GUPRO
–
Generische Umgebung zum Programmverstehen (Schlußbericht)

September 1998

Prof. Dr. Jürgen Ebert,
Andreas Winter
Institut für Softwaretechnik
Universität Koblenz-Landau
Koblenz

Hans H. Stasch
Volksfürsorge Unternehmensgruppe
Hamburg

Rainer Gimnich
IBM Deutschland Informationssysteme GmbH
Institut für Datenbanken und Software Engineering
am Wissenschaftlichen Zentrum
Heidelberg

Das diesem Bericht zugrundeliegende Vorhaben wurde mit Mitteln des Bundesministeriums für Bildung, Wissenschaft, Forschung und Technologie unter dem Förderkennzeichen **01 IS 504** gefördert. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autoren.

Vorwort 3

Aufgabenstellung 3

Problemlage 3

Anwendungsbezug 3

Lösungsidee 4

Voraussetzungen 5

Projektpartner 5

Volksfürsorge Unternehmensgruppe 5

Institut für Softwaretechnik der Universität Koblenz-Landau 5

Wissenschaftliches Zentrum der IBM Deutschland 5

Technische Rahmenbedingungen 6

Planung und Ablauf des Vorhabens 6

Zusammenarbeit 6

Projektphasen 7

Arbeitspakete 7

Zusammenarbeit mit anderen Stellen 7

Erzielte Ergebnisse 8

Übersicht 8

Wissenschaftlicher und technischer Ausgangspunkt 9

Graphentechnologie 10

Komponenten des GUPRO-Systems 10

Die GUPRO-Benutzungsschnittstelle 11

Die GUPRO Anfragekomponente 11

Die GUPRO-Parsingkomponente 12

Das GUPRO-Repository 13

Der aktuelle GUPRO-Prototyp 13

Voraussichtlicher Nutzen 16

Präsentationen und Publikationen 17

Literatur 18

Vorwort

Der vorliegende Bericht ist der Abschlußbericht zu dem vom BMBF unter dem Förderkennzeichen **01 IS 504** geförderten Verbundvorhaben GUPRO (Generische Umgebung zum Programmverstehen) gemäß BNBest-BMBF Ziffer 3.2, Stand Januar 1996.

Er wird ergänzt durch den **Erfolgsbericht** gemäß BNBest-BMBF Ziffer 3.2.3 vom 30. September 1998 und die **wissenschaftlichen Dokumentation** des Projektes in Buchform [Ebert et al., 1998a]:

> "GUPRO - Generische Umgebung zum Programmverstehen",
> herausgegeben von J. Ebert, R. Gimnich, H.H. Stasch und A. Winter,
> erschienen im Fölbach-Verlag, Koblenz 1998,

Das Buch [Ebert et al., 1998a] ist als der wissenschaftliche Schlußbericht des Projekts aufzufassen. Um überflüssige Redundanzen der Darstellung zu vermeiden, wird an zahlreichen Stellen dieses Berichts auf das Buch verwiesen.

Aufgabenstellung

Problemlage

Die Wartung und Weiterentwicklung großer Anwendungssysteme ist häufig geprägt durch gewachsene Programmstrukturen, unvollständige Dokumentation und schwer aufdeckbare Abhängigkeiten. Dadurch wird das **Verstehen** der Anwendungen ihrer Komponenten, ihrer Zusammenhänge und ihrer externen Schnittstellen - in unterschiedlichem Detaillierungsgrad - zu einer zentralen Aktivität in der Software-Evolution.

Die Notwendigkeit der Methoden- und Werkzeugunterstützung in der Softwarewartung zeigt sich deutlich bei den zur Zeit in allen Unternehmen anstehenden Aufgaben im Zusammenhang mit dem Jahrhundertwechsel (Jahr-2000-Problem) und der Einführung der gemeinsamen Währung (Umstellung auf Euro). Die Bearbeitung dieser, wie auch anderer Softwarewartungsaufgaben, sollte durch individuelle, auf den jeweiligen Problemkontext abgestimmte **Werkzeuge** unterstützt werden.

Im Projekt GUPRO wurde eine **Generische Umgebung zum Programmverstehen**, d.h. ein benutzerkonfigurierbares Programmverstehenswerkzeug entwickelt. Neben der Unterstützung beim Verstehen einzelner Programme wurde auch das Nachvollziehen und Verstehen heterogener Software beliebiger Sprachen (Programmiersprachen, Anfragesprachen, Sprachen der "4. Generation") berücksichtigt.

Wie im Konkreten die Idee für das Projekt GUPRO entstand ist in [Ebert et al., 1998a, Kapitel 1] berichtet.

Anwendungsbezug

Als **beispielhafte Systemumgebung** wurde die des Anwendungspartners Volksfürsorge Unternehmensgruppe betrachtet. Sie ist grob charakterisiert durch

- mehrere Betriebssystem-Plattformen: MVS, DOS, OS/2;
- mehrere Sprachen (Programmiersprachen, Generator-Sprachen, Anfragesprachen, Job-Control-Sprachen): COBOL, DELTA-COBOL, PL/I, DIABOLO, Assembler, CSP, SQL, MVS-JCL;
- mehrere Arten der Interaktionsbeschreibungen: MFS, CSP-Maps;
- mehrere Arten der Datenhaltung: DB2, IMS/DB, VSAM-Dateien, sequentielle Dateien.

Eine Betrachtung des **Mengengerüsts** allein der ausführbaren Programme zeigte, daß die COBOL-, DELTA-COBOL, Assembler- und PL/I-Programme sowie die CSP-Applikationen zusammen über 16.000 Programmeinheiten bilden. Diese Anwendungslandschaft ist laufend zu warten und weiterzuentwickeln. und das in einem sich rasch entwickelnden Markt.

Bei diesen hohen Anforderungen an die Flexibilität der Systeme, der großen Zahl von Anwendungskomponenten und dem hohen Zeitdruck in der Entwicklung ist es nicht verwunderlich, daß vorhandene Programm-Quelltexte trotz besseren Wissens meist schwach dokumentiert oder gar unkommentiert blieben. Zudem führte die korrektive und adaptive Wartung der Software mit der Zeit nahezu zwangsläufig zu **weniger gut strukturierten** Programmtexten, da der ursprüngliche Programmentwurf durch die notwendigen Korrekturen, Anpassungen und Erweiterungen erheblich verändert wurde. Diese Situation erschwert das Verstehen und die Wiederverwendung von Programmen bzw. Programmteilen.

Lösungsidee

Das Projekt GUPRO zielte daher auf eine **flexible, problemangemessene Unterstützung** der Wartung und Weiterentwicklung auch heterogener Anwendungslandschaften. Dies wurde durch die problembezogene Modellierung der Anwendungszusammmenhänge und die darauf aufbauende Generierung von Werkzeugen erreicht.

Insgesamt besteht ein **GUPRO-Werkzeug** aus

- **Parsingwerkzeugen**, die Programmtexte unterschiedlicher Programmiersprachen in interne, allgemein verarbeitbare Strukturen übersetzen,
- **Analysewerkzeugen**, die auch globale versteckte Abhängigkeiten zwischen Bausteinen erkennen. Dieses sind
    - **Anfragewerkzeuge**, die es erlauben, verborgene Informationen aus dem gesamten System auf eine kompakt gestellte Frage hin zusammengefaßt und übersichtlich strukturiert zu liefern, und
    - **Navigationswerkzeuge**, mit denen speziellen Abhängigkeiten interaktiv nachgegangen werden kann.

Diese Werkzeuge wurden in einer integrierten, auf die Anforderungen der Anwender hin konfigurierten **Wartungsumgebung** zusammengefaßt.