

Objektorientiertes Reverse Engineering

René Klösch Harald Gall

Objektorientiertes Reverse Engineering

Von klassischer
zu objektorientierter Software

Mit 53 Abbildungen



Springer

Dr. René Klösch/Dr. Harald Gall
Technische Universität Wien
Institut für Informationssysteme
Abteilung für Verteilte Systeme
Argentinerstraße 8/184-1
A-1040 Wien

ISBN-13: 978-3-540-58374-5 e-ISBN-13: 978-3-642-79221-2
DOI: 10.1007/978-3-642-79221-2

CIP-Aufnahme beantragt

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der jeweils geltenden Fassung zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

© Springer-Verlag Berlin Heidelberg 1995

Die Wiedergabe von Gebrauchsnamen, Handelsnamen, Warenbezeichnungen usw. in diesem Werk berechtigt auch ohne besondere Kennzeichnung nicht zu der Annahme, daß solche Namen im Sinne der Warenzeichen- und Markenschutz-Gesetzgebung als frei zu betrachten wären und daher von jedermann benutzt werden dürften.

Umschlaggestaltung: Künkel + Lopka, Ilvesheim
Satz: Reproduktionsfertige Vorlagen von den Autoren
SPIN 10475883 33/3142 – 5 4 3 2 1 0 – Gedruckt auf säurefreiem Papier

Vorwort

Die Software-Entwicklung war bisher vor allem auf die Neuerstellung von Software-Systemen konzentriert. Daneben hat in den letzten Jahren jedoch die Wartung von insbesondere großen, gewachsenen Systemen zunehmend an Bedeutung gewonnen.

Die steigende Bedeutung der Wartung solcher gewachsenen Applikationen, sowie deren Komplexität hat zur Entwicklung verschiedener, sogenannter Re-Engineering oder Reverse-Engineering-Ansätze sowohl in der Forschung als auch in der Praxis geführt.

Mit der zunehmenden Verbreitung der objektorientierten Konzepte werden auch die Bereiche des Re-Engineerings bzw. Reverse Engineerings von diesen erfaßt.

In diesem Buch versuchen wir nicht nur eine Einführung in die vielfältige Terminologie im Bereich des Reverse Engineerings zu geben, sowie einen Überblick über den aktuellen Stand dieses Bereichs der Software-Technologie zu präsentieren, sondern zeigen darüber hinaus einen konkreten objektorientierten Reverse-Engineering-Prozeß – untermauert durch ein praktisches Beispiel. Der in diesem Buch vorgestellte objektorientierte Reverse-Engineering-Prozeß ermöglicht – unter Einbeziehung eines menschlichen Experten – die Transformation von gewachsenen Applikationen in eine objektorientierte und somit leichter wartbare und für die Wiederverwendung besser geeignete Architektur.

Diesem Buch liegt eine praktische Fallstudie zugrunde, die ein Fakturierungssystem für das Baunebengewerbe, speziell bei der Kies- und Sandgewinnung, realisiert. IFAS (Integriertes Fakturierungs- und Abrechnungssystem) wurde im Jahre 1991 entwickelt und befindet sich seitdem erfolgreich im Einsatz. Im Laufe der Zeit wurde das Programm durch Wartungstätigkeiten unübersichtlich (seitens Größe und Komplexität) und somit schwer wartbar.

Diese Fallstudie dient als Grundlage für die Beschreibung des objektorientierten Reverse-Engineering-Prozesses und versucht, die einzelnen Schritte wie auch die Ergebnisse mittels Beispielen aus dem IFAS-System anschaulich zu demonstrieren.

Dieses System wurde aus mehreren Gründen als Fallstudie für den COREM-Prozeß herangezogen: Die Größe und Komplexität des Programms ist mit ca. 30 K LOC repräsentativ für eine solche Transformation, die Verständlichkeit ist, da es sich um eine kommerzielle Anwendung mit weit verbreitetem Charakter handelt, für einen breiten Leserkreis gegeben und es handelt sich um ein im Laufe der Zeit gewachsenes System.

Das Buch führt durch seine Kapitelstruktur von den allgemeinen Grundlagen und Konzepten des Reverse Engineering und der objektorientierten Anwendungsmodellierung, sowie des objektorientierten Entwurfs hin in die ausführliche Beschreibung der einzelnen Schritte eines objektorientierten Reverse-Engineering-Prozesses.

In Kapitel 1 wird eine Begriffsbestimmung für mit Reverse Engineering verwandte Begriffe durchgeführt, sowie die allgemeinen Vorteile und Einsatzbereiche von Reverse Engineering beschrieben. Es wird dabei auch kurz auf die Probleme bei der Software-Wartung und der Wiederverwendung eingegangen, die die eigentliche Ursache für das wachsende Interesse der Entwickler und Wartungsingenieure an solchen Reverse-Engineering-Methoden und -Werkzeugen darstellen.

Design Recovery stellt eine spezielle Methode im Rahmen des Reverse Engineerings dar, welches für das objektorientierte Reverse Engineering von besonderer Bedeutung ist. Dem Themenkreis des Design Recovery wird daher im Kapitel 1 breiter Raum gewidmet, was für das Verständnis der nachfolgenden Kapitel wichtig ist.

Das folgende Kapitel 2 führt in den Bereich der objektorientierten Analyse und des objektorientierten Entwurfs ein, soweit dies für das Verständnis des objektorientierten Reverse Engineering erforderlich ist. Darüber hinaus beschreibt dieses Kapitel die Fallstudie in Form einer einfachen Systemspezifikation. Den Abschluß des Kapitels bildet die Anwendung der objektorientierten Analyseverfahren von Coad und Yourdon [CY91a] auf die Fallstudie.

Kapitel 3 gibt einen Überblick über eine konkrete objektorientierte Reverse-Engineering-Methode, die sogenannte *Capsule Oriented Reverse Engineering Method* (COREM) und versucht, deren Zielsetzungen hinsichtlich Software-Wartung und Software-Wiederverwendung zu erläutern. Dabei

werden sowohl die Voraussetzungen für eine erfolgreiche Anwendung von COREM als auch die Einschränkungen diskutiert. Da COREM nicht vollständig automatisierbar ist, wird in diesem Abschnitt auch die Rolle des sogenannten Reuse Engineers und dessen notwendiges Wissen und Fähigkeiten definiert.

Die folgenden Kapitel beschreiben ausführlich die Durchführung eines objektorientierten Reverse-Engineering-Prozesses mittels COREM und verdeutlichen diesen jeweils durch Anwendung auf die Fallstudie.

Kapitel 10 faßt die Ergebnisse dieser objektorientierten Reverse-Engineering-Methode zusammen und gibt einen kurzen Ausblick auf eine mögliche Teilautomatisierung der *Capsule Oriented Reverse Engineering Method*.

Wir möchten allen Beteiligten danken, die an der Entstehung dieses Buches in unterschiedlichster Weise mitgewirkt haben. Besonderen Dank möchten wir Prof. Roland Mittermeir, Universität Klagenfurt, sowie unserem Kollegen Robert Barta für viele interessante und zielführende Diskussionen und Anregungen aussprechen. Ein Dank geht auch an Prof. Helmut Kerner für seine wohlwollende Unterstützung. Weiters gebührt unserem unermüdlchen L^AT_EX-”Guru” Manfred Hauswirth sowie Martin Kufner und Markus Schranz Dank für ihre Bemühungen um stilistische und layouttechnische Fragen. Nicht zuletzt möchten wir uns beim Springer-Verlag, insbesondere dem Team von Herrn Rossbach, für die freundliche und unkomplizierte Unterstützung bei der Erscheinung dieses Buches bedanken.

Wir danken all unseren Lieben und Bekannten, die uns während der schweren Zeit der Entstehung dieses Buches unterstützt und ermuntert haben und auf viele gemeinsame Abende und Wochenenden verzichteten. Wir hoffen, es hat sich gelohnt.

Wien, im Februar 1995

René Klösch,
Harald Gall

Inhaltsverzeichnis

1	Reverse Engineering	1
1.1	Einführung in das Reverse Engineering	1
1.2	Design Recovery	17
2	Objektorientierte Anwendungsmodellierung	41
2.1	Objektorientierte Analyse und Design – Eine Einführung .	43
2.2	Objektorientierte Systemanalyse	45
2.3	Grundlegende Konzepte der objektorientierten Analyse . .	48
2.4	Modellierungsansätze für das objektorientierte Anwen- dungsmodell	61
2.5	Anwendung von OOA auf die Fallstudie IFAS	62
2.6	Objektorientierter Systementwurf	77
3	COREM – Objektorientiertes Reverse Engineering	91
3.1	Einführung und Motivation	91
3.2	Die Systemtransformation	94
3.3	Zielsetzungen von COREM	101
3.4	COREM im Vergleich zu anderen Reverse-Engineering- Methoden	102
3.5	Voraussetzungen und Einschränkungen von COREM . . .	103
3.6	Die Rolle des Reuse Engineers	106

4	Die reverse Generierung von E/R-Diagrammen	109
4.1	Einführung	109
4.2	Notation	110
4.3	Identifikation der Entitäten	112
4.4	Identifikation von Relationen	123
4.5	Bestimmung der Relationen	124
4.6	Fallstudie IFAS	151
5	Generierung eines statischen objektorientierten Anwendungsmodells	165
5.1	Einführung	165
5.2	Unterschiede zwischen E/R-Diagramm und den ooAMs . .	166
5.3	Der Umformungsprozeß	168
5.4	Fallstudie IFAS	183
6	Das reverse generierte objektorientierte Anwendungsmodell	191
6.1	Identifikation der Services von Objekten	193
6.2	Kategorisierung der Service-Kandidaten	195
6.3	Die Service-Relationen	203
6.4	Fallstudie IFAS	206
7	Objekt-Mapping zwischen den Anwendungsmodellen	211
7.1	Unterschiede zwischen forward und reverse ooAM	212
7.2	Abbildung zwischen den Modellen	214
7.3	Auflösung von Mehrdeutigkeiten bei Service-Kandidaten .	232
7.4	Festlegung von Services des Ziel-ooAMs	234
7.5	Behandlung nicht abbildbarer Elemente	245
7.6	Beschreibung des Ziel-Anwendungsmodells	249
7.7	Fallstudie IFAS	250

8	Generierung eines vereinfachten objektorientierten Ziel-Entwurfs	261
8.1	Die Generierung eines vereinfachten objektorientierten Entwurfs	262
8.2	Die Behandlung der Mensch-Maschine-Schnittstelle	265
8.3	Abbildung zwischen Analyse- und Design-Objekten	283
8.4	Ergebnisse nach der Abbildung zwischen Analyse- und Design-Objekten	287
8.5	Fallstudie IFAS	288
9	Organisation des Ziel-Systems	293
9.1	Adaptierung der Objekte	294
9.2	Adaptierung des prozeduralen Rest-Systems	308
9.3	Resultierende Organisation des Ziel-Systems	313
9.4	Fallstudie IFAS	315
10	Zusammenfassung und Ausblick	323
	Glossar	331
	Literaturverzeichnis	339
	Index	350

Abbildungsverzeichnis

1.1	Forward und Reverse Engineering	2
1.2	Kostenverteilung im Software-Life-Cycle	3
1.3	“quick-fix”-Wartungsmodell	4
1.4	“iterative enhancement”-Wartungsmodell	5
1.5	Aktivitäten in der Software-Wartung	6
1.6	Verteilung der Arten von Wartung	7
1.7	Software-Re-Engineering	10
1.8	Reverse Engineering und verwandte Prozesse	11
1.9	<i>Domain Model</i> eines Multitasking-Window-Managers . . .	20
1.10	Hash-Tabelle in <i>Plan-Calculus</i> -Repräsentation	24
1.11	Structure Chart	31
1.12	Nesting Tree	34
2.1	IFAS-Geschäftsfall	64
2.2	Objektorientiertes Anwendungsmodell von IFAS (statische Aspekte)	75
2.3	Objektorientiertes Anwendungsmodell von IFAS (<i>message connections</i>)	76
3.1	Systemtransformation und Paradigmenwechsel	92
3.2	Der COREM-System-Transformationsprozeß	95

3.3	Zuordnung von Elementen	98
3.4	Unterschiedliche Zielsetzungen von COREM	102
3.5	COREM-System-Transformation und Anwendungsmodellierung	108
4.1	Reverse generiertes ERD der Fallstudie	164
5.1	Zwischenergebnis des statischen ooAMs der Fallstudie	184
5.2	Statisches ooAM der Fallstudie	190
6.1	<i>Reverse</i> generiertes ooAM für die Fallstudie	218
7.1	Objekt-Mapping zwischen <i>forward</i> und <i>reverse</i> ooAM	220
7.2	Ähnlichkeiten hinsichtlich <i>part-of</i> -Relation	223
7.3	Ähnlichkeiten hinsichtlich <i>is-a</i> -Relation	224
7.4	Unterschiedliche Darstellung in den Anwendungsmodellen	225
7.5	Keine Entsprechung im <i>reverse</i> ooAM	227
7.6	“Null-Initialisierung” und Redefinition	227
7.7	Keine Initialisierung, aber Verzweigung	228
7.8	Beschränkter Wertebereich und Redefinition	228
7.9	Wechselseitige ausschließliche Verwendung	229
7.10	Getrennte Objekte im <i>reverse</i> ooAM	230
7.11	Ableitung von Struktur-Ähnlichkeiten	233
7.12	Struktur der Service-Definition eines Objektes	251
7.13	Message connection aufgrund einer Prozedurzuordnung	251
7.14	Kommunikation zwischen Objekten und prozeduralem Rest-System	252
7.15	Objektorientiertes Ziel-ooAM nach dem Mapping	258
7.16	Objektorientiertes Ziel-ooAM für die Fallstudie	268

8.1	Struktur des untersuchten Systems	273
8.2	Erweiterung des Anwendungsmodells um Design-Objekte .	274
8.3	Phasenstruktur	285
8.4	Geschachtelte Phasen	286
8.5	Phasen anhand eines Beispiels	287
8.6	Delegation und Phasendefinition	289
8.7	Erweitertes Ziel-Anwendungsmodell für die Fallstudie . .	299
9.1	Struktur des entstandenen Ziel-Systems	303
9.2	Globale Daten zwischen Objekten	304
9.3	Kapselung von globalen Variablen	306
9.4	Datenobjekt im Ziel-System	308
9.5	Struktur des Ziel-Systems nach den Adaptierungen	322
9.6	Objektstruktur des Ziel-Systems für die Fallstudie	331