

1 Vorwort

Dieses Buch wendet sich an ABAP-Programmierer, die die neuen Möglichkeiten der objektorientierten Programmierung in ihren Anwendungen nutzen wollen.

Doch warum sollten ABAP-Programme nun objektorientiert entwickelt werden? Hierzu gibt es mehrere wichtige Gründe.

Seit Release 4.6 hat SAP die Benutzeroberfläche des R/3-Systems durch den vermehrten Einsatz grafischer Elemente deutlich ergonomischer gestaltet. Mit Unterstützung dieser **Enjoy-SAP-Initiative** soll die tägliche Arbeit am SAP-System für den Anwender Spaß machen. Allerdings sind die grafischen Elemente – Controls genannt – vollständig in objektorientierter Technologie aufgebaut. Derzeit arbeitet man in Walldorf mit Hochdruck daran, große Bereiche der Anwendung zu »enjoyen«, d.h. auf die Verwendung von Enjoy-SAP-Controls umzustellen. Dies wird zweifellos das Erscheinungsbild der SAP-Module tiefgreifend verändern. Um nun diese Controls auch in eigenen Programmen zu nutzen, muss man ABAP-Objects beherrschen.



Abb. 1-1
Enjoy-SAP-Logo

Auch die mächtigen Anwendungen der MS-Office-Reihe lassen sich über Controls mit Hilfe der OLE2-Technologie in SAP-Transaktionen integrieren. Stellen Sie sich eine SAP-Transaktion vor, in der ein Teilbereich des Dynpros z.B. mit einem Word-Dokument gefüllt ist. Innerhalb dieses Word-Controls können Sie fast alle Aktionen vornehmen,

die MS-Word beherrscht – sogar Makros können ausgeführt werden. Man nennt diese Technologie **Office-Integration**, sie zu realisieren ist nur in objektorientierter Programmierung möglich.

In diesem Zusammenhang muss der **Business-Document-Service** (BDS) erwähnt werden: Über diesen Dienst lassen sich beliebige Dokumente als Binärdaten auf der Datenbank des SAP-Systems ablegen. Die dahinter stehende Philosophie hat weit reichende Folgen für die Datenorganisation eines jeden Unternehmens, welches SAP für seine Betriebswirtschaft verwendet. Die Möglichkeiten des BDS selbst können ebenfalls nur in objektorientierter Programmierung genutzt werden.

Ein wesentlicher Vorteil von objektorientierter Programmierung ist die größere Code-Unabhängigkeit der Software-Komponenten gegenüber der klassischen Programmierweise. Eines der Ziele von SAP für die Zukunft ist die Schaffung von **releaseunabhängigen SAP-Modulen**. So soll der SAP-Anwender einmal beispielsweise in der Materialwirtschaft mit Release X arbeiten können, während im Vertrieb die neuen E-Commerce-Funktionen von Release Y eingesetzt werden. Das Zusammenspiel der betroffenen Module soll trotz unterschiedlicher Releasestände in voller Integration funktionieren! Auch hier ist die Objektorientierung der Schlüssel zum Erfolg. So werden derzeit in Walldorf einige Modulbereiche einem tief gehenden Reengineering unterzogen, das Ergebnis sind objektorientierte Software-Komponenten für die jeweiligen betriebswirtschaftlichen Objekte. Bereits vor einiger Zeit hat SAP damit begonnen, der Betriebswirtschaft in R/3 eine objektorientierte Sicht überzustülpen, das Ergebnis waren die sog. BAPIs (Business Application Program Interfaces), die eine Verwendung von SAP-Software von außerhalb des SAP-Systems ermöglichten (neben RFC, was es schon länger gibt). Demnächst wird SAP die BAPIs umwandeln in SAP-Klassen, die in vollständiger objektorientierter Sicht im Class Builder vorliegen. Der BAPI-Builder wird daher vermutlich in einem Folgerelease nicht mehr zur Verfügung stehen oder im Class-Builder integriert sein.



Wie eben bereits erwähnt stellt die objektorientierte Programmierung auch die Basis für die software-technische Öffnung des SAP-Systems dar. Nur objektorientierte Software kann ihre Möglichkeiten über die COM-Schnittstelle oder auch CORBA für Fremd-Software zur Verfügung stellen. Die objektorientierte Programmierung in ABAP ist somit die Schlüsseltechnologie, um moderne **E-Commerce-Lösungen** bzw. ein echtes SCM (Supply-Chain-Management) mit SAP R/3 zu ermöglichen.



© SAP AG

Abb. 1-2

mySAP.com-Logo

Doch die bisher vorgestellten pragmatischen Gründe für eine objektorientierte ABAP-Programmierung verdecken die viel weit reichenderen strategischen Überlegungen: Nur unter objektorientiertem Design und Programmierung lassen sich komplexe Software-Systeme unter einem hohen Qualitätsstandard herstellen. Die Objektorientierung ist hier der Schlüssel für einen hohen Wiederverwendbarkeitsgrad und eine bestmögliche Stabilität der entstandenen Software-Komponenten.

Die eben beschriebenen Entwicklungen lassen mich nur ein klares Fazit ziehen:

Nur wer auch die objektorientierten Sprachelemente in ABAP beherrscht, kann zukünftig in SAP R/3 Software herstellen, die den hohen qualitativen und funktionalen Anforderungen der Zukunft genügt.



1.1 Release-Stand und Software-Beispiele

Die zahlreichen Navigationshilfen und Screenshots basieren auf dem SAP-Release 4.6c, die beschriebenen Konzepte und Verfahren funktionieren jedoch ebenso auf älteren SAP-Release-Ständen ab Release 4.6a. An denjenigen Stellen, wo es Unterschiede in dieser auch für SAP neuartigen Technologie gibt, wird gesondert darauf hingewiesen.

Sie können die Beispielprogramme, welche auf www.abaps.de zum Download bereitstehen, mit Hilfe des Programms ZREPTRAN_46C inklusive Textelemente etc. auf einfachste Weise auf Ihr eigenes SAP-Entwicklungssystem bringen, um die Beispiele selbst durchspielen zu können.

1.2 Notwendige Vorkenntnisse

Da die Verwendung von Enjoy-SAP-Controls auf einer dialogorientierten Verarbeitung beruhen, benötigen Sie nicht nur das Wissen um die Bedienung der Werkzeuge der ABAP-Workbench (mit Ausnahme des Class-Builders), sondern auch der Dialogprogrammierung, wie sie z.B. bei SAP im Seminar BC400 vermittelt wird.



1.3 Warenzeichen

Die in diesem Buch verwendeten Begriffe SAP, R/3 und ABAP sind eingetragene Warenzeichen der SAP AG, Walldorf. Aufgrund der besseren Lesbarkeit wurde auf eine Kennzeichnung der zahlreichen Textstellen mit diesen Wörtern mit dem entsprechenden Symbol (®) verzichtet.

Die Strichmännchen an den Seitenrändern heißen »Screen-Beans« bzw. »Strichmännchen«, sie stammen ebenso wie die verwendeten Fotos aus der Clipart-Gallery von Microsoft Office. Wir bedanken uns für die Erlaubnis, diese hier verwenden zu dürfen.

1.4 Zum Inhalt



Da die meisten der ABAP-Programmierer die objektorientierte Technologie nicht beherrschen, werden zunächst die **Grundlagen der objektorientierten Programmierung** vermittelt. Dies ist eines der wichtigsten Kapitel, da hier gegenüber der klassischen Programmierweise eine völlig neue Denkwelt erschlossen wird. Entwickler, die diese bereits beherrschen, können das betreffende Kapitel recht schnell überfliegen.

Im dritten Kapitel werden die **OO-Sprachelemente der ABAP-Sprache** im Detail vorgestellt. In einer praktischen Übung werden diese Konstrukte verwendet und gleichzeitig die Beherrschung der »neuen Denke« in ihrer praktischen Anwendung überprüft. Zum Abschluss gibt es noch einige Überlegungen zum **objektorientierten Design**.

Lokal deklarierte OO-Konstrukte gelten analog lokaler Datendeklarationen nur im jeweiligen Programm. Ähnlich wie das Data-Dictionary für Datendeklarationen wirkt der **Class-Builder** und stellt die OO-Elemente allen Programmen zur Nutzung zur Verfügung.

Nachdem wir eigene OO-Elemente deklarieren und verwenden können, betrachten wir die von SAP bereitgestellten OO-Klassen. Es erfolgt ein weit reichender Überblick über die wichtigsten **Enjoy-SAP-Controls**. Zu jedem Control gibt es ein eigenes Kapitel mit Beispielprogramm. Da die Controls intelligente Software-Komponenten des Präsentationsservers sind, werden die Problemstellungen der verteilten Datenverarbeitung anhand der **Synchronisation der Automation Queue** ausführlich besprochen. Ebenso wichtig sind die Verfahren, wie man in ABAP auf **Ereignisse**, die ein solches Control ausgelöst hat, reagieren kann.

Die Krönung der Control-Verarbeitung stellt die **Office-Integration** dar. Beispiele zeigen die Verwendung von MS-Word und MS-Excel als integrierte Komponenten innerhalb von SAP-Transaktionen.

Die Business-Object-bezogene Speicherung von solchen Office-Dokumenten auf der SAP-Datenbank hat weitreichende Konsequenzen. Dem **Business Document Service** ist daher ein eigenes Kapitel zgedacht.

In einem abschließenden Kapitel werden die Möglichkeiten des **Datenaustauschs** und der weiteren Integration von Office-Anwendungen kurz vorgestellt: Die Office-Dokumente können Makros (in Visual Basic) enthalten und ihrerseits auf Ressourcen zugreifen, die innerhalb des SAP-Systems liegen. Hier haben wir dann eine komplexe Client-Server-Anwendung mit interaktiver Verarbeitung in ABAP-Objects und Visual Basic.

Zu einigen der Themen dieses Buches – insbesondere zu den wichtigen Grundlagen – habe ich **Übungsaufgaben** vorgesehen. Bei den anderen sind die jeweils wichtigen Programmcodings direkt im Kapitel wiedergegeben, so dass Sie keine Schwierigkeiten haben sollten, eigene Anwendungen zu erstellen. Mit Hilfe der Übungsaufgaben erreichen Sie einen besseren Lerneffekt und erlangen etwas Praxiserfahrung in der objektorientierten Arbeit unter ABAP. Ich empfehle Ihnen daher auf jeden Fall, diese Übungen durchzuführen. Bei Schwierigkeiten finden Sie die entsprechenden Musterlösungen im Anhang. Doch beachten Sie: Man lernt besser durch eigene Fehler, als durch das bloße Nachvollziehen von Musterlösungen.

1.5 Grenzen?

Die Summe der Möglichkeiten von ABAP Objects lassen nur noch eine einzige Schranke für die Gestaltungsmöglichkeiten von SAP-Programmen zu: Horizont und Qualifikation des Programmierers. Ansonsten ist gemäß einem Werbeslogan von Toyota: **nichts mehr unmöglich.**



