

Entwickler suchen Integration von Objekt- und Prozessbeschreibung

## Softwariemodellierung ringt um Einheitlichkeit

**München (fg) – Die Unified Modeling Language (UML) stößt an ihre Grenzen. Kritiker werfen der Standard-Entwurfssprache vor, zur Formung von Geschäftsprozessen ungeeignet zu sein. Das Manko droht das Modellierungslager zu spalten.**

„Der UML-basierte Ansatz ist stark bei der Datenmodellierung, weist aber bei der Prozessmodellierung große Schwächen auf.“ So umreißt der Informatikprofessor Oscar Pastor von der Universität Valencia die Kritik am Standard. Pastor propagierte auf der Entwicklermesse OOP einen Modellierungsansatz, der nur Teile der UML verwendet und eine komplette Code-Generierung aus Modellen anstrebt. Aus demselben Grund plädieren Hersteller wie Microsoft und Metacase schon länger für domänenspezifische Sprachen.

Gerade zur Beschreibung von Geschäftsprozessen (BPM) existieren viele Werkzeuge und technische Ansätze. Dies führt dazu, dass in Projekten Geschäftsobjekte wie Kunden oder Verträge häufig in UML, Prozesse aber separat mit BPM-Tools modelliert werden. „Modelle für Geschäftsobjekte und Prozesse sind komplementär“, betont Marc Gille, Chef des BPM-Spezialisten Carnot: „Beide werden benötigt, um

dazustellen, was die Software tun muss. Man braucht aber Mechanismen, um diese Paradigmen zu kombinieren.“ Eine mögliche Lösung sieht Gille in der Orchestrierungssprache BPEL (Business Process Execution Language): „In ihr sind die Geschäftsobjekte zumindest rudimentär modelliert.“

Der Carnot-Chef verschweigt nicht, dass auch die UML-Version 2 Prozesse modellieren kann. Die BPMN, die offizielle Notation für BPEL-Diagramme, enthalte aber darüber hinausgehende Beschreibungselemente. Gille erwartet daher, dass BPMN bald Eingang in die UML finden wird. Pragmatisch bricht Chris Rupp, Geschäftsführerin des Beratungshauses Sophist Group, eine Lanze für den UML-Standard: „Die Aktivitätsdiagramme der UML eignen sich sehr gut für die Geschäftsprozessmodellierung, sofern man nicht den Anspruch hat, alle Notationselemente zu nutzen.“ In Projekten bereite es aber oft Probleme, wenn UML-Entwicklungswerkzeuge in den Fachabteilungen zur Analyse der Geschäftsabläufe benutzt werden: „Tools zur Workflow-Modellierung, die nicht UML-geeignet sind, haben manchmal bei der Bedienbarkeit Vorteile, weil sie für das Fachpublikum geschrieben wurden.“



Trotz unterschiedlicher Auffassungen über quelloffene Programme:

## Ideologische Gräben sind überwindbar

**München (fg) – Der Apache-Mitbegründer Brian Behlendorf propagiert die transparente Zusammenarbeit von Projektgruppen als eines der wichtigsten Open-source-Prinzipien, das sich auch auf das Geschäftsleben im allgemeinen übertragen lässt.**

„Ich habe durch Zuschauen und durch die Teilnahme am Apache-Projekt mehr über Software-Engineering gelernt als in meinem Informatikstudium“, gesteht der Open-source-Pionier freimütig: „Man kann den Dialog verfolgen, in dem sich Ent-



**Open-source-Pionier Behlendorf hält quelloffene Projekte für sehr innovativ.** Foto: Collabnet

wickler austauschen.“ Behlendorf, der heute als Chief Technical Officer für Collabnet tätig ist, weist den Vorwurf zurück, quelloffene Projekte ahmten nur kommerzielle Produkte nach und seien nicht innovationsfähig. Der Kalifornier verweist auf das Web als Beispiel für die Innovationskraft quelloffener Technologien. Außerdem nennt er Bittorrent, eine Peer-to-Peer-Software: „Konzeptionell ist es allem überlegen, was Firmen in dieser Richtung erfunden haben.“ Behlendorf gibt zu, dass ein philosophischer Antrieb von Anfang an Teil der Open-source-Community gewesen sei. Aus seiner Sicht gibt es zwei Lager – eines, das proprietäre Software für ein Übel hält und die GPL-Lizenz (General Public License) benutzt, um den Anteil freier Software zu vergrößern. Die Haltung des Apache-Lagers sei dagegen, möglichst vielen Beteiligten die Zusammenarbeit zu erlauben – selbst wenn die Resultate in ein proprietäres Produkt einfließen. Diese beiden Welten seien aber nicht unversöhnlich.

Informatikprofessor Oscar Pastor sieht den Durchbruch entwurfsbasierter Softwareentwicklung gekommen – Kritik an Schwächen des Standards:

# „Wir nutzen die UML nur eingeschränkt“

München (fg) – Die Fülle an Konzepten der Unified Modeling Language (UML) hält Professor Oscar Pastor, Leiter des Instituts für Informationssysteme der Universität Valencia, für problematisch. Um Code komplett aus Modellen generieren zu können, plädiert er für eine Konzentration aufs Wesentliche. **Würden Sie sich als Rebellen in der Modellierer-Community sehen?**

Ja. Ich halte mich schon für eine Art Rebell. Denn ich bin davon überzeugt, dass die Transformation von Modellen in Code praktikabel ist. Dahinter steckt ein grundlegendes menschliches Prinzip: Hinter jeder Programmierentscheidung steht ein Konzept. Es sollte also möglich sein, alle Konstrukte, die einem solchen Konzept zugrundeliegen, zu sammeln, zu katalogisieren und zu dokumentieren. Und wenn dieser Katalog komplett ist, müssen wir definieren, wie diese Dinge auf die jeweilige Softwareplattform abgebildet werden. Wenn wir das haben, ist der Compiler in Reichweite. Warum soll man nicht dafür streiten, dass die Idee, das Modell direkt in Code umzusetzen, praktikabel ist?

**Wo liegt der Hauptunterschied zwischen herkömmlichen Techniken und dem Ansatz, den Sie in dem Tool *Olivanova Model Executiv* realisieren haben?**

Die meisten Tools, die es derzeit gibt, könnte man modellgetrieben anstatt modellbasiert nennen. Der Entwurf treibt zwar die Entwicklung voran, aber die Programmierung bleibt die zentrale Aufgabe. In unserem Fall bildet die Modellierung die zen-

trale Aufgabe, denn wenn alle Anforderungsinformationen im Entwurf dokumentiert sind, ist man quasi fertig. Denn alle Informationen, die nötig sind, um das Modell zu kompilieren und die fertige Software zu bekommen, liegen bereits vor.

**Die UML gilt als etablierter Standard. Sollten Entwickler sich davon wirklich verabschieden?**

Wenn man modellgetrieben arbeitet, möchte man natürlich ein Stück weiter gehen. Man ist zwar versucht zu sagen: Wir können nicht alles modellieren. Aber das würde bedeuten, dass irgendwo im Code Zauberei steckt – also dass sich etwas nicht aus einem Konzept herleiten lässt. Wir fühlen uns als Teil der MDA-Community (Model Driven Architecture). Der vielleicht wichtigste Unterschied ist aber, dass wir die UML in eingeschränkter Weise nutzen, weil wir nur die Konstrukte auswählen, die wirklich nötig sind, um konzeptionelle Schemata zu bauen. Das macht es leicht-



Oscar Pastor streitet für komplette Code-Generierung aus Modellen. Foto: Care Technologies

## Meister motivieren Entwickler



Foto: Graser

München (fg) – Passend zum WM-Jahr 2006 haben die Darmstadt Dribbling Dackels, das Roboter-Fußballteam der Technischen Universität Darmstadt, bei der Entwicklertagestunde OOP in München für Stimmung gesorgt. In den Konferenzpausen war das Spielfeld der Roboterhunde stets dicht umlagert. Die spielstarken Krabblers, die in den Jahren 2005 und 2004 jeweils den Weltmeistertitel im Roboterfußball errangen, stellten sich auch menschlichen Kontrahenten – diese steuerten dann per Joystick eines der beiden Dackel-Teams. Die OOP fand heuer zum 15. Mal statt und feierte damit ein kleines Jubiläum. Zu den dominierenden Themen zählten neben serviceorientierten Architekturen auch die Softwaremodellierung sowie Qualitätsaspekte in der Entwicklung.

ter, die UML in der Praxis zu nutzen.

**UML-Modelle sind häufig daten- und objektzentriert. Wie betten Sie die Applikationslogik in Ihrem Tool ein?**

Die UML hat einen großartigen Job gemacht, weil sie eine Standardnotation für Entwürfe bietet. Aber der UML-Ansatz ist stark bei der Datenmodellierung und eher schwach in der Prozessmodellierung. Wir suchen uns – ausgehend von einer strengen formalen Basis – die grundlegenden Bausteine aus, die nötig sind, um ein Informationssystem zu spezifizieren. Dazu liefern wir eine UML-basierte Notation, um damit arbeiten zu können. Ansonsten

würde man sich in den Aberhunderterten von UML-Konzepten und Diagrammen verlieren, die für den Unterricht gut sein mögen, aber nur schwer in die Praxis umzusetzen sind.

**Was würden Sie den hartnäckigen Skeptikern entgegnen?**

Benutzen Sie einfach das Werkzeug Olivanova. Der beste Weg, um zu prüfen, ob etwas funktioniert, ist doch, es auszubasteln. Einige sehr zurückhaltende Leute wollten nicht glauben, dass die Codegenerierung, die sie beobachteten, wirklich passierte. Das war sehr kurios. Es kommt also wirklich darauf an, das Werkzeug zu benutzen.