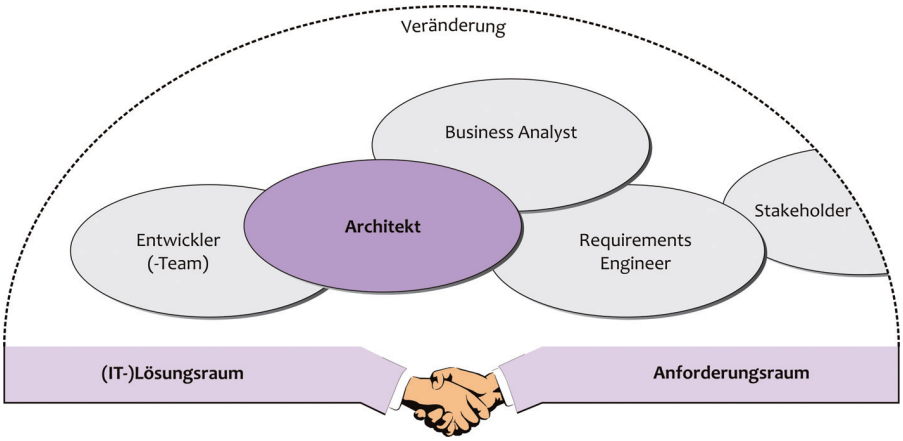

Abb. 1: Rollen und ihre Einordnung in den Lösungs- und Anforderungsraum.

als Mittler zwischen dem geschäftlichen Bedarf, der sich aus Problemen, Chancen und Einschränkungen ergibt, und den Lösungen. Letztere enthalten typischerweise einen hohen IT-Anteil, sind jedoch nicht darauf beschränkt. So können prozessuale oder organisatorische Lösungen auch alleine zum Erfolg führen.

Unter Berufung auf den anerkannten „Guide to the Business Analysis Body of Knowledge" (BABOK Guide) schlägt [Ger15] ein Vorgehensmodell für die Business Analysis vor. Daraus leiten sich folgende Aufgaben ab:

- Informationen ermitteln,
- den Bedarf analysieren,
- mögliche und tatsächliche Lösungen bewerten und evaluieren,
- die Anforderungen definieren,
- bei der Veränderung unterstützen.

Gerade in IT-Projekten bringt man dieses Aufgabenspektrum auch mit der Rolle des Requirements Engineer in Verbindung. In der Tat ist eine Abgrenzung müßig, da sich beide Rollenbilder „irgendwie" um Anforderungen drehen und entsprechende Stellen- und Berufsbezeichnungen austauschbar erscheinen. Obige Auflistung zeigt jedoch: Anforderungen bilden nicht den alleinigen Dreh- und Angelpunkt des Business Analyst. Er nimmt das gesamte Unternehmen und seinen Bedarf in den Fokus.

Zunächst ermittelt der Business Analyst Informationen, indem er Fragen stellt, Daten sammelt und dokumentiert, ohne sie sofort zu analysieren. Hierbei spielen die Stakeholder eine große Rolle, das heißt alle Personen oder Gruppen, die das Vorhaben beeinflussen oder daran interessiert sind. Entgegen einer verbreiteten Auffassung liefern Stakeholder jedoch keine Anforderungen im engeren Sinne. Vielmehr äußern sie Wünsche, Meinungen und Fakten aus ihrer Perspektive. Aufgabe des Business Analyst ist es nun, die verschiedenen Perspektiven in ein Gesamtbild zu integrieren, um daraus den *tatsächlichen* geschäftlichen Bedarf abzuleiten. Nur unter dieser Voraussetzung kann er anschließend die *richtigen* Anforderungen explizit machen.

Hierzu greift er selbstverständlich auf das Repertoire des Requirements Engineering zurück. Wie u. a. [Poh15] zu entnehmen ist, umfasst dieses nicht nur das Ermitteln und die sprachliche oder modellbasierte Dokumentation von Anforderungen. Wichtig sind auch die Prüfung, Abstimmung und Verwaltung von Anforderungen über ihren gesamten Lebenszyklus hinweg.

Bezüglich möglicher IT-Lösungen schlägt der Business Analyst die Brücke zur Softwareentwicklung. Hauptansprechpartner ist hier meist ein Architekt! Das Requirements Engineering zählt ihn für gewöhnlich zu den Stakeholdern. Diese Betrachtungsweise greift jedoch zu kurz, da sich ein Architekt in variierender Intensität selbst um Anforderungen kümmern muss. Folgt man den zitierten Quellen, so „definiert" ein Business Analyst die Anforderungen, während sie ein Architekt abhängig von dieser Vorarbeit „klärt". Dies betrifft insbesondere, aber nicht nur Qualitätsanforderungen,

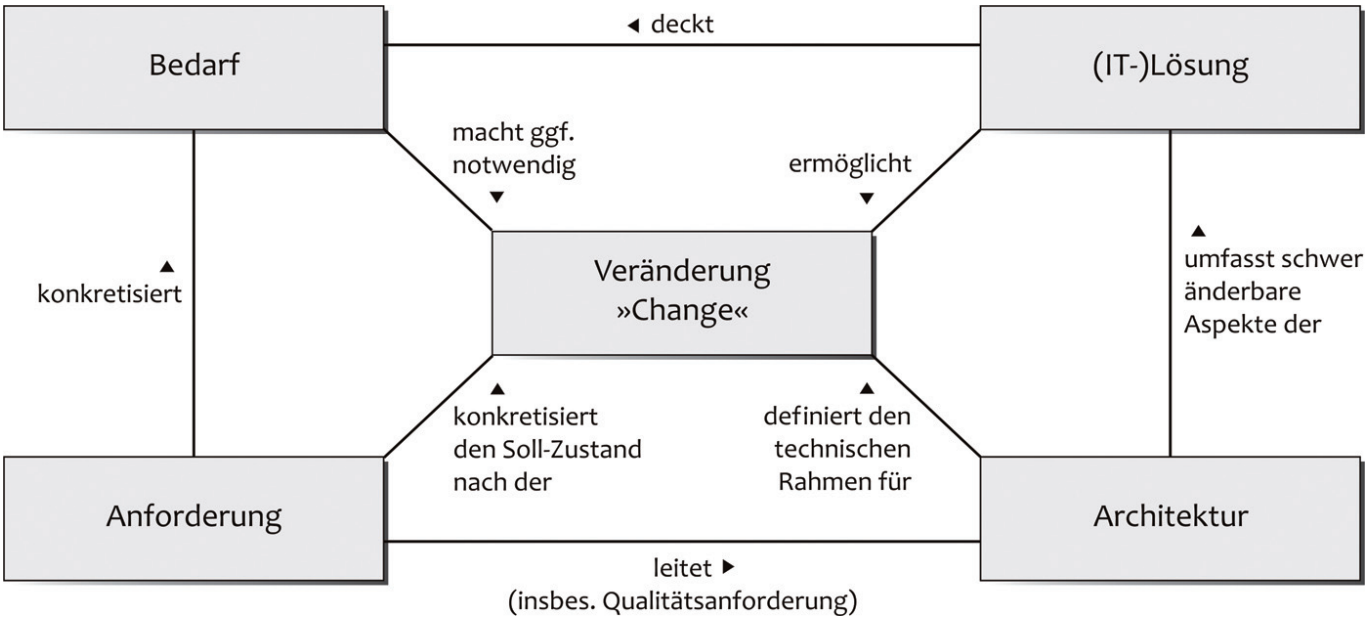

Abb. 2: Begriffswelt des Artikels.

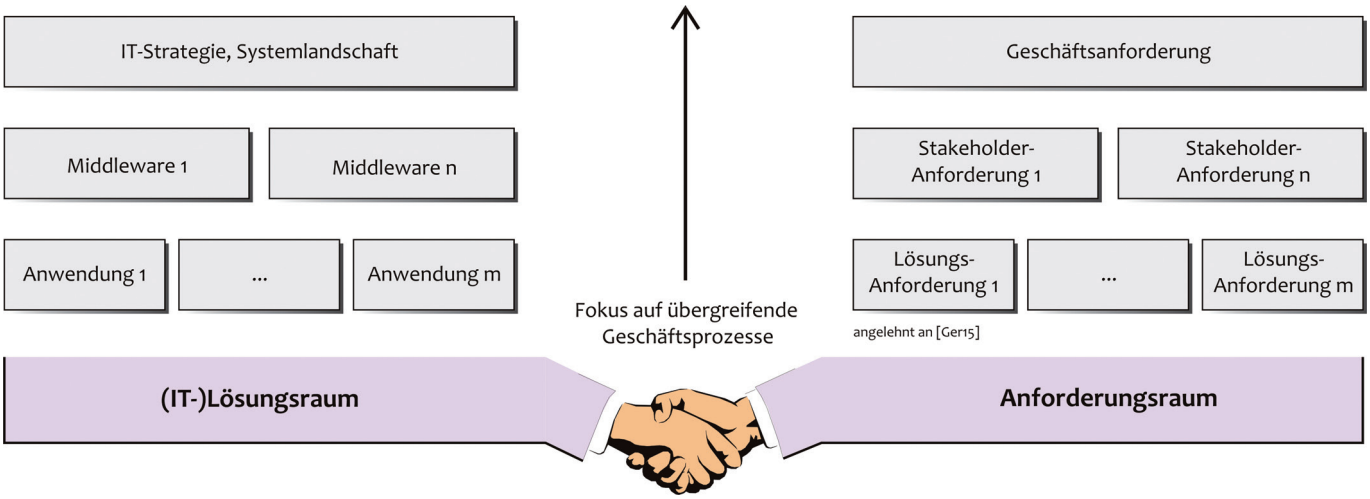

Abb. 3: Ebenen im Lösungs- und Anforderungsraum.

worauf der Artikel im nächsten Abschnitt eingeht.

Abbildung 1 Illustriert das beschriebene Rollenverständnis. Der Architekt ist wie das Entwicklerteam im Lösungsraum der IT angesiedelt, macht jedoch immer wieder Ausflüge in den Anforderungsraum. Dort trifft er auf den Requirements Engineer in Reinform. Umgekehrt begibt sich der Business Analyst regelmäßig in den Lösungsraum und kommt damit dem Architekten entgegen. Stakeholder fügen sich naturgemäß auf der Anforderungsseite ins Bild. Zusammen mit anderen Rollen ermöglichen sie in einem Projekt oder einem kontinuierlichen Verbesserungsprozess die Veränderung vom Ist-Zustand hin zum geplanten Soll-Zustand.

Da „Veränderung" nicht nur in der Business Analysis, sondern auch in agilem Gedankengut und in der IT-Architektur ein Schlüsselbegriff ist, setzt ihn die Concept Map aus **Abbildung 2** in den Mittelpunkt. Die Grafik liest sich wie ein einfaches UML-Fachklassendiagramm, das die Begriffe des Artikels miteinander in Beziehung setzt. „Veränderung" meint hier primär einen signifikanten Wandel in der Aufbau- oder Ablauforganisation, an den Geschäftsprozessen oder den IT-Systemen eines Unternehmens.

Diese Begriffsauslegung rührt vom betriebswirtschaftlichen Change Management her, welches bisweilen als Schwesterdisziplin der Business Analysis gesehen wird. Sie umfasst weiter gehende Methoden und Rollen, die insbesondere die sozialen Aspekte eines solchen Wandels berücksichtigen[2]). Diesbezüglich kann man die Architektur einer IT-Lösung als Rahmen sehen, innerhalb dessen künftige technische Veränderung möglich ist. Der agile Slogan „embracing change" bezieht sich zwar nicht explizit auf denselben Change-Begriff, kann jedoch in diesem Sinne gedeutet werden.

Funktionale oder Qualitätsanforderungen?

Dort, wo sich ein Architekt um das richtige Verständnis des geschäftlichen Bedarfs bemüht, trifft er idealerweise auf einen Business Analyst, der sich um Lösungen bemüht. Gemeinsamer Gesprächsstoff sind Anforderungen, doch welche?

Das Requirements Engineering kennt verschiedene Klassifikationsschemata für Anforderungen. Am gebräuchlichsten ist wohl die Unterscheidung von funktionalen und nichtfunktionalen Anforderungen. Erstere formulieren Erwartungen an die Funktionalität, das heißt, das Verhalten der geplanten Lösung als Reaktion auf eine bestimmte Eingabe. Zu den nichtfunktionalen Anforderungen gehört der potenzielle Rest. Genauer gesagt formulieren sie Erwartungen an querschnittliche *Qualitätsmerkmale* der Lösung.

Jene Qualitätsmerkmale beziehen sich typischerweise auf mehrere Funktionen oder unabhängig davon auf die gesamte Lösung. Hierin mag der Grund liegen, warum der Begriff „nichtfunktional" an Akzeptanz verliert. Zum einen weist er nur darauf hin, was *nicht* gemeint ist. Zum anderen trifft selbst diese Negation nicht immer zu. Zum Beispiel fallen Anforderungen zur Sicherheit eines IT-Systems primär in die Kategorie „nichtfunktional". Gleichzeitig erfordern sie auch funktionale Änderungen des Systems: Der Benutzer muss sich authentisieren.

Im Umfeld des iSAQB, eines Gremiums zur Standardisierung der Ausbildung von Softwarearchitekten, hat sich daher die Begrifflichkeit „Qualitätsanforderungen" beziehungsweise „Qualitätsziele" etabliert. Sie bildet einen wichtigen Ausgangspunkt für die Methodik der Softwarearchitektur. Hierfür gibt es zwei entscheidende Gründe:

- **Häufige Vernachlässigung:** Die Erfahrung zeigt, dass Qualitätsanforderungen gegenüber den funktionalen Anforderungen oft vernachlässigt werden. Schließlich ist die äußere Funktionalität der Lösung viel greifbarer als die Qualität, die „inneren Werte" der Lösung. Letztere sind oft schwer zu messen und wirken sich eher langfristig aus. So hat die eingangs zitierte Aufgabe „Anforderungen und Randbedingungen klären" einen Hang zu Qualitätsanforderungen. Ein Architekt muss sie deutlich öfter einfordern als funktionale Anforderungen.
- **Basis für Architektur:** Wie [Zör15] und [Tot15] unter Berufung auf Eoin Woods respektive Martin Fowler darlegen, manifestiert die Architektur einer IT-Lösung letztlich alle Entscheidungen, die sich im Nachhinein schwer revidieren lassen. Deren Herleitung aus Qualitätsanforderungen erscheint vollkommen plausibel, da sie weite Teile der

[2]) Hiervon abzugrenzen ist der Change-Begriff der IT Infrastructure Library (ITIL), der sich auf die IT-Infrastruktur im engeren Sinne bezieht.

Ebene	Anforderung		
Geschäftsanforderung	Steigerung der Kosteneffizienz im Kunden-Service ohne negative Auswirkungen auf die Kunden		
Stakeholder-Anforderung	Als Mitarbeiter(in) im Kunden-Service möchte ich nicht manuell für jeden Kunden überprüfen, ob er die fehlenden Angaben schon eingereicht hat. Das soll das System machen, damit ich mich auf die schwierigen Fälle konzentrieren kann.	Als Team-Leiter(in) im Kunden-Service möchte ich eine Übersicht der Fälle, die das System nicht automatisch verarbeiten konnte. Die gehen wir dann in der Team-Besprechung durch.	
Lösungsanforderung	Wenn nach Ablauf von 10 Kalendertagen ab Bestelldatum noch notwendige Kundenangaben fehlen, muss das System dem Kunden automatisch eine Erinnerungs-E-Mail zusenden.	Wenn der Kunde Angaben nachgereicht hat, diese aber unvollständig oder widersprüchlich sind, muss das System einen Klärfall anlegen und die weitere automatische Verarbeitung unterbinden.	Das System muss muss dem Front-end des Kunden-Ser vice alle Klär fälle mit einer maximalen Verzögerung von 4 Stunden zur Verfügung stellen.

Tabelle 1: Beispielhafte Kaskade von Anforderungen.

Lösung betreffen. Hierdurch stecken sie den Rahmen für künftige technische Veränderung ab, sofern das Unternehmen nicht gewillt ist, jene „weiten Teile der Lösung" kontinuierlich anzupassen.

Bezüglich der neueren, berechtigten Fokussierung auf Qualitätsanforderungen mag der Eindruck entstehen, der Architekt kümmere sich ausschließlich um sie. Die funktionalen Anforderungen sind jedoch nicht minder wichtig. Meist begründen sie den primären Nutzen für die Stakeholder und somit den größten Geschäftswert, weshalb der Architekt ein natürliches Interesse an ihnen haben sollte. Zudem muss er sie oft anderen Entwicklern nahebringen. Abgesehen davon gehört es zum Standardrepertoire eines Architekten, innerhalb der Funktionalität künftige Veränderungen abzuschätzen. Welche Funktionen werden sich in absehbarer Zeit ändern? Was muss ohne Codeanpassung konfigurierbar sein? Welche funktionalen Aspekte bilden andererseits den stabilen Kern der Lösung?

In der Praxis schließt sich eine interessante Frage an: *Wo* werden die Anforderungen, die ein Architekt beeinflusst oder aufbringt, dokumentiert? Folgt man [Zör15] und [Arc42] gibt es die klare Präferenz, die architekturrelevanten Anforderungen zumindest überblicksartig in der Architekturdokumentation darzustellen. Die Erfahrung zeigt darüber hinaus, dass ein Architekt „seine" Anforderungen mit Nachdruck in das offizielle Anforderungsdokument einbringen sollte. Andernfalls sind sie für die meisten Stakeholder zu wenig sichtbar. Hierzu gehören auch Qualitätsszenarien, welche die Qualitätsanforderungen beispielhaft illustrieren und somit für alle Beteiligten besser greifbar machen.

Lösungsraum = Flachland?

Funktionale und Qualitätsanforderungen beschreiben gewissermaßen, *wie* die Lösung aussehen soll. [Ger15] zählt sie daher zu den „Lösungsanforderungen". Im Gegensatz dazu beschreiben die abstrakteren Geschäfts- und Stakeholder-Anforderungen eher, *was* überhaupt gelöst werden soll. Geschäftsanforderungen setzen dazu bei den Unternehmenszielen an. Stakeholder-Anforderungen hingegen beziehen sich auf einen oder mehrere Stakeholder und damit auf Teilbereiche des Unternehmens. Es entsteht eine Kaskade von wenigen abstrakten bis hin zu zahlreichen konkreten Anfor-derungen, wie die rechte Seite von **Abbildung 3** zeigt. Als Beispiel führt **Tabelle 1** Anforderungen aus dem Umfeld des Business Process Management (BPM) auf, die prinzipiell auf den Kunden-Service einer Bank, Versicherung oder Telekommunikationsfirma anwendbar sind.

Den IT-Lösungsraum durchzieht ebenfalls eine Kaskade, wie die linke Seite von **Abbildung 3** illustriert. So betreffen manche Aspekte der IT-Lösung die IT-Strategie oder die Systemlandschaft, andere wiederum die Middleware-Komponenten zur Anwendungsintegration. Auf dieser Zwischenebene sind Methoden und Werkzeuge für Enterprise Application Integration (EAI), Service-Oriented Architecture (SOA) und Business Process Management (BPM) angesiedelt. Es folgt eine Vielzahl von Anwendungen, von denen normalerweise jede einzelne einen Satz an domänenspezifischen Funktionen umfasst.

Die skizzierten Architekturebenen korrespondieren nicht 1:1 mit den abgebildeten Anforderungsebenen. Umgekehrt gilt dies ebenso wenig. Beispielsweise muss sich eine Geschäftsanforderung nicht auf die gesamte Systemlandschaft beziehen. Wichtig ist jedoch: Je grobgranularer und abstrakter eine Anforderung oder ein Aspekt der Lösung ist, desto mehr Teilbereiche und Geschäftsprozesse des Unternehmens sind potenziell betroffen. Es partizipieren zunehmend mehr Stakeholder und Anwendungen gemeinsam an komplexer Fachlichkeit. Während der Analyse und Umsetzung steigt der Abstimmungsbedarf. Ebenso ist der Geschäftswert der einen oder anderen Lösungsoption nicht trivial zu ermitteln.

Vor diesem Hintergrund verwundert es nicht, dass sich diverse Ausprägungen der Architektenrolle entwickelt haben, wie **Tabelle 2** überblicksartig zeigt. So hat ein Middleware oder Solution Architect typischerweise mehrere Anwendungen innerhalb einer Domäne im Fokus, die sich entlang von Geschäftsprozessen integrieren. Ein Enterprise Architect hingegen setzt auf der Ebene des Gesamtunternehmens an. Auch wenn jene Rollenausprägungen sich zunehmend von der gelebten Softwareent-

Ebene aus Abb. 3	Architekturdisziplin	Rollenbezeichnung
IT-Strategie, Systemlandschaft	Enterprise Architecture Unternehmensarchitektur IT-Architektur *(im Allgemeinen)*	Enterprise Architect
Middleware zur Anwendungsintegration	Solution Architecture Systemarchitektur *(je nach Betrachtungshorizont)*	Solution Architect
Anwendung	Softwarearchitektur Systemarchitektur *(je nach Betrachtungshorizont)*	Softwarearchitekt Applikationsarchitekt

Tabelle 2: Mögliche Zuordnungen von Architekturebenen, -disziplinen und -rollen (ohne Gewähr für Vollständigkeit).

wicklung entfernen, haben sie zumeist einen starken technischen Hintergrund. Anders verhält sich dies bei der relativ neuen Rolle des Business Architect, die zur Fachlichkeit tendiert. Gegenüber der Rolle des Business Analyst grenzt sie sich durch eine stärkere Orientierung an der konkreten Umsetzung ab.

Und was ist nun mit agil?

Wie eingangs angedeutet, üben Agilisten Kritik an der expliziten Vergabe der Architektenrolle. Was steckt dahinter und warum haben Architekten dennoch nicht ausgedient?

Primär ist Agilität kein Vorgehensmodell, sondern eine Geisteshaltung. So sagt das viel zitierte Agile Manifest unter [Agile] nicht etwa: „We have come to value Scrum over Waterfall." Stattdessen drücken der berühmte Wertevergleich und die zwölf begleitenden Prinzipien eine Geisteshaltung aus, die sich von traditionellen, aber trügerischen Sicherheiten löst. Zu letzteren gehören auch starr gelebte Rollenbilder. „Was kann ich als Architekt dafür, wenn der Analyst mir unvollständige Anforderungen über den Zaun wirft?" Eine klare Aufgabenverteilung und Abgrenzung der Zuständigkeiten sind wichtig. Dem Geist des agilen Manifests folgend sind Individuen und Interaktionen im Zweifel jedoch wichtiger.

Was die Architektenrolle angeht, stellt [Cop10] fest: „Domain experts often bear the title of Architect." In diesem Kontext erfährt das bereits 2004 formulierte und in [Cop10] wieder aufgegriffene Vorgehensmuster „Architect also implements" eine neue Bewertung: Technische und fachliche Expertise sollen zwar Hand in Hand gehen, nach agiler Auslegung aber nicht in Form der Architektenrolle. Nicht nur ein einzelner Domänenexperte soll zu den grundlegenden Strukturen der IT-Lösung beitragen, sondern jeder im Team. Auch im Agilen Manifest heißt es: „Fachexperten und Entwickler müssen während des Projektes täglich zusammenarbeiten." Von Architekten ist dort nicht die Rede.

Allerdings haben agile Konzepte und Methoden zumeist die Ebene der Softwarearchitektur im Blick, weniger aber höhere Ebenen wie Solution oder Enterprise Architecture. Weiterhin bringt nicht unbedingt jeder im Team das notwendige Interesse und Verständnis für den geschäftlichen Bedarf auf, um die Architektur darauf ausrichten zu können. Vielmehr geht die Vision der „architektenlosen Agilität" von folgenden Voraussetzungen aus:

- *Beherrschung des Handwerks:* Der Blick für die Fachlichkeit wird erst dann wirklich frei, wenn das Coding, die Technologien und die Methodik im Griff sind. Wenn ein Entwickler oder Architekt stattdessen kämpft, um den Code seiner Kollegen zu verstehen, den Build ans Laufen zu bringen und seine Rolle im Projektvorgehen des Unternehmens einzunehmen, wird er für geschäftliche Belange nicht sehr offen sein.
- *Persönliche Reife:* Beherrscht ein Entwickler das Handwerk, benötigt er dennoch ein gewisses Maß an Eigenverantwortung und Disziplin. Die Schuldigen nicht pauschal auf der Fachseite zu vermuten, sondern proaktiv ein Verständnis für das „Business" zu entwickeln, ist Ausdruck von Eigenverantwortung. Die eigene Experimentierfreude und Lust an moderner Technologie in geordnete Bahnen zu lenken, ist Ausdruck von Disziplin.

Nicht jeder Mitarbeiter bringt diese Voraussetzungen mit, auch wenn sie jeder Vorgesetzte mehr oder minder erwartet. Daher mag es je nach Situation im Team sinnvoll erscheinen, einen primären Ansprechpartner für Business Analysts und Stakeholder festzulegen. Wenn solch ein benannter Architekt jedoch der einzige bleibt, der sich mit dem geschäftlichen Bedarf auseinandersetzt, skaliert die Aufgabenverteilung nicht. Im Spannungsfeld zwischen Technik und Fachlichkeit kämpft der Architekt mit chronischer Überlastung, während die anderen Entwickler auf fachlichen „Input" warten.

[Tot15] begegnet diesem Problem mit der Rollenausprägung des Architecture Owner, nicht ohne deren Kompromisscharakter zu betonen. Ein Architecture Owner agiert mehr als Ansprechpartner und Teilzeit-Coach, denn als domänenkundiger Allein-

Literatur & Links

[Agile] Manifesto for Agile Software Development (Agiles Manifest), siehe: `agilemanifesto.org`

[Arc42] Ressourcen für Softwarearchitekten, siehe: `arc42.de/aufgaben/process.html`

[Cop10] J. Coplien, G. Bjørnvig, Lean Architecture for Agile Software Development, John Wiley & Sons, 2010

[Ger15] I. Gerstbach, P. Gerstbach, Basiswissen Business-Analyse – Probleme lösen, Chancen nutzen, Redline, 2015

[Poh15] K. Pohl, C. Rupp, Basiswissen Requirements Engineering – Aus- und Weiterbildung nach IREB-Standard zum Certified Professional for Requirements Engineering Foundation Level, dpunkt.verlag, 2015

[Tot15] S. Toth, Vorgehensmuster für Softwarearchitektur – Kombinierbare Praktiken in Zeiten von Agile und Lean, Hanser, 2015

[Zör15] S. Zörner, Softwarearchitekturen dokumentieren und kommunizieren – Entwürfe, Entscheidungen und Lösungen nachvollziehbar und wirkungsvoll festhalten, Hanser, 2015

herrscher. Er agiert nicht ausschließlich an der Schnittstelle zur Fachlichkeit, kann dort aber von besonderem Nutzen sein. Auf jeden Fall bleibt er in der Technik verwurzelt. Welchen Stellenwert das Coaching in agilen Vorgehensmodellen hat, zeigt bereits die Rolle des Scrum Master: Er soll agiles Gedankengut und Know-how verbreiten sowie zu Eigenverantwortung und Disziplin ermuntern.

Jenseits der Ebene der Softwarearchitektur steht die Architektenrolle weit weniger infrage. Wie im vorangegangenen Abschnitt erläutert, ist ein Architekt mit steigendem Abstraktionsgrad der Architekturebene zunehmend mit anwendungsübergreifender Fachlichkeit konfrontiert. Indem er den Überblick wahrt, ermöglicht er den anderen Beteiligten, in „ihrer" Domäne in die Tiefe zu gehen.

Fazit

Ist in einem Entwicklerteam kein ausreichendes Interesse und Verständnis für den geschäftlichen Bedarf vorhanden, konterkariert dies in letzter Konsequenz die Unternehmensziele. Unter anderem darin mag die Architektenrolle ihren Ursprung haben: Wer Experte in einer bestimmten Domäne ist, wird wohl die besten Strukturen für eine IT-Lösung in dieser Domäne entwerfen können. [Cop10] stellt diese Annahme zurecht infrage. Fakt ist jedoch, dass die meisten Ausprägungen der Rolle „Architekt" im Spannungsfeld zwischen Technik und Fachlichkeit angesiedelt sind.

Die agile Bewegung hat ebenfalls erkannt, dass die Beziehung zur Fachseite essenziell ist. Ihr ist die explizit vergebene Architektenrolle jedoch ein Dorn im Auge. Jeder im Team soll zu den grundlegenden Strukturen einer IT-Lösung beitragen. Unerwähnt bleibt meist: Dies funktioniert nur, wenn jeder im Team auch ausreichendes Interesse und Verständnis für den geschäftlichen Bedarf aufbringt. Schuldzuweisungen an die Fachseite oder unangemessene Technikverliebtheit sind hier fehl am Platz.

Architekten haben daher nicht immer ausgedient, wenn „agil" vorgegangen wird. Daneben ist vor allem die Architekturebene entscheidend: Mit steigendem Abstraktionsgrad ist ein Architekt zunehmend mit anwendungsübergreifender Fachlichkeit konfrontiert. So hat die agile Bewegung hauptsächlich die Ebene der Softwarearchitektur im Blick, weniger aber die der Solution oder Enterprise Architecture.

Unter Entwicklern kann sich ein Architekt auf jedweder Ebene als Coach engagieren, um fachliches Überblickswissen zu streuen und seinerseits in der Technik verwurzelt zu bleiben. An der Schnittstelle zur Fachlichkeit ist er auf einen dedizierten Business Analyst als Partner angewiesen. Steht ihm kein solcher Partner zur Seite, muss er im Rahmen seiner Möglichkeiten selbst die Anforderungen klären. Dass dies auf Dauer zu Überlastung oder zu unangemessenen IT-Lösungen führt, liegt auf der Hand.

Der Artikel betont deshalb die Relevanz der Business Analysis im Unternehmen. Ein Business Analyst geht über Requirements Engineering in Reinform hinaus, indem er den tatsächlichen geschäftlichen Bedarf und mögliche Lösungen aus verschiedenen Perspektiven eruiert. In der Praxis beschäftigt er sich tendenziell mit funktionalen Anforderungen, ein Architekt eher mit Qualitätsanforderungen. Nicht zuletzt deshalb zahlt es sich für das Unternehmen aus, beide Rollen näher zusammenzubringen! ||

Der Autor

|| Matthias Ostermaier
(info@ostermaier.online)
ist Solution Architect für Java-EE-basierte Middleware bei Vodafone Kabel Deutschland. Sein besonderes Interesse gilt den methodischen und kommunikativen Aspekten der Architekturarbeit.