Kapitel 0

{ // Vorwort

Der Werkzeugkasten der Methoden zur objektorientierten Softwarekonstruktion hat sich in der täglichen Praxis des Softwareingenieurs als Kollektion recht wirkungsvoller Hilfsmittel erwiesen. Diese Methoden helfen bei der Konstruktion korrekter, zuverlässiger und wiederverwendbarer Software. Sie begleiten die Konstruktion von der Analyse, also den ersten Ideen eines Programms, über den Entwurf bis hin zur Implementierung. Die Objektorientierung dient hierbei anfangs als Richtschnur zur angemessenen Beschreibung der Phänomene, die man modellieren und dann in einem Programm angemessen implementieren möchte. Sie wird in vielen Programmiersprachen durch sprachliche Hilfsmittel unterstützt — als Schlagworte mögen die Begriffe *Klasse*, *Geheimnisprinzip* und *Vererbung* dienen, so daß ein objektorientierter Entwurf auch objektorientiert implementiert werden kann.

Objektorientierung als Anfang

Da sich objektorientierte Methoden bewährt haben, besteht breiter Konsens darüber, daß die Einführung in die Informatik mit einer objektorientierten Programmiersprache beginnen soll. Das Argument, mit dem Proponenten der Objektorientierung arbeiten, lautet etwa so: Es gelingt, viele Phänomene der natürlichen Welt mit objektorientierten Hilfsmitteln auf natürliche Weise zu modellieren, daher sollte eine Einführung in die Informatik früh mit diesen Methoden vertraut machen. Im Tandem mit diesem Argument kommt dann auch eine objektorientierte Programmiersprache.
Nun mag man diskutieren, ob die Objektorientierung wirklich eine *natürliche* Modellierung ist. Sie ist vielen anderen Modellierungsmethoden überlegen. Warum das so ist, werden Sie in diesem Buch sehen. Der Beifahrer dieser Argumentation, nämlich die Programmiersprache, sollte auch objektorientiert sein. Das ist einsichtig, denn man möchte keinen methodischen Bruch zwischen der Modellierung und der Implementierung, also der Realisierung in einem Programm, entstehen lassen. Solche *methodischen* Brüche führen erfahrungsgemäß leicht zu *kognitiven* Brüchen, so daß die Vorteile, die man sich mühsam bei der Modellierung erobert hat, bei der Implementierung verloren zu gehen drohen.
Gegenwärtig werden zwei objektorientierte Programmiersprachen als Sprachen für den akademischen Anfangsunterricht bevorzugt, nämlich die Sprachen JAVA und C++. Die Programmiersprache JAVA begann im zweiten Drittel der neunziger Jahre des vorigen Jahrhunderts mit ihrem Siegeszug und scheint unbezwingbar zu sein. Ihre vielen Vorteile machen es ihr leicht, ein Anwendungsfeld nach dem anderen zu erobern. Sie kann ihre Abstammung von

C++ und, wenn man genauer hinschaut, von BETA nicht leugnen. Die Programmiersprache C++ hingegen basiert auf der kampferprobten Sprache C, der Basis für das populäre Betriebssystem UNIX (mit seinen modischen LINUX-Varianten). C++ ist die objektorientierte Tochter von C, also eine Spracherweiterung durch objektorientierte Konstruktionen.

So entstand dieses Buch

Der Verfasser durfte ab WS 97/98 die Dortmunder Erstsemester der Elektrotechnik in die Informatik einführen. Auf Bitten der Fakultät für Elektrotechnik wurde eine zweisemestrige Veranstaltung *Einführung in die Informatik für Ingenieure I/II* konzipiert und in einem zweiten Durchlauf validiert. Sie sollte und soll nach den Vorstellungen der Fakultät Algorithmen und Datenstrukturen sowie objektorientierte Konzepte zusammen mit der Programmiersprache C++ lehren. Die Entscheidung, C++ als Sprache vorzuschlagen, beruhte auf der Beobachtung daß viele Programmkomponenten und Bibliotheken in C oder C++ formuliert sind. Die Studenten werden durch die Veranstaltung unmittelbar in die Lage versetzt, diese Komponenten für ihre eigene Arbeit zu benutzen (und natürlich eigene Programme zu schreiben).

Aus diesen Vorlesungen ist das vorliegende Buch entstanden. Es richtet sich an Nebenfächler der Informatik und an solche Leserinnen und Leser, die sich mit der objektorientierten Programmierung und mit C++ vertraut machen wollen. Das Buch wurde so konzipiert, daß auch geistes- und sozialwissenschaftlich orientierte Leser den Ausführungen mit Gewinn folgen können. Der Verfasser hat dazu bewußt Beispiele gewählt, die nicht dem typischen Umkreis des Ingenieurs oder des Informatikers entstammen, sich vielmehr bemüht, die weitgespannten Anwendungsmöglichkeiten der objektorientierten Modellierung und Programmierung exemplarisch zu zeigen. Das schlägt sich auch in den Aufgaben nieder. Der Ton dieses Buchs wurde bewußt nicht-technisch gehalten. Der gelegentlich unausstehliche Jargon, der sich in der Programmierung breitgemacht hat, soll soweit als möglich vermieden werden.

Die Objektorientierung und die Programmiersprache C++ stehen zwar im Vordergrund dieses Buches, wir beginnen jedoch mit einer Einführung in diejenigen Teile von C++, die aus der Sprache C stammen. Wir fangen auch beim Programmieren nicht gleich mit der Objekt*orientierung* an. Der didaktische Ansatz dieses Buches geht vielmehr von der objekt*basierte* Programmierung in C aus. *Objektbasiert* bedeutet hier, daß mit Instanzen Abstrakter Datentypen gearbeitet wird, also mit Instanzen von *Kapseln*, die Daten und die Prozeduren auf diesen Daten in einer gemeinsamen Hülle einschließen. Der Begriff des Abstrakten Datentypen wird sehr früh eingeführt, um dem Leser möglichst früh Gelegenheit zu geben, Datenabstraktionen gemeinsam mit Funktionsabstraktionen zu benutzen. Technisch sieht das so aus, daß die Vorstufe von Klassen, nämlich zusammengesetzte Strukturen, um funktionale Komponenten erweitert werden. Damit ist eine objektbasierte Grundlage für die weiteren Diskussionen gelegt. Sie wird weidlich genutzt. Wir zeigen an Beispielen, daß dieses Konzept sehr tragfähig ist, und sich schon zur Realisierung recht mächtiger Anwendungen eignet. Erst dann, wenn diese sprachlichen Hilfsmittel eingeführt und ausreichend an Beispielen erprobt worden sind, werden objektorientierte Konzepte eingeführt. Klassen erscheinen als Ausprägung zusammengesetzter Strukturen, Vererbung und die damit verbundenen Möglichkeiten der Polymorphie werden dann ebenfalls eingeführt und an Beispielen erprobt. Die Tragfähigkeit des Konzepts erweist sich dann an einem größeren Beispiel, das

recht ausführlich diskutiert wird. Das Buch schließt mit einer Diskussion von Schablonen (die im eigentlichen Sinne ja nicht mehr objektorientiert sind) und einer kurzen Diskussion der Ausnahmebehandlung.

Und das steht in diesem Buch

Eine kurze Darstellung des Inhalts soll folgen. Fast alle Kapitel beginnen mit der Diskussion einer konkreten Problemstellung, die eine Erweiterung der sprachlichen Möglichkeiten vorschlägt. Die neuen Ausdrucksmöglichkeiten werden auf ihre Belastbarkeit geprüft, meist auch erweitert und auf andere Probleme angewendet. Fast jedes Kapitel enthält Übungsaufgaben, die in Anspruch und Umfang von leichten Etuden zur Einübung der neuen Techniken bis hin zu kleinen Projekten reichen. Insgesamt sind es 120 Aufgaben geworden.

Kapitel 1: Hofzwerge Das Buch beginnt mit der Geschichte der Hofzwerge in der Wiener Hofburg. Diese Hofzwerge — genauer: ihre Besoldungsstruktur — werden nach einigen Seiten hin untersucht. Die Eigenschaften der Hofzwergbesoldung werden als Klassifikationshierarchie gefaßt. Damit hat der Leser bereits am Anfang ohne jegliche Programmierkenntnisse einen wichtigen Entwurfsschritt getan. Der so aufgespannte Bogen dehnt sich weit. Er findet erst im letzten Viertel des Buchs sein (anderes) Ende, wenn nämlich die Klassifikationshierarchie mit allen ihren dynamischen Aspekten implementiert wird. Der Verfasser möchte auf diese Weise zeigen, daß es durchaus möglich ist, objektorientierte Konzepte einzuführen, ohne tiefergehende Eigenschaften einer Programmiersprache benutzen zu müssen. Daß dies möglich ist, belegt die Tragfähigkeit des Konzepts der Objektorientierung. Dieser Bogen dient natürlich auch dazu, das Buch zusammenzuhalten: Die Hofzwerge schauen immer mal wieder um die Ecke und tauchen in der einen oder anderen Übungsaufgabe auf, bringen sich also in Erinnerung, um die interessanten Eigenschaften, die dieses ulkige Völckchen hat, auch wirklich zum Funkeln zu bringen.

Kapitel 2, 3: C, elementar Die Programmiersprache C wird in ihren elementaren Komponenten eingeführt. Die sprachlichen Konzepte werden an stetig komplexer werdenden Problemstellungen eingeführt. In diesen beiden einführenden Kapiteln wird die Philosophie der Vorgehensweise deutlich: Konstruktionen werden nicht um ihrer selbst willen eingeführt, vielmehr sollen konkret vorliegende Probleme damit gelöst werden. Solche einführenden Kapitel sind fast immer ziemlich langweilig, aber diese Durststrecke will überwunden sein. Der Leser sollte dieses Kapitel vielleicht überfliegen, um bei Bedarf später nachschlagen zu können.

Kapitel 4: Funktionen Hier werden Funktionen eingeführt. Sie werden als wichtige Strukturierungsmöglichkeit für Programme diskutiert, die darüber hinaus auch die Möglichkeit geben, die Lokalität von Namen einzuführen und zu erkunden.

Kapitel 5: Separate Übersetzung Das Thema der Strukturierung von Programmen wird in diesem Kapitel noch einmal aufgenommen. Hier geht es nämlich darum, ein umfangreiches Programm so zu zerlegen, daß die einzelnen Teile in getrennten Dateien aufbewahrt werden können. Dazu muß man etwas über Namensräume wissen, und sie stehen darum in diesem Kapitel im Vordergrund.

- Kapitel 1 bietet zusammen mit den Kapiteln 13 – 19 eine kurzgefaßte Einführung in die objektorientierte Programmierung.
- Das gesamte Buch kann als elementare Einführung in die objektorientierte Konstruktion von Software mit C++ und als Einführung in die Sprache dienen.

Zur Literatur Als Anmerkung zur verwendeten Literatur sei vermerkt, daß sich der Verfasser an die Folklore in diesem Gebiet gehalten hat, sich also an den gängigen Beispielen orientieren konnte. Daher sind also meist nicht die Quellen angegeben. Der Verfasser hat sich insbesondere gern von den Lehrbüchern [AHU73, AHU82, Die96, SK97, Knu94, Knu93a, Knu93b, MN99, OW90, Dij76, CC82, DD99][1] inspirieren lassen und seine eigenen Variationen angebracht. Diese Texte waren bei der Suche nach Ideen für Übungsaufgaben hilfreich.

Materialien

Dieses Buch ist aus Vorlesungen hervorgegangen, und die Vorlesungsmaterialien stehen zur Verfügung. Sie umfassen etwa 630 animierte und anotierte Powerpoint-Folien sowie etwa 300 Folien mit Programmtexten und die Programmtexte selbst. Der Verfasser hat sie im ursprünglichen Zustand belassen; Sie können unter `http://www.eVerlage.de` darauf zugreifen. Wie das genau geht, sehen Sie, wenn Sie sich auf jener Seite befinden. Unter dieser Adresse finden Sie auch eine Schnupperversion und eine volle Version des vorliegenden Buchs.
Die Programmausschnitte in diesem Buch sind direkt aus den Programmtexten der oben genannten Folien in das Manuskript kopiert worden. Die Programme sind ohne Ausnahme mit dem C++-Compiler von BORLAND in der Version 4.5.2 übersetzt worden; Stichproben haben gezeigt, daß auch der GNU-Compiler die Programme ohne Probleme übersetzen konnte. Die Programme sind in der Vorlesung zu Demonstrationszwecken ausgeführt worden, deshalb ist der Verfasser zuversichtlich, daß sich dort nicht allzu viele Fehler finden.

Danksagungen

Für die freundliche Hilfe, die mir angeboten wurde, bin ich vielen Kollegen und Studenten dankbar; gern habe ich die Möglichkeiten, über das Konzept des Buchs zu diskutieren, angenommen. Kritische und konstruktive Anregungen kamen insbesondere von Frau Prof. Dr. Sigrid Schubert, die das Manuskript aus der Sicht der Informatik-Didaktikerin gelesen und kommentiert hat. Mein Dortmunder Kollege Prof. Dr. Gisbert Dittrich war ein hilfreicher Sparringspartner, wenn es um den Zugang ging; auch er hat Anmerkungen zum Manuskript gemacht. Einige Anregungen und Hinweise kamen von Prof. Dr. Hermann Stever (Landau) und Prof. Dr. Udo Kelter (Siegen). Dr. Stefan Dißmann, bewährter Koautor, hat in vielen Diskussionen mit und ohne Kaffee (d.h.: er ohne, ich mit) dabei geholfen, das didaktische Konzept zu schärfen und auf sichere Füße zu stellen. Heiko Falk, Dr. Eike Riedemann, Stefan Steinke und Hamza Tatlitürk waren hilfreiche Gesprächspartner, auf die auch die Idee zu der einen oder anderen Übungsaufgabe zurückgeht. Prof. Dr. Klaus Schumacher sorgte als aufmerksamer und kritischer Kollege dafür, daß die Studenten der Elektrotechnik praktische Aspekte nicht vernachlässigten.

[1] Angaben in eckigen Klammern beziehen sich auf das Literaturverzeichnis, das auf Seite 322 beginnt.

Alla Stankjawitschene schrieb den Text und wunderte sich oft, daß manche Satzkaskade doch noch zum Abschluß kam. Klaus Alfert half mir dabei, meine rostigen LaTeX-Kenntnisse zu aktualisieren und überzeugte mich von den Vorteilen von Emacs (so daß ich den Satz des Buchs in einer angenehmen Umgebung durchführen konnte). Julia Kathrin Doberkat war eine wichtige und angenehme Hilfe bei den redaktionellen Arbeiten; der frische Blick der jungen Geisteswissenschaftlerin hat manche verkorkste Formulierung ins Lot gebracht. Ihnen allen möchte ich meinen herzlichen Dank sagen: Probleme, Schreibfehler oder logische Pannen liegen ausschließlich in meiner Verantwortung. Bedanken möchte ich mich auch und nicht zuletzt bei Dr. Peter Spuhler für die wie gewohnt gute Zusammenarbeit mit dem Verlag.
Und bei Constantin & Maximilian.

Bochum und Dortmund, Pfingsten 2000