Understanding of Informatics Systems: A Theoretical Framework Implying Levels of Competence

Peer Stechert
Didactics of Informatics and
E-learning, University of Siegen,
Germany

stechert@die.informatik.uni-siegen.de

ABSTRACT

Informatics education is concerned with how learners make sense of computational processes and devices in secondary institutions. In this article, we describe a learner-centred cognitive approach to informatics system comprehension for upper secondary education. It is part of a broader research study initiated by experience in first year informatics (CS 1) education at university level. The approach is based on object-oriented design patterns as knowledge representation carrying networked fundamental ideas of informatics and supporting the learning process. We develop a framework for informatics system comprehension consisting of three dimensions, namely learners' competencies (the learner), principles of informatics systems (Informatics) and knowledge representation (Didactics of Informatics). We conclude by describing means to achieve different levels of competence for informatics system comprehension and assign learners' activities to every level of competence.

Keywords

Education Model, Knowledge Representation, Understanding of Informatics Systems.

1. INTRODUCTION

In the knowledge society information technologies are essential in everyday life and mature citizen need informatics system comprehension. But learners have to overcome the cognitive barrier built by the complexity of networked informatics systems and their hidden principles. From a historical perspective, informatics has been a science considering an isolated computing machine. Since the 1990s, regarding computers as single machines is no longer appropriate. They are part of local networks and of the Internet. We make the need of new cognitive relations evident considering the Model Curriculum for K-12 Computer Science of the ACM that describes how to integrate the "study of computers and algorithmic processes, including their principles" into the curriculum [14]. The demand for informatics system comprehension becomes obvious in the mandatory Level II courses: "A major outcome (...) is to provide students with general knowledge about computer hardware, software, languages, networks, and their impact in the modern world" [14]. Accordingly, we define the terms informatics system and informatics system comprehension:

Definition 1: "An informatics system is the specific combination of hardware, software and networking facilities needed to solve some application problem. (...) Informatics, then, is the scientific discipline addressing construction and design of informatics systems" [4].

Definition 2: Informatics system comprehension comprises that learners have knowledge, capabilities and skills with respect to design, construction, application and network structure of informatics systems.

The didactic challenge is to present a successful education model for informatics system comprehension that will foster networked thinking for new cognitive relations.

2. AIMS AND METHODOLOGY

2.1 Related Work

This article is part of a broader research study [10]. Our approach to understanding of informatics systems at upper secondary level is motivated by experience in CS 1 education. From 2003 to 2005, our institute was responsible for the CS 1 course at the University of Siegen including class and lab lessons. Object-oriented modelling was the basis of the course and object-oriented design patterns were introduced, i.e., Composite, Singleton, Observer, Factory Method, Proxy, Model-View-Controller architecture. The work is also based on Schneider's work concerning object-oriented patterns in lower secondary education [9]. Schneider recommends the design patterns Composite, Observer, Interpreter and State [6] and formulates criteria for the selection of design patterns. Layered architectures and design patterns can be connected for the understanding of informatics systems [10]. Our approach to understanding of informatics systems combines educational research on fundamental ideas of informatics [12] and object-oriented design patterns [6]. For this purpose the author developed a didactical classification of design patterns for informatics system comprehension [13].

2.2 Scientific Aims

In the following we outline the main research questions and hypotheses, which are expatiated in Section 3.1, 3.2, and 3.3.

Research Question 1: To understand informatics systems, which characteristics of informatics systems have to be investigated and to which extent?
Hypothesis 1: There are three characteristics of informatics systems to be investigated: external behaviour, internal structure, and specific qualities (implementation details) [4].
Research Question 2: To understand informatics systems, how can we bridge the gap between learning isolated fundamental ideas and informatics system comprehension?
Hypothesis 2: We achieve informatics system comprehension via learning networked fundamental ideas of informatics. Therefore, we need an adequate knowledge representation. Object-oriented design patterns support the learning process for informatics system comprehension because they represent experts' knowledge, they are solutions to recurring design problems and carry networked fundamental ideas of informatics.
Research Question 3: What are levels of competencies according to informatics system comprehension?
Hypothesis 3: For each characteristic in Hypothesis 1, learners' activities have to be described at each level of competence.

2.3 Research Methodology

Our research methodology is inspired by successful researchers, who investigate their theories in practice, i.e., intervention by performing field studies. Prominent examples are Dagiene presenting the activities of Young Programmer's School in Lithuania [5], Hubwieser and Broy introducing a new informatics curriculum, which was "being tested in Bavaria currently" [7], as well as Schulte and Niere who present teaching strategies for secondary schools and evaluate them with teachers [11]. Such research methodology permits critically reflecting the research results.

Definition 3: We call research including field studies *Intervention Based Didactics of Informatics.*

In secondary education, there is the following situation specific to informatics education: one school seldom has enough courses in parallel to enable researchers to design studies involving experimental and control groups. Furthermore, informatics education depends on the school where it takes place, e.g., because of the informatics infrastructure. We apply case study research in the Intervention Based Didactics of Informatics, because it adds knowledge and insights regarding a phenomenon or problem identified by the researcher [2].

We also include the Didactic System [3] in the research methodology. It formalizes the learning process and is a "mapping of a didactic concept to a learning supporting series of informatics modules" [3] to make the learner active. It consists of a) knowledge structures, b) exercise classes, c) exploration modules. Prerequisites of learners, methods, and concepts are represented as nodes in a graph, i.e., a knowledge structure. Exercise classes help defining levels of competencies. Construction, description and use of learning aids like exploration modules have to be integrated into our research methodology. Our research methodology consists of six phases.

Phase 1. Analysis of the research field. We analyse the demands of learners to specify the scientific aims. Studying existing literature offers reasons why the demands are not satisfied. Finally, we analyse Informatics and Didactics of Informatics with respect to possible solutions.
Phase 2. Hypotheses. Identification of research questions demands for coarse-grained hypotheses to improve informatics system comprehension. Result of this phase is a small set of fine-grained hypotheses permitting promising research and curricula intervention.
Phase 3. Development of an education model. The hypotheses have to be combined with common theories of Informatics, Didactics of Informatics and learning theories. We use the theories to construct a learner-centred theoretical framework for informatics system comprehension and derive an education model, which forms the theoretical basis for the learning process in upper secondary education.
Phase 4. Intervention Based Didactics of Informatics. The education model for informatics system comprehension has to be implemented at upper secondary level. We apply a case-based research methodology. It offers results with respect to acceptability by learners and general feasibility. Different learning phases demand for learners' activities and new learning aids, which have to be described and developed. Therefore we consider the Didactic System. The application of the education model including the use of learning software can be implemented by the researcher and together with student teachers. The effect of being researcher and teacher in one person has to be discussed: quantitative studies can suffer from such a constellation, but we aim at qualitative results.
Phase 5. Evaluation. The learners have to take an examination. The results offer feedback concerning general feasibility. Additionally, for the evaluation of the acceptance through the learners, a written questioning is carried out and we interview the teachers of the informatics courses. The results are used for the evaluation of the Didactic System and the hypotheses. Exercise classes are validated and competencies for informatics system comprehension are formulated as a contribution to standards of informatics education.
Phase 6. Feedback. Finally, theory and learning aids will be refined. There will also be further reflection on and contribution to theory. The objective is to create workshops for teachers.

In the remainder of this article, we describe the theoretical framework and derive the education model.

3. THEORETICAL FRAMEWORK

3.1 Principles of Informatics Systems

The first dimension corresponds to the science Informatics. With respect to informatics systems, there are three characteristics [4] to be considered in a holistic way:

External Behaviour. The external behaviour of an informatics system can be investigated by informatics experiments. Experiments can apply a concrete informatics system, e.g., running a small program (or a sequence of slightly different programs) that implements and illustrates fundamental ideas of informatics by its behaviour. Learners describe the results, e.g. by functional models and use case diagrams. Animations of behaviour can be provided by learning software. Fundamental ideas, necessary concepts and design problems are identified.
Internal Structure. In general, the internal structure is only known by developers but not by users. It can be investigated rather through analysis of the components than through experiments. Learners apply different diagrams, e.g., class, object, state, and sequence diagrams to visualize the internal structure. It is necessary to consider dynamic and static representations. Design patterns as solutions to previously identified problems can be applied and connected.
Specific Qualities. They can be understood with the aid of a specification, which can serve a concrete realization (implementation details). Implementation of selected aspects of informatics systems has to be done. Learners must know programming concepts and they need to connect them. Studies show [8] that learners often fail to implement programs even if they know essential programming concepts, because they do not see the whole picture. They fail to network the concepts. It is essential to connect the implementation to results describing the internal structure of an informatics system.

These characteristics of an informatics system lead to Hypothesis 1. As mentioned above, investigation of the external behaviour has to be done in informatics experiments:

Definition 4: An informatics experiment comprises describing question and hypotheses about expected behaviour, preparation of the (technical) environment, realization / execution of the experiment, and refinement of hypotheses if needed. It is contrary to trial-and-error approaches.

The educational value of learning objectives is essential. We choose fundamental ideas of informatics as principles of informatics systems, because they provide objective criteria. A concept of informatics is called fundamental idea if it fulfils the following criteria: 1) it is observable in different areas of informatics; 2) it may be demonstrated and taught on every intellectual level; 3) it can be observed in the historical development of informatics and will be relevant in the longer term; 4) it is related to everyday language and thinking [12]. Fundamental ideas are consensus with regard to their educational value. According to the chain of reasoning in the motivation, learning isolated fundamental ideas has not been the key to informatics system comprehension. Therefore, we suggest focusing on networked fundamental ideas (Hypothesis 2). To network fundamental ideas we need an adequate knowledge representation.

3.2 Knowledge Representation

The second dimension deals with knowledge representations, which make experts' knowledge available for learners. This is necessary to analyze situations and to solve problems. Artificial intelligence uses knowledge representations as formalisms representing rules and facts about a situation or a problem. Concepts and relations between concepts are of interest, which is also the focus of Didactics of Informatics. Examples are virtual machines and models of layers, which are helpful in advanced courses at higher education [10]. Former approaches applying block diagrams fail describing the network aspect of informatics systems, i.e., they describe isolated computing machines. Object-oriented design patterns are solutions to recurring design problems. Applying those sets emphasis on modularization and interfaces, which are essential for informatics systems. Most studies of software design behaviour "relied on one characteristic of designing, the enactment of problem solving skill" [8]. So, we conclude that design patterns are an adequate knowledge representation, because they carry networked fundamental ideas, visualize design heuristics and are solutions to design problems, i.e., we want to improve the learning process towards informatics system comprehension by offering patterns

1. to classify the behaviour of a system,
2. to structure the functioning of a part of a system,
3. to network fundamental ideas inherent in patterns,
4. to network design patterns as (parts of) informatics systems (pattern language),
5. to represent essential principles and heuristics, because they are identified hot-spots of the system, where specific problems and changes of the system occur.

We integrate design patterns into the learning process for a better understanding of informatics systems; not to qualify software engineers. For the learning process, we need a quality factor, e.g. fundamental ideas of informatics. Schwill identified them in the software engineering process [12]. So, they are consistent with Definition 1, because of a strong connection to construction and design of informatics systems. Design patterns represent networked software systems, but they also represent fundamental ideas that also occur in the context of hardware facilities and non-technical issues. They are observable in different areas of informatics. Schneider has shown that design patterns can be learned at different levels of complexity [9].

3.3 Learners' Competencies

The third dimension describes cognitive states, i.e., levels of informatics system comprehension. The levels can be described by exercise classes of the Didactic System [3] and general problems the learner is able to solve. We apply Bloom's Revised Taxonomy [1]. It distinguishes between 1) factual, conceptual, procedural, metacognitive knowledge, and 2) six succeeding cognitive processes, i.e., remember, understand, apply, analyze, evaluate, create. To every cognitive process we assign exercise classes. The characteristics of informatics systems imply the competencies for informatics system comprehension (Figure 1).

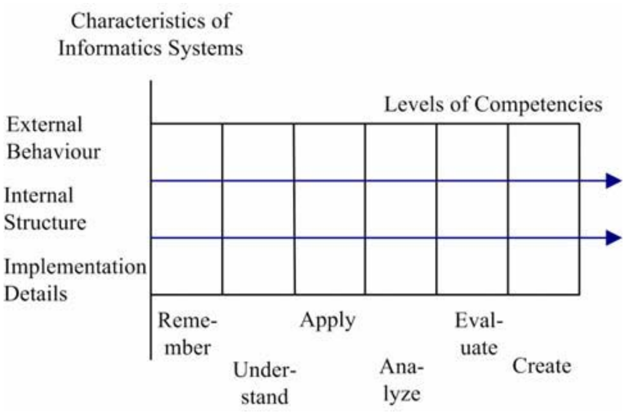

Figure 1. Levels of competencies to be achieved for informatics systems comprehension

The consideration leads to Hypothesis 3.

4. EDUCATION MODEL

A learner-centred classification of design patterns for understanding of informatics systems based on networked fundamental ideas has been done in [13]. For informatics system comprehension, we divide the learning process into three phases (S_i):

S_1: Understanding of essential aspects of external behaviour of the informatics system,

S_2: Understanding of essential aspects of the internal structure of the informatics system that are based on fundamental ideas,

S_3: Understanding of selected qualities from the set of qualities given by the specification of the informatics system (implementation details).

The three phases recur at every level of investigation and form a picture of the whole informatics system. For the learning process towards access control, we exemplarily describe learners' activities according to the levels of Bloom's Revised Taxonomy and apply Proxy, which is a structural design pattern based on object aggregation. It describes a placeholder to control access to another object [6]. The exercise classes are based on results of [3], but they have to be proved according to their effectiveness towards informatics system comprehension. It is important to mention that the education model needs a real-life scenario. e.g., music shop, and the combination of design patterns. An example for a sub-objective of S_1 is

$S_{1,1}$: Understanding of the fundamental idea access control for a list data structure with Proxy design pattern.

S_1 has to be realized by investigating input and output, which is done in informatics experiments. Different prepared programs can show unexpected behaviour, e.g., faults. Learning software showing real-life examples of fundamental ideas is adequate to explore the behaviour. Advanced learners apply functional modelling techniques and case diagrams to describe the behaviour of the informatics system. Exercise classes for $S_{1,1}$ are:

Remember: Questions about the intended behaviour of the informatics system, e.g., a small program "music shop" counting access to songs stored in a list data structure.

Understand: Learners describe the value of access control and answer understanding questions after dealing with an animation of the concept with a strong connection to real-life experiences.

Apply: Learners apply their knowledge, specify the general behaviour of a program and arrange experiments with the program implementing different aspects of the music shop.

Analyze: Learners discuss possible underpinning fundamental ideas, which permits understanding the organizational structure. They cope with unexpected behaviour by distinguishing between faults and correct behaviour.
Evaluate: Experiments with slightly changed programs, e.g., to show variations of access control, imply transformation of knowledge. That means, experiences of previous informatics experiments can be combined with unknown behaviour.
Create: -

Examples for sub-objectives of S_2 are:

$S_{2,1}$: Understanding of access control by designing an object-oriented model of a list and Proxy as validated in $S_{1,1}$,

$S_{2,2}$: Understanding of interfaces and inheritance by applying the Proxy design pattern to Composite design pattern.

For S_2, a documentation of the system, different kinds of modelling (data, functional) and diagrams (class, object, sequence, state diagram) should be used. Exercise classes for $S_{2,1}$ are:

Remember: Questions about involved classes, objects, access control, and problem solving strategies. To be a placeholder of a list, Proxy needs the same interface, which is realized by inheriting from the same abstract class. Such object-oriented concepts belong to the previous knowledge of the learners.
Understand: Understanding questions according to the relation between Proxy and the list. The learners have to eliminate irrelevant relations between classes.
Apply: Arranging the correct relations between the participating classes in a class diagram within a learning environment, e.g., Proxy needs a relation to a list to call it after counting the accesses. Learners construct a state diagram of access control.
Analyze: Role-playing typical scenarios results in understanding the internal processes, e.g., accessing a song.
Evaluate: Learners modify given models and transform them to another representation or another context.
Create: Learners construct an object-oriented model for a music shop, which only allows a certain number of accesses.

An example for a sub-objective of S_3 is

$S_{3,1}$: Understanding of programming parts of the designed object-oriented model including Proxy design pattern.

There has been much research on programming competencies, which we will not explain for every level. It is essential for learners to connect programming concepts and to see the whole informatics system while programming an isolated line of code. So, the exercises of S_3 should include links to S_1 and S_2. In the example, counting access applying a loop and controlling access by programming to a common interface of Proxy and a list have to be connected. We are developing the learning software "Pattern Park" to connect the described phases [13].

5. OPEN QUESTIONS AND DISCUSSION

We present our approach to informatics system comprehension and argue for a design, intervention, evaluation cycle of curriculum development in secondary schools, which has to be discussed. Furthermore, we construct a theoretical framework for informatics system comprehension that demands for regarding different characteristics of informatics systems, choosing valuable informatics content (fundamental ideas) and learning it in a networked way within design patterns as knowledge representation. In combination with the Didactic System the framework permits developing hypotheses for the research questions. It has to be discussed whether and how the research questions have to be refined for scientific investigation of informatics system comprehension at upper secondary level. In particular, we assign learners' activities to the levels of Bloom's Revised Taxonomy. It has to be discussed how to foster networked thinking, how to engage students at the upper levels of the taxonomy, and how to assess their understanding of informatics systems.

6. REFERENCES

[1] Anderson, L. W. and Krathwohl, D. R. (eds.). *A taxonomy for learning, teaching and assessing: A revision of Bloom's Taxonomy of educational objectives.* Addison Wesley Longman, New York, 2001.

[2] Bassey, M. *Case study research in educational settings.* Chapter 7: Methods of enquiry and the conduct of case study research, UK: Open University Press, 1999.

[3] Brinda, T. and Schubert, S. Didactic System for Object-oriented Modelling. In: Watson, D. and Andersen, J. (eds.): *Networking the Learner. Computers in Education.* Kluwer Academic Publisher, Boston, 2002, 473-482.

[4] Claus, V. and Schwill, A. *Duden Informatik*. 4. Auflage, Duden Verlag, Mannheim, 2006.

[5] Dagiene, V. Programming-Based Solutions of Problems In Informatics Curricula. In *IFIP WG 3.1 and 3.5 Open Conference "Communications and Networking in Education: Learning in a Networked Society.* 1999, 88-94.

[6] Gamma, E., Helm, R., Johnson, R., and Vlissides, J. *Design Patterns. Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA, 1995.

[7] Hubwieser, P. and Broy, M. Educating Surfers Or Craftsmen: Introducing An ICT Curriculum For The 21st Century. In *IFIP WG 3.1 and 3.5 Open Conference "Communications and Networking in Education: Learning in a Networked Society. 1999*, 163-177.

[8] McCracken, W. M. Research on Learning to Design Software. In: Fincher, S. and Petre, M. (eds.) *Computer Science Education Research*, Taylor & Francis, London, 2004, 155-174.

[9] Schneider, M. Design pattern, a topic of the new mandatory subject informatics? In van Weert, T. and Munro, R. (eds.): *Informatics and The Digital Society: Social, Ethical and Cognitive Issues*, Kluwer 2003, 157-171.

[10] Schubert, S. From Didactic Systems to Educational Standards. In *Proceedings of the 8th IFIP World Conference on Computers in Education.* Cape Town, South Africa, 2005, Documents/397.pdf.

[11] Schulte, C. and Niere, J. Thinking in Object Structures: Teaching Modelling in Secondary Schools. In: *Proceedings of the ECOOP Workshop on Pedagogies and Tools for Learning Object-Oriented Concepts*, Spain, 2002.

[12] Schwill, A. Computer science education based on fundamental ideas. In Passey, D. and Samways, B. (eds.) *Information Technology - Supporting change through teacher education*. Chapman Hall, 1997, 285-291.

[13] Stechert, P. Informatics System Comprehension - A learner-centred cognitive approach to networked thinking. In: Education and Information Technologies, ISSN: 1573-7608, Springer Netherlands, 2006.

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