14

A new approach to teaching information technologies: shifting emphasis from technology to information

Peter Hubwieser, Manfred Broy, and Wilfried Brauer
Fakultät für Informatik der Technischen Universität München
D-80290 München, Germany

Abstract

Being aware of the increasing pedagogical challenges posed when using and teaching IT in schools, we have begun a widespread initiative at the Technical University of Munich to improve the practice of teaching IT in secondary schools in Bavaria. We offer a curriculum to students who intend to become secondary school teachers. The goal of the curriculum is to offer students a more general viewpoint, leaving behind the more narrow hard- and software specific approaches that schools have followed most often. Guided by the process of developing software, we propose to concentrate on modelling techniques, especially object-oriented versions. We intend, furthermore, to educate as many teachers at a university level in computer science as possible. To that end we have developed a new, lean curriculum for the education of teachers, which also serves as a basis for an in-service training project. In addition, we plan to improve the access that schools have to modern information technologies. We have built an Internet node for a variety of rural schools in Rosenheim.

Keywords

Secondary education, teacher education, curriculum development, information technology, modelling, software engineering

1 INTRODUCTION

Recognising the steady growth in the importance of modern information processing concepts in everyday life, visible, for instance, in the number of multimedia computers and levels of Internet access in the homes of our students, it can no longer be accepted that there is almost no teaching of modern concepts of information technologies in our schools.

With the intention of improving the practice of teaching IT in secondary schools in Bavaria, the Technical University of Munich has begun a widespread initiative, strongly supported by the Bavarian Ministry of Education, Culture, Science and Art.

1.1 The Bavarian 'Gymnasium' as our main target

Today the majority of children seek to enter higher educational levels. Therefore, we have concentrated our efforts currently at this school level, which means that we are dealing with the so-called 'gymnasium'. This type of school covers 9 years of education, and is placed at the highest of the three educational levels that are offered to students in Bavaria once they have completed primary school at the age of 10 years.

1.2 Current education in IT

At the moment, the only obligatory IT education in Bavarian gymnasiums is a requirement termed "IT Basic Instruction". In this, each student gains about 60 hours of IT lessons, distributed through regular classes in mathematics, economics and German language, mainly dealing with aspects of use within the hard- and software provision of the schools. The teachers responsible for these courses generally lack any educational background in IT. Therefore, this system produces results which are rather varied.

Building on this weak basis, it is possible for students to choose a range of optional IT courses. The variety that is offered depends to a large extent upon the number of teaching hours remaining after the timetabling of obligatory classes.

2 THE INFORMATION-CENTRED APPROACH

First of all, we considered which underpinning philosophy of education in IT should be used. We decided to choose the process of software development as a guide, because it is the ideal paradigm for the task when structuring information. Observing how this process moves forward, starting from a rough real-life situation, passing through the construction of proper models, producing data structures, algorithms, program code, and finally to an implementation on a specific machine, we identify at each step an increasing dependence upon the details of the environment in which the software is implemented. In teaching IT, we firmly believe that the first two steps of this development process are the most beneficial, as they are the most general and the least environment-specific.

2.1 Abstraction of general concepts

In emphasising modelling techniques as the main topic of the suggested IT lessons, we offered general techniques to the students so that they could handle various types of complex information, even if there was no computer or any other hard- or software involved. Starting from a real-life situation example, the students were instructed in the development of proper models without considering any aspects of implementation. The modelling is based on application and on the problem to be solved. This allows us

to concentrate on the structure of the information, not on a specific problem solving technique such as an algorithmic, object-oriented, set-based, or rule-based one. Specific techniques should be chosen after the completion of the modelling-phase, and not before.

2.2 Data models

The structural components of the system chosen are described by data models. From a traditional point of view these structures would be static, with the dynamic aspects modelled separately by global functions. This technique might be appropriate when acquainting students with a rough structure of the application problem. We suggest they use a graphical method to represent static data structures, using a special vertex- and edge-marked graph where the vertices represent data types, the edges denote by their particular form the method of composition, and by their titles the name of the components.

When describing the dynamic aspects of the data model, we prefer the object oriented method, which aims at combining static data structures (attributes) and functions (operations) to form classes. This approach offers a 'narrow semantical gap' (Jacobson, 1989), which avoids the separation of data and functions that arises from the von Neumann concept with program and data storages, allowing the modelling to be determined by the hardware structure. Human beings prefer to think in terms of interacting subjects and objects (Anderson, 1985), and this should be respected within modelling techniques. We suggest using an object modelling technique analogous to Rumbaugh (1991), and Booch (1994).

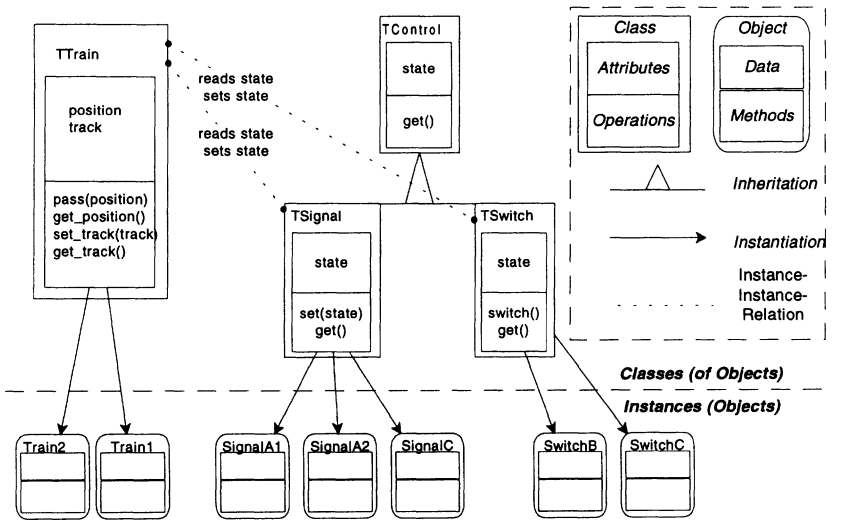

Figure 1 Representation of an object modelling technique

Teachers should be aware that from a mathematical point of view, a data model is based upon a family of carrier sets and a family of functions, as described by Broy (1995) or Goos (1994). The students are instructed to create a graph with nodes representing objects and edges defining relationships between the objects, which could be hierarchical relationships like inheritance and instantiation or other associations like communication relationships.

The object model provides the framework for the other model types which follow.

2.3 The dynamic model

The behaviour of a system or subsystem can be described by a number of well-known state transition diagrams, which can be represented by graphs, the vertices defining states, the edges representing transitions. An extensive description of the mathematics is found in Goos (1994). Each diagram represents a process which transforms information. The edges can be marked with three types of information: in the first instance, the initial (triggering) action must be written, then either the condition for the transition can be specified, or the action that is carried out by the transition, or both.

2.4 The functional model

Not knowing the precise way in which the information is transformed, the functional model (see Figure 2 below), consisting of well-known data flow diagrams, describes only the communication aspects between processes (information transformers), active objects (data sources and consumers), and passive objects (data storage). This type of modelling will not be applicable once processes or objects are created dynamically. As long as the flow of information is static, it illustrates well the interactions between the processes. An exact description of data-flow models using stream processing functions can be found in Broy (1995).

2.5 Action diagrams

In the functional model the transformation of information is carried out by processes, which ignores their inner structure. This can be illustrated by action diagrams, which are again graphs, consisting of events (which are instances of actions) as vertices, and causal relationships as edges. For simplicity, the nodes are marked with the name of the action. For the mathematical model see Broy (1994).

2.6 Correlation of the different modelling techniques

Putting together the different description techniques, the internal structure of the system is separated into objects which interact according to their operations. The operations represent actions. The instances of the operations are called methods, corresponding to events. A process is a combination of events, connected by causal relationships in the action diagram, as well as a combination of methods. The state transition diagrams provide information about the behaviour of processes, transitions representing events, and states describing the content of attributes. The data flow diagrams carry information about the communication and interaction between the single parts of the system.

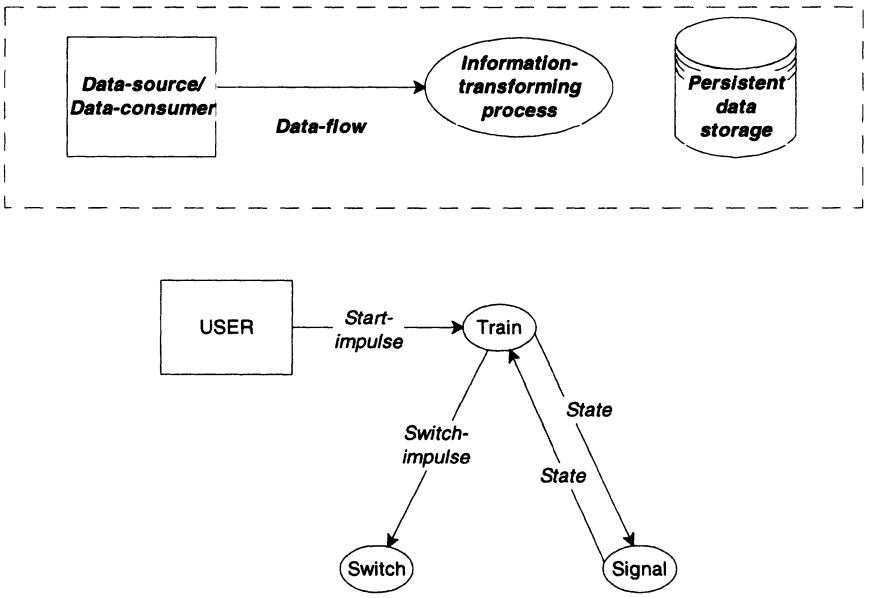

Figure 2 Representation of a functional model

2.7 Method

The IT lessons we propose are based on examples and real-life situations to a great extent, in order to illustrate the abstract concepts as much as possible. Therefore, no standard path through the teaching process can be defined. We indicate outlines of how a lesson could proceed, trying to emphasise the most important aspects.

Often a sequence of lessons will start with the introduction of the problem, presenting the situation in as illustrative a way as possible, and aiming to motivate the students.

The introduction should be followed by a detailed but informal description of the problem, using word processors wherever possible. This ensures that the description can be used again, and makes it possible to combine the work of parallel teams, each dealing with a particular set of the problems.

After discussing the informal description, the central part of the lesson begins: the modelling phase. In most cases it will be sufficient to construct perhaps two of the models described above, depending upon the characteristics of the problem. The students should be supported during this phase by the use of a flow-chart program, which leads to designed representations and avoids frustrations arising when needing to merely repeat the drawing of nearly identical pictures.

The review of the models produced will lead to a method of complementation that is appropriate to the situation. The main goal at this point is to illustrate the modelling results, not to train perfect users of specific programs or sophisticated programmers. Thus we use generic software, such as spreadsheets, and databases, as well as specific software. Using programming languages may sometimes be helpful, but the level of syntax should be kept to a minimum. If this is not possible, the teacher should use

preprogrammed routines to ensure this. An interesting option for realisation would be to use CASE (Computer-aided-software-engineering) tools that directly transform the graphic representation of a model into program code. Ideally the output would be a ready-to-run program.

The last step is a critical review of achievements, possible alternatives, possible improvements, associations with those problems already solved, as well as social consequences of IT use in this specific situation.

3 EDUCATING TEACHERS

As indicated above, one of the main reasons for the stagnation of IT education in Bavarian schools during the last decade has been the lack of university-educated teachers. In co-operation with the Bavarian Ministry of Education, Culture, Science and Art we aim to educate at least one teacher per gymnasium, which means some 400 teachers in total in the next decade. These graduated teachers would serve as consultants and promoters for the development and realisation of this new concept of teaching IT.

3.1 Our curriculum for teacher education

Teacher education in IT in Bavaria is suffering currently from the fact that it is only possible to undertake it alongside a regular university education in two other scientific disciplines, such as mathematics and physics.

We expect students to complete the course in 3 years, to accompany their education in the two other main disciplines. We suggest that they attend the full set of 4 introductory lectures which cover the basics of automata, formal and programming - languages, algorithms and data structures, theories of complexity and calculability, machine architecture and operating systems, the building of interpreters and compilers, distributed systems, and communication. Building on these basic lessons, we expect students to attend special classes about database systems, data structures, software development, operating systems, networking, and the teaching of IT.

In order to get an appropriate mathematical background, we suggest that students attend basic lessons in analysis and linear algebra, which is obligatory for all students of engineering or natural sciences.

3.2 The compact course of study

Considering the current economic situation in Bavarian schools, it is not likely that there will be more than about 200 IT teachers entering service in Bavaria during the next decade. So we have offered a form of in-service training to increase the number of university-trained IT teachers in employment. We have developed a specific in-service course of study with all lessons provided on one specific day of the week. The teaching load of the participants is reduced by 6 hours each week. Using the free lesson time between the semesters to run special classes, we were able to reduce their studies to 2 years. At the moment, the first course is running with 20 participants, the next will be launched in September 1996 with about 25 members. The University of Erlangen-Nuernberg will duplicate this concept and start a parallel training course in Autumn 1996.

4 REFERENCES

Anderson, J.R. (1985) *Cognitive Psychology and Its Implications*. Freeman and Co, New York, Oxford.

Booch, G. (1994) *Object-Oriented Analysis and Design*. Benjamin/Cummings, Redwood City, CA.

Broy, M. (1994) *Informatik. Eine grundlegende Einführung, Teil III*. Springer München.

Broy, M. (1995) Mathematics of Software Engineering. Invited Talk at MPC 95, in B. Möller (ed.) *Mathematics of Program Construction*, July 1995, Kloster Irsee, Lecture Notes of Computer Science. Springer, Berlin Heidelberg.

Goos, G. (1995) *Vorlesungen über Informatik, Band 1*. Springer, Berlin Heidelberg.

Jacobson I., Ericsson M. and Jacobsson A. (1995) *The Object Advantage*. ACM Press, New York.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W. (1991) *Object-Oriented Modeling and Design*. Prentice-Hall, Englewood Cliffs, NJ.

5 BIOGRAPHIES

Dr. Peter Hubwieser gave classes in mathematics, physics and computer science for 10 years at Bavarian gymnasiums, until he moved out of the teaching service in 1992 in order to write his dissertation in theoretical physics, completed in 1995. For the last two years he has been seconded to the Technical University of Munich, with the remit to improve the situation with regard to IT use in Bavarian gymnasiums.

Prof. Dr. Manfred Broy is ordinarius for software technology and was recently awarded the "Bundesverdienstkreuz am Bande" for his contribution to improving the quality of commercial software.

Prof. Dr. Wilfried Brauer, ordinarius for theoretical informatics, is well-known for his works concerning IT teaching, especially within the German "Gesellschaft für Informatik" as well as within IFIP, acting as vice chair from 1979 and as chair from 1985. In 1984 he was one of the authors of the UNESCO-IFIP Modular Curriculum in Computer Science and was invited as a lecturer at the IFIP World Congress in 1989. Recently he was elected as a member of the very honourable Bavarian Academy of Sciences.