

# Investitionen in Flexibilität

IT-Systeme und Software müssen stabil und zuverlässig sein, gleichzeitig wird aber auch Flexibilität verlangt. Damit dies erreicht werden kann, sind Investitionen notwendig.

VON PETER MOOR UND MARKUS SCHÖNBÄCHLER

**M**it Software ist es ein wenig wie mit Wein: Der spezifische Vorteil eines bestimmten Weins führt inhärent zu einem «Lock-in» auf bestimmte Arten von Gerichten. Man sollte sich also gut überlegen, in welche Weine man investiert. Denn: Ist der Weinkeller einmal mit Burgunder gefüllt, wird man nicht mehr so leicht zum Fischrestaurant. Und wie sieht es bei Software aus? Kann man Software so gestalten, dass sie so flexibel nutzbar ist wie das Wasser beim Essen? Oder haben IT-Systeme eher den Charakter von distinktierten Weinen? Es wäre zu schön, wenn auch komplexe IT-Landschaften die Flexibilität von Wasser hätten. Manch ein Experte strebt dieses Ideal an – denn ist es nicht die Kernqualität von Software, dass sie soft, «weich» ist?

Die tägliche Praxis zeigt ein komplett anderes Bild: Software ist nur so lange «soft»,

spricht leicht änderbar, solange man nicht verlangt, dass sie irgendwelche vorgegebene Anforderungen wie zum Beispiel die korrekte und zeitgerechte Abwicklung eines Geschäftsprozesses erfüllt. Real existierende Systeme sind hingegen eher schwer zu ändern, haben also mehr den Charakter von Wein – oder Essig, bei den ungeliebten Legacy-Systemen.

Wo aber liegt das Problem bei der Software? Software kann schnell sehr komplex werden, und selbst kleine Fehler können grosse Folgen haben: Dies ist eine Folge der «Nichtlinearität» oder «Schwachen Kausalität» von Software: Kleine Ursachen können kleine, aber auch beliebig grosse Wirkungen entfalten. Dazu kommt, dass innere Zusammenhänge in der Software (z.B. Kontrollflüsse und Datenstrukturen) hochdimensional sein können – sie lassen sich nicht leicht und eindeutig auf ein Blatt Papier projizieren, so dass einem alle wichtigen Zusammenhänge intuitiv ins Auge stechen. Dies führt dazu, dass man meistens nicht bewusst wahrnimmt, wenn Software so komplex wird, dass sie einem entgleitet. Ob man sie noch im Griff hat, zeigt sich erst später bei Änderungen. So verhindert paradoxerweise ausgerechnet die enorme Leichtigkeit, mit der Software geändert werden kann, das Durchführen von tatsächlichen Änderungen, weil das Erhalten der Korrektheit beziehungsweise die Erfüllung von gegebenen Anforderungen umso schwieriger ist. Das kann bis zum «If it works don't fix it»-Syndrom führen: Lieber mit unangemessener Software arbeiten, als zu riskieren, dass die Situation durch Änderungen noch schlimmer gemacht wird.

## Agilität heisst Flexibilität

Informationssysteme flexibel zu gestalten, damit sie überhaupt und gleichzeitig auch innerhalb der geforderten Zeit änderbar sind, ist gleichbedeutend mit Investitionen: Investitionen in die Architektur dieser Informationssysteme. Das Fällen von Architekturentscheiden und das Defi-

nieren von Architekturvorgaben und -massnahmen nämlich verursachen Kosten und stellen damit nichts anderes als Architekturinvestitionen dar. Der Nutzen einer Architekturinvestition kann sich durch Kosteneinsparungen (geringerer Realisierungs- und Betriebsaufwand) oder durch bessere Time-to-Market ergeben. Die Frage stellt sich nun, wo investiert werden soll? Es ist in die Informationssysteme zu investieren, die wirtschaftlich und strategisch wichtige Geschäftsprozesse unterstützen und bei denen die Wahrscheinlichkeit gross ist, dass eine (Prozess-)Änderung eintritt.

## Flexibilität heisst auch Modularität

Ein Schlüssel für die Flexibilität ist die Modularität der IT-Unterstützung. Die IT-Unterstützung ist so in Informationssysteme aufzuteilen, dass Änderungen nicht über die ganze Anwendungslandschaft gemacht werden müssen, sondern dass sie mehrheitlich lokal, sprich systembegrenzt vorgenommen werden können. Der Modularitätsgedanke erstreckt sich weiter bis zum konkreten Design und zur Implementierung eines Informationssystems.

## Modularität heisst Komplexität

Die Business-Domäne und die Anforderungen an die benötigte IT-Unterstützung geben die Komplexität vor. Die Komplexität reduzieren heisst in Tat und Wahrheit, die Komplexität beherrschbar zu machen. Je grösser die geforderte IT-Unterstützung – gemessen in geforderter Qualität innerhalb der geforderten Zeit und zu angemessenen Kosten –, desto grösser ist die inhärente Komplexität, die eine Informatikorganisation bewältigen muss. Natürlich kann eine Informatikorganisation, welche ihr Handwerk nicht versteht und folgedessen die geforderte IT-Unterstützung schlecht konzipiert und bereitstellt, selbst zu einem beträchtlichen Komplexitätstreiber werden. Die Ingredienzien für die Beherrschung der durch die Business-Anforderungen inhärent vorgegebenen Komplexität sind sehr gut ausgebildete und erfahrene Software-Ingenieure, das Betreiben einer IT-Architektur in einer der Unternehmensgrösse angemessenen Form, hohe Prozessmaturität – und ein tief in der Organisation verwurzeltes unternehmerisches Denken.

Wenn aber in der Organisation bei mehr oder weniger jedem Entwicklungsvorhaben Fragen gewälzt werden wie: In welcher Form und Ausführlichkeit soll die Software dokumentiert werden? Wie wird das Software Configuration und Build Management aufgebaut (Folge: fehlende Prozessmaturität)? Oder: Welche Anwendungs- und Plattformarchitekturvorgaben sind zu berücksichtigen (Folge: fehlende IT-Architekturmaturität)? In diesem Fall ist man definitiv nicht agil. ■

## DIE AUTOREN

Peter Moor ist Leiter Geschäftsbereich IT-Entwicklung sowie Mitglied Geschäftsleitung Schweiz bei der Swiss Life und besitzt einen MBA-Titel der Universität St. Gallen. Seine berufliche Laufbahn führte Moor von der TRO Treuhand & Revisions AG über die Credit Suisse und die Klinik «Im Schachen» schliesslich zur Swiss Life, zu der er 2000 als Head Information Technology wechselte.



Markus Schönbächler ist seit mehr als 20 Jahren in der Informatik in verschiedenen Funktionen (Entwickler, Methodiker, Architekt, Projektleiter und Abteilungsleiter) tätig. Seit 1997 arbeitet er für die IT-Entwicklung der Swiss Life Schweiz.

