10

10 Bewertung von Software-Architekturen

To measure is to know.
James Clerk Maxwell[1]

Fragen, die dieses Kapitel beantwortet:

- Was können Sie in der IT überhaupt bewerten?
- Warum sollten Sie Architekturen bewerten?
- Was unterscheidet qualitative von quantitativer Bewertung?
- Wie sollten Sie bei der Bewertung von Architekturen vorgehen?
- Welche Stakeholder sollten bei der Bewertung mitwirken?

Bewerten Sie schon, oder raten Sie noch?

Bewertung unterstützt Steuerbarkeit

„*You cannot control what you cannot measure*“,sagt Tom DeMarco treffend. Positiv formuliert, bedeutet diese Aussage für IT-Projekte und Systementwicklung: Wenn Sie kontrolliert auf Ihre Projektziele zusteuern möchten, benötigen Sie dafür Messungen. Sie müssen das Maß Ihrer jeweiligen Zielerreichung ermitteln, um bei Bedarf steuernd eingreifen und korrigierende Maßnahmen einleiten zu können.

Letztlich müssen Sie für Ihre Systeme die Frage „Ist die Architektur gut genug?“ beantworten.

[1] http://de.wikipedia.org/wiki/James_Clerk_Maxwell: Formulierte die Maxwellgleichungen, Grundlagen der Berechnung elektromagnetischer Wellen.

Was Sie in der IT bewerten können

Sie können in Software-Projekten zwei Arten von „Dingen“ bewerten:

- Prozesse, wie etwa Entwicklungs- oder Betriebsprozesse: Hierbei können Sie organisatorische Aspekte betrachten oder aber den Einsatz von Ressourcen bewerten. Leider können Sie auf Basis der Prozesse kaum Aussagen über die Qualität der entwickelten Systeme treffen[2]. Dieses Thema werde ich hier nicht weiter vertiefen.
- Artefakte, wie beispielsweise Anforderungen, Architekturen, Quellcode und andere Dokumente. Einige Arten dieser Artefakte (wie beispielsweise Quellcode) können Sie quantitativ, das heißt in Zahlen, bewerten. Andere (wie etwa die Software-Architektur) entzieht sich der rein zahlenmäßigen Bewertung. Diese Gruppe von Artefakten bewerten Sie qualitativ, also ihrer Beschaffenheit oder Güte nach. Zur letzten Gruppe gehören Software-Architekturen, deren Bewertung *qualitativ* erfolgt.

Die Details qualitativer Bewertung von Architekturen stelle ich Ihnen ab Abschnitt 10.1 vor. Zuvor möchte ich Sie kurz mit den Möglichkeiten quantitativer Bewertung durch Metriken vertraut machen.

Quantitative Bewertung durch Metriken

Es gibt eine ganze Reihe quantitativer Metriken, die Sie für Ihre Projekte und Ihren Quellcode ermitteln können. Einige Beispiele:

- Für Anforderungen: Anzahl der geänderten Anforderungen pro Zeiteinheit.
- Für Tests und Testfälle: Anzahl der Testfälle, Anzahl der Testfälle pro Klasse/Paket, Anzahl der Testfälle pro Anforderung, Testabdeckung[3].
- Für Fehler: Mittlere Zeit bis zur Behebung eines Fehlers, Anzahl der gefundenen Fehler pro Paket.
- Für Prozesse: Anzahl der implementierten/getesteten Features pro Zeiteinheit, Anzahl der neuen Codezeilen pro Zeiteinheit, Zeit für Meetings in Relation zur gesamten Arbeitszeit, Verhältnis der geschätzten zu den benötigten Arbeitstagen (pro Artefakt), Verhältnis von Manager zu Entwickler zu Tester.
- Für Quellcode: Abhängigkeitsmaße (Kopplung), Anzahl der Codezeilen, Anzahl der Kommentare in Relation zur Anzahl der Programmzeilen, Anzahl statischer Methoden, Komplexität (der möglichen Ablaufpfade, *cyclomatic*

[2] Sie können jedoch durch gezielte Reflexionen oder Projekt-Retrospektiven signifikante Verbesserungen erreichen, die zumindest mit hoher Wahrscheinlichkeit positive Auswirkungen auf die Qualität von Systemen haben.

[3] Testabdeckung ist eine verbreitete Metrik, jedoch auch eine sehr kritische. [Binder2000] und [Marick97] beschreiben einige der damit verbundenen Probleme im Detail.

complexity), Anzahl der Methoden pro Klasse, Vererbungstiefe und einige mehr.

- Für ganz oder teilweise fertige Systeme: Performanceeigenschaften wie Ressourcenverbrauch oder benötigte Zeit für die Verarbeitung bestimmter Funktionen oder Anwendungsfälle.

Sie sehen, eine ganze Menge von Eigenschaften können Sie durch solche quantitativen Metriken charakterisieren.

Dennoch: Metriken als Unterstützung der Entwicklung haben sich nach meiner Erfahrung in der Praxis bisher kaum durchgesetzt. Insbesondere liegen in vielen Unternehmen noch keine Erfahrungswerte oder Vergleichszahlen für diese Metriken vor, was ihre Nützlichkeit deutlich mindert (und bei Managern die Akzeptanz erschwert).

> *Beispiel:* In einem mittelgroßen Client/Server-Projekt gab der Kunde einem Forschungsinstitut den Auftrag, die etwa 1000 entwickelten und bereits produktiv (und zur Zufriedenheit des Kunden) eingesetzten Java-Klassen durch Metriken zu analysieren. Ziel dieser Untersuchung sollte sein, spätere Wartungsaufwände besser abschätzen zu können.
>
> Im ersten Bericht der Forscher wurde die Qualität der Software heftig kritisiert, weil verschiedene Metriken starke Abweichungen vom wissenschaftlichen Ideal zeigten.
>
> Kunde, Architekt, Entwickler und Projektleiter waren entsetzt – die Software funktionierte einwandfrei, dennoch wurde die Qualität bemängelt. Damit war die Akzeptanz von Software-Metriken im Entwicklungsteam auf ein Allzeit-Tief gesunken.
>
> Übrigens stellte sich später heraus, dass ein übereifriger Forscher neben den Klassen der Anwendung den gesamten GUI-Framework sowie die zugekaufte Middleware mit analysiert hatte und die bemängelten Metrik-Defekte ausschließlich in den Schnittstellen zu diesen Frameworks lagen.

Einige Ratschläge zum Einsatz quantitativer Metriken (insbesondere Code-Metriken):

- Metriken benötigen einen fachlichen und technischen Kontext, um vergleichbar zu sein. Sammeln Sie Daten aus Vergleichsprojekten.
- Metriken können gute Hinweise für strukturelle Veränderungen von Software geben – über die Funktionsfähigkeit und Qualität zur Laufzeit sagen Codemetriken hingegen nichts aus.
- Metriken können bei unvorsichtiger Anwendung wichtige strukturelle Aussagen über Entwürfe im Zahlendschungel verbergen. Daher: Minimieren Sie die Anzahl der Metriken, und konzentrieren Sie Analysen auf überschaubare Ausschnitte. Weniger ist oft mehr!

- Insbesondere sollten Sie Abhängigkeiten im Quellcode durch ein Werkzeug bei jedem Build[4] messen und überwachen (siehe Abschnitt 6.4).

Systeme nur auf Basis von Quellcodemetriken zu bewerten, ist riskant, weil grundlegende *strukturelle* Schwachstellen dadurch möglicherweise überhaupt nicht aufgedeckt werden. So kann es dann passieren, dass Sie qualitative Defizite erst spät im Entwicklungsprozess finden.[5] Möglicherweise notwendige Änderungen der Architektur sind dann in der Regel teuer und aufwändig. Daher möchte ich mich auf den folgenden Seiten auf die qualitative Bewertung von Architekturen konzentrieren.[6]

Messungen am laufenden System liefern nur Indikatoren

Metriken können jedoch zur Systemlaufzeit (etwa: Performancewerte, Ressourcenverbrauch oder auch Fehlerzahlen) wertvolle Indikatoren für Probleme liefern. In jedem Fall müssen Sie auf Basis solcher Messwerte noch *Ursachenforschung* betreiben!

Beginnen Sie bereits in frühen Entwicklungsphasen mit der Erfassung von Laufzeitmetriken, insbesondere Performancedaten (etwa: Durchlaufzeiten wichtiger Anwendungsfälle und Ressourcenverbrauch). Ermitteln Sie diese Werte automatisiert und achten Sie auf Trends.

Bewerten Sie insbesondere Architekturen

Von den vielen Artefakten, die innerhalb von Software-Projekten entstehen, eignen sich Architekturen meiner Meinung nach besonders für die Bewertung:

- Architekturen entstehen als eines der ersten Artefakte innerhalb der Software-Entwicklung deutlich früher als die vollständige Implementierung.
- In der Architektur werden Entwurfsentscheidungen von großer Tragweite getroffen, sowohl für die zu entwickelnden Systeme als auch für die Projekte und Organisationen, die diese Systeme realisieren, benutzen und betreiben. Viele dieser Entwurfsentscheidungen fallen unter Unsicherheit oder (vagen) Annahmen und sind daher risikobehaftet. Architekturbewertung hilft, diese Risikofaktoren zu identifizieren.

[4] Sie haben doch hoffentlich einen *Daily Build* etabliert, inklusive täglicher Durchführung sämtlicher Unit-Tests, oder? Solche Werkzeuge (wie (n)ant, CruiseControl oder Maven) können Abhängigkeitsmetriken berechnen.

[5] Iterative Entwicklungsprozesse mit frühzeitigem Feedback vermeiden dieses Problem.

[6] Ich rate Ihnen aber, in Entwicklungsprojekten Codemetriken zur Steigerung der Qualität einzusetzen. Sie sollten in jedem Fall Komplexitäts- und Abhängigkeitsmaße auf Codeebene messen und überwachen – viele Entwicklungsumgebungen unterstützen Sie dabei.

Bewerten Sie regelmäßig

Während der Systementwicklung sollten Sie regelmäßig die Frage „Gut genug?“ stellen und auch beantworten. In Kapitel 3 haben Sie das Vorgehen der Architekturentwicklung kennen gelernt – darin kommt die (regelmäßige) Aufgabe der Architekturbewertung vor. Abbildung 10.1 zeigt Ihnen zur Erinnerung eine kompakte Version des gesamten Vorgehens.

In der Praxis finden Bewertungen (unter dem Titel „Audit“ oder „Review“) oftmals in späten Entwicklungsphasen statt, wenn das betroffene System bereits im produktiven Einsatz läuft. Viel zu selten investieren laufende Projekte in die Bewertung ihrer Architektur – obwohl das zumindest mittleren oder großen Projekten guttäte.

> Bewerten Sie so früh wie möglich. Die Ergebnisse qualitativer Architekturbewertung sind für Projekte und Systeme meistens von langfristigem Wert. Investition in Architekturbewertung lohnt sich.[7]

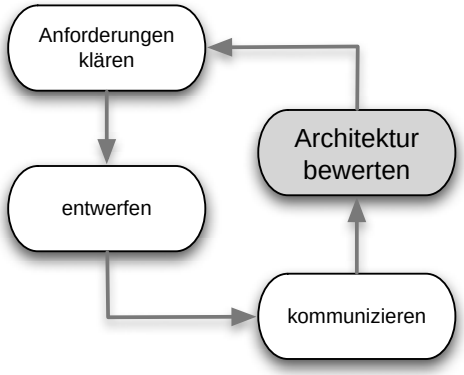

Abbildung 10.1
Architekturbewertung als Teil der Architekturentwicklung

10.1 Was Sie an Architekturen bewerten können

Bewertung von Qualitätsmerkmalen

Im Gegensatz zur oben beschriebenen *Messung* von Quellcode anhand bestimmter Metriken liefert die Bewertung von Software-Architekturen keine Zahlen, sondern qualitative Aussagen: Sie bewerten Architekturen danach, inwieweit sie die Erreichung bestimmter Qualitätsmerkmale ermöglichen.

Beispielsweise treffen Sie Aussagen darüber, in welchem Maß das jeweilige System den Anforderungen an Wartbarkeit, Flexibilität, Performance oder Sicherheit genügt.

[7] Zumal auch kurze Bewertungsworkshops von wenigen Stunden Dauer oftmals schon kritische Risiken in Architekturen aufdecken können – erheblich schneller als umfangreiche Code- oder Schwachstellenanalysen.

Dabei liefert eine Architekturbewertung kein absolutes Maß, d.h. sie bewertet nicht die „Güte-über-alles“, sondern nur im Hinblick auf spezifische Kriterien. Architekturbewertung hilft, Risiken zu identifizieren, die sich möglicherweise aus problematischen Entwurfsentscheidungen ergeben.

Geeignete Messgrößen für Software-Architekturen

In Kapitel 3 (Vorgehen bei der Architekturentwicklung) haben Sie wichtige Qualitätsmerkmale von Software-Systemen kennen gelernt. Ich habe Ihnen gezeigt, wie Sie mit Hilfe von Szenarien diese Qualitätsmerkmale beschreiben und operationalisieren können.

Mit Hilfe von Szenarien können Sie spezifische Messgrößen für Ihre Systeme definieren oder, noch besser, von den maßgeblichen Stakeholdern definieren lassen.

Qualität ist spezifisch für Stakeholder

Dabei gilt es zu ermitteln, welche spezifischen Kriterien die Architektur zu erfüllen hat. Anhand dieser Kriterien können Sie dann die „Güte“ oder „Eignung“ der Architektur ermitteln.

Eine Architektur ist in diesem Sinne geeignet, wenn sie folgende Kriterien erfüllt:

- Das System (das mit dieser Architektur entwickelt wird oder wurde) erfüllt sowohl seine funktionalen als auch nichtfunktionalen Anforderungen (Qualitätskriterien).
- Insbesondere erfüllt das System die spezifischen Anforderungen an Flexibilität, Änderbarkeit und Performanceverhalten.
- Das System kann mit den zur Verfügung stehenden Ressourcen realisiert werden. Sowohl das Budget wie auch der Zeitplan, das Team, die beteiligten Fremd- und Legacy-Systeme, die vorhandene Basistechnologie und Infrastruktur sowie die gesamte Organisationsstruktur beeinflussen die Umsetzbarkeit einer Architektur.

Zur Bewertung einer Architektur ist es daher von entscheidender Bedeutung, die spezifischen Qualitätskriterien des betroffenen Systems zu kennen. In der Praxis werden diese Kriterien oder Qualitätsziele oft erst bei einer Architekturbewertung expliziert.

> Bereits in Kapitel 3 habe ich Ihnen empfohlen, die Qualitätsanforderungen an Ihre Systeme explizit durch Szenarien zu dokumentieren. Diese Szenarien bilden eine wertvolle Orientierungshilfe für weitere Entwicklungsschritte und eine ausgezeichnete Basis für die Architekturbewertung!

10.2 Vorgehen bei der Bewertung

Ich möchte Ihnen im Folgenden ein methodisches Vorgehen zur Architekturbewertung vorstellen, das sich an der ATAM[8]-Methode von [Bass+03] und [Clements+03] orientiert. Abbildung 10.2 gibt Ihnen einen Überblick.

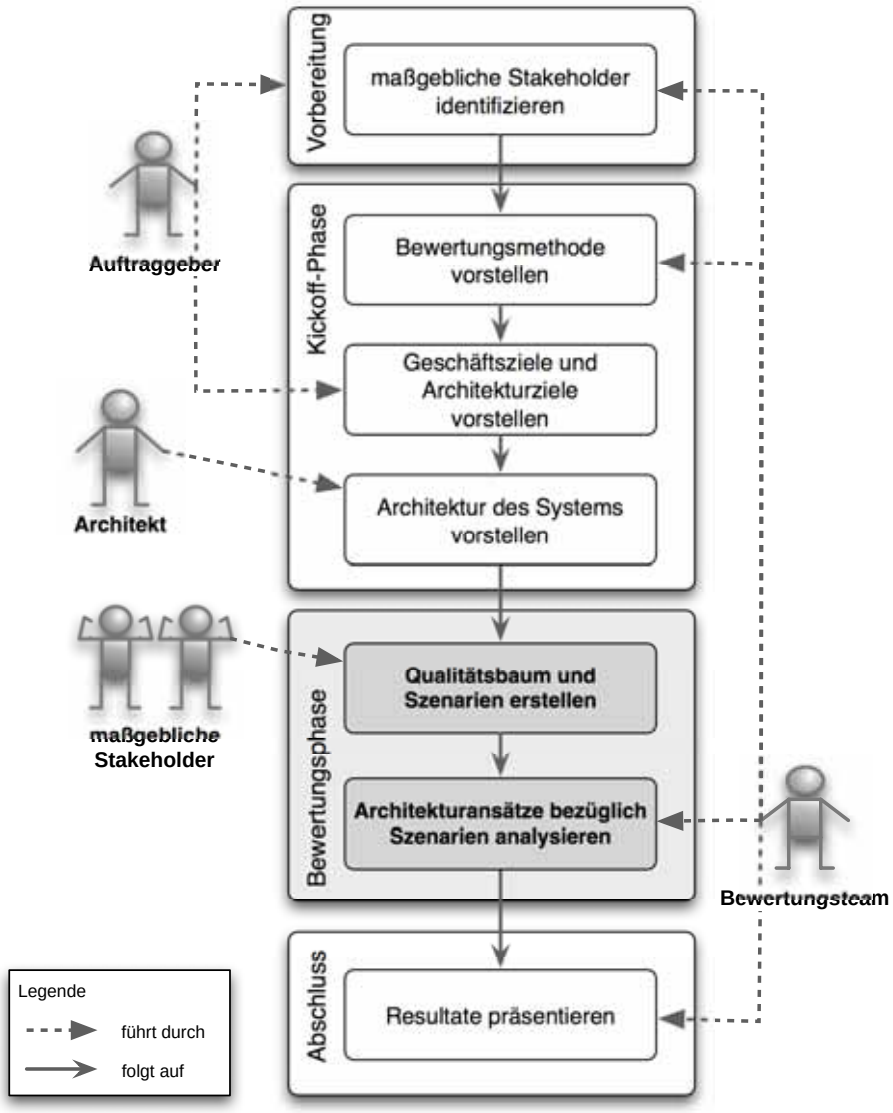

Abbildung 10.2 Vorgehen bei der Architekturbewertung

[8] Architecture Tradeoff Analysis Method.

Der fundamentale Schritt bei der Architekturbewertung ist die Definition der von den maßgeblichen Stakeholdern geforderten Qualitätsmerkmale in möglichst konkreter Form. Als Hilfsmittel dazu verwenden Sie Qualitätsbäume und Szenarien (siehe auch Abschnitt 3.6.2). Ein Beispiel sehen Sie in Abbildung 10.3.

Maßgebliche Stakeholder identifizieren

Da die konkreten Ziele Ihrer Architekturen ausschließlich durch die spezifischen Ziele Ihrer Stakeholder vorgegeben werden, muss der Auftraggeber oder Kunde im ersten Schritt jeder Architekturbewertung die hierfür entscheidenden („maßgeblichen“) Stakeholder identifizieren. In der Regel geht eine Architekturbewertung von wenigen Stakeholdern aus, häufig aus dem Management oder der Projektleitung.

Stimmen Sie mit Ihren Auftraggebern gemeinsam ab, welche Stakeholder bei der Festlegung der Bewertungsmaße mitarbeiten sollen.[9] Denken Sie – neben Ihren Auftraggebern oder Kunden – in jedem Fall an Endanwender und Betreiber.

Bewertungsmethode vorstellen

Bevor Sie mit der Definition der Bewertungsziele beginnen, sollten Sie den maßgeblichen Stakeholdern die Bewertungsmethode vorstellen. Insbesondere müssen Sie die Bedeutung nachvollziehbarer und konkreter Architektur- und Qualitätsziele klarstellen.

In dieser Vorstellung sollten Sie den Beteiligten verdeutlichen, dass es bei der qualitativen Architekturbewertung um die Identifikation von Risiken und Nicht-Risiken sowie um mögliche Maßnahmen geht, nicht um die Ermittlung von *Noten*.

Geschäftsziele vorstellen

Im Idealfall stellt der Auftraggeber die geschäftlichen (Qualitäts-)Ziele des Systems vor, das zur Bewertung ansteht. Dabei sollten der gesamte geschäftliche Kontext zur Sprache kommen, die Gründe für die Entwicklung des Systems sowie dessen Einordnung in die fachlichen Unternehmensprozesse.

Falls diese Ziele in Ihren Projekten bereits in den Anforderungsdokumenten aufgeführt sind, lassen Sie dennoch die *maßgeblichen Stakeholder* zu Wort kommen: Bei der Bewertung geht es um deren aktuelle Sichtweise. Außerdem sind die Zielformulierungen aus den Anforderungsdokumenten von Systemanalytikern gründlich gefiltert worden.

[9] Ich nenne diese spezielle Gruppe von Stakeholdern gerne die *maßgeblichen* Stakeholder.

Ich teile die Erfahrung von [Clements+03], die von *Aha-Erlebnissen* bei der Vorstellung der aktuellen Ziele berichten: Die unterschiedlichen Teilnehmer von Bewertungsworkshops sitzen häufig das erste Mal gemeinsam an einem Tisch und lernen aus erster Hand die Zielvorstellungen anderer Stakeholder kennen. Dabei kommt es zu wertvollen „Ach sooo war das gemeint"-Erlebnissen.

Architektur vorstellen

Anschließend sollte der Architekt des Systems die Architektur vorstellen. Dazu eignet sich die Gliederung der Architekturpräsentation, die Sie in Kapitel 4 kennen gelernt haben.

Es sollte in jedem Fall der komplette Kontext des Systems ersichtlich werden, das heißt sämtliche Nachbarsysteme. Weiterhin gehören zu einer solchen Vorstellung die Bausteine der oberen Abstraktionsebenen sowie Laufzeitsichten einiger wichtiger Use-Cases oder Änderungsszenarien.

Einen spezifischen Qualitätsbaum erstellen

Lassen Sie Ihre Stakeholder im Anschluss an die Vorstellung der Geschäftsziele sowie der Architektur im Rahmen eines kreativen Brainstormings die wesentlichen geforderten Qualitätsmerkmale erarbeiten. Ordnen Sie diese anschließend in hierarchischer Form an: Die allgemeineren Merkmale stehen weiter links (oder oben im Baum), die spezielleren Anforderungen weiter rechts (oder unten).

Ein Beispiel eines solchen Qualitätsbaumes finden Sie in Abbildung 10.3. Dort sind als globale Qualitätsziele Performance, Erweiterbarkeit und Verfügbarkeit eingetragen und durch Latenz, Durchsatz etc. verfeinert.

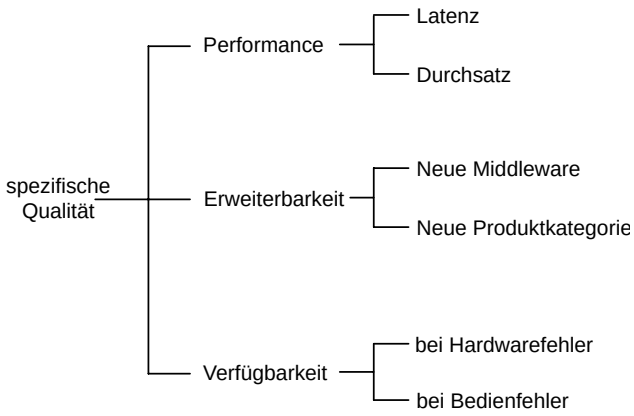

Abbildung 10.3 Beispiel eines Qualitätsbaumes

Entwickeln Sie Ihre Qualitätsbäume in kleinen und streng fokussierten Workshops, im Stile von Brainstorming-Sitzungen. Lassen Sie die Teilnehmer zu Beginn Architektur- und Qualitätsziele in ihren eigenen Worten beschreiben, ohne sie unmittelbar einzuordnen oder zu bewerten.

Falls Ihre Stakeholder etwas Starthilfe benötigen, können Sie die Architektur- und Qualitätsziele sowie Mengengerüste aus Anforderungsdokumenten in diese Diskussion einbringen. Ich finde es jedoch produktiver, die Teilnehmer einer solchen Diskussion möglichst wenig durch bestehende Dokumente zu belasten.

Anschließend strukturieren Sie gemeinsam mit den Teilnehmern die Beiträge in Form des Qualitätsbaumes. Dabei können Sie die Wurzel des Baumes auch anders benennen, etwa: „*Nützlichkeit*“ oder „*wichtigste Architekturziele*“.

> Mindmaps können bei der Erstellung und Dokumentation von Qualitätsbäumen gute Dienste leisten. Sie sind leicht verständlich und insbesondere für ein *Brainstorming* von Qualitätsmerkmalen geeignet.

Damit haben Sie den ersten Schritt zur Beschreibung der Bewertungsziele bereits erledigt.

Qualitätsmerkmale durch Szenarien verfeinern

Szenarien: Abschnitt 3.6.2

Im folgenden Schritt entwickeln Sie gemeinsam mit den maßgeblichen Stakeholdern Szenarien für die wichtigsten Qualitäts- und Architekturziele. Wichtig ist dabei, die Beschreibung der Szenarien möglichst konkret und operationalisiert zu halten. In Abbildung 10.4 sehen Sie einen um zwei Szenarien angereicherten Qualitätsbaum.

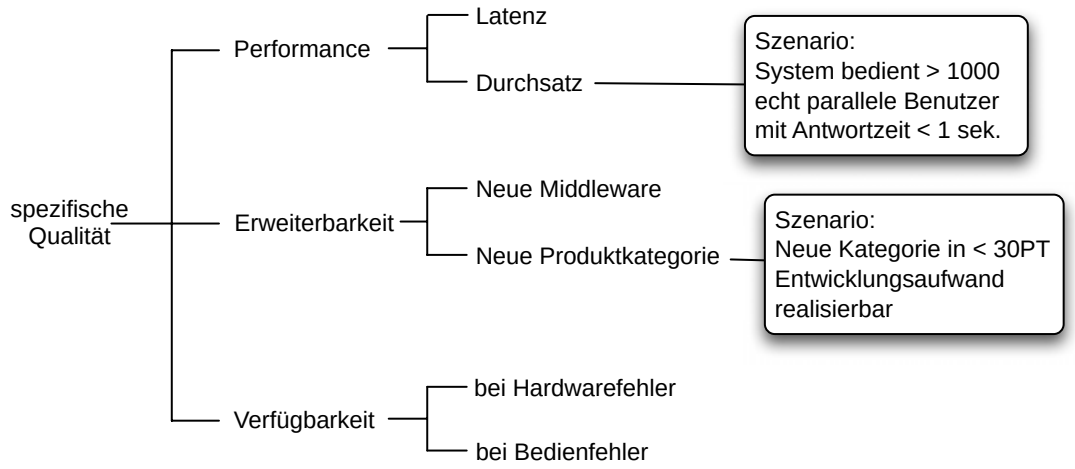

Abbildung 10.4 Qualitätsbaum mit Szenarien

Oftmals fällt es den Projektbeteiligten leichter, in konkreten Szenarien zu denken als in abstrakten Qualitätsmerkmalen. Beginnen Sie in solchen Fällen ruhig mit einem Brainstorming von Szenarien, und fügen Sie die zugehörigen Qualitätsmerkmale erst im Nachgang hinzu.

Bereits für mittelgroße IT-Systeme können Sie in solchen Brainstorming-Sitzungen durchaus 30 bis 50 verschiedene Szenarien finden. Im Normalfall wird Ihnen für die detaillierte Bewertung derart vieler Szenarien jedoch keine Zeit bleiben, so dass Sie sich auf einige wenige beschränken müssen. Dabei helfen Prioritäten.

Qualitätsmerkmale und Szenarien priorisieren

Lassen Sie Ihre maßgeblichen Stakeholder die gefundenen Szenarien nach ihrem jeweiligen geschäftlichen Wert oder Nutzen priorisieren. Ich bevorzuge dabei eine kleine Skala, etwa A = wichtig, B = mittel, C = weniger wichtig, exemplarisch dargestellt in Abbildung 10.5. Bei umfangreichen Sammlungen von Qualitätszielen kann es einfacher sein, die Ziele fortlaufend zu nummerieren. Diese Prioritäten bestimmen später die Reihenfolge der Bewertung.

Zusätzlich sollten die beteiligten Architekten die Szenarien nach ihrer technischen Schwierigkeit ebenfalls in einer kleinen Skala (etwa: A = sehr schwierig oder sehr riskant, B = normal, C = leicht, ohne Risiko) einstufen. Diese zweite Dimension der Einstufung von Szenarien ermöglicht es, die gleichermaßen *wichtigen* und *schwierigen* Szenarien zuerst zu bewerten.

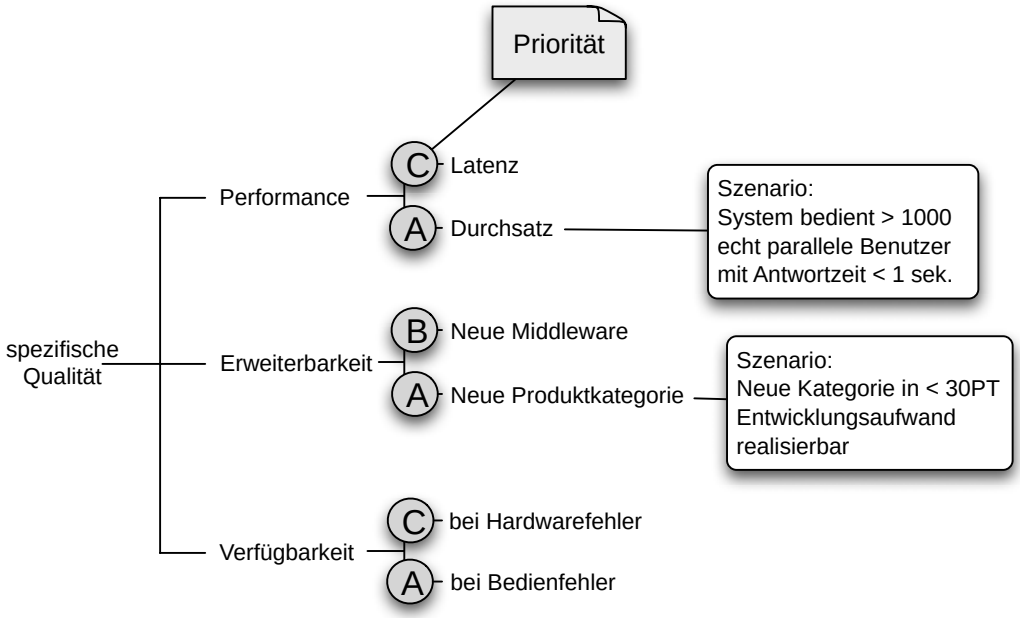

Abbildung 10.5 Qualitätsbaum mit Szenarien und Prioritäten

Konzentrieren Sie sich bei der Bewertung auf die Szenarien, die in beiden Dimensionen der Priorisierung ein „A“ erhalten haben, die also sowohl geschäftlich wichtig als auch technisch sehr schwierig sind. Wenn die Architektur schon diese Szenarien nicht unterstützt, können Sie sich die Bewertung der restlichen ersparen.

Architektur hinsichtlich der Qualitätsmerkmale bewerten

Mit den priorisierten Szenarien für die aktuellen Qualitäts- und Architekturziele besitzen Sie nun einen Bewertungsmaßstab für Ihre konkrete Architektur. Jetzt folgt die eigentliche Bewertung, die Sie in der Regel in einer kleinen Gruppe gemeinsam mit den Architekten des Systems durchführen. Die maßgeblichen Stakeholder sind dabei im Normalfall nicht anwesend.

Gehen Sie bei dieser Kernaufgabe gemäß den Prioritäten der Szenarien vor, beginnend mit den wichtigen und schwierigen! Lassen Sie sich in Form eines *Walkthrough* von den Architekten erläutern, wie die Bausteine des Systems zur Erreichung dieses Szenarios zusammenspielen oder welche Entwurfsentscheidungen das jeweilige Szenario unterstützen.

Leider muss ich Ihnen an dieser Stelle eine schlechte Nachricht überbringen: Es gibt keinen Algorithmus, der Ihnen die Zielerreichung einer Architektur hinsichtlich der Szenarien bestimmt. Ihre Erfahrung (oder die des Bewertungsteams) ist gefragt – auch Ihre subjektive Einschätzung.

Als Bewerter oder Auditor einer Architektur untersuchen Sie die zentralen Architektur- und Qualitätsziele auf Basis der jeweils definierten Szenarien gemeinsam mit dem Architekten des Systems. Lassen Sie sich die zugehörigen Architekturentscheidungen und -ansätze erläutern, und beantworten Sie die folgenden Fragen:

- Welche Architekturentscheidungen wurden zur Erreichung eines Szenarios getroffen?
- Welcher Architekturansatz unterstützt die Erreichung des Szenarios?
- Welche Kompromisse wurden durch diese Entscheidung eingegangen?
- Welche anderen Qualitätsmerkmale oder Architekturziele werden von dieser Entscheidung noch beeinflusst?
- Welche Risiken erwachsen aus dieser Entscheidung oder aus diesem Ansatz?
- Welche Risiken gibt es für die Erreichung des jeweiligen Szenarios und des damit verbundenen Qualitätszieles?
- Welche Analysen, Untersuchungen oder Prototypen stützen diese Entscheidung?

Als Resultat erhalten Sie einen Überblick über die Güte der Architektur hinsichtlich konkreter Szenarien, und damit auch hinsichtlich der spezifischen Architektur- und Qualitätsziele.

Meiner Erfahrung nach ist Auftraggebern (und maßgeblichen Stakeholdern) schon viel damit geholfen, für die wichtigsten Szenarien zu erfahren,

- welche Risiken bezüglich ihrer Erreichung oder Umsetzung existieren,
- welche Maßnahmen es zur Umgehung oder Milderung dieser Risiken gibt und
- welche Szenarien auf jeden Fall (d.h. risikolos) erreicht werden ([Clements +03] spricht hier von *Non-Risks*).

Auswirkungen von Architekturbewertungen

Neben den qualitativen Aussagen hinsichtlich Zielerreichung und Risiken erwachsen aus der Architekturbewertung in der Regel eine Reihe weiterer Vorteile:

- Die maßgeblichen Stakeholder diskutieren über ihre eigenen Ziele[10] und deren Prioritäten. Das erleichtert in Entwicklungsprojekten die Vergabe von Prioritäten erheblich.
- Zentrale Architekturentscheidungen werden explizit dargestellt und offen diskutiert.
- Architekturansätze werden dokumentiert, weil Bewertungen häufig auf Basis der Architekturdokumentation stattfinden.

Schließlich kommt es vor, dass maßgebliche Stakeholder sich des hohen Risikos bewusst werden, das durch einige ihrer Anforderungen und Qualitätsziele entsteht.

10.3 Weiterführende Literatur

[Bass+03] und [Clements+03] beschreiben die ATAM-Methode (*Architecture Tradeoff Analysis Method*) ausführlich. Meiner Erfahrung nach ist diese Methode jedoch durch die wiederholte Bewertung der Architekturansätze in vielen Fällen zu aufwändig in der Anwendung. Trotzdem legt sie die Grundlage für den Ansatz der szenariobasierten Bewertung.

[Kruchten+02] fassen den State of the Art bezüglich der Architekturbewertung zusammen. Sie geben viele Hinweise, wie eine Bewertung praktisch ablaufen kann.

[10] Diese Diskussion der Stakeholder hätte bereits in der Analysephase stattfinden sollen. Leider besteht für die wesentlichen Stakeholder in der Realität hinsichtlich ihrer Architektur- und Qualitätsziele allzu oft weder Klarheit noch Einigkeit.

[Ionita+02] stellen unterschiedliche szenariobasierte Bewertungsmethoden vor. Sie kommen dabei zu interessanten Schlussfolgerungen; beispielsweise kann keine Methode sicherstellen, *alle* vorhandenen Risiken aufzudecken.

[Bosch2000] beschreibt außer der szenariobasierten Bewertung einen simulationsbasierten Ansatz. Ich habe hier erhebliche Einwände bei Kriterien wie Flexibilität, Sicherheit und Wartbarkeit, die sich meiner Meinung nach nicht durch Simulation bewerten lassen.

[Henderson-Sellers96] ist eine ausführliche Darstellung objektorientierter Metriken.

[Lorenz94] ist eine praktische Einführung in Metriken für objektorientierte Systeme.