

Dr. Johannes Martin

Ovid – Technologien zum automatisierten Software-Reengineering

Viele Firmen investieren hohe Beträge in speziell auf sie zugeschnittene Anwendungen, die den reibungslosen Ablauf ihrer Geschäftsprozesse gewährleisten sollen. Die Anforderungen, die diese Anwendungen erfüllen müssen, ändern sich laufend. Unvorhergesehene Änderungen der Anforderungen wie die Jahr 2000 Umstellung oder der Umstieg auf Internet-Plattformen machen nicht selten größere Änderungen der Anwendungen erforderlich. Das Ovid-Projekt stellt Werkzeuge zur Verfügung, die diese Änderungen erleichtern, zum Beispiel durch automatisierte Migration von Anwendungen nach Java.

Einführung

Das Jahr 2000 Problem und die Euro-Umstellung haben vielen Unternehmen deutlich gemacht, wie wichtig die Zuverlässigkeit ihrer DV-Systeme für die korrekte Ausübung ihrer Geschäftsprozesse ist. Wenn größere Änderungen der Anforderungen an die DV-Systeme anstehen, wird oft vorgeschlagen, die bisherigen DV-Systeme (Altsysteme) komplett durch neue Systeme zu ersetzen, die sowohl in der Hardware als auch in der Software die neuesten und besten Technologien verwenden.

Ein genauerer Blick auf die betroffenen DV-Systeme und Geschäftsprozesse zeigt jedoch meist recht schnell, dass dieser Weg nicht gangbar ist. Nicht nur die Software der Altsysteme ist komplex, auch die Geschäftsprozesse, die sie erleichtern oder ausführen, sind vielschichtig und anspruchsvoll. Die Implementierung dieser Prozesse in den Altsystemen ist das Ergebnis jahrelanger Zusammenarbeit von Software-Entwicklern mit Kennern der Geschäftsprozesse und selten ausführlich und zusammenhängend dokumentiert. Die Altsysteme komplett durch neue Systeme zu ersetzen, hiesse also, das Ergebnis jahrelanger Arbeit in einer kurzen Zeit zu reproduzieren.

Selbst wenn ausreichend erfahrene Entwickler und Branchenkenner verfügbar sind, ist es riskant ein großes DV-System durch ein neues zu ersetzen.

Mehr Erfolg verspricht die Anpassung des Altsystems an die neuen Gegebenheiten. Das Risiko dieses Prozesses lässt sich noch verringern, indem man jeweils nur kleine Teile des Altsystems an die neuen Anforderungen anpasst oder ersetzt. Diese kleinen Teile lassen sich einfacher testen, da im Falle einer Fehlfunktion die Fehlerquelle leicht auf das ausgetauschte Teilstück des Gesamtsystems eingegrenzt werden kann.

In vielen Fällen lassen sich die notwendigen Änderungen mittels speziell geschriebener Programme automatisieren. Durch die Automatisierung können zum einen die Änderungen schneller vollzogen und dokumentiert werden. Zum anderen können Flüchtigkeitsfehler vermieden werden, die einem Programmierer bei der Ausführung vieler ähnlicher Änderungen unterlaufen könnten.

Das Ovid-Projekt stellt Werkzeuge zur Verfügung, mit deren Hilfe Programme zur automatisierten Anpassung von DV-Systemen erstellt werden können. Eines der Ziele ist es dabei, die Werkzeuge so flexibel zu gestalten, dass sie in vielen verschiedenen Situationen einsetzbar sind, dabei aber trotzdem genügend Funktionalität zur Verfügung stellen, um ihre Anpassung an die jeweiligen Gegebenheiten sinnvoll und lohnend zu machen.

Motivation

In einem früheren Projekt (Ephedra) befassten wir uns mit den Problemen und möglichen Lösungswegen der Einbindung von C und C++ Programmen ins Internet, um deren Nutzung zum Beispiel über einen Webbrowser zu ermöglichen. Für eine bestimmte Gruppe dieser

Programme erkannten wir, dass die Migration (Umstellung) auf die Programmiersprache Java eine gute Lösung ist. Wir untersuchten und entwickelten daher Strategien zur Durchführung der Migration von C und C++ nach Java und stellten Programme her, die diese Migration automatisieren.

Im Laufe dieser Arbeiten gewannen wir einige interessante Erkenntnisse: Unsere Strategien zur Umwandlung von Programmquelltext von C/C++ nach Java machten teilweise eine Umstrukturierung des Programms notwendig. Das Ergebnis der Umwandlung war ein besser strukturiertes Programm. Da Java im Vergleich zu C und C++ eine sehr restriktive und sichere Programmiersprache ist und daher während der Ausführung eines Programmes viele Fehlersituationen bemängelt, die in anderen Programmiersprachen nur schwer zu erkennen sind, wurden bei der Umwandlung von Programmen nach Java oft Fehler in den ursprünglichen C/C++ Programmen gefunden.

Im Ovid-Projekt möchten wir die Ergebnisse des Ephedra-Projekts in einem allgemeineren Rahmen, also nicht nur zur Migration von C/C++ nach Java, darstellen und weiterentwickeln. Da der Aufwand einer Programmumwandlung ohne genaue vorherige Analyse des DV-Systems nur schwer abzuschätzen ist, werden in Ovid auch Analysewerkzeuge integriert. Um eine hohe Flexibilität zu erreichen, stellt Ovid einzelne kleine Komponenten zur Verfügung, aus denen nach dem Baukastenprinzip leistungsfähige Werkzeuge zur automatisierten Analyse und Umwandlung von DV-Systemen erstellt werden können.

Möglichkeiten der Programmumwandlung

Programmumwandlungen lassen sich in verschiedene Kategorien einteilen. Eine

Auswahl dieser Kategorien wird in den folgenden Absätzen vorgestellt.

Austausch der Benutzeroberfläche

Die Benutzeroberfläche eines DV-Systems spielt eine bedeutende Rolle für den Erfolg dieses Systems. Ist sie funktional und ansprechend, dann akzeptieren die Benutzer ein System schnell. Ein System, das in der Bedienung kompliziert ist, wird dagegen oft verworfen, auch wenn es sehr leistungsfähig ist. DV-Entwickler sind daher bestrebt, Ihren Kunden stets eine Benutzeroberfläche zu bieten, die den neuesten Standards und Trends genügt.

In der Vergangenheit haben wir verschiedene Typen von Benutzeroberflächen kennengelernt. Auf traditionellen Großrechnersystemen wurden und werden transaktionsorientierte Benutzeroberflächen eingesetzt, d.h. Benutzer füllen Formulare aus und senden dieses zur Bearbeitung an den Großrechner.

Mit den ersten Personalcomputern kamen ablaufgesteuerte Benutzeroberflächen auf den Markt. Typische DV-Systeme auf diesen Rechnern präsentierten und verlangten Informationen genau dann, wenn sie dem Programm verfügbar wurden oder vom Programm benötigt wurden. Der Benutzer hatte normalerweise keinen Einfluss darauf, in welcher Reihenfolge er die nötigen Informationen an das Programm weitergab.

Die graphischen Benutzeroberflächen heutiger DV-Systeme werden von Ereignissen gesteuert. Wie bei transaktionsorientierten Benutzeroberflächen werden dem Benutzer Formulare und Menüs angeboten, diese stehen jedoch zu jedem Zeitpunkt unter der Kontrolle des Programms, das den Benutzer deshalb beim Gebrauch der Formulare unterstützen und etwaige Fehleingaben sofort erkennen kann.

Internet-basierte DV-Systeme erforderten ursprünglich einen Rückschritt zur transaktionsorientierten Benutzeroberfläche. Durch die Einführung von Programmiersprachen wie Java und JavaScript, die von Webbrowsern unterstützt werden, sind sie von der Bedienung heute ereignisgesteuerten DV-Systemen ähnlich - in der Programmierung unterscheiden sie sich aber deutlich

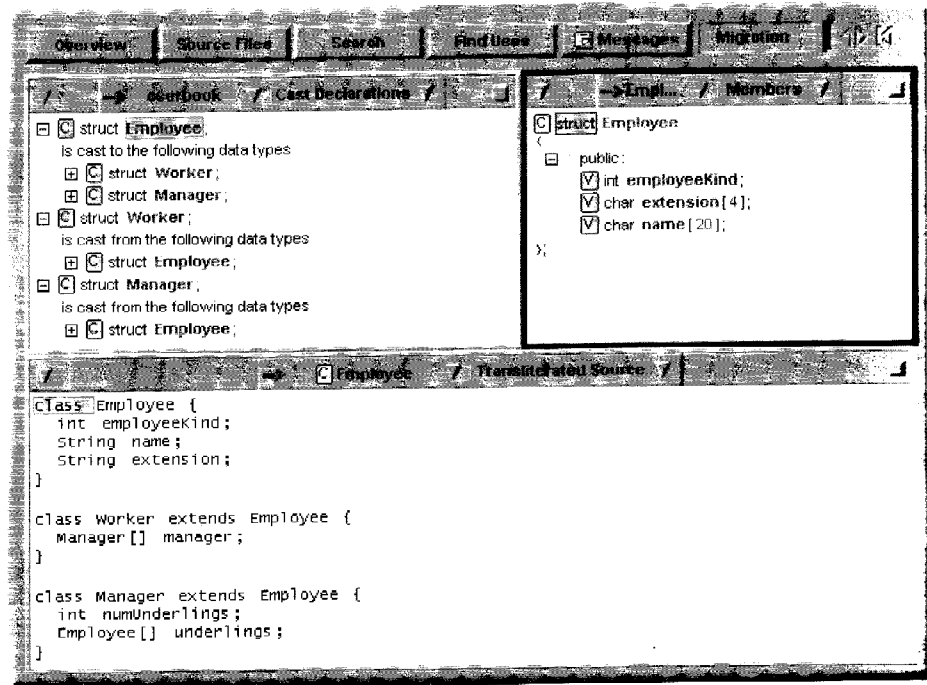


Abb. 1: Ein Blick auf das Ephedra-Migrationswerkzeug: C-Datenstrukturen werden analysiert und in Java-Datenstrukturen umgewandelt.

sowohl von transaktions- wie auch ereignisgesteuerten DV-Systemen.

Da die Art der Kommunikation mit dem Benutzer so grundsätzlich verschieden ist, erfordert die Umstellung von einer Benutzeroberfläche auf eine andere oft eine große Umstrukturierung des betroffenen DV-Systems. Programmcode für Benutzeroberfläche und Geschäftsprozesse ist nicht selten eng verflochten und muss vor einer Umstellung getrennt werden. Software-Engineering Werkzeuge wie die des Ovid-Projekts können bei dieser Aufgabe helfen.

Aktualisierung von Programmbibliotheken und Programmiersprachen

Sicherheitslücken in älteren Programmbibliotheken oder eine Aufrüstung auf neue Hardware erfordern gelegentlich den Umstieg auf neue Versionen der verwendeten Programmbibliotheken oder Programmiersprachen. Diese sind nicht immer mit den bisher eingesetzten Versionen kompatibel, so dass Änderungen am Programm Quelltext vorgenommen werden müssen. Oft sind viele kleine Änderungen nötig, die ein Programmierer nur mit viel Zeit und Mühe, ein automatisiertes Werkzeug jedoch sicher und schnell ausführen kann. Solche Werkzeuge werden mit den

Komponenten des Ovid-Projektes einfach zu erstellen sein.

Umstieg auf eine andere Programmiersprache

Manchmal wird es notwendig, ein Programm teilweise oder komplett von einer Programmiersprache auf eine andere umzustellen. Beim Umstieg auf neue Hardware kann es vorkommen, dass die ursprüngliche Programmiersprache nicht mehr unterstützt wird. Soll ein DV-System im Internet verfügbar gemacht werden, bietet die Umstellung auf die Programmiersprache große Vorteile. Manche Programmiersprachen sind nur für kleinere Projekte geeignet, so dass ein Umstieg notwendig wird, wenn das Projekt wächst und für eine Implementation in der ursprünglichen Programmiersprache zu komplex wird. Das Ephedra-Projekt befasste sich mit dem Spezialfall des Umstiegs von C/C++ auf Java. Im Ovid-Projekt wird dieses Thema allgemeiner betrachtet werden.

Technologien

Ephedra nutzt den IBM VisualAge C++ Compiler, um den zu bearbeitenden C/C++ Quelltext zu analysieren. Ursprünglich war dies ein großer Vorteil in der Entwicklung von Ephedra: IBM

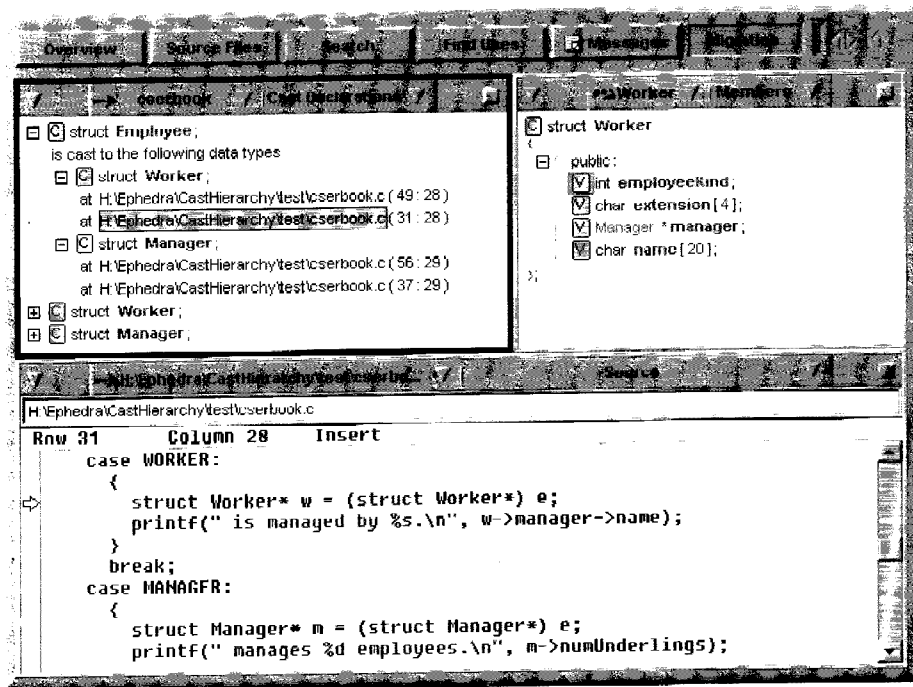


Abb. 2: Umwandlungen zwischen Datentypen werden in C und Java unterschiedlich vorgenommen. Das Ephedra-Migrationswerkzeug findet Typumwandlungen, die in Java Probleme bereiten können.

VisualAge C++ stellt wohl durchdachte und ausreichend dokumentierte Schnittstellen zur Verfügung, die eine einfache und effiziente Programmierung der Ana-

lyse erlauben. Da IBM VisualAge C++ jedoch nur noch auf IBM AIX Betriebssystemen erhältlich ist, schränkt dies den Einsatz von Ephedra heute stark ein.

Im Blick auf diese Erfahrungen entschieden wir uns im Ovid-Projekt für den Einsatz von Standardprotokollen für Softwareanalyse und -umwandlung, die mittlerweile entwickelt worden sind. Zu nennen ist hier vor allem das GXL Dateiformat, das den Austausch von Informationen über Programme zwischen Werkzeugen unterschiedlicher Art und Herkunft erlaubt. Es existieren auch bereits einige GXL-kompatible Produkte, die die Stelle von IBM VisualAge C++ im Ovid-Projekt einnehmen können. GXL wird auch zur Kommunikation zwischen den Ovid-Komponenten genutzt. Da GXL unabhängig von der Programmiersprache der analysierten DV-Systeme ist, fördert sein Einsatz die angestrebte Universalität der Ovid-Komponenten.

Kollaboration und Meinungsaustausch haben einen hohen Stellenwert in Wissenschaft und Forschung. Das Ovid-Projekt wird als Open-Source Projekt durchgeführt, um anderen Forschern und industriellen Anwendern schon in den Anfangsstadien seiner Entwicklung zu erlauben, erste Ergebnisse zu evaluieren und Rückmeldungen zu geben.

Der Autor

Johannes Martin, Jahrgang 1971, studierte Mathematik an der Johannes-Gutenberg Universität Mainz und später Informatik an der Northern Illinois University (DeKalb, IL, USA) mit Abschluss als Master of Science (1996). Als Doktorand an der University of Victoria (BC, Kanada) beschäftigte er sich eingehend mit Techniken zum Software Reverse-Engineering und Reengineering. Während mehrerer Forschungsaufenthalte im IBM Toronto Laboratory nahm er maßgeblich an der Reali-

sierung und Durchführung eines Projektes zur Migration eines IBM Softwareproduktes von PL/I nach C++ teil. In seiner Dissertation (2002) entwickelte und implementierte er Strategien zur Migration von C/C++ Programmen nach Java im Ephedra-Projekt. Seit Oktober 2002 arbeitet Johannes Martin an der Migration statistischer Anwendungen des Psychologischen Instituts der Johannes-Gutenberg Universität Mainz zur Nutzung im Internet und im Ovid-Projekt.

Ansprechpartner

Dr. Johannes Martin
Johannes-Gutenberg
Universität Mainz
Psychologisches Institut
Abteilung Methodenlehre
und Statistik
Staudingerweg 9
D-55099 Mainz
Telefon:
++49 (0) 6131-39-25069
Telefax:
++49 (0) 6131-39-24341

E-Mail: jmartin@tigris.org
Internet: <http://ovid.tigris.org>