Bei der SAP-Wartung sparen

BRANDNEU!

Modelle gegen die Kostenspirale

Modellbasierte Entwicklung setzt sich in immer mehr Industriebereichen durch und ist nun auch in unterschiedlichste Richtungen für SAP-Projekte einsetzbar. Damit können sowohl Entwicklungs- wie auch Wartungskosten deutlich reduziert werden.

Software kostet über die gesamte Lebensdauer Geld, denn zu den reinen Entwicklungskosten kommen während der Nutzungsphase erhebliche Wartungskosten hinzu. Sobald die Software im Produktivbetrieb läuft, werden anfallende Kosten unter „Softwarewartung" verbucht. Und diese Kosten haben es in sich: Sie können über den Lebenszyklus betrachtet bis zu 80 Prozent der Gesamtkosten ausmachen. Nach Berechnungen der Softwarefirma Cast Research betragen allein die Wartungskosten 3,61 US-Dollar für jede einzelne Zeile Quellcode. Interessant ist auch, dass 47 Prozent des gesamten Wartungsaufwandes in die Einarbeitung in ein Problem investiert werden müssen. 28 Prozent entfallen auf die Verifizierung und nur 25 Prozent auf die Implementierung von Änderungen.

Entscheidend für eine schlanke Kostenstruktur in der Wartung sind neben einem guten Softwaredesign auch die Qualität des Quelltextes und – natürlich – die Abdeckung der reinen Funktionalität der Software. Je höher die Qualität der Softwareentwicklung ist, desto geringer sind die Kosten für den Betrieb der Software. Um der durch schlechtes Softwaredesign und niedrige Qualität des Quelltextes entstehenden Kostenspirale zu entgehen, ist die Verbindung mit der modellbasierten Entwicklung ein erfolgversprechender und bereits durch Studien und praktische Erfolge untermauerter Ansatz. Die klare Dokumentation und Visualisierung durch die Modelle ermöglicht eine zeitsparende Ein- und Bearbeitung der Software, wodurch die Wartungskosten sinken. Auf diese Weise kann bei einem Refactoring auch von Beginn an gut konstruierte Softwarearchitektur entwickelt und laufend kontrolliert werden. Diese Möglichkeit führt zu einer längeren Lebensdauer der Software und des Systems.

>> Es entsteht eine Brücke zwischen Enterprise Architect und Abap-OO-Entwicklungen in SAP. <<

Hans Bartmann, Geschäftsführer von SparxSystems Central Europe.

Modellbasierte Entwicklung und SAP

Gobas ist ein Verbund verschiedener IT-Firmen aus dem Großraum Braunschweig und Hannover mit den Schwerpunkten SAP-Entwicklung und Mobile Development. Bis vor einiger Zeit war die modellgetriebene Entwicklung von Software im SAP-Umfeld für das Unternehmen noch kein Thema. Das wachsende Interesse daran führte aber nun dazu, mit Gobas Q.Trans für modellgetriebene Softwareentwicklung in SAP ein neues Werkzeug zu entwickeln. „Viele unserer Kunden haben SAP im Einsatz und bewegen sich gleichzeitig hin zur modellbasierten Entwicklung. Um diese beiden Welten optimal verbinden zu können, haben wir Q.Trans mit Enterprise Architect verknüpft und bieten die Lösung zur Verbindung dieser beiden Welten an", so Geschäftsführer Markus Abel von Gobas.

Mit weltweit über 350.000 Nutzern, davon etwa 60.000 im deutschsprachigen Raum, ist Enterprise Architect von SparxSystems als UML-Modellierungsplattform weitverbreitet. Der durch Gobas Q.Trans verstärkte Einsatz im SAP-Umfeld erschließt nun einen ganz neuen Anwenderkreis: „Wir freuen uns, dass Q.Trans unter Einsatz des Enterprise Architect erstellt wurde. So entsteht eine Brücke zwischen Enterprise Architect und Abap-OO-Entwicklungen in SAP, womit ein sehr interessanter neuer Markt erschlossen wird. Einmal mehr wird dabei der hohe Nutzen von Enterprise Architect bei der Optimierung und einem besseren Verständnis von komplexen Softwareprojekten deutlich", unterstreicht Hans Bartmann, Geschäftsführer von SparxSystems Central Europe.

UML verbreitet sich immer mehr

Die modellbasierte Entwicklung auf Basis der grafischen Sprache UML/SysML (Unified Modeling Language/System Engineering Modeling Language) hält derzeit in immer weiteren Bereichen der Industrie Einzug. Bei der Industrieinitiative Industrie 4.0 steht modellbasierte Entwicklung ebenfalls ganz im Mittelpunkt des Konzepts. Die Ausbreitung der modellgetriebenen Entwicklung verstärkt sich nicht zuletzt durch die stetige Verbesserung der Modellierungssprache UML/SysML.

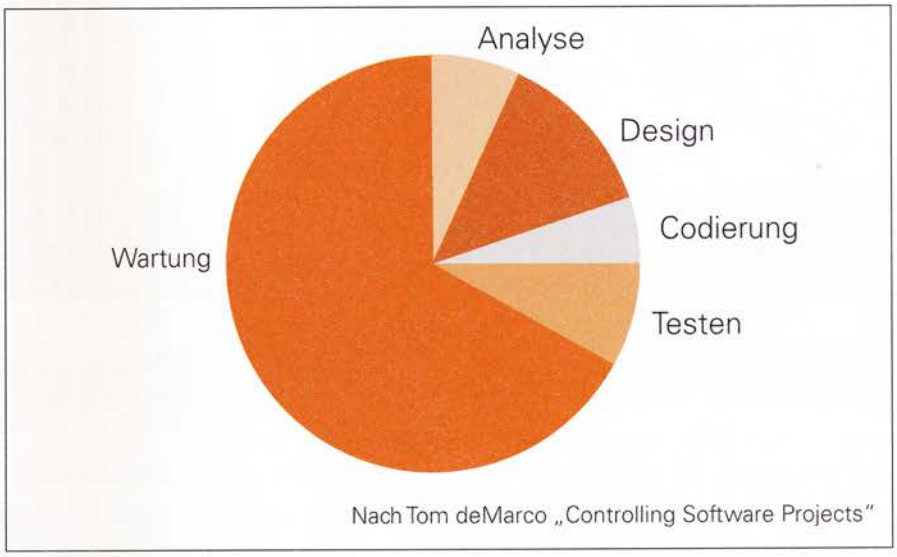
Vor 17 Jahren veröffentlichte das Industrie-Standardisierungsgremium OMG (Object Management Group) eine erste Beschreibung der UML. Inzwischen ar-

beitet man dort bereits an der Version 2.5, die den Entwicklern noch detailliertere Beschreibungsmöglichkeiten für ihre Modelle an die Hand geben wird. Mit der Einführung der SysML weitete man vor einigen Jahren die Anwendungsmöglichkeiten des modellbasierten Entwickelns auf ganze Systeme aus. Immer mehr Wissenschaftler und Anwender sehen die Zukunft der Software- und Systementwicklung ganz klar in der Arbeit an den Modellen, aus denen dann etwa Code automatisch generiert werden kann.

Kernidee der modellbasierten Entwicklung ist die Erstellung eines Funktionsmodells für das zu entwickelnde Produkt. Dieses Funktionsmodell kann bereits während der Erstellung als Basis für die Simulation dienen und nach Fertigstellung als Grundlage zur Generierung des Seriencodes. Schätzungen gehen davon aus, dass durch intensive Modellierung und Verifikation auf Modellebene Einsparungen im Bereich von 30 bis 50 Prozent möglich sind.

Im Bereich der SAP-Entwicklung kann die modellbasierte Methode sowohl für Forward als auch Reverse Engineering zum Einsatz kommen. Gerade im letztgenannten Bereich zeigte sich bisher großes Interesse von Kunden, die ihre SAP-Lösungen gezielt weiterentwickeln wollen. Mit diesem Ziel wurden verschiedene am Markt verfügbare Werkzeuge zur modellbasierten Entwicklung getestet. Das Ergebnis zeigte sehr deutlich, dass Enterprise Architect über die XMI-Schnittstelle die meisten Informationen aus den SAP-Lösungen herausholen kann.

Mit Q.Trans lassen sich aber auch UML-Modelle einfach in ein SAP-System importieren, um dort die entwicklungsrelevanten Objekte zu erzeugen, oder vorhandene Objekte nach den Modellvorgaben zu ändern („Forward Engineering"). Neben dem Quellcode gehört allerdings zu einer Software auch eine zuverlässige Dokumentation. Daher ermöglicht es die Lösung, den aktuellen Software-Entwicklungsstand in ein Modell zu exportieren und so direkt in die revisionssichere Dokumentation des im Projekt entstandenen Quelltextes einzubinden („Reverse Engineering"). Das Tool erlaubt darüber hinaus den Abgleich zwischen einem Modell und bestehendem Abap-OO-Code. Darüber hinaus bietet die neue Lösung ausgefeilte Metriken, um die Qualität von Modellen und Abap-OO-Code zu beurteilen.



Kostenarten im Verlauf eines SAP-Projekts.

Metriken zur Qualitätsbeurteilung

Belastbare Daten und Zahlen für eine Bewertung von Kosten und Qualität einer Software liefert die wissenschaftliche Methode der Softwaremessung. Diese befasst sich mit der quantitativen Behandlung von Eigenschaften von Softwareprodukten, -prozessen und -projekten. „Dabei geht es um die Nutzung von Softwaremessungen ... um bestimmte Ziele, wie beispielsweise Projektkontrolle, Fehlerreduktion oder Effizienzsteigerung, zu erreichen."

Die Transformation der Theorie in die Praxis steckt jedoch noch in ihren Anfängen. Viele Unternehmen stehen vor der Herausforderung, die richtigen Softwaremetriken zur Messung auszuwählen und zutreffende Schlüsse aus den ermittelten Zahlen zu ziehen, um so die Risiken ihrer Softwareprojekte zu minimieren. Basierend auf aktuellen wissenschaftlichen Erkenntnissen wurde in das neue Entwicklungstool Gobas Q.Trans die Anwendung verschiedenster Softwaremetriken integriert. Das ermöglicht es Abap-OO-Entwicklern, aus vorgegebenen Maßzahlen die richtigen Schlussfolgerungen zu ziehen. Gleichzeitig werden moderne Softwareentwicklungsmethoden wie agile und inkrementelle Vorgehensmethoden unterstützt. Mithilfe der eingesetzten Softwaremetriken wird der Entwicklungsstand des Softwareprojekts fortlaufend analysiert und dokumentiert. Die Fortschritte werden sichtbar, Projektvorgaben können überprüft, Fehlentwicklungen bereits im Ansatz erkannt werden.

www.sparxsystems.de
www.gobas.de

Funktionen und Leistungsumfang von Gobas Q.Trans im Überblick

- **Forward Engineering:** Generierung von Abap-Quellcode aus UML-Modellen; paketübergreifend; Generierung von Getter- und Setter-Methoden; Generierung von Kommentaren; automatischer Syntax-Check; Konsistenzcheck für DDIC-Objekte; Versionierung; Anlegen einer Import-Historie
- **Reverse Engineering:** Generierung eines UML-Modells aus Abap-Quellcode; Selektionskriterien für Objekte; Customizing für Objektnamen; paketübergreifend
- **Qualitätsmanagement:** Abgleich von Objekten und Paketen zwischen UML und SAP; detaillierte Auflistung der Attribute, Methoden und Interfaces; paketübergreifend; Beurteilung von Code mittels objektorientierter Softwaremetriken; Vererbungstiefe (DIT), Gesetz von Demeter (LoD), Kohäsion LCOM, Instabilität und diverse andere, wie z. B. McCabe- und Halstead-Metrik
- **Unterstützte Objekttypen:** Domäne, Datenelement, Struktur, Tabellentyp, Datenbanktabelle, Klasse, Interface sowie Funktionsgruppen sowie Pakete