

Softwaremodellierung verbindet Applikationen

Integrationservices werden nicht mehr programmiert, sondern per UML (Unified Modeling Language) modelliert – so lautet die Vision des Schweizer Softwerkers E2E. Eine virtuelle Maschine, die die Modelle ausführt, fungiert dabei quasi als Enterprise Service Bus.

„Integration von Anwendungen war bisher schmerzhaft, teuer und nicht erfolgreich“, urteilt Serge Gansner, Vorstandschef des Basler Softwarehauses E2E. Denn es ist sehr komplex, Applikationen miteinander zu verbinden, die oft auf unterschiedlichen Architekturprinzipien basieren. „Es wäre schön, eine generische Integrationsplattform zu haben“, überlegte der E2E-Chef bereits in den achtziger Jahren.

Die Anforderungen daran ließen sich relativ einfach umreißen: Gansner schwebte eine Art virtueller Maschine vor, die mit den Definitionen von Integrationsdiensten gefüttert werden kann – und die in der Lage ist, diese Beschreibungen selbstständig auszuführen. Erste Versionen dieser virtuellen Maschine wurden mit textbasierten Servicedefinitionen gefüttert. Schließlich bot sich die UML als alternative Notation an. Im Unterschied zu anderen



In UML (Unified Modeling Language) entworfene Integrationsdienste führen die IT-Systeme des Schweizer Logistikkonzerns DKSH in Asien zusammen – ohne dass Code erzeugt werden muss. Foto: DKSH

UML-basierten Ansätzen ist hier jedoch nicht die Codegenerierung das Ziel, sondern die direkte Ausführung der Programmmentwürfe. Obwohl die Entwurfssprache über ein Dutzend Diagrammtypen kennt, nutzt E2E nur fünf (siehe Kasten). Der Grund: „Viele Diagrammtypen sind irrelevant für die Arbeit, die wir machen wollen“, erläutert Chris Henn, Vice President Business Development bei E2E.

Ergänzt werden die Entwürfe durch die UML Action Semantics. Diese von der Object Management Group (OMG) entwickelte Spezifikation bildet sozusagen eine Skriptsprache, die die Softwaremodelle präzisiert und dadurch ihre Ausführung unterstützt. Hilfreich ist die Sprache zum Beispiel bei Stringmanipulationen, wie Henn weiß: „Das sind kleine, haarige Detailoperationen – wenn Sie die alle mit den entsprechenden

Kästchen ausmodellieren müssen, zeichnen Sie sich zu Tode.“ Die Skriptbefehle, die die grafischen Modelle begleiten, gelten als sehr mächtig – laut Henn lassen sich zum Beispiel vier Zeilen Java-Code in einem Statement zusammenfassen. Für die Praxistauglichkeit des Bridging-Ansatzes von E2E gibt es bereits Projekterfahrungen – und auch einige Marktbeobachter halten die Vorgehensweise für viel versprechend. Eine der

umfangreichsten Implementierungen ist derzeit bei dem Schweizer Logistikdienstleister DKSH im Einsatz, der seine asiatischen Länderorganisationen in ein IT-Servicezentrum im Malaysia integriert. Die Dienste, welche die einzelnen Anwendungen verbinden, werden in UML modelliert und von der virtuellen Maschine ausgeführt. Letztere übernimmt dabei die Rolle eines Enterprise Service Bus.

Aspekt-Konzept greift bei Sicherheit

Vorteile verspricht E2E-Manager Henn speziell für die Umsetzung von Sicherheitsanforderungen. Dabei kommen ähnliche Grundsätze wie bei der aspektorientierten Softwareentwicklung zum Tragen. So werden die Geschäfts- und Sicherheitslogik der Services in getrennten Modellen implementiert: „Wenn die Security-Leute auf neue Ideen kommen“, so Henn, „dann können sie sicherstellen, dass schon beim nächsten Serviceaufruf die neuesten Richtlinien angewendet werden.“ Die jüngst angekündigte Partnerschaft mit dem Saarbrücker Geschäftsprozess-Spezialisten IDS Scheer dürfte dem E2E-Verfahren breitere Aufmerksamkeit

bescheren. Das Prozess-Tool Aris Bridge greift auf die Schweizer Technik zurück, um Geschäftsabläufe auszuführen. „Wir können zum ersten Mal die Brücke von den Prozessdefinitionen über die Orchestrierung bis zur Ausführung schlagen“, freut sich Technikexperte Henn. „Bis jetzt sind die Prozesse eigentlich nur dokumentiert und gezeichnet worden. Dann ging man zur den IT-Leuten – und die haben dann gesagt: Bleibt, wo ihr seid, ihr versteht eh nichts davon.“ fg

Darstellungen im Überblick

- E2E setzt nur fünf Diagrammtypen der UML ein:
- **Das Use-Case-Diagramm** beschreibt Anwendungsfälle und Benutzerrollen.
 - **Klassendiagramme** dienen unter anderem der Definition von Objekttypen.
 - **Aktivitätsdiagramme** bilden Prozesse und Logik ab.
 - **Deployment-Diagramme** dokumentieren Verbindungen der Dienste zum Backend, etwa zu Datenbanken.
 - **Sequenzdiagramme** dienen zum Testen der Services in verteilten Umgebungen. fg