.NET BIBLIOTHEK

hrsg. von dr. holger SCHWICHTENBERG

www.IT-Visions.de

manfred STEYER
holger SCHWICHTENBERG
matthias FISCHER
jörg KRAUSE

VERTEILTE SYSTEME UND SERVICES MIT .NET 4.5

KONZEPTE UND LÖSUNGEN MIT WCF 4.5 UND ASP.NET WEB-API

2. Auflage

Zusammenspiel von WCF, ASP.NET Web API, SignalR, Workflow Foundation, Identity Foundation, Entity Framework und Azure Service Bus

EXTRA: Mit kostenlosem E-Book

HANSER

Steyer/Schwichtenberg/Fischer/Krause

Verteilte Systeme und Services mit .NET 4.5

Manfred Steyer
Holger Schwichtenberg
Matthias Fischer
Jörg Krause

Verteilte Systeme und Services mit .NET 4.5

Konzepte und Lösungen für WCF 4.5 und ASP.NET Web-API

2., überarbeitete und erweiterte Auflage

HANSER

Die Autoren:

Manfred Steyer, IT-Visions.de, FH CAMPUS 02, Graz

Holger Schwichtenberg (Herausgeber und Autor), IT-Visions.de, Essen

Matthias Fischer, Rathenow bei Berlin

Jörg Krause, Berlin

Bibliografische Information der Deutschen Nationalbibliothek:

Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.d-nb.de abrufbar.

Lektorat: Sieglinde Schärl
Copy editing: Sandra Gottmann, Münster-Nienberge
Herstellung: Irene Weilhart
Umschlagdesign: Marc Müller-Bremer, www.rebranding.de, München
Umschlagrealisation: Stephan Rönigk
Gesamtherstellung: Kösel, Krugzell
Ausstattung patentrechtlich geschützt. Kösel FD 351, Patent-Nr. 0748702
Printed in Germany

print-ISBN: 978-3-446-43443-1
e-book-ISBN: 978-3-446-43565-0

Inhalt

Geleitwort des Herausgebers

Liebe Leserinnen, liebe Leser,

das .NET Framework ist inzwischen mehr als zehn Jahre alt, und man kam ihm Reife nicht nur altersbedingt, sondern vor allem auch aus fachlichen Gründen bescheinigen. Das Besondere an .NET ist, dass es immer mehr Anwendungsarten gibt, die sich mit der gleichen Sprachsyntax, den gleichen Bibliotheken und den gleichen Werkzeugen erstellen lassen. Zu Desktop- und Standard-Web-Applikationen haben sich inzwischen Multimedia- und Office-Anwendungen sowie Rich Internet Applications und Apps gesellt. Und auch auf der Serverseite gibt es zahlreiche Möglichkeiten für den Einsatz von .NET, vom Microsoft SQL Server über Biztalk bis hin zu SharePoint. Mit der Version 4.5 liefert Microsoft wesentliche Verbesserungen für zahlreiche Teilbibliotheken des .NET Frameworks sowie auch neue Bausteine, insbesondere im Bereich der verteilten Systeme und Webservices.

Anlässlich von .NET 4.5 aktualisieren wir natürlich auch wieder die erfolgreiche Buchreihe .NET-Bibliothek, die ich für den Carl Hanser Verlag seit 2006 als Herausgeber betreue. Die Fachbücher dieser Reihe liefern fundiertes Wissen zu zentralen Bausteinen der Klassenbibliothek im .NET Framework. Die Reihe zeichnet sich durch prägnant gehaltene Bücher aus, die das elementare Wissen zu einem Fachgebiet für den professionellen Entwickler aufbereiten. Ein weiteres Merkmal der Reihe sind die Autoren, die seit vielen Jahren mit .NET-Technologien arbeiten und ihre umfangreiche Praxiserfahrung aus .NET-Projekten in die Bücher einfließen lassen.

Die Welt der verteilten Systeme ist wieder einmal im Umbruch. SOAP wird an immer mehr Stellen durch vermeintlich einfachere REST-Dienste abgelöst. Das macht sich auch im .NET Framework bemerkbar, indem es nun mit dem ASP.NET Web API eine neue Bibliothek für HTTP-basierte Dienste gibt, mit denen man REST flexibler umsetzen kann als mit der Windows Communication Foundation (WCF). Dennoch ist SOAP und damit auch die WCF in vielen Szenarien (insbesondere, wenn .NET mit .NET oder einer anderen Plattform mit starker SOAP-Unterstützung kommuniziert) überlegen, weshalb die WCF auch weiterhin den Löwenanteil in diesem Buch ausmacht.

Als neue Themen behandelt das Buch ASP.NET Web API (und die damit zusammenhängenden Sicherheitsmechanismen insbesondere OAuth), ASP.NET SignalR sowie den Service Bus für Windows Server und Windows Azure. Der Rest des Buchs wurde auf die zum Zeitpunkt der Drucklegung verfügbaren stabilen Versionen (WCF 4.5, ADO.NET Entity Framework 5.0, WCF Data Services 5.1) aktualisiert.

Ich wünsche Ihnen, dass dieses Buch zu Ihrem Projekterfolg beitragen kann.

Essen, im Januar 2013

Dr. Holger Schwichtenberg
www.IT-Visions.de

Vorwort

Die Geschichte verteilter Softwaresysteme ist fast so alt wie die Geschichte programmierbarer Rechner, die mittlerweile in verschiedensten Formen Einzug in unser tägliches Leben gehalten haben. Gerade im Zeitalter von Internet und mobilen Anwendungen sind verteilte Softwaresysteme spannender denn je. Sei es die Kommunikation zwischen Personen, zwischen unternehmensinternen und -externen Applikationen zur Automatisierung von Abläufen, oder zwischen Bestandteilen eines Systems, zum Beispiel Client und Server, – verteilte Systeme sind omnipräsent. Während sich die Prinzipien hinter diesen Applikationen in den letzten 30 Jahren kaum verändert haben, liegt der Teufel – wie so häufig – im Detail. Es gilt, verschiedene Protokolle und Formate in Einklang zu bringen, auf die mannigfaltigen Aspekte der Sicherheit zu achten, Transaktionen zu realisieren, Fehler zu kompensieren und Datenverluste trotz Kommunikationsproblemen zu verhindern. Hinzu kommt, dass immer häufiger Systeme miteinander kommunizieren müssen, die ursprünglich gar nicht dazu konzipiert wurden.

Mit der *Windows Communication Foundation* (WCF) hat Microsoft den Versuch unternommen, die unterschiedlichen Möglichkeiten, die es zur Lösung der aufgezeigten Herausforderungen gibt, unter einen Hut zu bringen. Außerdem steht nun mit der *ASP.NET Web API* eine schlanke Alternative zur WCF zur Auswahl, welche sich zwar auf bestimmte Arten der Kommunikation beschränkt, dafür aber auch einfacher einzusetzen ist.

Die Erfahrung hat jedoch gezeigt, dass man WCF und Web API nicht isoliert betrachten darf. Für die erfolgreiche Implementierung von verteilten Systemen sind weitere Frameworks notwendig, die mit WCF bzw. Web API in Einklang gebracht werden müssen. Aus diesem Grund behandelt das vorliegende Werk auch Technologien wie die *Workflow Foundation* (WF), die zur Automatisierung von Geschäftsprozessen dient, das *Entity Framework* (EF), die WCF Data Services und WCF RIA Services zum Zugriff auf Datenbanken, die *Windows Identity Foundation* (WIF) zur Realisierung von erweiterten Sicherheitsszenarien, die dem Ruf nach Single Sign-On-Lösungen gerecht werden, und den *Azure Service Bus*, welcher mittlerweile auch zur lokalen Installation als Windows Server Service Bus verfügbar ist, zur standortübergreifenden Integration verschiedener Systeme.

Wer ist die Zielgruppe dieses Buchs?

Dieses Buch wendet sich an Personen, die bereits mit .NET gearbeitet haben, und nun verteilte Systeme sowie damit einhergehende Services entwickeln oder auf technischer Ebene planen möchten. Ebenso werden Entwickler mit Erfahrung in den genannten Technologien adressiert, die das eine oder andere Thema vertiefen oder einfach nur bestimmte Aspekte

nachschlagen möchten. Die hier präsentierten Beispiele sind in C# gehalten, da C# die führende Sprache auf der .NET-Plattform darstellt. Allerdings finden sich im Web zahlreiche Tools zur automatischen Übersetzung in andere .NET-basierte Sprachen. Die verwendete C#-Version ist die Version 5; die verwendete Version von Visual Studio ist 2012. Um den Beispielen optimal folgen zu können, sollte der Leser ebenfalls die besagten Versionen oder neuere einsetzen. Sofern jedoch nicht explizit erwähnt ist, dass die beschriebenen Features .NET 4.5 benötigen, ist davon auszugehen, dass sie auch bei Verwendung von .NET 4.0 zur Verfügung stehen.

Wie soll dieses Buch gelesen werden?

Obwohl die einzelnen Kapitel des vorliegenden Werks miteinander korrelieren und deswegen auch immer wieder aufeinander verweisen, müssen sie nicht strikt sequentiell gelesen werden. Einsteigern empfehlen wir jedoch, zumindest das erste Kapitel zu bearbeiten, bevor sie sich den jeweiligen Interessensgebieten in den Folgekapiteln zuwenden. Lesern, die sich in die WCF einarbeiten möchten, sei auch Kapitel 2 und 3 ans Herz gelegt. Jene, die mit der ASP.NET Web API starten möchten, können nach dem Bearbeiten von Kapitel 1 mit den Kapiteln 6 und 7 fortfahren.

Einige Kapitel gehen davon aus, dass der Leser den Paket-Manager NuGet verwendet, da Microsoft bestimmte Bibliotheken vorzugsweise darüber zum Download anbietet. Deswegen befindet sich ein Überblick zu NuGet im Anhang.

Die einzelnen Abschnitte erläutern die behandelten Themen meist anhand von Beispielen. Gerade das Experimentieren mit fertigen Beispielen hat sich beim Erarbeiten neuer Stoffgebiete im Selbststudium bewährt. Die Beispiele zu diesem Buch finden Sie unter *http://downloads.hanser.de*.

Wir freuen uns darüber hinaus über Ihr Feedback und Ihre Fragen. Hierfür wurde unter *www.it-visions.de/leser* ein eigener Bereich eingerichtet.

Danksagung

Dank für ihre Mitwirkung und Unterstützung an diesem Buch möchten wir aussprechen an:

- unsere Familienangehörigen, die uns neben unserem Hauptberuf das Umfeld geschaffen haben, auch an manchen Abenden und Wochenenden an diesem Buch zu arbeiten.
- Frau Sieglinde Schärl vom Carl Hanser Verlag, die dieses Buchprojekt betreut hat.
- Frau Sandra Gottmann und Frau Julia Stepp, die das Buch sprachlich verbessert haben.
- Frau Irene Weilhart, die sich um die optischen Aspekte des Buchs gesorgt hat.

Über die Autoren

Manfred Steyer

FH-Prof. Manfred Steyer (*www.softwarearchitekt.at*) ist freiberuflicher Trainer und Berater bei *www. IT-Visions.de*. Darüber hinaus ist er für den Fachbereich Software Engineering der Studienrichtung IT und Wirtschaftsinformatik an der FH CAMPUS 02 in Graz (www.campus02.at) verantwortlich.

Er schreibt regelmäßig Fachartikel im .NET-Umfeld und ist Autor mehrerer Bücher. Manfred Steyer hat berufsbegleitend IT und IT-Marketing in Graz sowie Computer Science in Hagen studiert und kann auf mehr als zehn Jahre an Erfahrung in der Planung und Umsetzung von großen Applikationen zurückblicken. Er ist ausgebildeter Trainer für den Bereich der Erwachsenenbildung und spricht regelmäßig auf Fachkonferenzen.

In der Vergangenheit war Manfred Steyer mehrere Jahre für ein großes österreichisches Systemhaus tätig. In der Rolle als Entwicklungsleiter hat er gemeinsam mit seinem zehnköpfigen Team Geschäftsanwendungen konzipiert und umgesetzt.

Seine E-Mail-Adresse lautet manfred.steyer@softwarearchitekt.at. Auf Twitter findet man ihn unter *http://www.twitter.com/ManfredSteyer*.

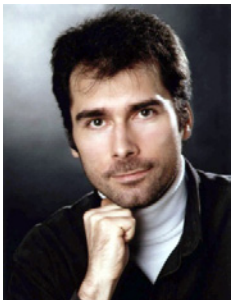
Dr. Holger Schwichtenberg

Dr. Holger Schwichtenberg ist Leiter des .NET-Expertennetzwerks www.IT-Visions.de (*http://www.IT-Visions.de*), das zahlreiche Unternehmen in Europa durch Beratung, Schulung, Coaching und Support unterstützt. Zudem ist er Entwicklungsleiter bei der 5minds IT-Solutions GmbH & Co. KG, die Software im Kundenauftrag entwickelt. Die persönlichen Schwerpunkte von Dr. Holger Schwichtenberg sind Webanwendungen, verteilte Systeme und Datenbankzugriffe.

Dr. Holger Schwichtenberg gehört durch seine Auftritte auf nationalen und internationalen Fachkonferenzen sowie zahlreiche Fachbücher für Addison-Wesley, Microsoft Press und den Hanser Verlag zu den bekanntesten .NET-Experten in Deutschland. Darüber hinaus ist er ständiger Mitarbeiter der Fachzeitschriften dotnet magazin, dotnetpro und iX sowie bei heise.de. Von Microsoft ist er für sein .NET-Fachwissen als Microsoft Most Valuable Professional (MVP) für ASP.NET/IIS ausgezeichnet worden.

Sein Weblog finden Sie unter *http://www.dotnet-doktor.de*. Bei Twitter finden Sie ihn unter *http://www.twitter.com/DOTNETDOKTOR*.

Matthias Fischer



Matthias Fischer ist seit einigen Jahren Softwareentwickler und -architekt. In seiner Tätigkeit als Technical Expert für .NET beschäftigt er sich insbesondere mit dem .NET-Framework, ASP.NET, WCF sowie Netzwerk- und Kommunikationstechnologien in C# und F#.

Des Weiteren ist Matthias Fischer zertifizierter Softwaretester mit Erfahrung im Embedded-Umfeld. Sein umfangreiches Wissen gibt er in diversen Usergroup-Vorträgen, auf Fachkonferenzen sowie als Berater und Fachbuchautor wieder.

Neben seiner beruflichen Tätigkeit beteiligt sich Matthias bei der Organisation und Ausrichtung der .NET Usergroup Berlin-Brandenburg.

Wenn es die Zeit neben den Projekten erlaubt, unternimmt Matthias gern Radtouren, oder er beschäftigt sich mit Fotografie.

Jörg Krause

Jörg Krause ist Partner des .NET-Expertennetzwerks HYPERLINK „http://www.IT-Visions.de“ www.IT-Visions.de. Er arbeitet als freier Fachautor, Trainer, Consultant und Softwareentwickler aus Leidenschaft. Schwerpunktthemen sind die Webapplikationen mit ASP.NET, Microsoft SQL Server sowie die SharePoint-Programmierung. Zu allen Themen hat er Bücher und zahlreiche Artikel in Fachzeitschriften verfasst und bietet individuelle Workshops und Trainings auf hohem Niveau an. Seit 1998 tritt er auf zahlreichen Konferenzen als Speaker auf.

Wenn er sich nicht mit Computern beschäftigt, was eher selten vorkommt, liest er bei schlechtem Wetter Bücher (Science-Fiction, Thriller) oder spielt bei besserem Wetter Golf und Badminton.

1 Serviceorientierung

Bevor es in den nachfolgenden Kapiteln um die Realisierung von Services mit Windows Communication Foundation (WCF) gehen wird, bietet dieses Kapitel einen kompakten Überblick zum Thema Serviceorientierung. Neben den unterschiedlichen Sichten auf dieses Thema wird auch auf Protokolle zur Realisierung von Services eingegangen.

1.1 Konzeptionelle Ebene

Sowohl für Techniker als auch für Betriebswirte ist Serviceorientierung von Bedeutung. Allerdings haben beide Gruppen eine unterschiedliche Sicht auf diese Thematik. Dieser Abschnitt geht auf diese beiden Sichten ein und informiert darüber, was allgemein unter Service verstanden werden kann.

1.1.1 Betriebswirtschaftliche Sicht

Auf das Thema Serviceorientierte Architekturen (SOA) gibt es zwei Sichtweisen – eine betriebswirtschaftliche und eine technische. Aus betriebswirtschaftlicher Sicht helfen serviceorientierte Architekturen beim Unterstützen von Geschäftsprozessen. Einzelne Schritte eines Geschäftsprozesses werden dabei von Services, die von verschiedenen Systemen angeboten werden, realisiert. Zur Koordinierung dieser Services wird ein weiterer Service, ein sogenannter Prozessservice, eingesetzt. Dieser ist beispielsweise mit einer Workflow-Engine implementiert, sodass Anpassungen einfach möglich sind und die Umsetzung auch gleichzeitig die Dokumentation widerspiegelt. Bild 1.1 veranschaulicht dies. Dargestellt ist hier ein sehr einfacher, mit den Mitteln der *Business Process Modeling Notation* (BPMN, vgl. *www.bpmn.org*) modellierter Prozess, der eine Vorgehensweise zur Angebotslegung für Reisen beschreibt und in vier Bereiche geteilt ist. Diese Bereiche werden in der BPMN als *Pools* bezeichnet. Die Aktivitäten in den Pools *System*, *Fluggesellschaft* und *Hotelreservierung* werden idealerweise von Services implementiert. Diese können entweder innerhalb der eigenen Organisation zur Verfügung stehen oder von Geschäftspartnern angeboten werden. Ersteres ist bei den Aktivitäten im Pool *System* der Fall; Letzteres bei den Aktivitäten in den Pools

Fluggesellschaft und *Hotelreservierung*. Die Aktivitäten im Pool *Reisebüro* würden die Applikation darstellen, die durch Kombination der restlichen Services geschaffen werden soll, und der Prozess an sich würde mit den Mitteln einer Workflow-Engine, welche die einzelnen Services koordiniert, umgesetzt werden. Hierbei ist auch von *Orchestrierung* die Rede.

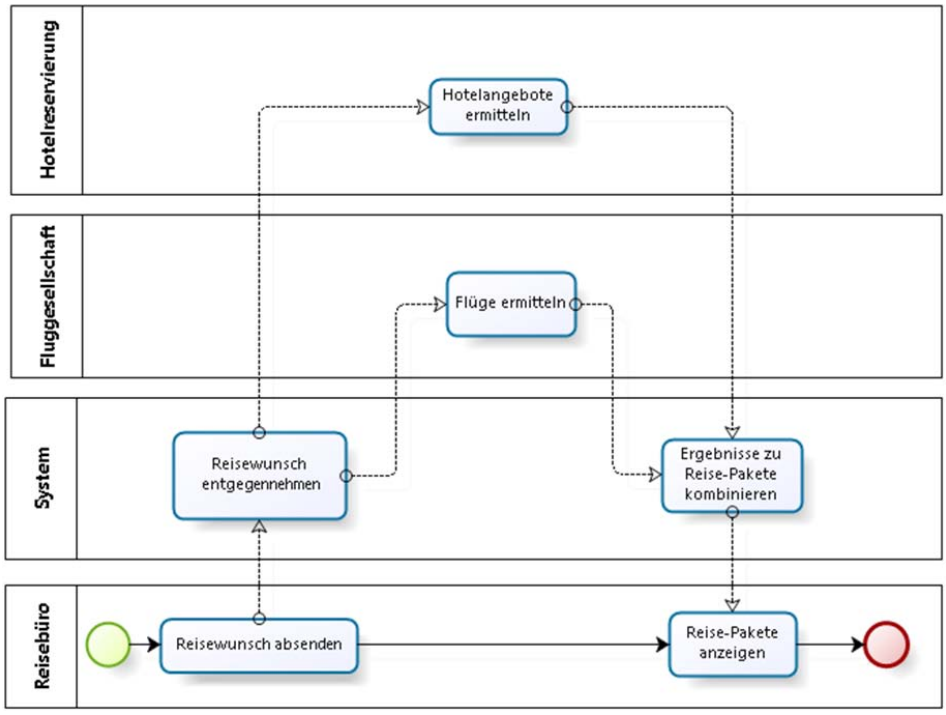


Bild 1.1 Beispielhafter Geschäftsprozess

1.1.2 Technische Sicht

Aus technischer Sicht geht es beim Thema SOA um verteilte Systeme und Systemintegration. Einzelne Dienste werden, meist über das Netzwerk, in Anspruch genommen. Dazu sendet ein Client eine Anfrage an einen Service. Dieser führt daraufhin die gewünschte Aufgabe durch und sendet das Ergebnis retour an den Client. Je nach Kommunikationspartner können dabei die zu verwendenden Protokolle und Datenformate variieren. Letztere gilt es bei Bedarf zu „übersetzen", sodass sie in einer für das Gegenüber bearbeitbaren Form vorliegen. All diese Konzepte sind wahrscheinlich so alt wie das Konzept von Computernetzwerken und firmierten vor den Zeiten von SOA zuletzt unter dem Begriff *Enterprise Application Integration* (EAI) und *Komponentenorientierung*. Der Übergang zwischen diesen Paradigmen und SOA ist, nüchtern betrachtet, fließend, zumal auch die verwendeten Technologien großteils dieselben sind. SOA unterscheidet sich von seinen Vorgängern vor allem darin, dass der Aspekt der Interoperabilität, also das Zusammenspiel verschiedener Systeme, stärker im Vordergrund steht. Dies manifestiert sich in den bevorzugt verwendeten offenen Protokollen und Datenformaten, wie zum Beispiel HTTP, XML oder auch SOAP. Daneben

schlägt sich dieser Aspekt auch in den vorzufindenden Mustern und Best Practices wieder. So wird beispielsweise häufig auf serviceübergreifende Transaktionen verzichtet, da sich deren interoperable Implementierung in der Regel als sehr herausfordernd darstellt. Stattdessen werden im Fehlerfall Kompensationslogiken, welche die ursprünglichen Aktionen rückgängig machen, angestoßen. Bezogen auf das zuvor betrachtete Beispiel könnte es sich dabei um das Stornieren eines Fluges handeln.

1.1.3 Was ist ein Service?

Neben der Tatsache, dass es sich bei einem Service um eine Sammlung von Operationen, die von einzelnen Clients über ein Netzwerk konsumiert werden können, handelt, geht damit auch ein bestimmtes Mindset einher. Im Gegensatz zum klassischen Ansatz der *Remote Procedure Calls* (RPC) ist ein Service in sich geschlossen. Das bedeutet, dass das Innenleben eines Services für den Aufrufer eine Blackbox darstellt, sodass dieser nicht über technische Details Bescheid wissen muss. Aus diesem Grund sind Service-Operationen auch stark Use-Case-orientiert und selten technisch bzw. generisch. Dies führt auch dazu, dass Service-Operationen mitunter grobgranularer als herkömmliche Methoden sind. Ein gutes Beispiel hierfür bietet die nachfolgend dargestellte Operation `BucheFlug`.

```
public Ticket BucheFlug(Buchung buchung) { … }
```

Einige dazu passende Negativbeispiele gehen aus Listing 1.1 hervor, zumal sich der Aufrufer hier mit technischen Konstrukten des Systems, wie zum Beispiel *FlugHandles* oder *SeatHandles*, beschäftigen muss. Gute Services würden hingegen fachliche Konstrukte verwenden, z. B. Flüge anstatt *FlugHandles*.

Diese Operationen könnte es im vorhin betrachteten Positivbeispiel zwar auch geben, allerdings wären diese hier zum einen nicht öffentlich zugänglich und würden zum anderen von der Operation `BucheFlug` koordiniert werden. Services mit solchen Koordinierungsmethoden werden auch als *Fassaden* bzw. *Servicefassaden* bezeichnet, da sie dahinter liegende Details verbergen. Dabei soll nicht verschwiegen werden, dass die Verwendung von Fassaden auch schon vor dem Aufkommen des Begriffs SOA gängige Praxis im Bereich verteilter Geschäftsanwendungen war. Der Vorteil dieser Vorgehensweise liegt neben der Tatsache, dass sich der Aufrufer nicht mit Interna belasten muss, in einer geringeren Netzwerkbelastung, da anstatt vieler kleiner Nachrichten wenige oder nur eine größere zu übersenden sind. Daneben führt dieses Muster dazu, dass die von der Fassade gekapselten Details ohne Anpassungen des Clients ausgetauscht werden können. Dies erlaubt es, die verwendeten Software-Systeme einfacher an sich ändernde Geschäftsregeln und Abläufe anzupassen, was in weiterer Folge auch die Agilität des Unternehmens erhöht. Hierbei ist auch von loser Kopplung die Rede, was im betrachteten Fall bedeutet, dass der Client möglichst wenig über den konsumierten Service wissen muss.

Listing 1.1 Negativbeispiele für Service-Operationen

```
public FlightHandle GetFlightHandle(string flugNummer, DateTime datum) { … }
public SeatHandle CheckFreeSeats(FlightHandle h) { … }
public bool CheckFlightCancelled(FlightHandle h) { … }
public bool CheckCustomerExists(String customerNumber) { … }
```