Exploration Module for Understanding the Functionality of the Internet in Secondary Education

Stefan Freischlad

Universität Siegen
Didactics of Informatics and E-Learning,
Hölderlinstraße 3, 57068 Siegen,
freischlad@die.informatik.uni-siegen.de

Abstract

In developing and successfully implementing a classroom project on 'Internetworking' in secondary education, the author found that there is a need for adequate educational software. Abstract informatics concepts have to be connected to real-life experiences. In this paper the concept of views that support different perspectives on the subject is applied to meet the requirements for educational software in secondary education in terms of activities and abstraction. The educational software developed allows analysis, simulation, and construction of network infrastructures and connects the visible behaviour, inner structure, and implementation details of Internet applications. Thus it provides a motivating approach based on the concept of discovery learning.

Keywords: educational software, exploration, Internet applications

1 Learning About the Internet in Secondary Education

In June 2005 we started a research project, promoted by the German Research Foundation (DFG), with the aim of analysing the requirements of informatics education and developing a didactic concept in the field of 'Internetworking' for secondary education. Although there is a broad consensus that knowledge about the Internet has to be part of informatics education (Tucker 2003, van Weert 2000), we found that there is no relevant science-based didactic concept for secondary education. Therefore we are developing the didactic system 'Internetworking', which comprises didactic concepts and learning material. In developing and successfully implementing classroom projects about Internetworking in secondary education, the author discovered the need for practice-oriented learning material. The research question in this paper is how learning material about 'Internetworking' can be designed based on the constructivist learning theory.

Kurose and Ross (2005) describe their approach for higher-education teaching about the Internet by starting with applications. They give three reasons for applying a top-down approach to the Internet layer model. First, students are really motivated and interested in how the Internet works if they start with the applications and the application layer and then proceed in more detail through underlying layers. The second argument is the exceptional meaning of the application layer for actual developments. And the third reason is that it is possible to introduce network application programming earlier in the course. In fact, linking both aspects – informatics concepts and applications – is necessary for the understanding of informatics systems (Stechert 2007). Therefore, practice-oriented learning material that bridges the gap between abstract informatics concepts and Internet applications is necessary for informatics courses in secondary education. Classroom activities would start with a motivating example of an Internet application or another phenomenon within the scope of our project.

We started analysing the characteristics of Internet-based informatics systems to identify the necessary competencies for using Internet applications (Freischlad 2006) and described our theoretical approach towards three components of the didactic system 'Internetworking'. The 'knowledge network' describes the learning objectives and necessary prerequisites of learners. 'Exercise classes' represent the abstraction of specific exercise contexts and are hierarchically structured (Freischlad and Schubert 2007). The third component is 'learning aids', particularly educational software. We developed learning materials based on the didactic system and implemented them into classroom practice, that is, we engaged in curriculum intervention. The first project was primarily about communication and privacy protection on the Internet. The second project focused on Internet structures. And the third project combined several aspects of the preceding projects. The results of the projects led to modification and refinement of the didactic system. The curriculum intervention brought us valuable information about students' prior knowledge, attitudes, and misconceptions. At the same time we have started the development of the educational software for 'Internetworking' as a student project, which we call FILIUS[1]. This software is required for practical exercises and experiments.

2 How to Understand Internet Applications and Services

2.1 Understanding of Informatics Systems

According to Stechert (2007) the analysis of characteristics of informatics systems must focus on the visible behaviour (A), the inner structure of a system (B), and specific internal details (C). Therefore, educational software that fosters understanding of the functionality of Internet applications has to consider these perspectives. The visible behaviour is represented by user agents such as web browsers and email

[1] "Freie Interaktive Lernumgebung für Internetworking der Universität Siegen"
(URL: http://www.die.informatik.uni-siegen.de/pgfilius/)

software, but it doesn't include data exchange with protocols. The inner structure is composed of several informatics concepts, for example, protocols as sets of rules for data exchange, and the format and meaning of data units that are exchanged between the entities on different hosts. Specific internal details are represented, for example, by the source code of programs.

Brinda (2006) describes how the concept of exploratory discovery learning was applied to learning with informatics systems. Informatics experiments are identified as a starting point. He then derives requirements for 'discovery friendly application software', which must support exploration and experiment. He concludes that 'discovery friendly learning' must provide motivating learning activities, the possibility of continuation, and linking of construction and simulation. Therefore, when it is introduced for discovery learning, educational software must allow a high level of interactivity, comprising the engagement categories of Viewing, Changing, and Constructing according to the engagement taxonomy of Naps et al. (2002).

Bruner (1966) specifies three representations that are used within the learning process. "At first the child's world is known to him principally by the habitual actions he uses for coping with it. In time there is added a technique of representation through imagery that is relatively free of action. Gradually there is added a new and powerful method of translating action and image into language, providing still a third system of representation" (Bruner 1966, p. 1). He calls these modes of representation 'enactive', 'ikonic' and 'symbolic'. Even though these representations are developed step by step in a child's cognitive development, they are important not just for adolescents but also for adults: "their interplay persists as one of the major features of adult intellectual life" (Bruner 1966, p. 1). Jank and Meyer (2005) explain that even though these representations were developed step by step they were kept in learners' minds throughout their lives. Furthermore, complex mental representations require integration of the three modes – including the enactive mode which is defined by learners' activities. They conclude that the interplay between cognition and activities is integral to the development of mental representation, and therefore to learning (Jank and Meyer 2005, p. 322). Synchronisation of activities with ikonic and symbolic representations provides students with the means for this interplay.

2.2 Related Works

There are already approaches published that aim at learning about informatics concepts linked to the Internet. We implemented a concept within a course at university. In order to provide an enactive approach to the issue of computer networks in the course 'Didactics of Informatics', students had to build a local area network with six notebooks and one hub. They had to configure the notebooks and test their configuration with simple tools like ping or a tool for sending messages to other notebooks. Furthermore, they integrated the notebooks into a user domain. Students were very motivated because they could test their configuration with applications and we provided the students with the output of Wireshark[2] on a whiteboard. This hands-on approach is not feasible in those schools where we conducted curriculum intervention. These schools have no notebooks that can be used for this purpose in informatics courses, because learners would need administrative privileges for network configuration. Corbesero (2003) describes a similar concept for higher education.

Steinkamp (1999) analysed how discovery learning with experiments could be transferred from the natural sciences to informatics. As a prototype for necessary interactive learning material he developed an exploration module for computer networks. The software makes it possible to build computer networks out of different components. After the network is constructed learners can test their environment with a web browser. A second view allows the observation of status messages which can be used to analyse the situation. The components comprise both hardware and software, for example, DNS server, web server and web browser, Internet Protocol, socket, switch, and the abstract component Internet represented by a cloud. These components are connected on the desktop. With this prototype Steinkamp showed how experiments could be performed in informatics education. But there are several restrictions linked to this software. Software and hardware components are not distinguished. Therefore, students cannot see that several services are processes on the same hardware. The abstract component Internet represented as a cloud does not explain the characteristics of the Internet. And it is not possible to configure web services.

Kornelsen et al. (2005) describe an educational software package for discovery learning about Internet services. The concept is based on a generic framework with a layered architecture composed of the three layers user interface, service broker, and Internet services. The user interface provides views for input of requests for Internet services and for representation of received answers. This concept allows extensibility with illustrative and learner specific views, but it provides no complete picture of the Internet. Services are not explored within the application context. Therefore the visible behaviour and the inner structure of the informatics system are not linked. Besides, it is restricted to Internet services at the application layer.

A third approach is the use of virtual machines. It supports the facility to run several machines on the same computer. Each host can be configured and any software can be used. Students work within a familiar environment with original versions of applications. A network analyser such as Wireshark can be used to observe the data exchange. Preconfigured virtual machines are used for given scenarios. The realisation in school practice is limited by problems with hardware and license restrictions. Furthermore, this approach does not provide an ikonic representation of the computer network.

3 Concept of the Educational Software

3.1 View Conception

From the need for students to attain an overall picture of the subject, Brinda (2006) concludes that exploratory software should provide different views on it. While students are familiar with graphical user interfaces of Internet applications, these interfaces hide the (physical) structure of the Internet and data exchange between the components, which are important aspects of the overall picture of Internet applications. The aim of FILIUS is to support students in understanding the functionality of the Internet and applying it. Therefore, the following items have to be considered:

- the items A-C describing characteristics of an informatics system to give students an overall picture;

[2] see http://www.wireshark.org

- construction and simulation to support discovery learning;
- variation of program execution through parameters and modification of the structure of the 'world';
- views assigned to the enactive, ikonic and symbolic modes of representation;
- synchronisation of activities with views providing ikonic and symbolic representations; and
- persistent saving of scenarios for continuation.

We defined four views for the realisation of these requirements. The 'Network View' (I) makes it possible for students to construct local area networks and interconnected networks out of hardware components using a visual representation. These components can also be configured with parameters, for example, the IP addresses of hosts. Connections between hardware components are highlighted during data exchange. The 'Operation View' (II) makes available the virtual graphical user interfaces of hosts, making it possible for users to install or remove software and to start, configure, and use the applications. The 'Message View' (III) allows the observation of data units that are exchanged between hardware components on different layers of the Internet protocol stack, and shows the status messages of selectable components. The 'Code View' (IV) enables students to develop and implement applications of their own or to manipulate these software elements. It provides an editor for source code amendment and manipulation and the facility to compile these programs for direct use.

With these different views learners are able to discover implications caused by changes or manipulations in one view or another. One example is the effect when cutting the connection between two local networks while just one network includes a domain name server. In the network without a domain name server it is no longer possible to retrieve a web page from a web server by its domain name, because the domain name cannot be resolved. Students can analyse this situation by observing the data exchange between the components.

3.2 FILIUS – a Practice-Oriented Approach

FILIUS comprises basic generic functions and two different modes: a design mode for the construction of computer networks and software and a simulation mode for the simulation of Internet applications. The basic functionality involves persistent saving of scenarios comprising computer networks, configurations, and applications. Therefore, it is possible for learners to analyse, extend, modify, or simulate given networks. Additionally, it is possible to compare student solutions in the classroom to explain the solution to a given task.

The design mode enables students to construct, extend or modify computer networks. Hosts, switches, routers, and connections are available to construct local and interconnected networks. Additionally, there is a Modem component, which allows the connection of two instances of this software to build up a larger network with components on different computers. It is possible to configure the components in design mode. Each host must be configured either manually, by explicitly configuring IP address, network mask, and default gateway, or by dynamic host configuration protocol (DHCP) if the local network includes a server that supports DHCP. Each host can provide a DHCP server. Furthermore, advanced students can extend a generic client-server application and modify given components. The simulation mode offers a view of the entire network. A virtual desktop can be opened for each host. The configuration of each host can be modified by installing or removing applications. Several generic tools such as a file explorer and Internet applications and services are supported. Each application provides a simple graphical user interface for configuration and use or starting and stopping of services. Data exchange can be observed within the network representation, with connections highlighted during data exchange and transferred messages shown within a message dialogue box similar to network analysis tools such as Wireshark.

The Internet protocol stack is implemented to ensure that students can do their own experiments. The application layer is implemented with several Internet applications, that is, email, World Wide Web, and peer-to-peer software for file sharing, and Internet services, that is, DNS and DHCP. The transport layer comprises transmission control protocol (TCP), which is used for Internet applications, and user datagram protocol (UDP), which is used for the services DNS and DHCP. The implementation of the Internet layer comprises datagram forwarding with static forwarding tables that are part of the router. Dynamic routing is not supported. This layer also includes the address resolution protocol (ARP), which is necessary for data exchange on local area networks based on IP addressing. The underlying subnet layer is implemented as a simple protocol.

4 Activities of Students

Freischlad and Schubert (2007) describe a hierarchical classification of exercise classes, derived from an analysis of textbooks, for the didactic system 'Internetworking'. Apart from previous knowledge about computer networks and information security they discern five classes at the first level which are concretised at the second level. The first major class is 'Applications', and includes their characteristics, their uses, and their specific realisations as distributed systems on the Internet. The other classes are designed to foster understanding of the functionality of these applications. The 'Protocols' class also includes Internet security because Internet-specific security mechanisms are implemented through protocols. The 'Addressing' class comprises identification of hosts within computer networks. Additionally, this class includes directory services that support information about an address. The 'Data Transfer' class describes what is going on when data has left one end-system and before it reaches the targeted end-system. The last class, 'Architecture', is composed of models and informatics systems that are essential for the architecture of the Internet, considering both hardware and software.

The approach of Stechert and Schubert (2007) for the learning process starts with the analysis of an informatics system. Brinda (2006) demands construction and simulation for discovery learning. Therefore we discern the three activities analysis, simulation (including modification of parameters), and construction. Table 1 displays the supported activities depending on the exercise classes. In the first column the observable characteristics of informatics systems are presented, while in the second and third columns the corresponding views are listed. Applications are constructed in the Code View. The Operation View allows users to manipulate parameters, for example, email configuration or DNS records. The graphical user interface or the Operation View allows observation of visible behaviour, and the Code View allows analysis of implementation details. Protocols

	Analysis	Simulation	Construction
Application	A, C	Operation	Code
Protocols	B	Code	Code
Addressing	A, B	Network	Network
Data Transfer	B	Network	Network
Architecture	A, C	Network, Operation	Network, Operation

Table 1: Support of Student Activities (A is visible behaviour, B is inner structure, C is specific internal details)

of the application layer are constructed and parameters are modified in the Code View. Protocols are not part of the visible behaviour of an informatics system. Therefore, students can analyse the internal structure just by means of the Message View and implementation details via the Code View. The Network View supports construction and modification of networks, that is, Addressing, forwarding (which is part of Data Transfer), and interconnected networks (Architecture). Overlay networks that are necessary for peer-to-peer file sharing are constructed and modified in Operation View. They are also part of the exercise class Architecture. Thus every exercise and activity is supported, while Applications, Addressing, and Architecture are emphasised in consideration of their role in the understanding of informatics systems.

5 Conclusions

The educational software enriches classroom practice, such as the learning process about how local area networks are built up, with student activities. Without this software we required an analytical approach, because it was not possible for students to configure or modify components. This software enables students to build up their own computer network out of different components, which must be configured. After building a network they can test it with applications such as email or WWW. Having simulated a computer network, students can validate their hypotheses by testing its functionality through Internet applications.

The different views of the learning software enable students to acquire an overall picture of selected informatics systems, such as email. For this purpose, the author proposes that learners build up an entire infrastructure with at least two email servers. Furthermore, they should include two computers, each with email software, so that different users can send mails to each other. A DNS server is needed for domain name resolving. Thus, learners are able to connect the user-visible behaviour and the inner structure of the informatics system, the latter being represented by the static view of the computer network and the dynamic view of exchanged data.

6 Acknowledgements

The author gratefully thanks the members of the student project team FILIUS: André Asschoff, Johannes Bade, Carsten Dittich, Thomas Gerding, Nadja Haßler, Johannes Klebert, and Michell Weyer.

References

Brinda, T. (2006), Discovery Learning of Object-Oriented Modelling with Exploration Modules in Secondary Informatics Education, Springer Science, Educ Inf Technol (2006) 11: 105-119.

Bruner, J. S.; Olver, R. R.; Greenfield, P. M. (1966), Studies in Cognitive Growth, John Wiley, New York.

Corbesero, S. G. (2003), Teaching System and Network Administration in a Small College Environment, in JCSC 19, 2 (December 2003), pp. 155-163.

Freischlad, S. (2006), Learning Media Competences in Informatics. in Proceedings of Second International Conference on "Informatics in Secondary Schools. Evolution and Perspectives - ISSEP", Vilnius, Lithuania, pp. 591-599.

Freischlad, S.; Schubert, S. (2007), Towards High Quality Exercise Classes for Internetworking, in Proceedings of IFIP-Conference "Informatics, Mathematics and ICT (IMICT2007): A golden triangle". Boston, USA.

Jank, W.; Meyer, H. (2005), Didaktische Modelle, Cornelsen, Berlin.

Kornelsen, L.; Lucke, U.; Tavangarian, D. (2005), Expedition in das Datenreich: Exploratives Erlernen von Internetdiensten, in Haake, J. M.; Lucke, U.; Tavangarian, D., eds., Proceedings of DeLFI 2005, LNI, GI e. V., Rostock, pp. 271-282.

Kurose, J. F.; Ross, K. W. (2005), Computer Networking: A Top-Down Approach Featuring the Internet, Addison-Wesley, Amsterdam.

Naps, T. L.; Rößling, G.; Almstrum, V.; Dann, W.; Fleischer, R.; Hundhausen, C.; Korhonen, A.; Malmi, L.; McNally, M.; Rodger, S.; and Velazquez-Iturbide, J. A. (2002), Exploring the role of visualization and engagement in computer science education. In Working Group Reports From ITiCSE on innovation and Technology in Computer Science Education.

Stechert, P. (2007), Understanding of Informatics Systems - A theoretical framework implying levels of competence, in A. Berglund; M. Wiggberg, eds., Proceedings of the 6th Baltic Sea Conference on Computing Education Research, Koli Calling.

Stechert, P.; Schubert, S. (2007), A Strategy to Structure the Learning Process Towards Understanding of Informatics Systems, in Proceedings of IFIP-Conference "Informatics, Mathematics and ICT (IMICT2007): A golden triangle". Boston, USA.

Steinkamp, D. (1999), Informatik-Experimente im Schullabor, Diploma thesis, University of Dortmund, Faculty of Informatics, URL: http://www.die.informatik.uni-siegen.de/lehre/diplom/steinkamp/

Tucker, A., ed., (2003), A model curriculum for K-12 computer science: final report of the ACM K-12 task force curriculum committee, ACM, New York.

van Weert, T., ed., (2000), Information and communication technology in secondary education - A Curriculum for Schools, UNESCO, Paris.