mehr zum thema:
www.arc42.de

SOFTWAREARCHITEKTEN: DIE ZEHNKÄMPFER DER IT

Softwarearchitekten tragen – wenn sie ihre Aufgabe ernst nehmen – eine Menge Verantwortung im Projekt und benötigen dazu eine Menge an Fähigkeiten. Wir vergleichen in diesem Artikel die Fähigkeiten eines Softwarearchitekten mit den vielseitigen Fähigkeiten, die ein erfolgreicher Zehnkämpfer im Sport mitbringen muss. Einige davon werden nach unseren Erfahrungen in Organisationen und Projekten immer noch unterschätzt. Mit konkreten Praxistipps helfen wir angehenden und erfahrenen Softwarearchitekten, ihr Berufsbild noch besser auszufüllen.

die autoren

Peter Hruschka
(E-Mail: hruschka@b-agile.de)
ist Trainer, Berater und Coach für Software- und Systemarchitekturen und IT-Strategie.

Gernot Starke
(E-Mail: gs@gernotstarke.de)
unterstützt als unabhängiger Berater Unternehmen bei anspruchsvollen IT-Projekten, insbesondere in den Bereichen Softwarearchitektur, iterative Entwicklungsprozesse und Technologiemanagement.

Wir teilen die verbreitete Meinung, dass Softwarearchitekten Technologien gut kennen müssen, um die wesentlichen technischen Entscheidungen für ein System treffen und verantworten zu können. Softwarearchitekten müssen jedoch über eine Vielzahl weiterer Fähigkeiten verfügen, die über reine Technologiekenntnisse hinausgehen. Wir vergleichen im Folgenden Softwarearchitekten mit sportlichen Zehnkämpfern, die in zehn Disziplinen Höchstleistungen erbringen müssen – und nicht nur in einer einzigen.

Beim Zehnkampf müssen Sportler vier unterschiedliche Distanzen laufen (100, 400 und 1.500 m sowie 110 m Hürden), drei unterschiedlichen Sprungbewerbe absolvieren (Weitsprung, Hochsprung und Stabhochsprung), sowie ihre Kraft und Technik in drei Wurfbewerben unter Beweis stellen (Kugelstoßen, Diskuswerfen und Speerwerfen). Dazu benötigen sie viele Fähigkeiten und müssen diese jeweils richtig einsetzen: Sie benötigen Kraft, Technik und Ausdauer, aber auch Schnelligkeit, Koordinations- und Konzentrationsfähigkeit sowie ein rasches Reaktionsvermögen.

Im Folgenden betrachten wir die sechs sportlichen Herausforderungen für Softwarearchitekten, d. h. die Tätigkeiten, die sie verantworten. Danach stellen wir die zehn Fähigkeiten vor, die Softwarearchitekten zur Bewältigung dieser Tätigkeiten benötigen.

Tätigkeiten von Architekten

Softwarearchitekten bilden in Projekten meist das Bindeglied zwischen so unterschiedlichen Stakeholder-Gruppen wie Projektmanagern, Requirements-Ingenieuren, Entwicklern, Testern und Betreibern. Ihre Aufgaben in eine lineare Reihenfolge zu bringen, entspricht dem Versuch der Quadratur des Kreises. Trotzdem haben wir uns in **Abbildung 1** zu einer Reihenfolge durchgerungen, auch wenn diese sechs Tätigkeiten fast immer parallel oder in sehr kurzen Zyklen stattfinden.

Anforderungen und Randbedingungen klären

Wir gehen davon aus, dass eine andere Person als der Softwarearchitekt die Ermittlung und Dokumentation von Anforderungen an das jeweilige System verantwortet, etwa ein Requirements-Engineer, ein Produktmanager (in Scrum-Terminologie: der *Product Owner*), die Fachabteilung, Kunden oder spätere Nutzer.

Tipp: Volere-Template für Anforderungen
Kennen Sie schon das praktische (und kostenfreie) Volere-Template (vgl. [Vol]) zur Systemanalyse? Wir haben damit seit vielen Jahren gute Erfahrungen gemacht und sind der Meinung, dass Volere Softwarearchitekten wertvolle Dienste bei der Überarbeitung von Anforderungen – insbesondere der Qualitätsanforderungen – leisten kann.

TIPP

Tipp: Die Sprache der Stakeholder sprechen
Die Akzeptanz der Kommunikation und Dokumentation steigt drastisch, wenn Sie dabei die Sprache der jeweiligen Stakeholder (Leser, Zuhörer) sprechen – statt in Ihrem eigenen Architekten-Slang zu verharren. Lernen Sie daher fachliche Dialekte und die Management-Sprache, um effektiver mit Auftraggebern und Managern zu kommunizieren. Erwarten Sie niemals, dass diese Personengruppen Ihre technische Sprache lernen!

Für Ihre Aufgabe als Softwarearchitekt müssen Sie sich jedoch ein genaues Bild von der Qualität und Stabilität der Anforderungen verschaffen. Schwachstellen in den Anforderungen müssen Sie nachbessern (oder nachbessern lassen). Dies trifft besonders auf die nicht-funktionalen Anforderungen zu, die gewünschten Qualitätsmerkmale des Systems. Nur all zu oft sehen die Anfordernden diese als selbstverständlich an und versäumen es, sie explizit zu dokumentieren. Qualitätsmerkmale sind jedoch die wesentlichen Architekturtreiber. Als Softwarearchitekt müssen Sie explizite, konkrete und operationalisierte Qualitätsmerkmale unbedingt einfordern oder aktiv nacharbeiten.

Zu dieser Aufgabe gehört es auch, das System sauber gegen Nachbarsysteme abzugrenzen: Klären Sie daher möglichst genau – d. h. exakt, präzise und eindeutig –, was in den eigenen Aufgabenbereich fällt und wo die Schnittstellen zu anderen Systemen liegen. Dabei ist zu hoffen, dass die Systemanalytiker bereits einiges an Vorarbeit geleistet haben. Diese Abgrenzung fällt am Anfang eines Projekts oft schwer. Deshalb sollten Sie diesen Punkt kontinuierlich im Auge behalten. Außerdem sollten Sie als Architekt noch einen kritischen Blick auf die Stakeholder-Liste werfen, falls bereits eine existiert. Ansonsten müssen Sie auch hier Hand anlegen und alle für die weiteren Schritte in der Systementwicklung maßgeblichen Personen, Rollen oder Institutionen ergänzen.

Vor allem aber müssen Sie in diesem Schritt explizite Architekturziele verhandeln, kommunizieren und dokumentieren, um die wesentlichen Qualitätsziele des Systems explizit zu fixieren. Dabei dürfen Sie die Architekturziele nicht mit den Projektzielen verwechseln:

- Die *Projektziele* definieren, wann ein Projekt erfolgreich war. Sie werden vom Auftraggeber und Projektleiter verantwortet und sind meist kurzfristig, d. h. auf die Dauer des Projekts ausgerichtet.

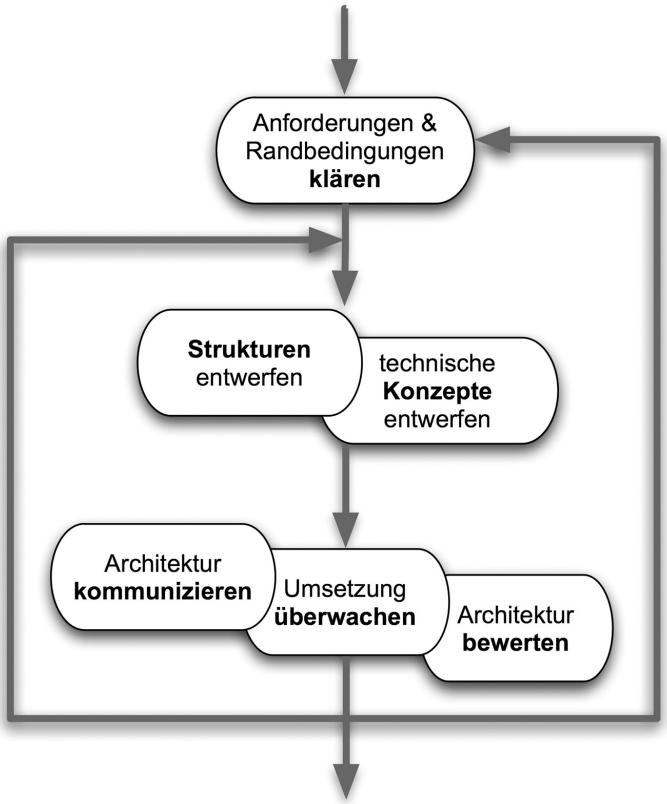

Abb. 1: *Tätigkeiten von Softwarearchitekten.*

- Die *Architekturziele* stellen den Maßstab für eine erfolgreiche Architektur dar. Architekturziele sind meist wesentlich langfristiger ausgelegt als die Projektziele, da die Architektur eine solide Basis für die Weiterentwicklung sein soll.

Ein Architekt wird nicht anhand der Projektziele beurteilt – diese müssen ohnehin erreicht werden. Er erhält seine Lorbeeren, wenn er darüber hinaus die langfristigen Architekturziele möglichst gut erfüllen kann. Dazu müssen sie aber bekannt, abgestimmt und anerkannt sein. Zwischen den Projektzielen und den Architekturzielen gibt es Konfliktpotenzial, wenn kurzfristige Projektinteressen längerfristigen Architekturinteressen eventuell entgegen stehen. In erfolgreichen Projekten werden diese potenziellen Konflikte einvernehmlich zwischen Projektleiter und Architekt aufgelöst. Das ist ein Grund dafür, warum wir selbst in kleineren Projekten für eine personelle Trennung der Rollen „Projektleiter" und „Softwarearchitekt" plädieren. Jeder kann sich dann voll auf seine Ziele konzentrieren und im Konfliktfall die Vor- und Nachteile von Alternativen abwägen.

Strukturen entwerfen
Der Entwurf von Strukturen gilt – zu Recht – landläufig als die Hauptaufgabe von Softwarearchitekten. Gemeinsam mit ihrem Team müssen sie die Strukturen des Systems entwerfen, Technologieentscheidungen treffen und das alles noch für alle Projektbeteiligten nachvollziehbar doku-

mentieren. Das Ergebnis dieser Tätigkeit – die Strukturen des Systems – ist vor allem für die Entwickler die maßgebliche Vorgabe, denn sie müssen das System gemäß diesen Strukturen und den Architektenentscheidungen implementieren. Ähnlich, wie Sie sich als Hauskäufer wahrscheinlich für mehrere Strukturen interessieren (z. B. Grundriss, Elektro-, Heizungs- und Wasserleitungsplan), stehen Ihnen für die Architektur mehrere relevante Strukturen oder Sichten auf das System zur Verfügung (vgl. [Sta08]).

Technische Konzepte entwerfen

Entwurfsentscheidungen, die Sie lokal für einen bestimmten Baustein treffen, sollten Sie dort motivieren und begründen. Manche Entwurfsentscheidungen gelten jedoch über den Kontext einzelner Bausteine hinweg und schlagen sich an vielen, teilweise verteilten Stellen im Quellcode nieder. Solche übergreifenden Themen sollten aber zentral und redundanzfrei dokumentiert werden. Wir schlagen für solche querschnittlichen Themen die Entwicklung technischer Konzepte vor. Damit dokumentieren Sie einen Lösungsansatz, der vom Entwicklungsteam in unterschiedlichen Bausteine gleichartig umgesetzt werden soll, um der Gesamtlösung einen bestimmten Stil aufzuprägen. Die technischen Konzepte verdienen in der Architektur möglicherweise detaillierte Erläuterungen, getrennt von den Strukturentscheidungen, um diese übergreifenden Festlegungen zu motivieren.

Kommen wir nun zu den drei Tätigkeiten, die häufig in der Architektenarbeit übersehen werden oder denen nach unserer Erfahrung oft nicht genügend Aufmerksamkeit gewidmet wird.

Architektur kommunizieren

„Wenn Sie glauben, dass Ihre Architektur gut ist, haben Sie diese noch niemanden gezeigt", sagt ein bekanntes Bonmot – dem wir voll zustimmen. Zu Ihren immer wiederkehrenden Tätigkeiten als Architekt gehört es, allen Stakeholdern – nicht nur dem Entwicklungsteam – die Architektur in geeigneter Weise zu vermitteln. Nur so erhalten Sie von den Stakeholdern wertvolles Feedback zu Entwurfs- und Technologieentscheidungen.

Tipp: Rückmeldungen aktiv einfordern

Fordern Sie von allen Personen im Projekt, deren Arbeit auf der Architektur aufsetzt, aktiv Feedback bezüglich der Tragfähigkeit und Brauchbarkeit der Architektur ein. Dieses Wissen fließt in die ständige Weitergestaltung der Architektur ein, d.h. in Änderungen von Strukturen oder Konzepten. Ebenso gehört die frühzeitige Abstimmung mit Migrations- und Inbetriebnahme-Teams sowie Produktion und Betrieb zu Ihren Überwachungsaufgaben.

Umsetzung überwachen

Wenn ein Entwicklungsteam die Architektur mit detailliertem Leben füllt, Quellcode schreibt oder durch Codegeneratoren erzeugen lässt und Sie dadurch konkretes Feedback zu Ihren Entscheidungen erhalten, ist Ihre Arbeit als Architekt noch lange nicht erledigt. Im Gegenteil: Ihre Rolle und Ihre Verantwortung erfordert die kontinuierliche Überprüfung, Überwachung und Anpassung der vorgegebenen Strukturen, sowie die zeitgerechte Einarbeitung berechtigter Änderungswünsche. Das heißt, Sie passen regelmäßig die Architektur (oder Teile davon) an neue Anforderungen, Randbedingungen oder Erkenntnisse im Projekt an.

Zu diesen Aufgaben gehören regelmäßige Gespräche mit allen Personen im Projekt, deren Arbeit auf der Architektur aufsetzt – also hauptsächlich mit dem Entwicklungsteam, dem Qualitätsmanagement, den Testern und den Administratoren und Betreibern des Systems – und möglicherweise noch vielen anderen.

Bitte versuchen Sie erst gar nicht, die Überwachung technischer Details an Ihre Projektleitung zu delegieren – das gehört zu den Aufgaben eines Architekten – schließlich geht es um Entwurfsentscheidungen und technische Konzepte.

Architektur bewerten

Viele Jahre haben wir nur unser Bauchgefühl genutzt, um festzustellen, ob eine Architektur gut ist oder nicht. Inzwischen sind systematische Verfahren bekannt (vgl. [Cle01]), um eine Architektur gegen vorgegebene Ziele und Qualitätskriterien zu überprüfen. Alle diese Ansätze beruhen darauf auf der Idee, aus den Zielen schrittweise operationelle Szenarien abzuleiten und die Architektur gegen diese Szenarien zu prüfen. Dabei geht es nicht um die Vergabe von Schulnoten für eine Architektur, sondern um Abwägungen, wie gut oder schlecht Entwurfsentscheidungen die Ziele und Qualitätskriterien erfüllen. Die Bewertungstechniken zeigen Ihnen als Architekt potenzielle Schwachpunkte, aber auch Stärken Ihrer Lösung auf, die Sie dann gezielt bei der Strukturbildung beachten können.

Beachten Sie in **Abbildung 1** die Rückkopplungspfeile. Beim Kommunizieren, Überwachen und Bewerten erhält ein Architekt viel Feedback in Form von Kommentaren, Anmerkungen, Kritik, Vorschlägen und Abwägungen von Chancen und Risiken. Diese fließen hoffentlich dauernd in Architekturentscheidungen zu Strukturen und technischen Konzepten ein (**linker Rückkopplungspfeil in Abb. 1**). Manchmal haben die Rückmeldungen der Stakeholder Auswirkungen auf Anforderungen und Randbedingungen, sodass der Architekt diese erneut klären muss (rechter Rückkopplungspfeil), bevor er die Auswirkungen auf Strukturen und technische Konzepte erkennen oder bewerten kann.

Zu jeder der Tätigkeiten gäbe es eine Menge mehr zu sagen und schreiben – einiges davon finden Sie in [Bas04] oder [Sta08]. Wichtiger ist an dieser Stelle: Ihr Erfolg bei diesen Tätigkeiten basiert auf einer Kombination aus den verschiedenen

Tipp: Entscheidungen durch undokumentierte Annahmen vermeiden

Ein schwer wiegender Fehler bei Entscheidungen besteht in der Annahme, Dinge wären „klar", ohne dies explizit zu hinterfragen oder zu dokumentieren. Gehen Sie niemals von wichtigen Sachverhalten aus, ohne diese im Entwicklungsteam klarzustellen. Explizieren Sie Ihre „Annahmen".

Tipp: Die Bedeutung von „architekturrelevant" im Team definieren
Welche Dinge oder Entscheidungen für Ihr System Architekturrelevanz besitzen, sollten Sie gemeinsam mit Ihrem Team entscheiden. Hier einige mögliche Definitionen: Architekturrelevant sind Entscheidungen,

- die Top-Level Strukturen betreffen.
- die das Qualitätsmerkmal „QM" beeinflussen (ersetzen Sie „QM" durch Performance, Sicherheit, Wartbarkeit oder ein anderes wichtiges Merkmal; diesen Satz können Sie in Ihre spezifische Definition mehrfach aufnehmen).
- die mit mehr als X Euro kosten-relevant sind.
- die Auswirkungen auf die externen Schnittstellen besitzen.
- die Auswirkungen auf den späteren Systembetrieb besitzen.
- die Risiken hinsichtlich der Implementierung oder Test besitzen.

Sie erkennen die große Bandbreite derjenigen Entscheidungen, die architekturrelevant sein könnten, aber nicht in jedem Falle sein müssen. Das sollten Sie a priori für Ihr System klären.

Fähigkeiten, die wir uns in den nächsten Abschnitten genauer ansehen.

Fähigkeiten von Architekten

Während beim Sportler Kraft, Ausdauer, Technik, Schnelligkeit, Koordination und Reaktionsvermögen zählen, müssen Architekten die folgenden zehn Eigenschaften in sich vereinen, um die genannten Tätigkeiten erfolgreich zu bewältigen.

Entwerfen

Die Architektur ist die Abstraktion des Quellcodes. Deshalb sollte ein Architekt dazu in der Lage sein, die Struktur im Großen vorzugeben, damit das System die gewünschten Qualitätseigenschaften – z. B. bezüglich Performance, Sicherheit, Wiederverwendbarkeit, Verständlichkeit oder Einbindung von Fertiglösungen – erreicht. Sie fungieren als Konstrukteur, der über einzelne Codezeilen hinaus die Gesamtstruktur des Quellcodes bestimmt oder diese gemeinsam mit dem Entwicklungsteam festlegt.

Sie können dabei auf einige Hilfsmittel zurückgreifen, weil in den letzten 10 bis 15 Jahren einige erprobte Architekturmuster publiziert wurden, die Sie abschreiben, nachahmen oder wiederverwenden können. In [Bus4] sind diese gut zusammengefasst.

Entscheiden

Zu jeder Problemstellung gibt es immer mehrere Lösungen:

- Lösungen mit unterschiedlichen Technologien
- Lösungen zur Optimierung unterschiedlicher, teils widersprüchlicher Ziele oder Qualitätsmerkmale
- Lösungen mit unterschiedlichen Kosten- oder Zeit-Restriktionen

Die Entscheidung für eine dieser vielen Lösungsmöglichkeiten stellt immer eine Entscheidung gegen sämtliche übrigen Alternativen dar. Sie müssen diese Lösung dokumentieren, kommunizieren, gegen anderen Meinungen verteidigen und vermarkten – und für die getroffene Entscheidungen später auch bereit sein, den Kopf hinzuhalten.

Sollten Sie keine Entscheidung bezüglich der Lösungsstrukturen und der eingesetzten Technologie treffen, müssen dies spätestens die Mitglieder Ihres Entwicklungsteams nachholen, weil im Quellcode sämtliche Details festliegen müssen.

Falls Sie auf Einheitlichkeit, konzeptionelle Integrität und übergreifende Verständlichkeit Wert legen – was wir unbedingt empfehlen – sollten Sie als Softwarearchitekt wichtige und architekturrelevante Entscheidungen treffen und nur Details offen lassen.

Vereinfachen

Einfachheit von Entwürfen und Systemen bedeutet leichte Implementierbarkeit, leichte Verständlichkeit, gute Testbarkeit und gute Wartbarkeit. Allerdings entsteht Einfachheit niemals von selbst, sondern muss explizit konstruiert werden. Aber wer im Projekt sorgt neben Einfachheit dafür, dass nicht zu viel Funktionalität gebaut wird, dass die gewünschten Funktionen so einfach wie möglich realisiert werden und dass die Gesamtkomplexität möglichst gering bleibt?

Kunden, Auftraggeber und Nutzer können hierbei nicht helfen. Sie haben Anforderungen und wollen, dass diese erfüllt werden. Und einzelne Entwickler verfügen – insbesondere bei größeren Projekten – häufig nicht über ausreichenden Überblick, um über eine Vereinfachung des gesamten Systems nachdenken zu können. Es bleibt also hauptsächlich an den Architekten hängen, die Frage aufzuwerfen, ob es dafür auch eine einfachere Lösung gibt.

Geben wir es doch zu: Als Techniker sind wir manchmal verspielt. Wir probieren gerne das eine oder andere aus, experimentieren mit neuen Technologien und perfektionieren unsere Lösungen. Als Architekt sollten Sie diesem Spieltrieb widerstehen. Sie sollten den gesamten Entwurf vereinfachen und nicht die schönste und neueste Lösung anstreben.

Tipp: Systematisch vereinfachen
Fragen Sie Ihr Entwicklungsteam regelmäßig, alle zwei bis drei Wochen, nach Möglichkeiten der Vereinfachung. Bringen Sie deren Vorschlägen hohe Wertschätzung entgegen – denn jegliche sinnvolle Vereinfachung reduziert die Komplexität und das Risiko.

Implementieren
„Architect also implements", hat Jim Coplien einmal gefordert. Dazu stehen wir. Als Architekt müssen Sie die Fähigkeit haben, Ihre Strukturen auch in Quellcode umzusetzen. Ansonsten besteht die Gefahr, dass Sie zwar schöne, aber unrealisierbare Pläne entwickeln.

Bei kleinen Projekten sind Sie als Architekt mit Entscheiden und Entwerfen sicherlich nicht ausgelastet. Wenn Sie im Projekt mit insgesamt fünf bis zehn Personen arbeiten, müssen Sie als Architekt zwar einerseits den Überblick über die komplette Lösung behalten, codieren aber bestimmt manche Bausteine oder Systemteile selbst. Bei großen Projekten – mit 50 oder mehr Personen – haben Sie als Chefarchitekt wahrscheinlich keine Zeit mehr, Quellcode selbst zu schreiben. Aber Sie benötigen die Fähigkeit, sich mit Ihren Entwicklern hinzusetzen und kritische Teile im Quellcode zu verstehen und zu beurteilen. Daher ist es wichtig, diese Basisfähigkeit nicht zu verlernen.

Dokumentieren
Ein Softwarearchitekt sollte angemessen, systematisch und methodisch dokumentieren können, denn Dokumentation beeinflusst die langfristige Verständlichkeit von Systemen maßgeblich.

Die Kunst besteht darin, weder zu viel, noch zu wenig zu dokumentieren. Eine zentrale Fähigkeit, die Sie als Architekt entwickeln und pflegen müssen, ist daher: gewusst was und wie! Sie benötigen nicht unbedingt eine vollständige Dokumentation, sofern Sie die wichtigen und relevanten Teile der Systeme dokumentieren. In der Dokumentation sollten alle Projektbeteiligten das Wesentliche der Architektur ohne viel Aufwand finden können. Wir stellen Ihnen dafür als Open-Source das bewährte arc42-Template (vgl. [arc42]) zur Verfügung, um Ihnen Hilfestellung bei der Entscheidung zu geben, was Sie auf welche Weise dokumentieren sollen und was Sie eher weglassen können.

Kommunizieren
Beim Entwerfen und Entscheiden versuchen Sie, eine für alle Betroffenen angemessene, lesbare und verständliche Dokumentation der Softwarearchitektur zu erstellen. Allerdings benötigen unterschiedliche Stakeholder ganz verschiedene Informationen – nicht jeder im Projekt will, soll und muss alles lesen, verstehen und kommentieren. Deshalb müssen Sie als Architekt Stakeholder-gerecht kommunizieren können.

Voraussetzung dafür ist es, dass Sie Ihre Stakeholder alle kennen und dass Sie wissen, wem Sie was zumuten können. Wichtig ist es auch, jedem die Architektur in geeigneter Weise nahe zu bringen. So freuen sich die Entwickler über Bausteinbeschreibungen mit prägnanten Zweckangaben und präzisen Schnittstellen; hingegen interessiert sich der Finanzvorstand eher für die Kosten bestimmter Teilbereiche des Systems und der Produktmanager wahrscheinlich für High-Level-Alternativen. Architekten von anderen Systemen möchten möglichst kompakt Ihre wesentlichen Entwurfsentscheidungen und die Begründungen dafür diskutieren.

Sie sollten zu jedem beliebigen Zeitpunkt die vorliegenden Erkenntnisse und Entscheidungen Stakeholder-gerecht zusammenfassen und präsentieren können. Dazu müssen Sie aus der Gesamtheit der Architekturinformationen die Teile, die für die jeweiligen Stakeholder wichtig sind, in der geeigneten Form (z. B. Präsentationen, Dokumente, Übersichtslisten) herausfiltern können. Achten Sie jedoch darauf, dass diese Extrakte kein Eigenleben entwickeln und aufwändig gepflegt und versioniert werden müssen. Holen Sie das Feedback der Stakeholder ein, um es entweder in die zentralen Sichten und Aspekte einbringen zu können oder die Ziele, die Randbedingungen oder den Kontext anzupassen.

Schätzen und bewerten
Als Architekt sollten Sie Ihre Fähigkeiten, zu schätzen und zu bewerten, ständig weiterentwickeln, um auch mit noch vagen Ideen zu möglichst guten Aussagen über potenzielle Aufwände zu kommen. Ihr Projektleiter und andere Manager dürfen das von Ihnen erwarten. Wenn Sie Vorschläge zur Gestaltung des Systems machen, muss Ihnen klar sein, was das kosten und wie lange es dauern wird. Mit technischer Brillanz alleine lässt sich kein Blumentopf gewinnen. Sie müssen die Kosten für Ihre Vorschläge auf den Tisch legen können.

Und natürlich müssen Sie nicht nur schätzen können, sondern Ihre Architektur auch gegenüber vorgegebenen Architekturzielen und Qualitätsanforderungen bewerten können. Wie sieht der Abgleich zwischen Aufwand und Nutzen aus? Ist diese Alternative im Hinblick auf die gestellten Anforderungen besser als jene? Nicht immer ist die erste Idee auch die beste! Lernen Sie daher explizit, mit Schätzungen und Bewertungen umzugehen.

Balancieren
Wir sind sicher, dass Sie als Architekt diplomatische Fähigkeiten brauchen. Sie werden oft in die Lage kommen, zwischen widersprüchlichen Interessen ausgleichen und vermitteln zu müssen. Denn Qualitätsziele stehen häufig im Widerspruch zueinander: So steht z. B. die Forderung nach Robustheit des Systems der Forderung nach beliebig leichter Flexibilität und Anpassbarkeit entgegen. Und Strukturiertheit und Austauschbarkeit arbeiten gegen Effizienz und Durchsatz. Außerdem treiben fast alle Qualitätseigenschaften die Kosten in die Höhe.

Sie werden daher im Projekt immer wieder Wünsche von Beteiligten gegeneinander abwägen müssen. Sie können es bestimmt nicht allen immer Recht machen. Daher sind Einfühlungsvermögen, Weitblick, politisches Geschick, Verhandlungstalent und Konfliktlösungsstrategien und natürlich Entscheidungskraft wichtige Fähigkeiten eines Softwarearchitekten.

Beraten
Ganz wichtig für einen Architekten ist, dass er die Stakeholder in seinem Projekt oder

Ein letzter Tipp: Die Rollen „Architekt" und „Projektleiter" trennen
Selbst bei kleinen Projekten empfehlen wir eindringlich die Trennung der Rolle des Projektleiters von der des Architekten. Machen Sie den besten Organisator und Politiker zum Projektleiter und den besten Techniker mit Hang zum Zehnkämpfer zum Chefarchitekten, damit die unterschiedlichen Ziele – einerseits Projektabschluss termin- und budgetgerecht, andererseits die oft längerfristigen Architekturziele, wie Ausbaubarkeit und Wiederverwendbarkeit – nicht von einer einzelnen Person verantwortet werden müssen. Sorgen Sie für ein kollegiales, offenes und produktives Verhältnis beider Rollen, um Zielkonflikte im besten Einvernehmen ausdiskutieren zu können.

Fähigkeiten / Tätigkeiten	Anforderungen klären	Strukturen entwerfen	Konzepte entwerfen	Architektur kommunizieren	Umsetzung überwachen	Architektur bewerten
Entwerfen		■	■			■
Entscheiden		■	■		■	
Vereinfachen	■	■	■			
Implementieren		■	■		■	■
Dokumentieren	■			■	■	
Kommunizieren		■	■	■		■
Schätzen	■	■	■			■
Balancieren	■	■	■	■		■
Beraten	■	■	■	■	■	
Vermarkten	■			■	■	

Tabelle 1: Fähigkeiten und Tätigkeiten von Softwarearchitekten.

System aktiv berät. Dazu gehören zum Beispiel die folgenden Bereiche:

- Helfen Sie Ihren Kunden, die sich vielleicht vorstellen können, was sie haben wollen, aber nicht, was davon wie aufwändig ist, dabei, eine Kosten/Nutzen-Abwägung zu treffen.
- Helfen Sie Ihrem Projektleiter bei der Release-Planung, indem Sie ihn beraten, welche Teile wie voneinander abhängig sind und welche daher zusammen implementiert werden müssen.
- Auch bei der Gestaltung der Teamstruktur sollten Sie den Projektleiter aktiv beraten, damit Conways Gesetz beachtet wird.[1]) Versuchen Sie also, dafür zu sorgen, dass Ihr Projektleiter diesen Einfluss kennt und Teilteams richtig zusammensetzen kann.

[1]) *Conway hat beobachtet, dass Projekte besser laufen, wenn die technische Struktur und die organisatorische Teamstruktur kongruent sind. Das ist leicht zu verstehen: Wenn Teilteams jeweils an in sich geschlossenen Aufgaben arbeiten können, gibt es weniger Kommunikationsaufwand untereinander. Wenn die Teamstruktur so unglücklich gewählt wird, dass jeder mit allen anderen sprechen muss, um seine Aufgabe zu bewältigen, leidet der Projektfortschritt erheblich.*

- Natürlich sind Sie als Architekt der Hauptansprechpartner für das Entwicklungsteam, dem Sie mit Rat und Tat in technischen Fragen zur Seite stehen müssen. Gleichzeitig sollten Sie diesen Spezialisten gut zuhören und Ideen von ihnen aufgreifen.
- Auch für die Systemanalytiker haben Sie eine Beratungsrolle, die durch die agile, iterative und inkrementelle Entwicklung noch viel intensiver geworden ist. Die Systemanalytiker benötigen Ihr aktives Feedback, wo die Anforderungen noch präzisiert werden müssen, um gute Entwurfsentscheidungen zu ermöglichen, und wo sie Arbeit sparen und Überspezifikation vermieden können – weil Sie bereits eine preiswerte und treffliche Teillösung parat haben.

Unterschätzen Sie daher nicht, wie viel von Ihren Fähigkeiten als Berater für andere Projektmitglieder abhängt. Warten Sie nicht darauf, gefragt zu werden, sondern treten Sie als Architekt aktiv und wiederholt an Ihre Kollegen heran.

Vermarkten

Es reicht nicht aus, Gutes zu tun – man muss es auch anderen erzählen. Wenn Sie glauben, dass Ihre Entscheidungen gut sind, dann sollten Sie diese aktiv kommunizieren. Erzählen Sie es vielen. Motivieren Sie andere und überzeugen Sie diese, dass die Lösung das Optimum unter den Randbedingungen ist. Kämpfen Sie auch manchmal – im Rahmen Ihrer Möglichkeiten – wenn kurzfristige Projektziele längerfristigen Architekturzielen entgegen stehen. Zeigen Sie aktiv Vor- und Nachteile von Alternativen auf und suchen Sie Verbündete, die die Entscheidungen mittragen.

Tabelle 1 zeigt im Überblick, wie die Zehnkämpfer-Fähigkeiten eines Architekten bei den unterschiedlichen Tätigkeiten zum Tragen kommen. Sie sehen, dass jede einzelnen Tätigkeit eine Kombination vieler Fähigkeiten erfordert.

Sicherlich beherrschen Sie einige der hier beschriebenen Fähigkeiten sehr gut, wenn Sie als Softwarearchitekt arbeiten. Unser Vergleich mit dem sportlichen Zehnkämpfer hat jedoch die Botschaft, dass Sie alle zehn Fähigkeiten in ausgeprägter Form benötigen. Es reicht nicht aus, in einer Disziplin sehr gut zu sein. Die Punkte im

sportlichen Zehnkampf orientieren sich am jeweiligen Weltrekord. Es gibt maximal 1.200 Punkte pro Disziplin, also theoretisch 12.000 Punkte zu gewinnen. Der derzeitige Weltrekordhalter Roman Sebrle bringt es auf mehr als 75 % davon. Im Spitzenfeld der Architekten spielen Sie nur dann mit, wenn Sie im Schnitt eine ähnlich hohe Leistung bringen und in keiner Disziplin zu große Schwachstellen zeigen.

Schwerpunkte je nach Projektgröße

Wir schätzen dass der Softwarearchitekt ca. 50 % der im Projekt anfallenden Aufgaben verantworten muss. Davon entfällt die Hälfte auf den Entwurf und die andere Hälfte auf die Implementierung und den Unit-Test. Für die koordinierenden Arbeiten eines Architekten (also Strukturen entwerfen, Konzepte vorgeben, Stakeholdergerecht kommunizieren, die Entwickler überwachen und die Architektur aktiv vermarkten) veranschlagen wir 3 % des Projektaufwands. Ein Team von 100 Personen kann sich daher drei hauptamtliche Softwarearchitekten leisten. Ein Team von 30 Personen immerhin noch eine Person, für die diese Rolle ein Ganztagsjob ist.

In kleineren Projekten, in denen 3 % des Projektaufwandes keine Vollbeschäftigung darstellen, wirken Sie als Softwarearchitekt neben der Managementunterstützung und den koordinierenden Aufgaben viel mehr im Detailentwurf und in der tatsächlichen Implementierung bzw. beim Test mit.

In größeren Projekten liegt der Schwerpunkt auf der Überwachung von Teilprojekt-Architekten oder Komponentenverantwortlichen.

In Projekten mit drei bis fünf Personen verschwimmt die Rollentrennung oft vollständig: Ein und dieselbe Person klärt die Anforderungen und verantwortet die Lösung – und erledigt nebenbei vielleicht auch noch die Projektleitung.

Wir haben uns in diesem Artikel auf die Architektenrolle konzentriert. Sollten Sie neben der Aufgabe als Softwarearchitekt auch noch andere Aufgaben wahrnehmen, so benötigen Sie noch weitere Fähigkeiten, auf die wir hier nicht eingehen. ■

Literatur & Links

[arc42] arc42 – Das freie Portal für Softwarearchitekten, siehe: www.arc42.de (Dieses frei verfügbare Template sowie dessen methodische Grundlagen haben wir gemeinsam entwickelt und setzen es seit einigen Jahren erfolgreich in Projekten ein.)

[Bas04] L. Bass, P. Clements, R. Kazman, Practical Software Architecture (2. Aufl.), Addison-Wesley 2004

[Bus4] F. Buschmann, K. Henney, D. Schmidt, Pattern Oriented Software Architecture, Vol. 4, Wiley 2007

[Cle01] P. Clements, R. Kazman, M. Klein, Evaluating Software Architectures. Methods and Case Studies, Addison-Wesley 2001

[Sta08] G. Starke, Effektive Software-Architekturen – Ein praktischer Leitfaden. 3. Auflage, Carl-Hanser Verlag, 2008

[Vol] Volere, Template für Requirements, siehe: www.volere.de (Erprobte Starthilfe für Anforderungen – erleichtert auch Softwarearchitekten ihre Arbeit.)