

Programmieren ist out – flexiblere Ansätze für agilere Unternehmen

Die zunehmende Verzahnung von Business und IT erfordert immer mehr Flexibilität. Maximale Automatisierung, volle Transparenz und messbar gesteigerte Wiederverwendbarkeit ist mit den etablierten Programmiersprachen nicht zu erreichen. Einen leistungsfähigeren Ansatz bieten direkt ausführbare Modelle.



Alex Büch

ist Vorstandsmitglied
von SwissICT und CTO von E2E

Seit der Internetrevolution ist IT kein Geheimtipp mehr. Sie steht im Zentrum von Wirtschaftlichkeit, Wachstum und unternehmerischer Anpassungsfähigkeit. IT ermöglicht die Umsetzung neuer Geschäftsmodelle. Wer seine IT nicht im Griff hat, ist nicht wettbewerbsfähig. Seit der Erfindung des Compilers durch John W. Backus und Grace M. Hopper haben immer wieder ähnliche Prinzipien die Evolution der Software-Entwicklung bestimmt: Im Vordergrund stand eine Steigerung der Produktivität bei gleichzeitiger Erhöhung der Qualität. Erreicht wurde dies durch verstärkte Automatisierung, Erhöhung des Abstraktionsgrades in Richtung der Problemstellung und verbesser-

te Wiederverwendbarkeit, da Zeit und Geld schlicht fehlen, um immer wieder von vorn zu beginnen. Die Summe dieser Prinzipien schaffte die Basis für Innovation und Kostenreduktion.

An diesen Grundprinzipien als treibende Kraft wird sich auch in Zukunft nicht viel ändern. Im Vergleich zu den Anfängen haben die heutigen Ansätze, aufbauend auf den klassischen Programmiersprachen, wesentlich dazu beigetragen, die Kernanforderungen in Bezug auf Produktivität und Qualität zu verbessern. Auch wenn es nicht möglich ist, deren Wertigkeit im Sinne der Produktivitätssteigerung exakt zu beziffern, so ist es doch intuitiv nachvollziehbar, dass diese Verbesserungen im Bereich mehrerer Grössenordnungen liegen. Hier ist mit den klassischen Programmiersprachen heute jedoch eine Grenze erreicht, da die Steigerung von Produktivität und Qualität auf der Ebene des einzelnen Entwicklers voll ausgereizt ist und Verbesserungen nur noch marginal möglich sind.

«Der Leitsatz «ein Bild sagt mehr als tausend Worte» zeigt klar auf, in welcher Richtung die Lösung zu suchen ist.»

Ein grafisches Modell sagt mehr als tausend Zeilen Code

Der nächste Schritt zur Steigerung von Produktivität und Qualität weist in eine Richtung, die sich weniger mit dem einzelnen Entwickler beschäftigt, dafür aber zum Ziel hat, mehr Effizienz auf Firmenebene und über Firmengrenzen hinweg zu erreichen. Architekturkonzepte wie Serviceorientierung oder Event Processing zeigen zwar prinzipiell den Weg auf, verlangen aber auch nach neuen Sprachkonzepten.

Der Leitsatz «ein Bild sagt mehr als tausend Worte» zeigt klar auf, in welcher Richtung die Lösung zu suchen ist. Diese führt uns in die Welt der grafischen Modellsprachen, die uns schon seit Jahren als Design-Hilfsmittel in Form der Unified Modeling Language (UML) oder neu in Form der Business Process Modeling Notation (BPMN) oder gar domänenspezifischer Designsprachen (Domain Specific Languages, DSL) begleiten. Waren diese grafischen Beschreibungssprachen in der Vergangenheit nur wenig mehr als Vorlagen für den Programmierer oder die Grundlage von teil-

weise automatisch erzeugtem Code, so bietet eine direkte Ausführbarkeit von Modellen den lang ersehnten Quantensprung. Denn Modelle erlauben eine stärkere Automatisierung, bieten einen höheren Grad an Abstraktion und ermöglichen mehr Wiederverwendbarkeit. Zum ersten Mal kann der Auftraggeber, beispielsweise eine Fachabteilung, dank der stark erhöhten Transparenz grafischer Modelle direkt mit in die Umsetzung der Software-Entwicklung einbezogen werden. Und ein weiteres Problem löst sich wie von selbst: Bei einem Ansatz basierend auf Model Execution gilt das Prinzip «Dokumentation gleich Code» - und nicht umgekehrt, so wie heute, wo chronischer Zeitmangel und permanent überschrittene Budgets dafür sorgen, dass der Code die einzige Dokumentation ist, die der Nachwelt übrig bleibt. Oder gibt es eine bessere Definition von Legacy-Software als: «Software, die unzureichend dokumentiert ist, und für die der ursprüngliche Programmierer nicht mehr verfügbar ist?»

Neue Entwicklungsmethoden können sich nicht nur darauf beschränken, immer wieder neue Anwendungen auf der grünen Wiese zu beginnen. Denn müssten wir alle bestehenden Softwaresysteme, die derzeit in Unternehmen ihren Dienst tun, allesamt durch neu entwickelte Services ersetzen, würde uns weder die Zeit noch das Geld dazu reichen. Diese Abhängigkeit von bestehenden Systemen, aber auch betriebskostenorientierte Businessmodelle wie Outsourcing oder Software-as-a-Service (SaaS), rücken das Thema Integration immer mehr in den Mittelpunkt. Gehen führende Branchenanalysten davon aus, dass schon heute geschätzte 30 Prozent der Gesamtausgaben in der Software-Entwicklung bei Integration und Schnittstellenpflege anfallen, so sehen dieselben Analysten voraus, dass sich in Zukunft mehr als 70 Prozent der Softwarekomponenten für Neuentwicklungen auf bereits bestehende Systeme stützen.



Die Zukunft ist hier: Model Driven Integration

Wer nun die zwei Grundelemente Model Execution und Integration zusammenführt, landet bei einem Trend, der schon heute in führenden Firmen wie Deutsche Post World Net oder der UBS als ein signifikanter Durchbruch in der Software-Entwicklung erkennbar ist: Model Driven Integration. Bei diesem Ansatz werden beide Grundelemente untrennbar miteinander verbunden. Gleichzeitig gesellt sich ein weiterer Aspekt hinzu, der einen massiven Beitrag zur Produktivitätssteigerung auf Organisationsebene und nicht nur auf der Ebene des Programmierers ermöglicht. Es ist dies eine ganzheitliche Betrachtung des Softwarelebenszyklus, von der Aufnahme der Anforderungen über das Design der einzelnen Komponenten bis hin zum Produktionseinsatz in der operativen IT und dem nachfolgenden Change Management, sowohl für technische Detailverbesserungen, als auch für umwälzende Prozessanpassungen. Es ist dieser ganzheitliche Ansatz («End-to-End Approach»), der als Summe weitere Größenordnungen an Produktivitätssteigerung ermöglichen wird.

