SIND MICROSERVICES DIE BESSERE SOA?

Microservices sind der neue Stern am Himmel der Software-Architektur. War vor zwei Jahren nur vereinzelt davon zu hören, gilt das Thema mittlerweile als einer der wichtigsten aktuellen Architekturtrends. Höchste Zeit also, sich auch in der öffentlichen Verwaltung damit zu beschäftigen.

| von ANDREAS RAQUET

„Microservices" bezeichnet einen Architekturstil, bei dem große Software-Systeme, sogenannte Software-Monilithen, in mehrere, unabhängig voneinander verteil- und betreibbare Einzelanwendungen zerlegt werden. Wer aus der Behörden-IT kommt, wird sich über diese Charakterisierung wundern: Geschieht dies mithilfe serviceorientierter Architekturen (SOA) nicht schon seit fast 15 Jahren? Tatsächlich bezeichnete die Firma Netflix, einer der Pioniere der Microservices, den Architekturstil früher als „loosely coupled, serviceoriented architecture with bounded context" (lose gekoppelte, serviceorientierte Architektur mit abgeschlossenem Kontext). Um zu verstehen, was Microservices sind, warum sie entstanden sind und worin sie sich von SOA unterscheiden, lohnt ein Blick in die Historie.

Ihren Ursprung fanden Microservices bei den Internetpionieren wie Amazon und Netflix. Diese hatten Anfang des 21. Jahrhunderts eine monolithische Webanwendung entwickelt, die quasi ihr gesamtes Geschäftsmodell trug. Mit dem explosionsartigen Erfolg der Unternehmen wuchsen auch die Anforderungen an die Anwendungen. Gleichzeitig wurden die Fehlerbehebung und Weiterentwicklung immer schwieriger und träger. So berichtet Adrian Cockroft, damals Chefarchitekt bei Netflix, dass die Netflix-Anwendung im Jahr 2008 einen Umfang erreicht hatte, der ca. 100 Software-Ingenieure für die Wartung und Weiterentwicklung erforderte[1]. Das Problem: Die Entwicklung eines einzelnen Software-Artefakts skaliert nicht zu dieser Größe. Die Entwickler

1 „Fast Delivery: Adrian Cockcroft @nginxconf 2014", https://youtu.be/5qJ_BibbMLw

blockieren sich gegenseitig. Die Lokalisierung von Fehlerursachen wird erheblich verkompliziert.

Ihre Anwendungen effizient und flexibel weiterentwickeln zu können, war für Netflix, Amazon und weitere sogenannte One-Application-Companies[2] überlebenswichtig. Daher entschieden sie sich für einen radikal anderen Ansatz, der im Wesentlichen auf drei Säulen beruht:

- Die monolithische Anwendung sollte in kleinere Anwendungen zerlegt werden, die unabhängig voneinander entwickelt, getestet und in Betrieb genommen werden können. Diese Anwendungen werden „Microservices" genannt.
- Für jeden Microservice ist ein Team zuständig, das sowohl Entwicklung als auch Test und Inbetriebnahme verantwortet. Dadurch sollten die üblichen Reibungsverluste und Verzögerungen zwischen Entwicklungsteam, QS-Team und Betriebsabteilung eliminiert werden – ein Ansatz, der in den aktuellen DevOps-Trend mündete.
- Die Betriebsinfrastruktur wird vollständig auf einer automatisierten Cloud-Infrastruktur aufgesetzt – im Falle von Netflix auf die Amazon Web-Services (AWS). Dadurch wird der lange Planungs- und Beschaffungsvorlauf für Hardware komplett eliminiert. Microservices können quasi spontan mit Betriebsinfrastruktur versorgt werden.

Cockroft stellt das enge Zusammenspiel dieser drei Säulen heraus, durch das der Ansatz letztlich zum Erfolgsmodell wurde. In der Folge habe viele andere Internetunternehmen wie Uber, Zalando oder Otto die Konzepte mit Erfolg übernommen.

Was sich bei jungen Internetunternehmen bewährt hat, lässt sich jedoch nicht einfach auf die klassische Behörden-IT übertragen. Trotz zunehmender Virtualisierung sind klassische Rechenzentren weit von der „Infrastruktur auf Knopfdruck" der Cloud-Anbieter entfernt. Und das verbreitete langfristige Outsourcen von Entwicklungskapazität an Dienstleister macht einen DevOps-Ansatz in diesem Umfeld zumindest kurzfristig mehr als unwahrscheinlich (siehe Infobox). Was bleibt, ist die Zerlegung in einzelne Services, die Kernidee der SOA. Allerdings unterscheiden sich Ziele und Ansätze deutlich von denen der SOA.

DER FACHLICHE SCOPE

SOA war von Beginn an darauf ausgerichtet, die IT eines ganzen Unternehmens in eine Servicelandschaft umzugestalten. Microservices konzentrieren sich hingegen auf eine einzelne Anwendung und zerlegen diese in sehr viel kleinere Services. Allerdings verliert dieser Unterschied zunehmend an Bedeutung.

- Werden Microservices auf komplexe Fachverfahren angewendet, können sie nicht beliebig klein geschnitten werden. Denn sonst zerschneidet man vielfältige Abhängigkeiten und muss erheblichen Aufwand betreiben, um im sich ergebenden Konglomerat an Teilanwendungen wieder die gewünschte Konsistenz herzustellen. Microservices sind also nicht zwingend „micro".
- SOA-Transformationen auf der Ebene der Gesamt-IT haben sich als sehr aufwendig und schwer beherrschbar erwiesen. Daher haben SOA-Architekten begonnen, SOA-Prinzipien auf überschaubare Teile der Gesamt-IT – auf einzelne Fachdomänen – anzuwenden.

Der Scope einer eher grob geschnittenen Microservices-Landschaft unterscheidet sich also gar nicht so sehr von dem einer lokalen SOA-Umgebung. Ein großer Vorteil der Microservices ist dabei allerdings, dass sie sich explizit mit der Frage nach dem geeigneten fachlichen Serviceschnitt beschäftigen. Häufig werden Conways Law[3] und Domain Driven Design[4] als wesentliche Treiber für den fachlichen Schnitt herangezogen. SOA-Initiativen rühren dagegen meist nicht an den gewachsenen Organisationsstrukturen.

EINGESETZTE TECHNOLOGIEN

Microservices nutzen moderne Technologien wie REST, Containerization oder NoSQL-Datenbanken. Dabei kann theoretisch jeder Microservice in einer anderen Sprache oder Technologie realisiert werden und beispielsweise eine andere Datenbank verwenden. Allerdings sind weder die verwendeten Technologien noch die heterogene Entwicklungsplattform zwingende Voraussetzung für Microservices. Die Technologieauswahl ist vielmehr der „kulturellen" Herkunft der Microservices geschuldet. Viele Microservice-Anwender nutzen weiterhin relationale Datenbanken, und die meisten konzentrieren sich auf eine einzige Storage-Technologie und Entwicklungsplattform.

Im SOA-Umfeld hat sich mit Process Servern, Governance Suites oder Enterprise Service Busses (ESBs) eine ganze Reihe technischer Infrastrukturen etabliert, die lange Zeit sehr eng mit SOA verbunden erschienen. Unternehmen haben in den letzten Jahren erheblich in diese Technologien investiert. Anstelle von Einsparungen brachten diese zentralen Infrastrukturen jedoch vor allem Reibungsverluste mit sich: Selbst einfachste Änderungen erfordern Eingriffe in diverse Infrastrukturelemente, die jeweils eigenes Spezial-Know-how voraussetzen und von eigenen Teams verantwortet werden. Mittlerweile findet hier ein Umdenken statt. Jüngere SOA-Initiativen besinnen sich wieder auf die Wurzeln der SOA – die Gestaltung der IT auf Basis miteinander interagierender Services.

2 Eine Bezeichnung für Internetunternehmen, deren Geschäft im Wesentlichen auf einer einzigen Webapplikation besteht, die sie ihren Kunden anbieten. Im Gegensatz dazu betreibt eine Behörde eine Vielzahl von Anwendungen für unterschiedlichste fachliche Verfahren.

3 https://de.wikipedia.org/wiki/Gesetz_von_Conway
4 https://de.wikipedia.org/wiki/Domain-driven_Design

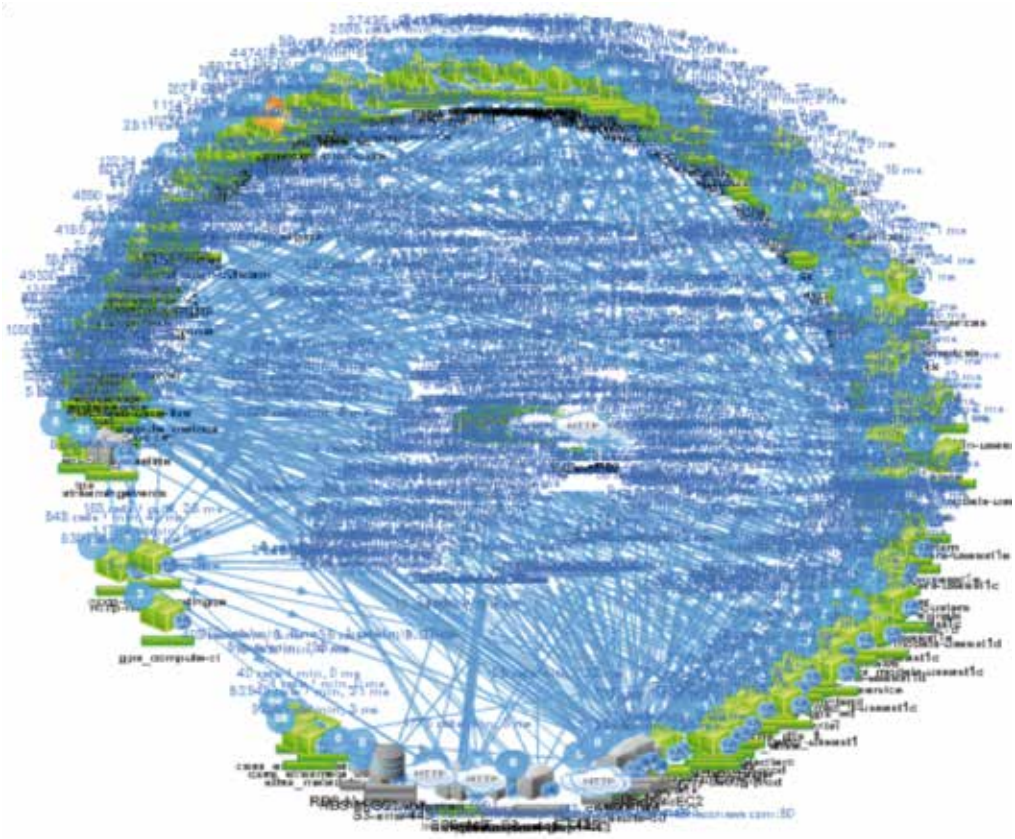

Abbildung 1: Microservice-Topologie bei Netflix (Quelle: A. Tseitlin, „Resiliency through failure", QCon NY, 2013). Abhängigkeiten von Services werden mit zunehmender Anzahl schnell unüberschaubar. In SOA versucht man, diesem Effekt durch Einsatz eines Enterprise Service Bus entgegenzuwirken.

AUSRICHTUNG AUF ENTWICKLUNGSEFFIZIENZ

SOA verfolgt primär die Ziele Agilität und Kosteneffizienz[5]. Microservices sind alleine auf Agilität ausgerichtet. Das klingt gar nicht so unterschiedlich. Tatsächlich hat die unterschiedliche Gewichtung jedoch zu grundsätzlich unterschiedlichen Herangehensweisen geführt.

Microservices setzen konsequent auf Dezentralisierung und Deregulierung. Sie erreichen ihr Ziel – die Agilität –, indem sie es ermöglichen, neue Services sehr schnell zu entwickeln und in Betrieb zu nehmen.

SOA verfolgt einen zentralistischen Ansatz: Services werden zentral angeboten, überwacht und gesteuert. Der einzusetzende Technologiestack und die Architektur werden durch Architektenboards explizit gesteuert und somit möglichst einheitlich und stabil gehalten. Neue Services können hier nur langsam in Betrieb genommen werden. Die Agilität entsteht vor allem dadurch, dass bereits eine große Zahl an Services existiert, die wiederverwendet werden können – zumindest in der Theorie. In der Praxis hat sich das jedoch kaum bewahrheitet. Der sehr fundamentale Ansatz „Agilität durch Wiederverwendung bestehender Services" kann heute als gescheitert angesehen werden.

SIND MICROSERVICES DIE BESSERE SOA?

Offenbar unterscheiden sich Microservices gar nicht so sehr von modernen SOA, sind ihnen aber in einigen Aspekten überlegen. Allerdings zeigen sie Schwächen in Bereichen, die in der SOA-Welt längst gelöst sind:

Um eine größere Servicelandschaft dauerhaft zu beherrschen, bedarf es einer gewissen Steuerung (Governance). Dazu gehört auch eine übergreifende Service-Registratur, die eine Wiederverwendung von Services in einer Behörde mit zahlreichen Fachfunktionen erst möglich macht. Microservices bieten hierfür bisher keine entsprechenden Konzepte oder vorgefertigten Werkzeuge.

5 http://soa-manifest.de/praeambel.html

Immerhin wurde mittlerweile erkannt, dass ein dezentraler Ansatz ohne irgendeine Art zentraler Steuerung nicht beherrschbar ist. Hier entsteht eine Reihe proprietärer Werkzeuge für Monitoring, Service Discovery oder Configuration Management. Wie robust diese sind und wie weit darauf basierende Microservice-Landschaften skalieren, ist derzeit kaum abzuschätzen (siehe auch Grafik Service-Topologie bei Netflix).

Auch der Einsatz moderner Technologien in Microservices hat seine Schattenseiten. So berichtet Matt Ranney, Chief Systems Architect bei Uber, dass der Einsatz von JSON für die Datenübertragung[6] einer der größten frühen Designfehler war, da eine syntaktische Nachrichtenvalidierung dadurch faktisch unmöglich wurde.[7]

Schließlich ist der große Themenkomplex der Service-Integration weitgehend offen. Eine Anwendung in viele kleine Services zu zerlegen, mag für den Entwickler attraktiv klingen, für den Anwender ist es das aber sicher nicht. Er erwartet weiterhin eine einheitliche Benutzeroberfläche. Hier finden sich zwei Ansätze:

- Die Benutzeroberfläche wird als eigener Client implementiert. Dieser ist hoch integriert, aber über sehr viele Schnittstellen mit allen anderen Microservices verknüpft. Der Ansatz skaliert nur für kleinere Anwendungslandschaften.
- Jeder Microservice bringt seine eigene Oberfläche mit. Die Integration erfolgt, wie beispielsweise bei OTTO.de, über krude und proprietäre Technologien wie Server Side Includes[8].

Die beiden Ansätze münden in die konkrete Frage, ob ein Microservice nun eine Webanwendung mit Benutzeroberfläche ist, oder wirklich nur ein Webservice, der auf anderem Wege integriert werden muss. In der Microservices-Community besteht darüber keine Einigkeit.

Ähnlich verhält es sich bei der Backend-Integration. Microservices akzeptieren ein gewisses Maß an Redundanz zwischen Services. Dazu, wie die Verteilung und der Abgleich der Daten erfolgen, machen Microservices jedoch keine Aussagen. In der Praxis findet man auch hier proprietäre Lösungen anstatt der von den SOA-Suiten implementierten EAI-Patterns.

Diese Beispiele zeigen: Microservices sind zwar pragmatischer als SOA, haben aber nicht denselben Reifegrad erreicht. Insbesondere gibt es für viele Herausforderungen keine Standardlösungen. Anwender müssen daher proprietäre Lösungen für Probleme finden, die in der SOA-Welt längst gelöst sind. Derzeit ist es also sicher zu früh, SOA zu begraben. Allerdings kann man von den Microservices so einiges lernen:

- die Reduzierung des Scopes auf einen beherrschbaren Umfang – sowohl funktional als auch (entwicklungs-)organisatorisch,
- die Konzentration auf pragmatische Ansätze mit unmittelbar messbarer Wirkung,
- der sparsamen Einsatz zentraler Infrastruktur.

Microservices entwickeln sich derzeit rasant weiter. Mit Eureka, Ribbon und Hystrix gibt es derzeit OSS-Bausteine für Service Discovery, Client-side Loadbalancing und Fault Tolerance, die in der SOA-Welt bisher in die Domäne der zentralen Infrastruktur gehörten.

DEVOPS UND CONTAINERIZATION

DevOps und Containerization sollen Software-Entwicklungsteams in die Lage versetzen, Software schneller in Betrieb zu nehmen. DevOps bezeichnet dabei das Vorgehen, Containerization bietet Werkzeuge dazu. Unterschätzt wird dabei häufig der Paradigmenwechsel, den dieser Ansatz mit sich bringt. Versteht sich die klassische Behörden-IT bisher als Betreiber von Anwendungen, um die herum ITIL-Prozesse umgesetzt werden, wird sie nun zum Betreiber von Infrastruktur „degradiert". Dieser Ansatz passt durchaus zu den Cloud-Anbietern wie Amazon und Strato, die sich ohnehin nie als etwas anderes gesehen haben als Infrastrukturbetreiber. In einem IT-DLZ wirft das jedoch einige Fragen auf:

- Verantwortlichkeiten wie IT-Sicherheit und Datenschutz gehen teilweise an die Entwicklung über und müssen neu geregelt werden.
- Große Teile des Incident-Managements verlagern sich in die Entwicklung. Die Schnittstelle zwischen Entwicklung und Betrieb muss neu gestaltet werden.
- Wer betreibt Software-Produkte, die nicht im Hause entwickelt werden? Wer betreibt querschnittliche Infrastrukturprodukte wie zum Beispiel LDAP-Server?

Nachdem sich die IT-DLZs mit der Anwendung von ITIL in den vergangenen Jahren in diesen Bereichen erheblich weiterentwickelt haben, ist nicht abzusehen, dass diese Kompetenzen nun an die Software-Entwicklung abgegeben werden.

6 Siehe Artikel über RESTful URL, .public 03-2016
7 https://www.infoq.com/articles/podcast-matt-ranney
8 Server Side Includes (SSI) sind in (meist HTML-)Dokumente eingebettete, einfach zu nutzende Skript-Befehle, die auf dem Webserver ausgeführt werden, bevor das Dokument an den Client ausgeliefert wird; siehe https://dev.otto.de/2015/09/30/on-monoliths-and-microservices

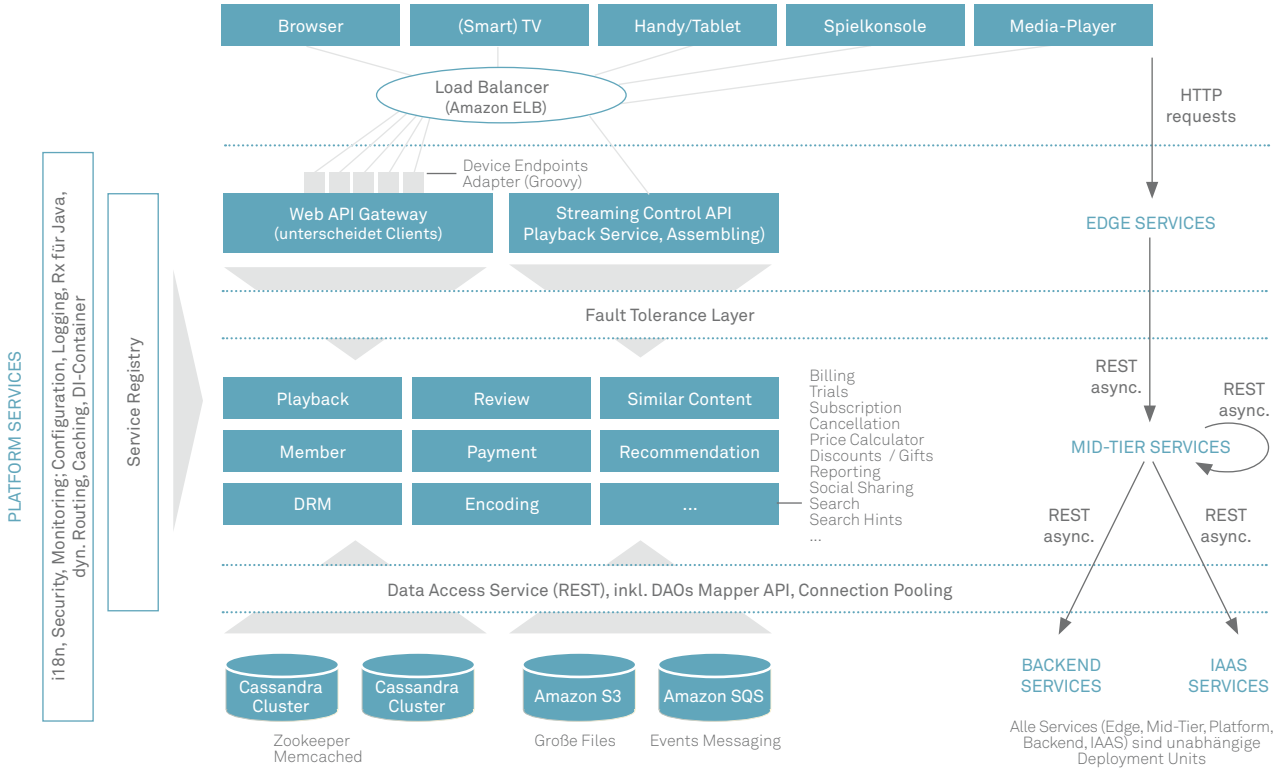

Abbildung 2: Übersicht fachliche Microservices und Servicekategorien bei Netflix (Quelle: S. Toth, www.embarc.de). Beispiel für eine Microservices-Architektur, wie sie bei Netflix entstanden ist. Im Unterschied zu monolithischen Anwendungen stellt jeder Kasten eine eigene Anwendung (einen Microservice) dar.

FAZIT

Microservices sind ein Architekturstil, der radikal auf Flexibilität der Anwendungsentwicklung ausgelegt ist. Er basiert auf Serviceorientierung, DevOps und Cloud-Technologien und findet in der Internet-IT mittlerweile große Verbreitung. Im Umfeld von Behördenanwendungen sind Microservices in ihrer ursprünglichen Form derzeit kaum praktikabel. Daher werden sie hier primär als leichtgewichtige und pragmatische Serviceorientierung ohne zentrale Infrastruktur verstanden. In jüngerer Zeit hat sich dafür der Begriff Self-Contained Systems (SCS) etabliert. Der Ansatz vermeidet einige Schwächen früherer SOA-Initiativen, der Reifegrad der Konzepte bleibt jedoch noch hinter denen von SOA zurück. Gleichzeitig entwickeln sich auch SOA Architekturen hin zu leichtgewichtigen, pragmatischen Ansätzen mit wenig zentraler Infrastruktur. Da die Zielsetzung von SOA besser in die Behörden-IT passt, dürfte man daher am besten mit einer modernen, leichtgewichtigen SOA beraten sein. Gleichzeitig kann es nicht schaden, die derzeit sehr aktive Entwicklung der Microservices zu beobachten und Konzepte oder Frameworks daraus zu übernehmen. ●

ANSPRECHPARTNER – ANDREAS RAQUET
Principal IT-Consultant
Public Sector Solutions Consulting

Reloaded!

Unsere Studie „IT-Dienstleistungszentren in der öffentlichen Verwaltung“ geht in die nächste Runde!

Die Landschaft der IT-DLZ in der öffentlichen Verwaltung entwickelt sich durch die Gründung des ITZBund und der weiter voranschreitenden Konsolidierung der IT-DLZ auf allen Ebenen der öffentlichen Verwaltung ständig weiter.

2015 haben wir begonnen, diese Entwicklung systematisch zu erfassen. Dieses Jahr schreiben wir unsere Studie mit der Ausgabe 2017 fort. In der aktuellen Ausgabe analysieren wir den Status quo und die Veränderungen seit 2015. Wir befragen dazu mehr als 100 Entscheidungsträger aus Bund, Länder und Kommunen – unter anderem zu folgenden Schwerpunktthemen:

- Erfahrungen mit der IT-Konsolidierung Bund
- Fachliche und technische Anforderungen der Behörden an IT-Unterstützung
- Leistungsangebote der IT-DLZ
- Fortschreibung der Bestandsaufnahme, Vergleich mit 2015 und Analyse der Veränderungen
- Identifikation von aktuellen Schwerpunkten und Trends, wie zum Beispiel eAkte, aufgrund des E-Government-Gesetzes, Open Data, Cloud, digitaler Transformation, EAM

In Expertengesprächen mit Behördenleitern und IT-Verantwortlichen in den Ministerien diskutieren wir die Ergebnisse und holen ihre Expertenmeinung dazu ein.

Alle Ergebnisse und Interviews können Sie in der kommenden Ausgabe unserer Studie lesen, die im September 2017 erscheint.

Sichern Sie sich Ihr kostenfreies Exemplar unter: www.msg.group/studie-public

.consulting .solutions .partnership

.msg