Service-Qualität managen

Klaus-Peter Eckert, Tom Ritter, Ina Schieferdecker

Die Auswahl geeigneter Services ist bei einem breiten Service-Angebot wesentlich – Service-Qualität und Service-Güte sind wichtige Kriterien zur Unterscheidung von Services gleicher Funktionalität.

Die Einbeziehung von Service-Qualität in eine SOA ist im Einzelfall zu entscheiden – bei vielen Anbietern vergleichbarer Services erlauben sie Alleinstellung.

Aber was ist eigentlich Service-Qualität?

Einführung

Während die meisten Ansätze zu service-orientierten Architekturen Services ausschließlich mittels ihrer Funktionalität charakterisieren, betrachtet der vorliegende Beitrag ebenso die Einbeziehung von Service-Qualität, die mittels *nicht-funktionaler Eigenschaften* beschrieben wird. So wird beispielsweise von einem Anbieter von Internet-Services nicht ein beliebiger DSL-Service (Anschluss) gewünscht, sondern ein DSL-Anschluss mit bestimmter Bandbreite, vorgegebenen Kosten und hoher Verfügbarkeit. Die funktionalen Service-Beschreibungen sind dazu um Beschreibungen nichtfunktionaler Aspekte, der sogenannten *Service-Güte* oder *Quality of Service* (QoS), zu ergänzen. Diese Ergänzungen sind sowohl bei der Ausformulierung des Architekturmodells zu beachten als auch von der Infrastruktur zur Laufzeit zu unterstützen.

Betrachten wir das Beispiel des DSL-Services genauer. Viele Anbieter kombinieren einen DSL-Anschluss mit der Bereitstellung einer e-Mail Kennung, Internet-Telefonie, Festnetz- und Mobil-Telefonie oder dem Zugriff auf multimediale Inhalte. Dazu werden dem Kunden konfigurierbare Service-Pakete angeboten. Dieses Konzept der Paketierung oder *Komposition* von Services ist ein weiteres, wesentliches Merkmal service-orientierter Architekturen. Im Zusammenhang mit der Einbeziehung von Service-Güte sind Fragen bezüglich der Definition und Berücksichtigung der Güte von komponierten Services sowohl zur Entwurfs- als auch zur Laufzeit zu beantworten.

Der vorliegende Beitrag erörtert Service-Güte aus verschiedenen Perspektiven. Es werden notwendige Erweiterungen zur Einbeziehung der Qualität von Services und die daraus resultierenden Eigenschaften SOA-konformer Systeme diskutiert. Ausführliche Darstellungen vorhandener Ausprägungen der Qualität von Services sowie Möglichkeiten zur Sicherstellung dieser Qualitäten runden den Beitrag ab.

Service-Güte in service-orientierten Architekturen

Service-orientierten Architekturen liegt das folgende Architekturmuster zugrunde. Ein *Service-Anbieter* beschreibt und registriert seine angebotenen Services bei einem *Service-Vermittler*. Ein potentieller *Service-Nutzer* sucht bei einem oder über einen Service-Vermittler nach gewünschten Services. Sofern ein geeigneter Service gefunden ist, wird eine Bindung zwischen Service-Anbieter als Server und Service-nutzer als Klient hergestellt und die eigentliche Service-Nutzung kann beginnen. Um dieses elementare Muster zu implementierten, müssen initial Service-Angebote und Service-Nachfragen beschrieben werden, der Grad der Übereinstimmung von Nachfrage und Angeboten ist zu ermitteln und das beste Angebot ist auszuwählen. Dabei ist nicht vorgegeben, ob die Auswahl durch den Service-Vermittler oder direkt durch den Service-nutzer erfolgt. Im Anschluss an die Auswahl

eines geeigneten Service-Angebots ist formal ein *Vertrag* zwischen Anbieter und Nutzer über die Konditionen der Nutzung des Services abzuschließen und eine technische Verbindung zum Service herzustellen, die als Grundlage für die eigentliche Nutzung des Services dient.

Aus Sicht service-orientierter Architekturen kann dieses Muster in verschiedene Richtungen erweitert werden. So können beispielsweise mehrere Service-Anbieter ihre Services bei einem Service-Vermittler anbieten, was den Auswahlprozess geeigneter Services interessanter macht und die Notwendigkeit der Einführung geeigneter Auswahlkriterien nahelegt. Die Berücksichtigung mehrerer Service-Vermittler erlaubt die natürliche Realisierung von Szenarien mit Wiederverkäufern (Retailer), und die Erweiterung von Auswahlprozessen auf kooperierende oder auch konkurrierende Vermittler.

Eine weitere Verallgemeinerung des grundlegenden Musters besteht im Vertauschen der Rollen von Service-Anbieter und Service-nutzer, wie man es von jedem „Schwarzen Brett" kennt. Um Angebot und Nachfrage aufeinander abzustimmen ist es vorstellbar, dass potentielle Service-nutzer ihre Service-Gesuche beim Vermittler registrieren und Service-Anbieter beim Vermittler nach vorhandenen Anfragen suchen. Auf den Zetteln am Schwarzen Brett steht nicht mehr „Biete:" sondern „Suche:". Auch Mischformen beider Ansätze sind vorstellbar.

All diese Variationen ändern aber nichts an der Notwendigkeit, Service-Angebote und Service-Nachfragen zu spezifizieren und Regeln sowie Metriken für Messung und Bewertung der Übereinstimmung zwischen Angebot und Nachfrage anzugeben.

Begriff Service-Güte

Der Begriff Service-Güte bzw. QoS wird in unterschiedlichen Bereichen des IT-Sektors in teilweise sehr unterschiedlichen Bedeutungen benutzt. Häufig trifft man auf diesen Begriff in der Netzwerktechnik. Dort wird er beispielsweise im Zusammenhang mit bestimmten Transportprotokollen wie den bekannten Internet-basierten Protokollen TCP, UDP oder RTP verwendet. Aber auch in der Telekommunikationsdomäne spricht man häufig über Service-Güte. Es gibt eine Reihe von weiteren Begriffen, die für die Bezeichnung der Service-Güte verwendet werden. So taucht häufig der Terminus nicht-funktionale Eigenschaften (*non-functional properties*) auf oder man spricht von sogenannten qualitätsorientierten Aspekten eines Systems. Spricht man mit IT-Experten über Service-Güte so fällt auf, dass es allgemein ein ausgeprägtes Bewusstsein für Service-Güte gibt. Sie wird oft als der entscheidende Punkt gesehen, um sich mit bestimmten Produkten am Markt erfolgreich platzieren zu können. Spricht man jedoch über Service-Gütegarantien und über technische Details, so bleibt vieles vage und sehr unbestimmt. Das konkrete Wissen über Service-Güte differiert stark zwischen den einzelnen Domänen und Tätigkeitsfeldern. Jedoch hat nahezu jeder IT-Experte und IT-Manager fast täglich Fragen der Service-Güte zu lösen. Beispielhaft sei hier eine der sicherlich am häufigsten auftretenden Frage der Skalierbarkeit genannt, bei der es darum geht, inwiefern ein System seine Funktionalitäten bei steigender Last aufrechterhalten kann und wann Engpässe auftreten.

Der Begriff Service-Güte ist vor allem deshalb oft sehr widersprüchlich, weil er sich manchmal nur schwer zuordnen lässt. Denn nicht immer ist klar, ob eine Wahrnehmung auf dem Service selber beruht und zu seinen Eigenschaften zählt oder ob es sich dabei um Güte-Eigenschaften des Services handelt. Aus diesem Grund wird im Folgenden der Begriff Service-Güte in seinen unterschiedlichen Facetten am Beispiel diskutiert, bevor die Bedeutung der Service-Güte für Mehrwert- bzw. Basis-Services und Service-Anbieter, Service-Vermittler und Service-Benutzer im Rahmen offener Service-Architekturen analysiert wird.

Nehmen wir ein ganz einfaches Beispiel, um zu verdeutlichen, was Service-Güte bedeuten kann. Nehmen wir die folgenden Aussage: „Eine EUR-Währungsumrechnung dauert eine Sekunde."

Es ist relativ schnell klar, dass der Service im Ermitteln eines Geldäquivalents in einer anderen Währung liegt und im Bereitstellen des Ergebnisses besteht. Oftmals spricht man an dieser Stelle auch von der Funktionalität des Services. Die Güte des Services besteht an dieser Stelle offenbar darin, das Ergebnis in einer Sekunde zu liefern, nicht in einer Stunde aber auch nicht in einer Millisekunde, sondern genau in einer Sekunde. Doch was wäre, wenn der Service das Ergebnis schon nach einer halben Sekunde liefern würde, und was wäre, wenn das Ergebnis erst in zwei Sekunden geliefert werden würde. Beides sind Abweichungen von der beschriebenen Service-Güte und dennoch scheinen sie in den meisten Fällen unterschiedlich bewertet zu werden. Abgesehen von bestimmten Realzeitsystemen ist es in aller Regel als positiv zu sehen, wenn ein Service schneller arbeitet als erwartet oder gar als zugesichert wurde. Dauert es hingegen länger, ist das meist negativ zu sehen. Das bedeutet, es gibt für eine konkrete Service-Güte auch immer eine Bewertungsfunktion oder eine so genannte Metrik. Diese gewichtet die Service-Güteeigenschaft nach ihrem Wert für den Service-nutzer. Im Falle der Berechnungszeit ist die Metrik bis auf die bereits genannte Ausnahme der Realzeitanwendungen trivial: eine kürzere Bearbeitungszeit bedeutet einen höheren Wert für den Nutzer.

Bis hierhin ist es relativ leicht den Service, also die Funktionalität, und dessen Eigenschaften, also die Service-Güte, klar zu identifizieren. An dem Beispiel der Währungsumrechnung lassen sich jedoch noch weitere interessante Dinge über Service-Güte zeigen. Oftmals wird Service-Güte nur implizit unterstellt; eine klare Abgrenzung von Funktionalität und Service-Güteeigenschaft ist nicht immer einfach. Nehmen wir einmal an, das gelieferte Ergebnis wäre auf 3 Stellen nach dem Komma genau. Wären wir mit diesem Ergebnis zufrieden? Bereits die Verordnung (EG) Nr. 1103/97 des Rates vom 17. Juni 1997 über bestimmte Vorschriften im Zusammenhang mit der Einführung des Euro, definiert eine Umrechnungsgenauigkeit von sechs signifikanten Stellen. War es also für eine Umrechnung von griechischen Drachmen akzeptabel, galt dies nicht für die deutsche Mark und andere Währungen. Hier darf/muss das Ergebnis also schon genauer sein. Bisher wurde das jedoch nicht formuliert. Und nun treffen wir auf das eigentliche Problem bei der Definition von Service-Güte. Ist eigentlich die Korrektheit bzw. Genauigkeit des Ergebnisses funktionaler Bestandteil des Services oder aber ist es eine Service-Güte, mehr oder weniger genaue Ergebnisse zu liefern. Ist es Teil des Services so würde man sagen, der Service liefert immer ein Ergebnis auf beispielsweise sechs signifikante Stellen genau. Würde man es zur Service-Güte zählen, so würde man sagen, der Service liefert ein Ergebnis, dass mindestens auf sechs signifikante Stellen genau ist. In diesem Fall kann man sich vorstellen, dass der Service der Währungsumrechnung von einer Reihe unterschiedlicher Service-Anbieter bereitgestellt wird, die bezüglich der Genauigkeit unterschiedliche Service-Güte anbieten. Einige sind vielleicht weniger genau, indem sie vielleicht nur 4 signifikante Stellen anbieten, während andere genauer sind.

Und nun wird es spannend: man muss sich nun die Frage stellen, warum wird nicht einfach immer der Service mit den besten Service-Güteeigenschaften benutzt. Die Antwort auf diese Frage ist der entscheidende Punkt, warum Betrachtungen zur Service-Güte in offenen Architekturen so wichtig sind. Service-Güteeigenschaften stehen in einem engen Zusammenhang zu verfügbaren Ressourcen. Eine höhere Service-Güte erfordert im Allgemeinen einen höheren Ressourceneinsatz. Eine schnellere Berechnung der Währungsumrechnung erfordert beispielsweise bessere Prozessoren mit höheren Taktraten. Zudem wird für eine höhere Genauigkeit zusätzlicher physikalischer Speicher benötigt. Aus diesem Grund muss ein Abwägungsprozess stattfinden, der die erforderliche Service-Güte im Verhältnis zu den zur Verfügung stehenden Ressourcen und den damit verbundenen Kosten

betrachtet. Ressourcen stehen nicht unbegrenzt zur Verfügung. Je mehr Ressourcen eingesetzt werden müssen, desto mehr Kosten entstehen. Daraus leitet sich ab, dass Services mit einer hohen Service-Güte im Allg. auch höhere Kosten verursachen. Dieser Zusammenhang ist offensichtlich und dennoch von enormer Bedeutung. Er zeigt, dass es wichtig ist, die zur Verfügung stehenden Ressourcen genau einschätzen zu können.

Neben der direkten Abhängigkeit der Service-Güte von Ressourcen muss man sich noch einer weiteren Abhängigkeit bewusst werden. Nimmt man die zur Verfügung stehenden Ressourcen als eine feste Größe an, so ergibt sich eine Abhängigkeit der Service-Güteeigenschaften untereinander. Wenn beispielsweise ein aktueller Umrechnungskurs für frei konvertierbare Währungen genutzt wird, so muss dieser abgefragt werden, was zusätzlichen Aufwand und eine Verzögerung der Antwortzeiten ergibt. Gleiches gilt natürlich auch in umgekehrter Richtung: ein fest eingestellter Umrechungskurs – beispielsweise von der letzten Woche – führt zu einem ungenaueren Ergebnis, doch kann das Ergebnis schneller geliefert werden. Darüber hinaus gibt es noch weitere Abhängigkeiten, die insbesondere in offenen Service-Architekturen eine entscheidende Rolle spielen: so können Services mehrfach aufgerufen werden und parallel stattfindende Service-Nutzungen um die zur Verfügung stehenden Ressourcen konkurrieren. Geht es bei der Service-Güte eines Services um eine optionale Nutzung der zur Verfügung stehenden Ressource, geht es bei der Service-Güte mehrerer Services um eine ergebnisoptimale oder kostenoptimale oder faire Ressourcenzuteilung.

Es gibt jedoch eine weitere Abhängigkeit, derer man sich bewusst sein sollte. Service-Güte steht immer in einer Beziehung zu Funktionalität. Service-Güte ergibt sich entweder direkt aus der Service-Implementierung oder sie wird durch zusätzliche Funktionalität erbracht. Um beispielsweise die Zuverlässigkeit von Services zu erhöhen, wird häufig die Strategie der Replikation benutzt. In diesem Fall werden einfach mehrere gleichartige Service-Instanzen gestartet, so dass der Ausfall einer Service-Instanz durch das Umlenken auf eine andere Service-Instanz ausgeglichen werden kann. Die Logik und Technologien, die für die Replikation benötigt werden, sind für sich genommen aber auch nur wieder ein Service, eine bestimmte Funktionalität, mit der sich letztlich die Service-Güte Ausfallsicherheit verbessern lässt. Beschäftigt man sich mit Service-Güte und mit der technischen Realisierung im Detail, so ist es oftmals nicht einfach, die Grenze zwischen Funktionalität und Service-Güte klar zu ziehen.

Je nach Blickwinkel ergeben sich für Service-Güte noch weitere Bezüge. So kann Service-Güte nur zu bestimmten Zeitpunkten relevant sein, oder nur für eine einzelne Service-nutzung. Service-Güte kann sich auf einen einzelnen Service oder sich auf eine Menge von Services beziehen.

Obwohl das hier benutze Beispiel der Währungsumrechnung vereinfacht ist, finden sich die genannten Zusammenhänge und damit verbundenen Probleme in der praktischen Anwendung service-orientierter Architekturen. Oftmals wird nicht klar definiert, was der Service und was die Service-Güte ist, die zur Verfügung stehenden Ressourcen können nicht eindeutig beurteilt werden oder die Wechselwirkungen der Service-Güteeigenschaften untereinander und zwischen den verschiedenen Services sind nur unzureichend bekannt und berücksichtig worden. Eine ganzheitliche Betrachtung, die die Service-Funktionalität und die Service-Güte einschließen, ist jedoch für einen reibungslosen und effizienten Einsatz von service-orientieren Architekturen in geschäftskritischen Abläufen entscheidend. Mit anderen Worten: man muss dem Service-nutzer, also dem Kunden, deutlich sagen, welchen Service er mit welcher Service-Güte erwarten kann.

Service-Beschreibungen heute

Wie sehen heutzutage Service-Beschreibungen aus? Das einführend beschriebene Szenario erinnert sicherlich viele an die Abläufe beim Ersteigern oder Erwerben beliebiger Produkte, hier als Synonym für Service verwendet, bei eBay oder dem Einkaufen von Büchern und anderen Waren bei Amazon. In beiden Fällen werden die angebotenen Produkte kategorisiert und mehr oder weniger detailliert beschrieben. Die grafischen Oberflächen des Web-Auftritts beider Anbieter ermöglichen dem potentiellen Käufer ein Stöbern aber auch ein gezieltes Suchen nach gewünschten Produkten. Hinter den Kulissen werden die Suchaktionen und Angebote unter Verwendung von Web-Services und zugehörigen XML-Datentypen (die „eXtended Markup Language“ – eine Datenstrukturbeschreibungssprache von W3C, dem World Wide Web Consortium) beschrieben und durchgeführt, wie man den WSDL-Spezifikationen (die „Web Services Definition Language“ – eine XML-basierte Beschreibungssprache von W3C für Web Services) von Amazon [3] und eBay [4] entnehmen kann. Eine Anfrage nach einem Buch bei Amazon wird beispielsweise auf eine detaillierte Beschreibung, bestehend aus mehr als 50 optionalen Attributen abgebildet („Details“ genannt). Darunter befinden sich spezifische Attribute wie Autor, Titel, Kategorie aber auch allgemeine Attribute wie Preis, Erscheinungsjahr, Seitenanzahl, Alterseinschränkungen, Verkaufsrang, Verfügbarkeit oder Bewertungen.

Zurück in die Welt service-orientierter Systeme: die Beschreibung von Services mittels erforderlicher Attribute, die die Funktionalität eines Services beschreiben, und mittels optionaler Attribute, die nicht-funktionale Eigenschaften beschreiben, ist ein naheliegender Ansatz für die Berücksichtigung von Service-Qualitäten bei der Beschreibung und Auswahl von Services. Dieser Weg wird ansatzweise in Verzeichnissen wie dem UDDI [5] (dem „Universal Description, Discovery and Integration“ – einer XML-basierten Registrator von OASIS für Services etc.) beschritten. Hier findet man Informationen über Service-Anbieter, Service-Kategorien, sowie über einzelne Services und deren Eigenschaften. Für die Festlegung von Service-Kategorien und Services können dabei Standards wie der vom „United Nations Development Programme“ (UNDP) definierte „United Nations Standard Products and Services Code“ (UNSPSC®) [6] verwendet werden. Die Verwendung zumindest domänenspezifischer Standards ist erforderlich, um ein einheitliches Verständnis zwischen Anbietern und Nutzern von Services zu ermöglichen. Im Falle von Amazon wird dieses Verständnis beispielsweise durch ein allgemeingültiges Verständnis der fokussierten Angebotspalette und die Festlegung der zur Beschreibung eines Produkts möglichen Attribute erreicht. Definitionen für allgemeingültige Systeme und Applikationen gibt es beispielsweise von der OMG, OASIS und anderen.

Für die Beschreibung angebotener und gewünschter Service-Qualitäten sind ebenfalls Standards zu entwickeln bzw. einzusetzen, die verbindliche, auf den Anwendungsbereich zugeschnittene Normen zu Angabe, Bewertung und Vergleich nichtfunktionaler Eigenschaften definieren. Erste Ansätze existieren mit Web Service Level Agreements (WSLA) von IBM [8], der Web Services Offerings Language (WSOL) der Carleton University [9] und der Web Services Management Language (WSML) von HP [10].

Service-Auswahl heute

Die Auswahl geeigneter Service-Angebote beruht auf einem Vergleich geforderter und angebotener Service-Attribute. Für jedes Attribut ist daher zu definieren, wann ein Angebot einer Anforderung genügt. Obwohl diese Forderung trivial klingt, ist sie im Einzelfall nicht immer einfach zu erfüllen.

Kommen wir zurück auf das Beispiel der UDDI. In diesem Verzeichnis findet man insbesondere technische Beschreibungen der angebotenen Services. Zu diesen Beschreibungen gehören unter anderem exakte Definitionen der Schnittstellen der

angebotenen Services sowie die bei der Nutzung der Services unterstützen Kommunikationsprotokolle. Ein Service ist von einem Klienten nur dann nutzbar, wenn er eine verträgliche Schnittstelle anbietet und mittels eines dem Klienten bekannten Kommunikationsprotokolls angesprochen werden kann. Um die Zulässigkeit eines Service-Angebots beurteilen zu können, ist es also erforderlich, Aussagen über die Verträglichkeit von Schnittstellen bzw. über das Vorhandensein von Protokollen zu machen. Es ist beispielsweise möglich, dass ein Server mehr Anforderungen bearbeiten kann, als der Klient benötigt, oder der Server weniger Ergebnisse liefert, als der Klient bearbeiten kann. Es ist nur zu vermeiden, dass ein Server durch Anfragen bzw. ein Klient durch Ergebnisse „überfordert“ wird. Vergleichbares gilt für die vom Service angebotenen Operationen, deren Parameter und logischen Vor- bzw. Nachbedingungen bei der Service-Nutzung. Pragmatisch gesehen ist sicherzustellen, dass das vorhandene Laufzeitsystem, die SOA Infrastruktur, eine Kommunikation zwischen Service-nutzer und Service-Anbieter unterstützt und Formate von Ein- und Ausgabeparameter intelligent anpasst und konvertiert.

Für die Einbeziehung von Service-Qualitäten bei der Service-Auswahl gelten die gleichen Kriterien. Es sind Datenstrukturen und zugehörige Metriken zur Beschreibung und Bewertung der Qualitätsmerkmale und zugehörige Verträglichkeitskriterien zu entwickeln. Die Infrastruktur muss in der Lage sein, diese nicht-funktionalen Eigenschaften von Services zu unterstützen.

Service-Auswahl morgen

Bislang sind wir in der Lage, Services und ihre Funktionalitäten und Eigenschaften zu beschreiben und auszuwählen. Aber haben wir damit nicht nur die Katze im Sack gekauft? Was machen wir, wenn sich das bei eBay ersteigerte Schmusekätzchen – Service = Katze, Eigenschaft = kuschelig - nach einiger Zeit als reißender Tiger entpuppt? Oder mit anderen Worten, was machen wir, wenn ein Service während der Nutzung nicht die versprochene Verfügbarkeit, den angepriesenen Durchsatz oder die erforderliche Datensicherheit bietet? Die Einbeziehung von Service-Qualitäten in die Beschreibung und Auswahl von Services macht nur dann wirklich Sinn, wenn am Ende der Service-Auswahl ein verbindlicher Vertrag zwischen Anbieter und Nutzer steht, dessen Einhaltung von der Infrastruktur zumindest überwacht werden kann. Das eingangs beschriebene Architekturmuster ist um einen *Service-überwacher* zu ergänzen, der sowohl als Notar bei der Erstellung von Service-Nutzungsverträgen als auch als Wächter (Supervisor) während der Nutzung von Services agiert. Service-Plattformen von IBM (WebSphere [11]), von Sun (SOA Governance Solution [12]) oder von PrismTech (OpenSplice [13]) bieten dafür eine erste Unterstützung.

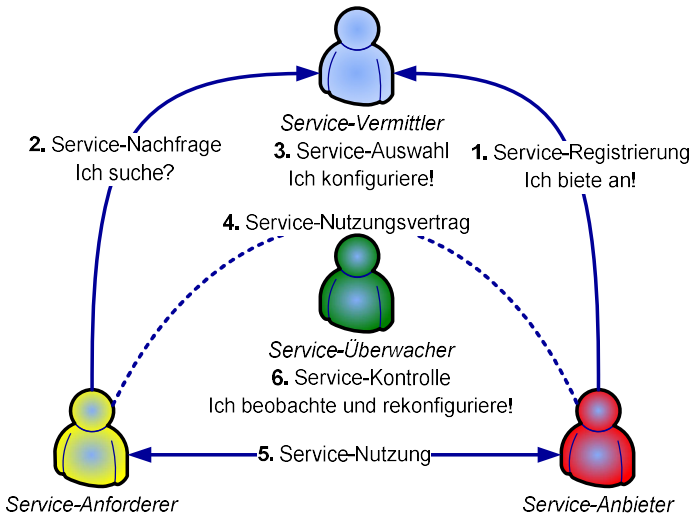

Abb. 1: Erweiterte service-orientierte Architektur

Stellt sich nun bei der Überwachung eines Services eine Verletzung des Nutzungsvertrags heraus, sind sowohl eine formale als auch eine technische Reaktion vorstellbar. Formal betrachtet sind die Vertragsverletzung festzustellen, zu dokumentieren und die vertraglich festgelegten Reaktionen wie eine mögliche Reduktion der Kosten zu veranlassen. Technisch betrachtet möchte man in vielen Fällen den eingeschränkten bzw. fehlerhaften Service durch einen besseren, korrekt arbeitenden Service ersetzen – das System ist wo möglich dynamisch zu rekonfigurieren. Die initial durchgeführte Auswahl und Bindung zwischen Service und Nutzer sind aufzulösen, ein neuer Service ist auszuwählen und die Nutzung ist möglichst verlustfrei fortzuführen. Ein derartiges Herangehen stellt hohe Anforderungen an die vorhandene Infrastruktur, die ein weitgehend autonomes Verhalten zulassen muss, um Fehler selbständig zu entdecken und beheben zu können. Eine derartige Verbindung von service-orientierten und autonomen Systemen steht heute noch auf der Wunschliste und Forschungsagenda der IT-Spezialisten.

Komposition von Services

Das bislang diskutierte Muster für service-orientierte Architekturen beschreibt nur einen der Grundbausteine, aus denen SOA-konforme Systeme aufgebaut werden. Ziel der Einführung einer SOA ist die Unterstützung bei der Identifikation und IT-gestützten Realisierung von Prozessen (also von Vorgängen und Abläufen) innerhalb und zwischen Unternehmen. Jeder Prozess setzt sich aus einzelnen Schritten (den Aktivitäten oder Aufgaben) zusammen, die ihrerseits mittels vorhandener Services durchgeführt werden. Die Beschreibung eines Prozesses besteht im Wesentlichen aus der Definition des Kontroll- und Informationsflusses zwischen den einzelnen Schritten. Es ist festzulegen, unter welchen Bedingungen ein Schritt durchzuführen ist, welche Information für seine Durchführung erforderlich ist und welche Ergebnisse er liefert. Je nach Vorhandensein einer kontrollierenden und steuernden Instanz spricht man von Service-Orchestrierung bzw. von Service-Choreographie.

Die Grundidee hinter diesem Vorgehen kann am eingangs verwendeten Beispiel eines Anbieters von Internet-Services verdeutlicht werden. Möchte ein Kunde das Gesamtpaket IP@Home erwerben und nutzen, müssen in bestimmter Reihenfolge Teil-Services und Anschlüsse bereitgestellt und freigeschaltet werden. Während der Nutzung ist beispielsweise sicherzustellen, dass die Qualität von Telefongesprächen nicht unter dem gleichzeitigen Konsumieren von Fernsehprogrammen leidet und Gespräche vom Mobiltelefon zum Festnetztelefon kostenfrei erfolgen. Für den Nutzer des Service-Pakets ist es dabei nicht relevant, ob der Anbieter des Pakets alle Teil-Services selber erbringt, an genau einen Anbieter weitervermittelt oder für jeden Teil-Service einen besonders geeigneten Anbieter auswählt. Die Logik des komponierten, gebündelten IP@Home Services ist daher unabhängig von den Anbietern der einzelnen Services. Aus diesem Grund wird die zur Komposition verwendete Service-Logik unabhängig von den auszuwählenden Anbietern von Teil-Servicen definiert. Zur Auswahl der einzelnen Anbieter wird in der Bereitstellungsphase des Service-Pakets (dem sogenannten Deployment) das SOA spezifische Architekturmuster verwendet.

Dieses Beispiel geht implizit von der Annahme aus, dass das Ergebnis der Komposition von Services seinerseits einen höher granularen Mehrwert-Service für potentielle Kunden definiert. Dabei stellt sich sofort die Frage nach dem Zusammenhang der Qualität des komponierten Mehrwert-Services und den Qualitäten der zur Komposition verwendeten Basis-Services. Man spricht hier auch von *Service Level Agreements* (SLA) zur Beschreibung der Nutzungsverträge des komponierten Services mit seinen Kunden und von Operation Level Agreements (OLA) zur Beschreibung der Nutzungsverträge des komponierten Services mit

den Basis-Services. Eine automatisierte Ableitung benötigter Service-Qualitäten von Basis-Services aus den für den Mehrwert-Service geforderten Qualitäten ist im Allgemeinen nicht möglich. Bei genauer Kenntnis des Nutzungsprofils des Mehrwert-Services ist es unter Verwendung von Simulationstechniken möglich, möglichst optimale Qualitätsanforderungen an die Basis-Services abzuleiten und zur Laufzeit durch eine intelligente, lernende Infrastruktur dynamisch anzupassen.

OASIS stellt hierfür mit dem Web Services Distributed Management (WSDM [7]) Standard, Kernkomponenten zur Verfügung. Es geht einerseits um das Management unter Nutzung von Web Services (Management Using Web Services, MUWS) und andererseits um das Management von Web Services (Management of Web Services, MOWS). MOWS baut auf MUWS auf und definiert WSDL Schnittstellen, über die Ereignisse und Metriken beobachtet, analysiert und kontrolliert werden können.

Auch hierbei gibt es offene technische Fragestellungen, was die praktische Nutzung in konkreten Situationen jedoch nicht ausschließt. Eine wesentliche Einschränkung bei der Einführung service-orientierter Systeme unter Berücksichtigung von Qualitätsmerkmalen stellt jedoch die Tatsache dar, dass sich nicht alle Prozesse in und zwischen Unternehmen vollständig automatisieren lassen. In vielen Fällen werden weiterhin einzelne Schritte von menschlichen Bearbeitern ausgeführt. Der Prozess übernimmt dann „nur“ die Steuerung und Kontrolle der einzelnen Schritte. Im Falle derartiger *human tasks* ist die Gewährleistung von Qualitäten des Gesamtprozesses kaum realisierbar, da es nur schwer nachvollziehbare Verfahren zur Messung und Garantie von Leistungsmerkmalen menschlicher Bearbeiter gibt.

Prozesse zwischen Unternehmen

Lassen Sie uns nun die Infrastruktur für service-orientierte Systeme betrachten. Hierbei steht die mit „Enterprise SOA“ bezeichnete Ausführung komplexer Prozesse zwischen eigenständigen aber kooperationsbereiten Unternehmen im Vordergrund. Vom technischen Gesichtspunkt gesehen, erscheint es zunächst egal, ob Mehrwert- und Basis-Services von unterschiedlichen Unternehmen oder ob diese von einem Unternehmen erbracht werden. Neben der technischen Kompatibilität der Beschreibungen von Services und ihrer Service-Qualitäten sind beim Zusammenspiel von Services zwischen Unternehmen auch spezifische Regeln zur Gewährleitung der gegenseitigen Sicherheitsanforderungen zu berücksichtigen. So können beispielsweise spezielle Verfahren zur Authentisierung und Autorisierung erforderlich sein; Verschlüsselungstechniken und Protokollierungen (so genannten Logs) können vorgeschrieben werden. Dieser zusätzliche Aufwand ist bei der Auswertung der Anforderungen an Services und ihre Qualitäten zu berücksichtigen, selbst wenn er nicht explizit in der Beschreibung der Services auftritt und nur durch entsprechende Eigenschaften der Infrastruktur zur Laufzeit auftritt.

Unter Umständen sind Anforderungen an Durchsatz und Antwortzeit an Mehrwert-Services in föderierten Umgebungen zwischen verschiedenen Unternehmen nicht erfüllbar, während dieselben Anforderungen innerhalb eines Unternehmens gewährleistet werden können. In diesem Zusammenhang wird deutlich, dass technische Umgebung und Infrastruktur, in der insbesondere Mehrwert-Services genutzt werden sollen, einen entscheidenden Einfluss auf deren Qualitätsmerkmale besitzen. Die Berücksichtigung von Service-Qualitäten bei der Service-Auswahl ist damit eng mit Anforderungen an bzw. mit Eigenschaften der genutzten Infrastruktur verbunden. Während beispielsweise mittels Web Services kommunizierende Partner nur sicherstellen müssen, dass auf beiden Seiten die gleiche WSDL-Kommunikationsanbindung in der genutzten Infrastruktur unterstützt wird, sind die Abhängigkeiten zwischen Services und Infrastruktur bei der Einbeziehung nicht-funktionaler Eigenschaften weitaus größer.

Qualitätssicherung in service-orientierten Architekturen

Nachdem wir nun Funktionalität und Güte von Services näher beleuchtet haben, betrachten wir abschließend wie diese analytisch nachgewiesen und gesichert werden können.

In SOA-basierten Systemen muss man sich von einigen alt hergebrachten Annahmen der Qualitätssicherung lösen: Die Systeme setzen sich typischerweise aus Services verschiedener Anbieter zusammen und werden daher nicht aus einer Hand gefertigt. Ein Service selber hat dabei nicht nur die eigentliche Service-Logik zu erfüllen, sondern muss ebenso korrekt in die SOA Infrastruktur eingebettet sein und die Kommunikationsanbindung korrekt umsetzen. Darüber hinaus können Systeme im Allgemeinen dynamisch die aktuell zu nutzenden Services einbinden – sie nutzen daher keine statischen Konfigurationen aus vorher bestimmten Services. Des Weiteren werden die Systeme oftmals in offenen Umgebungen eingesetzt, so dass der Nutzungskontext und die aktuelle Arbeitslast für ein System schwer vorhersagbar sind. Noch mehr tritt – wie in diesem Beitrag beschrieben – neben der Funktionalität und Korrektheit die Leistungsfähigkeit eines Systems in den Vordergrund, so dass eine Kombination von funktionalen und nicht-funktionalen Tests genutzt werden muss. Schlussendlich wird das System durch eine Reihe von Einstellungen wie Policies zur Zugangskontrolle permanent den aktuellen Anforderungen angepasst.

Auch wenn die Qualitätssicherung für Basis-Services einfacher ist als bei komplexer monolithischen Systemen, stellt doch die aus oben beschriebenen Eigenschaften resultierende Komplexität der Systeme auch die Qualitätssicherung vor neue Herausforderungen. Eine prominente Eigenschaft von SOA ist die der Wiederverwendbarkeit der Services über verschiedene Geschäftsprozesse hinweg. Dies reduziert die Redundanz in Funktionen und Daten, was in Bezug darauf einen geringeren Testaufwand mit sich bringt. Andererseits wird jeder Service damit geschäftskritischer, da eine Vielzahl von Geschäftsprozessen durch einen Service unterstützt wird. Damit erhöhen sich die Nutzungsszenarien für einen Service, auf die dieser hin getestet werden muss. Die hat insbesondere Implikationen auf Regressionstests, da bei Änderungen des Services, die Korrektheit der den Service nutzenden Geschäftsprozesse sichergestellt werden muss.

Der steigenden Zahl an Konfigurationen, Szenarien und Nutzungskontexten eines Services kann mittels einer angemessenen Testautomatisierung entgegengetreten werden. Mit TTCN-3 – der Testing and Test Control Notation – einer standardisierten Testspezifikations- und -implementierungstechnologie steht mit den SOAP- und WSDL-basierten Test Frameworks ein Ansatz zur Verfügung, der in idealer Weise den Anspruch reflektiert, integriert, automatisiert und über die verschiedenen Ebenen der Systemlogik bis zur Kommunikationsanbindung hinweg zu testen.

Es ändern sich aber nicht nur die Mittel der Qualitätssicherung sondern auch der Betrachtungsumfang bei derselben. Standen bislang einzelne Applikationen und Systeme im Zentrum der Betrachtung, geht es nun um Systemlandschaften und Systeme von Systemen. Hierbei ist beispielsweise sicherzustellen, dass die semantische Integrität über Systeme hinweg gewährleistet ist. Das ist einerseits eine Frage des Systementwurfs, aber gleichzeitig auch eine Frage der Qualitätssicherung, da geprüft werden muss, ob gleiche Daten von verschiedenen Services auch gleich interpretiert werden.

Die Systemlandschaft wird typischerweise über einen Enterprise-Service-Bus (ESB) realisiert, welcher damit selber zum Ziel der Qualitätssicherung wird. In Kombination mit den zu prüfenden Sicherheitsfragen ist der ESB so abzusichern, dass die Adaption, Integration und Interaktion zwischen den Services zuverlässig, gesichert und in einer skalierbaren Art und Weise erfolgen kann.

Schaut man sich den Aufbau einer SOA-Lösung an, so kann man neben dem ESB die folgenden Ebenen unterscheiden: die *Bestandskomponenten* zur Verwaltung von Datenbeständen und den Zugriff auf dieselben, die *Funktionskomponenten* für die Implementierung der fachlichen Funktionalität, die *Prozesskomponenten* für die implementierten Geschäftsprozesse und die *Interaktionskomponenten* für die Benutzerinteraktion mit der Systemlandschaft, siehe auch [15]. Dies erfordert eine Verfeinerung der bisherigen Testvorgehensmodelle. Beispielsweise kann sich das klassische V-Modell bei dem zwischen Komponenten-, Integrations-, System- und Abnahmetest unterschieden wird, in seiner Anwendung auf SOA so darstellen:

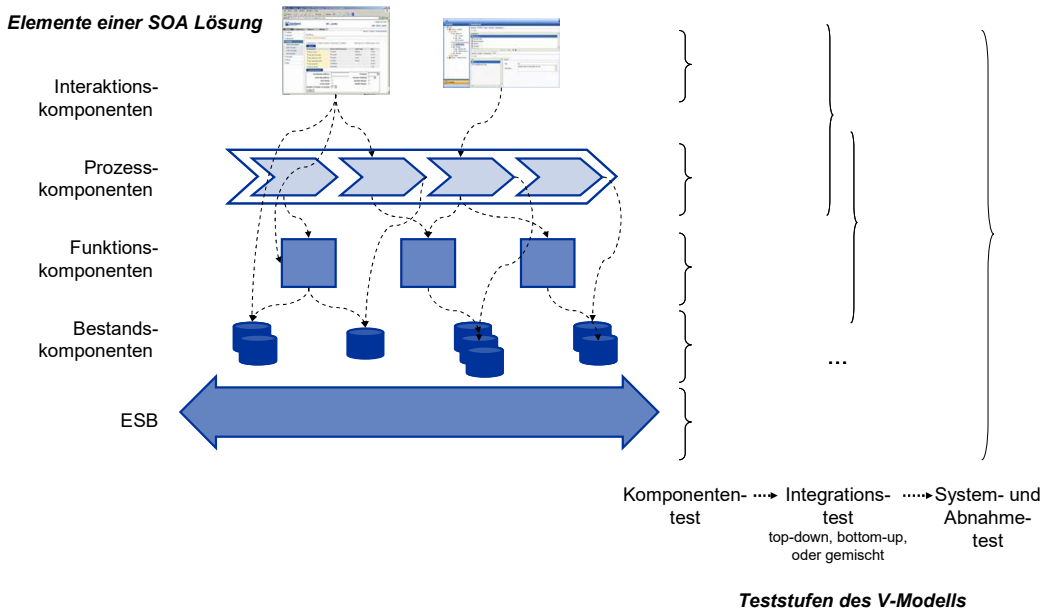

Abb. 2: Erweiterte service-orientierte Architektur

Die Komponenten werden individuell in der Komponententestebene getestet. Beim Integrationstests werden nach top-down, bottom-up oder Misch-Ansätzen die Komponenten integriert und resultierenden Komponenten-Kompositionen in ihrer Interaktion getestet. Auf System- und Abnahmetestebene wird das Gesamtsystem bewertet, was typischerweise über die Interaktionskomponenten unter Einbeziehung aller anderen Systemkomponenten erfolgt.

Neben den verschieden Teststufen sind auch die Testarten wie funktionale Tests, Leistungstests, Interoperabilitätstests, und Sicherheitstests zu unterscheiden. Funktionale Tests müssen in jeder Teststufe durchgeführt werden. Leistungstests wie Sicherheitstests werden typischerweise auf Systemebene durchgeführt, können aber ebenso gewinnbringend auf Komponentenebene durchgeführt werden. Interoperabilitätstests werden insbesondere bei der Integration der verschiedenen Komponenten(-arten) einer SOA Lösung an den ESB durchgeführt. Hier steht die Frage nach der Einbettung der Komponenten in die SOA Infrastruktur entsprechend der SOA Standards oder aber der Web Services Interoperability Organisation [16] im Vordergrund.

Es gibt mittlerweile eine Vielzahl von SOA-Test-Werkzeugen, die dazu genutzt werden können. Beispielsweise gibt es TestMaker von PushToTest [17], SOAtest von ParaSoft [18] oder SOAP Sonar von Crosscheck Networks [19], die aber oftmals nur SOAP und nicht flexible Kommunikationszugänge oder aber nur XML und nicht WSDL anbieten. Ein flexiblerer Ansatz wurde in [20] beschrieben und umgesetzt.

Referenzen

[3] CBDI-SAE™ Meta Model for SOA Version 2.0; September 2007; http://www.cbdiforum.com/public/meta_model_v2.php

[2] OASIS: Reference Model for Service Oriented Architecture; August 2006; http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=soa-rm

[3] WSDL Definition für Amazon: http://soap.amazon.com/schemas2/AmazonWebServices.wsdl

[4] WSDL Definition für eBay: http://developer.ebay.com/webservices/latest/eBaySvc.wsdl

[5] OASIS: Universal Description, Discovery, and Integration (UDDI); http://uddi.xml.org/uddi-org

[6] United Nations Standard Products and Services Code (UNSPSC®); http://www.unspsc.org/

[7] OASIS: Web Services Distributed Management (WSDM), Specification V1.1, August 2006, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsdm

[8] IBM: Web Service Level Agreements (WSLA) Project; http://www.research.ibm.com/wsla/

[9] Kruti Patel and Vladimir Tosic: Web Service Offerings Language (WSOL): Characteristics, Applications, and Tools, The Department of Systems and Computer, Carleton University; http://ottawa.ieee.ca/cs//index.php?option=com_content&task=view&id=26&Itemid=32

[10] Akhil Sahai, Anna Durante, Vijay Machiraju, Towards Automated SLA Management for Web Services, HP Laboratories, Palo Alto, California, HPL-2001-310R1, July 2002; http://www.hpl.hp.co.uk/techreports/2001/HPL-2001-310R1.pdf

[11] Beth Hutchison, Marc-Thomas Schmidt and Chris Vavra, IBM Software Group: Increasing IT flexibility with IBM WebSphere ESB software; http://www-07.ibm.com/in/university/relations/Pack_downloads/Enterprise_Service_Bus.pdf

[12] Sun Microsystems: SOA Governance Solution; http://www.sun.com/products/soa/soa_governance.pdf

[13] PrismTech: OpenSplice Version 3 middleware; http://www.prismtech.com/middleware

[14] ETSI: TTCN-3 –Testing and Test Control Notation; http://www.ttcn-3.org

[15] Bernhard Humm, Markus Voß, Andreas Hess: Regeln für serviceorientierte Architekturen hoher Qualität, Informatik-Spektrum, Springer Verlag, Juni 2006.

[16] WS-I: Web Services Interoperability Organization; http://www.ws-i.org/

[17] PushToTest: SOA-Test-Automation mit TestMaker 5.0; http://www.pushtotest.com/

[18] ParaSoft: SOAtest; http://www.parasoft.com/jsp/products/home.jsp?product=SOAP

[19] Crosscheck Networks: SOAP Sonar; http://www.crosschecknet.com/

[20] Ina Schieferdecker, Diana Vega und Cosmin Rentea: Import of WSDL Definitions in TTCN-3 Targeting Testing of Web Services, IDPT 2006, June 2006, San Diego, California, USA.

Die angegebenen Links entsprechen dem Stand November 2007

Autorenvita:

Dr. Klaus-Peter Eckert studierte Mathematik an der Technischen Universität Berlin und promovierte dort 1996 zum Thema verteilter, objekt-orientierter Systeme. Von 1977 bis 1988 war er als wissenschaftlicher Mitarbeiter am Hahn-Meitner-Institut tätig. Ab 1988 arbeit er am Fraunhofer Institut FOKUS auf den Gebieten Verteilte Systeme, Modellbasierte Entwicklung, Geschäftsprozessmodellierung und Middleware-Technologien wie CORBA, Web Services und SOA.

Tom Ritter studierte Informatik an der Technischen Universität Berlin.
Seit 1998 arbeitet er am Fraunhofer Institut FOKUS. Die Schwerpunkte seiner Arbeit liegen in den Bereichen verteilte Systeme und modellgestützte Entwicklung von Softwaresystemen. Sein besonderes Interesse gilt der Modellierung und Realisierung von Service-Güteaspekten in Softwaresystemen. Er war unter anderem an der Standardisierung und Implementierung verschiedener OMG Spezifikationen, wie beispielsweise QoS für CORBA-Komponenten, beteiligt und trägt mit seinen Veröffentlichungen zu Workshops und Konferenzen bei. Herr Ritter ist aktiver Entwickler bei der Eclipse Foundation.

Prof. Dr.-Ing. Ina Schieferdecker ist Leiterin des Kompetenzzentrums für Modellieren und Testen am Fraunhofer Institut für Offene Kommunikationssysteme (FOKUS), Berlin. Sie hält seit 2003 einen Fraunhofer Stiftungslehrstuhl an der Technischen Universität Berlin zum Entwurf und Testen kommunikationsbasierter Systeme. Frau Schieferdecker beschäftigt sich seit mehr als 10 Jahren mit Fragen des Entwurfs, der Analyse, des Testens und der Bewertung von kommunikationsbasierten, verteilten Softwaresystemen unter Nutzung von spezifikationsbasierten Techniken wie MDA, UML, MSC und TTCN-3. Sie ist Autorin vieler wissenschaftlicher Publikationen auf dem Gebiet des modellbasierten Systementwurfs, der Qualitätssicherung und des Testens.