Vorwort zur 8. Auflage

Bei der Realisierung von Anwendungs-Software sind Daten und Funktionen von Bedeutung. Je nachdem, woran sich der Anwendungsentwickler primär orientiert, sind die in Abb. 1 gezeigten Anwendungsentwicklungsarten zu unterscheiden.

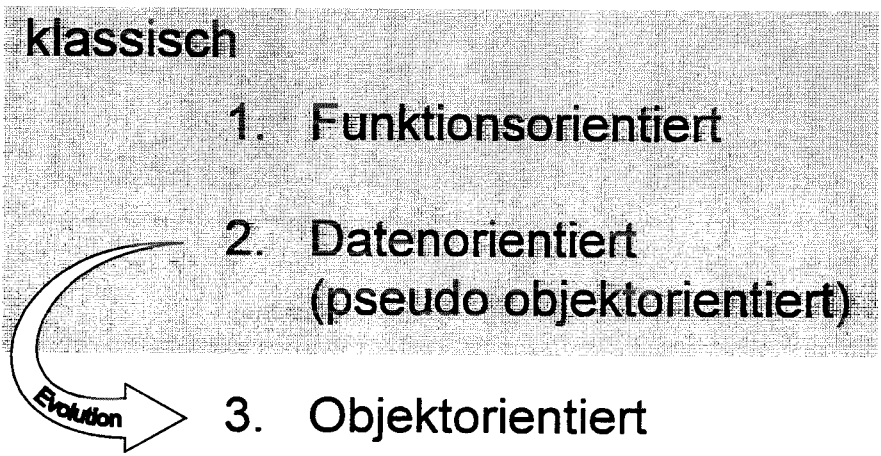

Abb. 1 Anwendungsentwicklungsarten

Dazu folgende Erläuterungen:

Bei der in Abb. 2 gezeigten *funktionsorientierten Vorgehensweise* stehen Funktionen (die für eine Anwendung relevanten Tätigkeiten repräsentierend) im Mittelpunkt der Überlegungen. Daten (zur Darstellung der für eine Anwendung relevanten Objekte, aber auch von Ergebnissen sowie Eingaben) haben sekundäre Bedeutung und sind erst in einem zweiten Schritt festzulegen. Dieses in der Steinzeit des Computerzeitalters praktizierte Vorgehen ist insofern verständlich, als das Entwickeln von Software mit der Erstellung von Programmen gleichgesetzt wurde, und ein Programm realisiert eben eine oder auch mehrere Funktionen. Nachteilig wirkt sich dabei aus, dass das Vorgehen zu Insellösungen und damit einhergehend zu anwendungsorientierten Datenbeständen mit *Redundanz-*, *Synonym-* sowie *Homonymproblemen* führt. In den meisten Unternehmungen, welche das funktionsorientierte Vorgehen praktizierten, stellte sich denn auch im

Verlaufe der Zeit ein verheerendes Datenchaos ein, welches kaum die Möglichkeit bietet, der Geschäftsleitung umfassende, den gesamten Geschäftsgang betreffende Informationen zur Verfügung zu stellen.

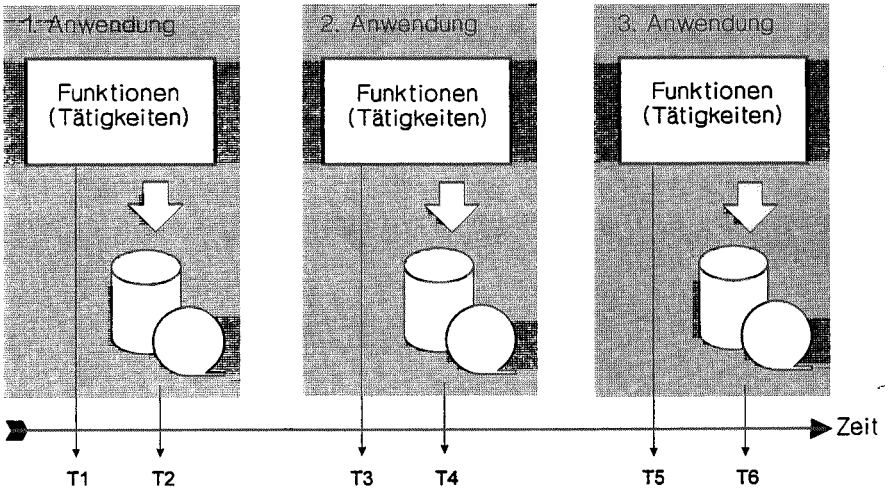

Abb. 2 Die funktionsorientierte Anwendungsentwicklung

Im vorliegenden Werk wird vorgeschlagen, sich bei der Anwendungsentwicklung nicht primär an Funktionen, sondern an den für eine Unternehmung relevanten Objekten wie Kunden, Mitarbeitern, Produkten, Produktionsmitteln, Lieferanten, etc. zu orientieren. Idealerweise sind besagte Objekte noch vor der Entwicklung einer ersten Anwendung in Form eines konzeptionellen (d.h. hardware- und softwareneutralen), möglichst unternehmungsweiten, groben Datenmodells darzustellen. In der Regel ist in diesem Zusammenhang von einer *globalen Datenarchitektur* die Rede. Eine globale Datenarchitektur ist entsprechend Abb. 3 mit einem Rohbau zu vergleichen, dessen Räume im Verlaufe der Zeit mit projektbezogen ermittelten Details datenspezifischer Art angereichert werden. Eine mit Details angereicherte globale Datenarchitektur ist ein *Muster* und keinesfalls mit der Datenbank gleichzusetzen. Die Speicherung der eigentlichen Daten erfolgt nach Massgabe des Verwendungsortes teils zentral teils dezentral, wobei sowohl die für die Speicherung wie auch Präsentation der Daten erforderlichen Strukturen vom Muster abzuleiten sind. Weil man sich beim geschilderten Vorgehen zunächst an den Objekten orientiert, müsste eigentlich von einer *objektorientierten Vorgehensweise* die Rede sein. Da der Begriff *Objektorientierung* in der Informatik mittlerweile eine - wie noch zu zeigen sein wird - umfassendere Bedeutung erlangt hat, sei hier zur Vermeidung von Missverständnissen von einer *datenorientierten* (resp. *pseudo objektorientierten*) *Vorgehensweise* die Rede. Die Bezeichnung ist

insofern naheliegend, als man die für eine Unternehmung relevanten Objekte zunächst in Form eines groben Datenmodells - eben einer Architektur - darstellt, bevor Überlegungen funktionsspezifischer Art in Angriff genommen werden.

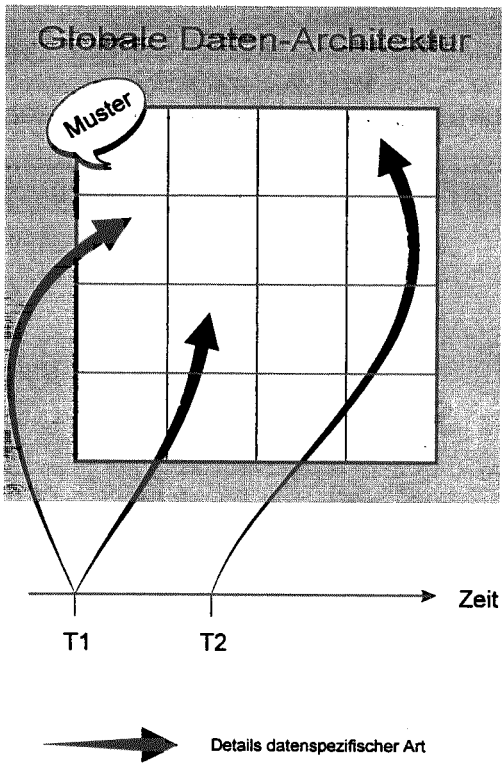

Abb. 3 Die datenorientierte Anwendungsentwicklung

Worin unterscheidet sich nun aber das *datenorientierte* (sprich *pseudo objektorientierte*) *Vorgehen* vom eigentlichen *objektorientierten Vorgehen*? Abb. 4 illustriert den Unterschied. Zu erkennen ist, dass im klassischen Umfeld Daten und Funktionen grundsätzlich getrennt sind. Entsprechend erstellt man einerseits mit Monolithen vergleichbare Programme und beschäftigt sich anderseits mit den eigentlichen Datenbeständen. Ganz anders im objektorientierten Umfeld, fasst man hier doch Daten und darauf operierende Funktionen (in der Regel ist diesbezüglich von sogenannten *Methoden* die Rede) als Einheiten - eben als *Objekte* - auf [1].

[1] Wie man sich diesen Sachverhalt vorzustellen hat, wird in Abschnitt 2.5 dieses Buches konzeptmässig behandelt. An Einzelheiten interessierte Leser seien auf das neue Werk von M. Vetter: *Objektmodellierung (Eine Einführung in die objektorientierte Analyse und das objektorientierte Design)*, B.G. Teubner, Stuttgart, verwiesen.

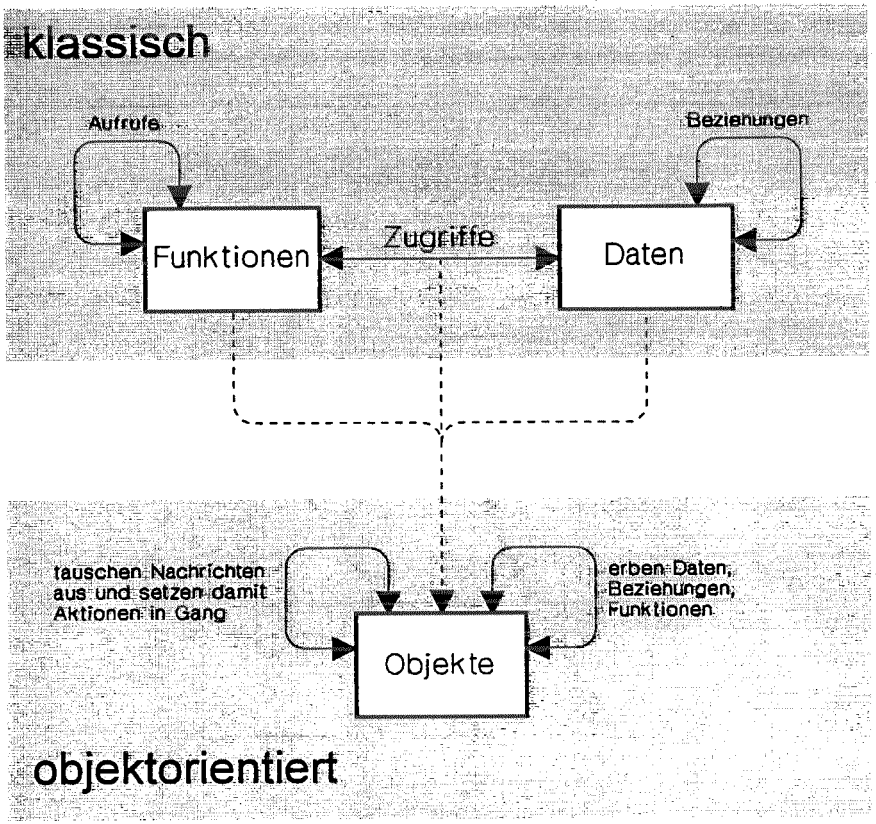

Abb. 4 Unterschiede zwischen klassischer und objektorientierter Anwendungsentwicklung

Globale Architekturen sind im objektorientierten Umfeld noch bedeutsamer als im klassischen, weil man damit entsprechend Abb. 5 nicht nur die im Verlaufe der Zeit projektbezogen ermittelten Details datenspezifischer, sondern auch solche funktionsspezifischer Art geordnet und in einer für jedermann wiederauffindbaren Weise versorgen kann. Damit ist angedeutet, dass die in diesem Buche dargelegten Überlegungen zur Erzielung von globalen Architekturen auch im objektorientierten Umfeld ihre Berechtigung behalten. Mehr noch: In Ermangelung effizienter objektorientierter Datenbankmanagementsysteme speichert man die Daten vielerorts nach wie vor relational (ja sogar netzwerkartig und hierarchieförmig) und bedient sich des objektorientierten Ansatzes lediglich zur Aufbereitung der zu präsentierenden Ergebnisse auf Workstations. Entsprechend sind die in diesem Buche zur Sprache kommenden Überlegungen, die zu relationalen, netzwerkartigen und hierarchieförmigen Datenstrukturen führen, nach wie vor von Bedeutung.