

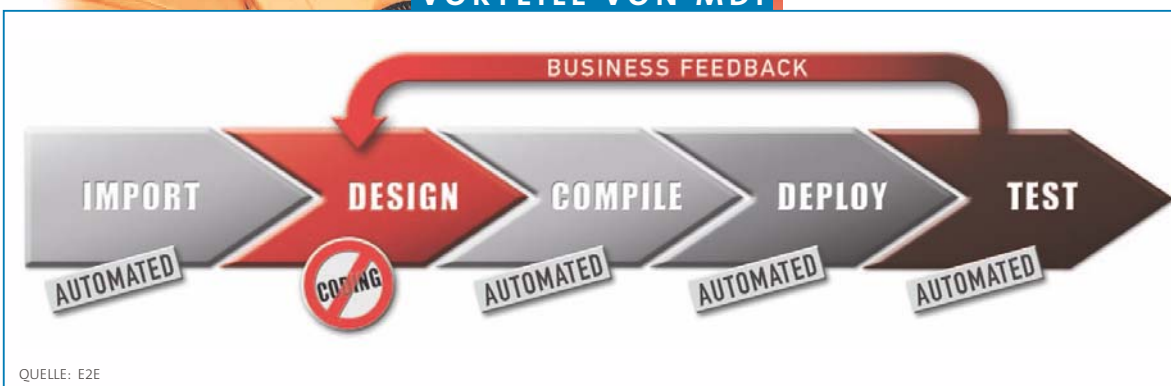
DURCH MODEL DRIVEN INTEGRATION KÖNNEN SYSTEMINTEGRATOREN PROJEKTE BESSER KALKULIEREN

Integrieren ohne programmieren

VEREINFACHT. Mit der Hilfe von modellbasierten Vorgehensweisen lassen sich zahlreiche Aufgaben automatisieren, die bei Integrationsprojekten bislang viel Zeit und Geld in Anspruch nahmen.



VORTEILE VON MDI



QUELLE: EZE

VEREINFACHT. Bei komplexen Integrationsprojekten lassen sich bis auf das Design die meisten Phasen automatisieren

Wie verdirbt man einem CIO den Tag? Ganz einfach, man spricht ihn auf sein letztes Integrationsprojekt an. Analysten zeichnen ein ebenso eindeutiges wie düsteres Bild vom Feld der Integration: In etwa der Hälfte der Projekte werden die ursprünglichen Ziele nicht erreicht, viele werden daher abgebrochen. Ebenfalls mindestens die Hälfte der Projekte werden nicht in der vorgesehenen Zeit abgeschlossen. Integrationsprojekte sind zudem so schwer zu kalkulieren, dass jedes zweite Projekt teurer wird, als anfangs veranschlagt.

Dies geht oft zu Lasten des IT-Dienstleisters. Und ebenso wie mancher CIO Integrationsprojekte so lange verschiebt und so klein hält wie möglich, denkt sich auch mancher IT-Consultant: Finger weg von komplexen Projekten! Zu oft müssen Systemintegratoren Projekte mit Verlust abschließen. Denn während die zu integrierenden IT-Komponenten immer komplexer werden, befinden sich die Margen aufgrund des weltweiten Wettbewerbs im freien Fall. Volle Kundenzufriedenheit und dennoch eine ausreichende Marge – dieser Spagat gelingt Systemintegratoren nur noch selten. Doch dabei muss es nicht bleiben.

Von der Vision zur Realität

Einen Silberstreif am Horizont zeigte bislang der Weg von der her-

kömmlichen Programmierung hin zur modellbasierten Architektur (Model Driven Architecture, MDA) auf. Durch eine erhöhte Abstraktionsebene und eine vermehrte Wiederverwendung von bereits getätigten IT-Investitionen könnte sich der Aufwand langfristig verringern lassen. Die Vision für Systemintegratoren: eine Software, die auf einem solchen Abstraktionsniveau arbeitet, dass sie bestehende Backend-Systeme weitgehend automatisch erschließt, selbstständig die gewünschten Informationen aus den beteiligten Systemen extrahiert und es dem Entwickler ermöglicht, die dabei entstehenden Service-Modelle zu abteilungsübergreifenden Prozessen zu verketteten. Diese Vorstellung ist keine Utopie mehr. Integration ohne herkömmliche Programmierung ist heute möglich.

Der Ansatz dazu heißt Model Driven Integration (MDI), also modellbasierte Integration. Was genau kann dieser Ansatz automatisieren? Teilen wir ein typisches Integrationsprojekt in Einzelschritte auf, so kristallisieren sich dabei fünf Phasen heraus:

1. Dokumentation und Definition bestehender Backend-Metadaten und -Funktionen,
2. Design der Integrations-Services und deren Orchestrierung,
3. Validierung der Service-Modelle und deren Umsetzung in eine ausführbare Form,
4. Deployment der lauffähigen Ser-

vices in die bestehende Serverlandschaft, und schließlich 5. die Qualitätssicherung und das Debugging

Jede Änderungsanfrage bedeutet einen erneuten Durchlauf der Phasen zwei bis fünf. Von diesen fünf Phasen kann MDI heute vier vollständig automatisieren. Übrig bleibt nur Schritt zwei, die Design-Phase, der kreative Kern jedes Integrationsprojektes. Die herkömmliche programmatische Implementierung oder manuelle Installation und das zeitaufwändige unkalkulierbare Debugging entfallen (siehe Grafik). Die messbare Konsequenz daraus liegt in einer massiven, um Faktoren reduzierten Gesamtkostenrechnung. Ein Beispiel: Bei einem klassischen, handgestrickten Integrationsprojekt kommen Designfehler, die erst in der Implementierungsphase erkannt werden, den Auftraggeber teuer zu stehen. Bis der Fehler identifiziert, der Code geändert, das neue Programm getestet ist, können

Tage, Wochen oder Monate vergehen. Mit modellbasierter Integration dagegen liegt der Integrations-Service komplett grafisch in UML (Unified Modeling Language) vor, und zwar nicht nur als Dokumentation, sondern als ausführbares Modell. Aufgrund des hohen Abstraktionsgrades werden logische Designfehler schon von der Software erkannt. Über eine Operation, die ins Leere geht, wird der Entwickler automatisch benachrichtigt. Über den UML Trace lässt sich der Fehler sofort lokalisieren und im UML-Modell sofort beheben. Die Validierung der Service-Modelle auf ihre logische Konsistenz erhöht in messbarer Weise die Qualität. Ein Designfehler macht sich bei MDI daher nur unwesentlich bei den Kosten bemerkbar, genauso werden Änderungswünsche innerhalb kürzester Zeit umgesetzt und live geschaltet. Die unberechenbare und ungeliebte Testphase von Integrationsprojekten kann durch Automatis-

FALLBEISPIEL
Ein Beispiel für die Vorteile einer MDI-basierten Vorgehensweise bei der Integration heterogener, mit einander nicht kompatibler IT-Infrastrukturen ist die Implementierung einer durchgängigen, länderübergreifenden Logistik-Lösung bei einer Tochter des Oetker-Konzerns. Die Agrano AG gehört zur Oetker-Gruppe und stellt Bäckerei-Halbfabrikate wie Backmittel, Vor- und Fertigmischungen für Spezialbrote sowie Sauerteigprodukte und Bio-Bäckerhefe her. Die Produkte des in Basel beheimateten Unternehmens werden in 30 Länder exportiert. Aufgabe des Integrationsprojektes war es, die bei Agrano eingesetzte Logistik-Software in das zentrale SAP-System der Oetker-Gruppe in Bielefeld zu integrieren, um einen reibungslosen Ablauf der Warenlieferungen zwischen den Oetker-Produktionsstandorten und dem Agrano-Verkaufslager zu gewährleisten. Die ursprünglich händisch programmierte Schnittstelle zwischen der Logistik-Software bei Agrano und dem zentralen SAP-System bei Oetker war bis kurz vor Projektende mit signifikanten Fehlern im Transaktionshandling behaftet. Der klassische Integrations-Ansatz führte zu einer viel zu geringen Transparenz an der SAP-Schnittstelle, verursachte hohe Kos-

ten bei Änderungsanfragen und machte es schwierig, kontinuierlich eine einigermaßen zufrieden stellende Transaktionsqualität aufrecht erhalten zu können. Durch den Einsatz einer MDI-basierten Lösung wurde die konsequente Trennung der Business-Logik von der Transaktionsverarbeitung ermöglicht und damit das Change Management im System vereinfacht. Entsprechend der MDI-Prinzipien wurden die bestehenden Systeme unverändert weiterverwendet und die Daten über MDI in der benötigten Form bereitgestellt. Ein transparentes, zertifiziertes SAP-Interface der eingesetzten MDI-Lösung E2E Bridge sorgt seitdem für die Abwicklung der täglich 2 000 bis 2 800 Transaktionen und ermöglicht gleichzeitig eine modellbasierte Harmonisierung der Metadaten. Die Transaktionsübermittlung erfolgt dabei über eine Standard-Internetverbindung. Installation und Roll-out erfolgten innerhalb von wenigen Stunden, das System läuft bis heute störungsfrei und problemlos. Der Einsatz von MDI bietet also auch bei der Integration von 3rd-Party-Softwareanwendungen in SAP-Systeme einen schnellen, kostengünstigen und vor allem zuverlässigen Weg zu einer erfolgreichen Umsetzung.

men ersetzt und damit beträchtlich verkürzt werden.

Die Modellierungssprache UML ist weltweit der am meisten verbreitete Standard für die Beschreibung von Software und Systemen. Beim Einsatz von UML im Integrationsumfeld wird die Beschreibung von Daten, Prozessen und Backend-Systemen vereinheitlicht und erleichtert damit den gesamten Integrationsprozess. Der Schlüssel zu der um Faktoren effizienteren Projektdurchführung mittels MDI ist die direkte Ausführbarkeit der UML-Modelle, die bislang nur als Vorlage für das „händische“ Programmieren oder allenfalls zur teilweisen Codegenerierung dienten. Die Codegenerierung entfällt mit MDI vollständig, ebenso das komplexe, fehleranfällige manuelle Programmieren der Service-Logik, die bei herkömmlichen Model-Driven-Development-Ansätzen (MDD) immer noch nötig ist. Bei MDI gibt es keinen Medien-

Schema dargestellt werden, das sowohl als Definition der Runtime-Umgebung, als auch als aktuelles Einsatzschema eingesetzt werden kann.

- Ein einziges Modell enthält alle Informationen vom Prozess bis zur Implementierung.

Das Paradoxe dabei ist, dass MDI die Integrationstechnologie revolutioniert – aber auf Basis von etablierten Standards. Es gibt keine neuen Plattformabhängigkeiten. Bereits existierende Software-Komponenten können vollständig wieder verwendet werden.

Neue Perspektiven für EAI

In einer zu Beginn des Jahres 2006 vorgestellten Marktstudie prognostiziert das Marktforschungsinstitut Pierre Audoin Consultants (PAC) ein Marktwachstum für den EAI-Markt in diesem Jahr von 18 Prozent. Die Analysten gehen dabei davon aus, dass das Projektgeschäft weiterhin den Hauptanteil davon einnehmen wird. Derzeit erweisen sich derartige Projekte für Systemintegratoren als schwierig und finanziell riskant. Das Projektgeschäft, das laut PAC den Hauptanteil des Integrationsmarktes ausmacht, ist somit weder für Anbieter noch Kunden zufriedenstellend.

Model Driven Integration bietet einen neuen Ansatz, der manuelle Programmierung überflüssig macht. Wie das Beispiel aus der Praxis verdeutlicht, lassen sich mit MDI Integrationsprojekte im Vergleich zu herkömmlichen EAI- und SOA-Ansätzen erheblich beschleunigen – gleichzeitig erhöht sich die Qualität, denn bei MDI wird die Dokumentation zum Code, und nicht umgekehrt. Für Systemintegratoren und Systemhäuser werden damit derartige Projekte hochattraktiv, denn sie sind in der Lage, ihre Kunden auch bei komplexen Integrationsprojekten zufrieden zu stellen, Projektermine einzuhalten und – das ist sicher die erfreulichste Neuigkeit – mit guter Leistung auch gutes Geld zu verdienen.

DER AUTOR



MICHAEL DRESCHER ist Vice President Sales Central & Eastern Europe bei E2E

bruch zwischen Dokumentation und Live-System. Dies bedeutet:

- Algorithmen, Prozesse und Workflows können gemeinsam in UML-Activity-Diagrammen zusammengefasst werden, die nicht nur für die Software-Entwickler nachvollziehbar sind, sondern auch für diejenigen, welche die Geschäftsprozesse inhaltlich verantworten.
- UML-Klassendiagramme von Daten-Modellen verschiedener unabhängiger Anwendungen können unabhängig vom ursprünglichen Format und Einsatzort in einem Dokument zusammengefasst werden und erhöhen damit die Transparenz.
- Der Zugang zu Backend-Systemen kann in einer von proprietären Protokollen, Middle-ware und Backend-Paradigmen unabhängigen Art und Weise dargestellt werden.
- System- und Service-Komponenten können in einem Architektur-