

SOA modernisiert Natural-Anwendungen

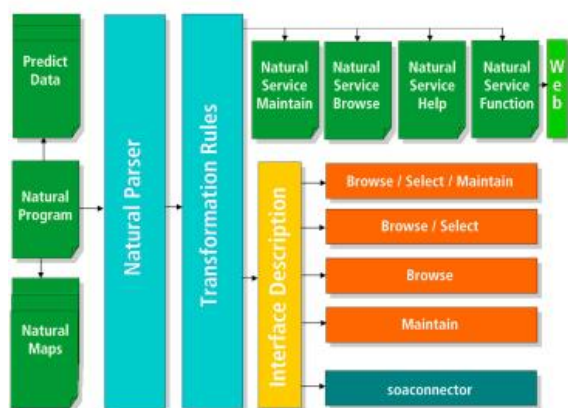
› Mit Hilfe von Generatoren lassen sich Natural- und Adabas-Programme in eine Service-orientierte Architektur überführen.

VON Thomas F. Wolf (16.01.2008 13:26:00)

Geht es um die **Modernisierung** (http://www.computerwoche.de/produkte_technik/software/540435/index.html) von Natural/Adabas-Anwendungen und den Aufbau einer **SOA** (<http://www.computerwoche.de/soa-trends/1849999/index.html>), stehen viele Unternehmen vor einer schweren Entscheidung: Bewährtes aufs Abstellgleis schieben und auf der grünen Wiese neu entwickeln oder die Altanwendungen sanft in die Moderne holen. Erstgenanntes ist in der Regel mit hohen Kosten verbunden. Letzteres schützt zwar die getätigten Investitionen, war bisher aber bei Natural/Adabas-Anwendungen aufwändig. Weil Anwender moderne Oberflächen mit einfacher und logischer Bedienung sowie Flexibilität in Bezug auf neue Services wünschen, böte sich eine Oberflächen-orientierte Modernisierung an.

Dieser Ansatz stößt bei immer mehr Nutzern auf breite Skepsis. Er führt nämlich entweder gar nicht oder nur unzureichend zu neuen Services. Ein Einstieg in die viel gepriesene **Service-orientierte Architektur** (<http://www.computerwoche.de/soa-trends/1848477/>) – sie ist der Schlüssel zur Flexibilität und Zukunftsfähigkeit der Anwendung – droht mit diesem Vorgehen zu scheitern. Die Alternative zur Oberflächen-orientierten Modernisierung von Natural-Anwendungen ist die Service-orientierte Modernisierung – also von unten nach oben. Dies setzt voraus, dass die vorhandenen Programme in Services zerlegt werden, die wiederum von anderen Systemen aufgerufen werden können.

› Inventarisieren



© Copyright by dab

Ein Service-Generator ermöglicht die teilautomatische Transformation von Altanwendungen in Services.

Entscheidend bei der Modernisierung von Natural-Anwendungen sind vor allem zwei Punkte: die vollständige Inventarisierung der Anwendung und die vollständige Definition des Zielsystems (Oberflächentypen, Servicetypen, Middleware). Grundlage für einen erfolgreichen Projektverlauf ist die gründliche Bestandsaufnahme der tatsächlich eingesetzten Programme und eine Isolierung der fachlichen Funktionen (Services) in den Programmen. Das Isolieren in grobe fachliche Funktionen reicht dafür aus. Viele IT-Verantwortliche legen hier die Latte unnötig hoch. Das Bestimmen und Isolieren der fachlichen Funktionen kann nur in enger Abstimmung mit dem Fachbereich und der IT-Abteilung erfolgen.

Die Optimierung von Geschäftsprozessen geschieht nicht innerhalb individueller Anwendungen. Daher

müssen Unternehmen bei der Inventarisierung alle intern und extern verwendeten Applikationen berücksichtigen. Durch eine vollständige Transformation der Anwendung in Services sind die Voraussetzungen erfüllt, um mit den neu gewonnen Komponenten Geschäftsprozesse zu optimieren (siehe auch: [Modernisieren statt Wegwerfen](http://www.computerwoche.de/index.cfm?pid=378&pk=549065) (<http://www.computerwoche.de/index.cfm?pid=378&pk=549065>)).

› Das Mengenproblem

Die Modernisierung von Natural-Anwendungen beinhaltet immer auch ein Mengenproblem. Die Zahl der verwendeten Programme und Masken liegt häufig bei weit mehr als 10 000. Benötigt man für die Modernisierung eines Programms fünf Projektstage, erreicht man in Summe Aufwendungen, für die in der Regel weder Budget noch Ressourcen vorhanden sind. Ein Servicegenerator ermöglicht in diesen Fällen die vollständige und teilautomatisierte Transformation der Programme in Services. Im Gegensatz zu herkömmlichen Modernisierungsansätzen werden dabei die Transformationen nicht Schritt für Schritt von einem Programmierer ausgeführt: Diese Aufgaben übernimmt der Servicegenerator vollautomatisiert für alle Programme. Dazu gehören mehrere Arbeitsschritte, darunter das vollständige Parsen aller Sourcen und deren Transformation vom Typ "Programm" in den Typ "Subprogramm" (siehe auch: [Legacy-Modernisierung auf Knopfdruck?](http://www.computerwoche.de/produkte_technik/596308/index.html) (http://www.computerwoche.de/produkte_technik/596308/index.html)).

Sourcen vom Typ Subprogramm können von anderen Systemen aufgerufen werden und somit zu Services werden. Sie werden ferner unterschieden in Sourcen vom Typ Browse und in Sourcen vom Typ Maintenance. Abhängig vom Typ werden unterschiedliche Transformationsregeln ausgeführt, beispielsweise die Transformation des Datenbereichs GLOBAL in einen Datenbereich PARAMETER, Eliminieren und Transformieren von Bildschirmausgaben, Eliminieren und Transformieren von Bildschirmsteueranweisungen, Generieren einer Logik für das Weiterlesen von Datensätzen bei Browse-Programmen oder Generieren einer Stateless-Logik bei Maintenance-Programmen. Dazu gehört auch das Ermitteln sämtlicher GETTER- (Servicefelder füllen) und SETTER-Anweisungen (GUI-Felder füllen) und das Generieren eines einheitlichen Message Handlings. Diese Transformationsregeln müssen an das jeweilige Natural-Modernisierungsprojekt angepasst werden. Der Aufwand dafür liegt in der Regel bei fünf bis 20 Projekttagen.

› Middleware

Der Servicegenerator transformiert nicht nur die Anwendungen in Services; er generiert auch die komplette Middleware für das Zusammenspiel zwischen modernen Benutzeroberflächen und Services inklusive WSDL-Beschreibungen (Web Services Description Language). Die Middleware sorgt dafür, dass die modernen Oberflächen mit den Services kommunizieren. Schließlich muss ein Klick auf einen Button der Benutzeroberfläche den Aufruf eines Service auslösen.

Model Driven Development

Softwareentwicklung mittels Model Driven Development (MDD) trennt nach Abstraktionsgrad geordnet Geschäftsmodell, fachliches Modell, technisches Modell und Programmiersprachenebene voneinander. Im Sinne eines Forward Engineering verfolgt MDD das Ziel, die fachlichen Modelle über die technische Ebene auf die Programmiersprachenebene automatisch zu transformieren. Das Modell ist damit der Ausgangspunkt für die Softwareentwicklung. Treten Änderungen auf, wird zuerst das Modell geändert. Durch automatische Transformation werden technisches Modell und Code synchron gehalten. So wird im Modell statt im Code gepflegt.

Der Charme von MDD besteht zum einen darin, auch Nichtentwickler am Entstehungsprozess zu beteiligen, weil die Spezifikation einer Software nicht in einer Programmiersprache erfolgt. Zum anderen legt man sich bei der technischen Umsetzung nicht auf eine aktuelle Variante fest, sondern kann immer wieder neue Varianten vorsehen. Das ist besonders dort wichtig, wo sich die Technik alle zwei bis drei Jahre grundlegend ändert, wie beispielsweise bei der Entwicklung von grafischen Benutzeroberflächen.

Der Servicegenerator erzeugt ferner auch die Benutzeroberflächen. Dazu generiert er Excel-Dateien. Diese werden in das Modell eines GUI-Generators importiert. Aus dem Modell heraus erzeugt dieses System wahlweise Swing- oder Ajax-Oberflächen inklusive Event-Steuerung. Innerhalb weniger Minuten lassen sich so komplette Benutzeroberflächen erstellen. Diese Art der GUI-Erzeugung mittels Model-Driven Development-Werkzeugen (siehe Kasten "Model Driven Development"), gepaart mit Automatisierung, empfiehlt sich nicht nur für die Modernisierung, sondern auch für Neuentwicklungen. Die Überführung von Natural-Anwendungen in eine SOA muss also nicht zur Sisyphosarbeit ausarten. Eine sanfte Migration in die Neuzeit ist möglich und schützt bei überschaubarem Aufwand die vorhandenen Investitionen.

› Fazit

- › Entscheidend bei der Modernisierung sind die Inventarisierung der Natural-Anwendung und die Definition des Zielsystems.
- › Bei der Inventarisierung müssen Unternehmen alle intern und extern verwendeten Anwendungen berücksichtigen.
- › Servicegeneratoren transformieren Programme in Services und generieren die Benutzeroberflächen.
- › Die Erzeugung von Benutzeroberflächen mit Model-Driven-Development-Werkzeugen, gepaart mit Automatisierung, eignet sich nicht nur für die Modernisierung, sondern auch für Neuentwicklungen

(wh)

IDG Business Media GmbH

Alle Rechte vorbehalten. Jegliche Vervielfältigung oder Weiterverbreitung in jedem Medium in Teilen oder als Ganzes bedarf der schriftlichen Zustimmung der IDG Business Media GmbH. DPA-Texte und Bilder sind urheberrechtlich geschützt und dürfen weder reproduziert noch wiederverwendet oder für gewerbliche Zwecke verwendet werden. Für den Fall, dass in Computerwoche unzutreffende Informationen veröffentlicht oder in Programmen oder Datenbanken Fehler enthalten sein sollten, kommt eine Haftung nur bei grober Fahrlässigkeit des Verlages oder seiner Mitarbeiter in Betracht. Die Redaktion übernimmt keine Haftung für unverlangt eingesandte Manuskripte, Fotos und Illustrationen. Für Inhalte externer Seiten, auf die von Computerwoche aus gelinkt wird, übernimmt die IDG Business Media GmbH keine Verantwortung.