Cooperative Creation of Concept Keyboards in Distributed Learning Environments

Benjamin Weyers[1], Nelson Baloian[2] and Wolfram Luther[1]
[1]*Department of Computer and Cognitive Science, University of Duisburg-Essen, Germany*
{luther, weyers}@inf.uni-due.de
[2]*Departement of Computer Science, University of Chile, Santiago de Chile, Chile*
nbaloian@dcc.uchile.cl

Abstract

In this paper we introduce CoBo, a collaborative system supporting cryptographic protocol learning, which offers an interface for the cooperative creation of concept keyboards. Concept keyboards are input interfaces created by the user during runtime with the goal of meeting the user's need for interaction. Experiments with an early version of CoBo have shown that, although students recognize the power of collaborative learning, they found a collaborative simulation of cryptographic algorithms should be associated with an additional step in the learning process: the collaborative design of the concept keyboards that will be used to perform the collaborative simulation of cryptographic protocols.

Keywords: Concept Keyboards, Distributed Learning, Cooperative Creation, Algorithm Visualization.

1. Introduction

Algorithm Visualization (AV) and algorithm animation have been successfully used as computer supported learning (CSL) activities in the past. Most of the approaches described in the literature (cf. [10, 13], etc.) have in common that user interaction with the software is limited to button clicking to control the algorithm execution step-by-step and manipulation of tools to define the initial data for the algorithm to work with, thus giving the learner a mere spectator role without active involvement.

Hundhausen [12] presents an experiment with students creating their own AVs using a basic prototype language. From the results of this experiment, Hundhausen concludes that AV software can improve students' comprehension of the algorithm and specifically that “what learners do, not what they see, may have the greatest impact on learning”.

Motivated by the results from previous research in algorithm visualization and Hundhausen’s observations, we developed an extension of the original approach to AV for classical algorithms like QuickSort and Dijkstra’s algorithm using Concept Keyboards (CK) called ConKAV (cp. Fig. 1). CKs are specialized configurable keyboards in which each key triggers a defined execution step of the algorithm. For each step of the algorithm, the student has to decide which action to choose, thus becoming directly involved in executing the algorithm.

The results obtained encouraged us to develop a new interface supporting the collaborative, distributed learning of cryptographic algorithms called CoBo [30]. With this system, each learner takes the role of an agent, and the whole group simulates the exchange of encrypted messages. Using the system, each agent chooses what operation to perform and when to trigger it. The evaluation of CoBo [5] replicates the good results of ConKAV, confirming the success of the CK approach and also showing the importance of group work.

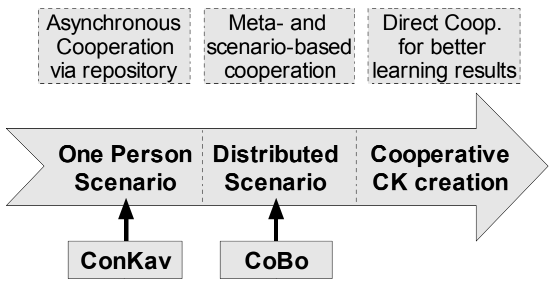

Asynchronous Cooperation via repository
Meta- and scenario-based cooperation
Direct Coop. for better learning results
One Person Scenario
Distributed Scenario
Cooperative CK creation
ConKav
CoBo

Fig. 1: Agenda of learning systems for teaching algorithms using concept keyboards—further and future work

First, the participants were asked to evaluate the software in the single-person scenario, where the CK contains all relevant keys to execute all steps of the protocol. Then, groups of three to four persons were formed to explore the algorithms collaboratively, where the students could use only those keys triggering the actions pertinent to their individual roles. Interestingly, the single-person scenario’s CK design was better rated than the group scenario, where each student could only use the parts of the keyboards relevant to his or her role. This is mainly because students prefer to learn the algorithm using the whole keyboard. However, simulation of the algorithm in the distributed mode with individually configured keyboards operated cooperatively by

978-1-4244-3535-7/09/$25.00 ©2009 IEEE

groups of users allows better exploration of the protocols.

To tackle this problem, we present in this paper a new cooperative approach to distributed learning environments. The system supports the students creating their own keyboards collaboratively since group learning has been shown to be a helpful learning concept in various publications (cf. [1]).

2. Recent work

Since 1988 many AV systems have been developed (see Stasko et al. [13] and Stephan Diehl et al. [10]). Various repositories exist all over the Web for algorithm animation, such as [11]. Kerren, Müldner and Shakshuki describe solutions for algorithm visualization and explanation [17] as well as for Web environments based on hypertext languages [16]. Basic research has also been conducted by Eisenberg [15] and others on algorithm visualization.

Collaborative learning has shown group work to have an important impact on achieving learning success [1, 18]. Small group work presents opportunities for students to share insights [19] and observe the strategies of others [20], which is helpful in the context of algorithm learning.

Different groups are working on the topic of cooperative learning in distributed environments. Most of them deal with cooperative modeling tools, designing cooperative applications, knowledge building, awareness, and so forth.

Our work concentrates mainly on interfaces and interaction methods. The problem with traditional keyboards is that the functionality of the basic keys and their layout do not correctly match the operations of an algorithm and their granularity. CKs are, in our opinion, a highly adaptable solution for algorithm learning [26] that can be even applied to interactive tables for face-to-face activities [27]. The CoBo system has been implemented for desktop PCs as well as mobile devices (PDAs) [6]. Mobile devices allow users to synchronize their actions through face-to-face communication when the system is used in a classroom scenario.

Concept keyboards have been used in the past for other learning environments. Moreover, they have been proved to be highly effective in the context of teaching complex algorithms [6, 14]. Fig. 2 shows an example of a hardware version of a concept keyboard. The touch-sensitive surface detects a finger tip on the sheet of paper lying on it. The sheet of paper is totally flexible and can be redesigned in any way. On a CK, every key can assume a special meaning through the appropriate choice of its label. It can also be customized with additional information or sound.

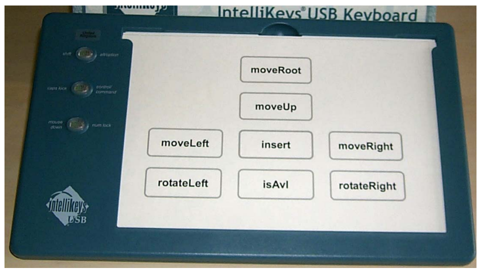


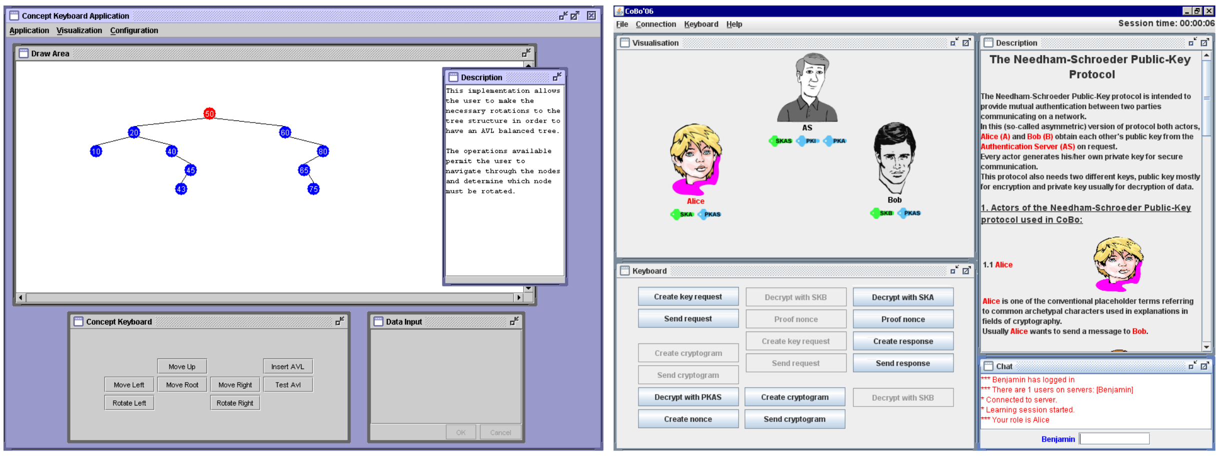
Fig. 2: Hardware CK with overlay

3. Motivation

The central aspect of and motivation for our research lies in computer-supported teaching of algorithms, which includes high dynamic interface creation and usage in the form of Concept Keyboards as well as algorithm visualization and animation. In our work, we have identified three major types of application scenarios:

1. Simulation of processes that solve problems with strict or loose rules.
2. Simulation of processes that involve more than one actor in a distributed environment.
3. Non-deterministic activities in a dynamic system.

In the following subsections we describe our systems and the collaborative CK creation for algorithm learning and give a short overview of future work concerning dynamic and adaptive interfaces in dynamic systems.



```
This implementation allows
the user to make the
necessary rotations to the
tree structure in order to
have an AVL balanced tree.

The operations available
permit the user to
navigate through the nodes
and determine which node
must be rotated.
```

The Needham-Schroeder Public-Key Protocol

The Needham-Schroeder Public-Key protocol is intended to provide mutual authentication between two parties communicating on a network.
In this (so-called asymmetric) version of protocol both actors, Alice (A) and Bob (B) obtain each other's public key from the Authentication Server (AS) on request.
Every actor generates his/her own private key for secure communication.
This protocol also needs two different keys, public key mostly for encryption and private key usually for decryption of data.

1. Actors of the Needham-Schroeder Public-Key protocol used in CoBo:

1.1 Alice

Alice is one of the conventional placeholder terms referring to common archetypal characters used in explanations in fields of cryptography.
Usually Alice wants to send a message to Bob.

```
*** Benjamin has logged in
*** There are 1 users on servers: [Benjamin]
* Connected to server.
* Learning session started.
*** Your role is Alice
```

Fig. 3: Screenshots of ConKAV's (left) and CoBo's (right) user interface

3.1. ConKAV—Simulation of classical algorithms in a one-person learning scenario

ConKAV [24, 25, 27] was our first step in developing systems using CKs for supporting algorithm learning. The idea was to develop a system for algorithm visualization and animation in an individual learning scenario for classical algorithms with new interaction and exploration facilities.

The ConKAV system deals with an interface based on CKs and algorithm visualization (see Fig. 3, left side). The student starts a session followed by direct interaction with the algorithm using the CK's keys in the right context. From the beginning, an important principle in ConKAV's design was for students to conceive and construct their own keyboards.

It is the logical relationship of the key with its functionality in the algorithm that fosters the student's process of algorithm understanding. The software offers the student a choice of more or less meaningful actions to solve underlying problems. The learner has to connect the various actions in the algorithm to individual keys to create a meaningful and helpful keyboard. This creation process helps the student understand the algorithm and offers another perspective on algorithm learning than single animation or visualization issues.

A second important aspect of the ConKAV system is the ability to communicate with other learners asynchronously by using a repository of already created CKs. In his diploma thesis, Kraft [28] describes a scenario showing how this repository can help students understand the algorithm and create appropriate keyboards even in a one-person scenario. The repository contains data and explanatory comments for keyboards that have already been created, and solutions contributed and recorded by other students can inspire new and even better solutions. This exchange of solutions results in knowledge distribution within the student group.

3.2. CoBo—Simulation of cryptographic algorithms in a distributed-group scenario

Evaluations at the University of Duisburg-Essen have shown the great and positive impact of using CKs in the context of learning algorithms [26]. This motivates us to extend our research on using CKs in distributed scenarios for cryptographic learning from one-person to group scenarios. The result is the CoBo system, a computer-supported collaborative learning system based on the visualization and simulation of cryptographic protocols [9] using concept keyboards [6]. It was first developed as a student project in 2006 [30] and then extended in [5] and [7]. This work is also based on the concept of the participatory simulation learning technique [8], which is a learning activity in which the student is directly involved in the learning context as an actor playing a certain role in a simulation.

Cryptographic algorithms and protocols deal mainly with exchanging information between two partners without joint knowledge of a third party.

Various types of algorithms and protocols exist, depending on their basic idea or their respective techniques. All cryptographic protocols have the following in common: Two or more participants are involved in the process, and the communication protocol is fully standardized to hinder intrusion. CoBo offers a framework for simulating and visualizing cryptographic protocols in a distributed way. Different actors in a protocol can be modeled as roles in CoBo and simulated on distributed computers or mobile devices.

In order to use CoBo, the student user has to connect his running CoBo instance to a session that has already started and chose a protocol and a role. After this, the interface presents the role's perspective (see Fig. 3; right side shows the view for the role of Alice in the Needham Schroeder Protocol). The perspective is connected to the role-dependent concept keyboard in the lower panel of the interface. Only the operations to be executed by the agent in the chosen role are enabled.

It is also possible to simulate the protocol using the "teacher's role"; in this role, the user can trigger all the actions of the protocol alone. This role is useful mainly in a one-person scenario, offering the student the possibility of learning alone. In the context of [6], a helpdesk system was implemented to introduce the student easily into the system and to give advice about the interaction.

When considering cryptographic protocols instead of exploring an algorithm, more complex supervision is needed during simulation and interaction with the system. This is even more critical in a distributed scenario. Complex action logic is necessary to meet the requirement of the complex behavior of the system and error recognition in the user's actions. In CoBo, an action logic system is implemented based on Petri nets (PN) [29]. In [22] the component implementing this functionality is described in detail. Fig. 4 shows a brief overview of CoBo's architecture and illustrates the correlation of all components. This component permits modeling and use of complex action logics in the system and therefore offers a great advantage compared to the basic approach in ConKAV. Different complex interaction scenarios in cryptographic protocols can be implemented involving cooperation between the students involved with meaningful and context-sensitive error messages.

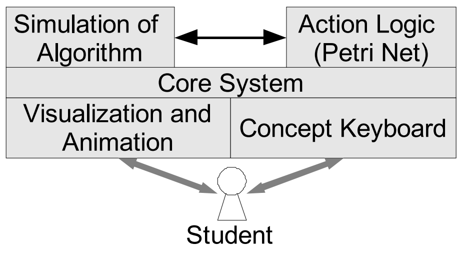


Fig. 4: Overview of CoBo's architecture

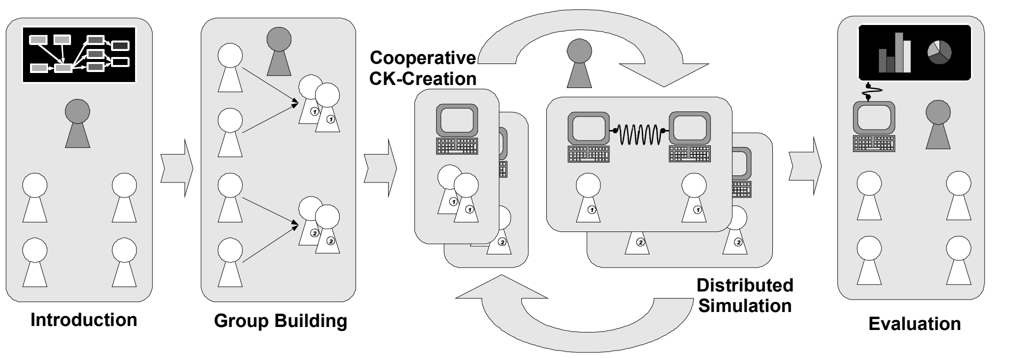


Fig. 5: A new approach to the teaching and learning of distributed algorithm in a distributed environment using an iterative and cooperative creation step of CKs

The following are possible scenarios for learning activities using CoBo.

1. Basic simulation of a "friendly" progress of the algorithm in order to achieve secure communication between partners. This scenario has been shown not typically to be cooperative [5].
2. An extended scenario involving an intruder trying to alter the normal communication. This scenario, which was developed as an example using the Needham Schroeder protocol in [7], involves cooperation because the two main participants in the algorithm have to find a way to lock the intruder out of the communication.

As mentioned in the introduction, the CoBo evaluation has shown that CKs stimulate comprehension, but configurations developed by the users have not really been tried. The intruder scenario suggests that cooperation is only possible in CoBo on a meta-level. The aim of this paper is to show how the system can be used to introduce cooperation into a learning scenario using CoBo and thereby enhance the learning process.

3.3. CK's in a non-deterministic environment—interaction with complex systems

Algorithms are more or less deterministic. Especially in cryptographic protocols, the allowed operations and their order are completely fixed. The idea is to give the user more free space to interact with the process in a more flexible way. The first step in that direction is the development of the intruder scenario, which introduces cooperation into the basic approach of distributed simulation. The two students playing the roles of the two communication partners Alice and Bob have to decide how to deal with the intruder and prevent any impact on the communication; for example, in the Needham-Schroeder protocol, they introduce Bob's name again. In this scenario, the action logic has to deal with more non-deterministic issues and the CK has to meet other requirements: Formally identical actions should be interpreted within their contexts or redefined completely.

The system should give the user the ability to modify the protocol or algorithm in order to try out alternatives to the original protocol that might solve its weakness. Fixing the communication process in the Needham-Schroeder protocol is only possible if the (inter)action logic is able to switch to a new layer in the Interface reconfiguration strategies and sophisticated adaptation of the (inter)action logic. These are topics of our current research but not of this paper. Initial concepts and results can be found in [23].

3.4. Lessons learned

In [5] an evaluation of CoBo with forty computer science students was made and analyzed. It has shown that learners can benefit from using CKs in a distributed learning scenario, but it has also shown some problems in the basic workflow of the system. In addition to other questions, the students were asked whether they preferred to take the teacher's role or an algorithm-specific one. It was interesting that 75% of the students voted for the teacher's perspective. They were arguing that this role ([5, 6])

- Permits better exploration of the protocols,
- Allows more time to consult the help screens without the time constraints that occur in a collaborative scenario, and
- Shows the complete keyboard with all necessary actions in the right order in the start-up configuration, thus leading to deeper understanding when using keys in a different order.

Despite the preference the students had for the teacher's perspective, the group version with its various actors and their roles was clearly perceived as enriching the learning process.

The question is how to move this experience phase in the learning process using the teacher's perspective to a cooperative task among students? In light of the second point, we can formulate the question in a different way: How can the initiation phase (learning by trial and error) be combined with the cooperative step in the learning process? Our answer is the cooperative creation of role-specific CKs that can subsequently be used for the distributed algorithm simulation.

The next step is to integrate both sequential phases of the process into an iterative learning process (cp. Fig. 5). In cooperation with the group and using collaborative tools like the chat in CoBo's interface, progress will be made in learning and information will spread throughout the whole group.

4. Cooperative Concept Keyboard Creation—Realization

Different models describe this knowledge distribution among the members of a group and have been used to implement various collaborative learning scenarios [1, 2, 3]. Based on this research and on well known principles from classical school education [4], we extended the original CoBo learning approach evaluated in [5] by adding a cooperative phase during which the CKs are created that will used in the distributed simulation by the same group (cp. Fig. 5). Combining this cooperative phase with the distributed simulation opens two new perspectives on the learning object, an overview of the simulated process and a chance to get involved with the algorithm at work.

During the creative process, the students have to talk about the algorithm to determine the correct mapping between the pool of operations and a role. An interesting point in the creation of CKs is the fact that the system proposes operations in different granularities. The group has to decide whether the performance of the protocol will be realized step by step following the formal description or whether it will have a set of fine-grained steps combined into a single action.

After this, the group will enter the distributed simulation phase (cp. Fig. 5). New experiences made during the simulation can be discussed collaboratively and then used in a second cooperative adaptation of the concept keyboards. A closing evaluation will consolidate the acquired knowledge.

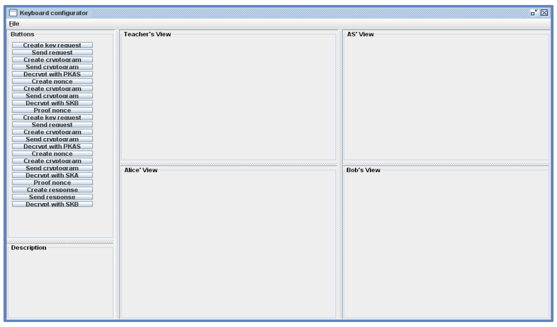


Fig. 6: Mockup for the new keyboard configurator

Cooperative CK creation requires a special extension of the basic CK creator implemented in CoBo. Fig. 6 shows a mockup of the interface. It is separated into two major groups of frames. The first group offers space for the creation of the role-dependent keyboards. The different buttons shown on the left side can be mapped to the different roles by drag and drop. The teacher's view serves as a configuration frame to create the basic positioning of all buttons. It must be decided whether, in the simulation perspective of CoBo, it is more meaningful to show the whole keyboard with disabled buttons or to show only the different keyboards created in the cooperative creation phase. The first approach was used during the evaluation described above. Further evaluation of the cooperative extension should include investigation into the relative advantages of the two approaches.

5. Planned evaluation

An evaluation of the new approach is planned for the 2009 summer term with the aim of determining the impact of the new learning process and CK editor on the learning result. The following scenario will be tested, which differs from the original one as described in [5].

Two groups will be formed. The first will be confronted with the basic scenario of a distributed cryptographic algorithm without intruder. The second will work with an extended version including intruder activity. In both scenarios, the students will be briefly introduced to the topic and to the protocol in standard courses. Once the groups have formed, the first exercise is to cooperatively create a CK (see Fig. 5). The helpdesk, together with information concerning the various operations available to the students, will help them to create keyboards for the algorithm. To do so, they have to understand the different roles and functions of the participants and the various operations they can execute. After they have finished the first creation phase, they will enter the simulation phase, in which they can test their keyboards and take note of possible errors (missing keys/operations, bad design and so on) in their keyboards. In the intruder scenario, this repetition has an additional benefit for the group. Not only can errors in the creation be detected and repaired, but a strategy against the intruder can also be developed in a cooperative way.

6. Conclusion

In this paper we have presented a new approach to algorithm visualization and animation using concept keyboards in an extended learning process with cooperation. Motivated by positive results from former evaluations of our systems for the use of CKs in interactive learning activities we started to think about an extended approach in distributed learning environments and complex process contexts using cooperative techniques for creation of CKs. Cryptographic protocols or algorithms in general are strongly specified and formally described constructs which rarely leave room for creative problem solving like real-life processes. We think that less strictly specified or informally described problems would offer a better scope for using flexible interfaces with adaptive interaction logic. In [23] some ideas concerning nondeterministic dynamic systems in combination with interface design issues are proposed.

A possible scenario could be a simulated interface for a nuclear power plant. A small extract of the process was developed in [21] and could serve as the basis for a learning system for plant operators. The interesting point in this scenario is the impact of dynamically reconfigured interfaces in exceptional circumstances and how to deal with the no predictable interaction behavior of the user. The process is complex enough that several modeling approaches for time-continuous processes with delayed reactions are possible.

Acknowledgement

The work was partially funded by CONICYT project Nr. 1060797.

References

[1] R.S. Nickerson, "On the distribution of knowledge in a group: some reflections", in Distributed Cognition, ed. by Gavriel Salomon, Cambridge University Press, Cambridge, pp. 229–259, 1993.

[2] M. Muehlenbrock, F. Tewissen, H. U. Hoppe, "A Framework System for Intelligent Support in Open Distributed Learning Environments", Proceedings of 8th World Conference on Artificial Intelligence in Education (AIED'97), Kobe, Japan, 1997.

[3] M. Mühlenbrock, A. Loesch, "Eine Architektur zur Unterstützung kollaborativen Lernens in offenen Umgebungen", Proceedings of KI-1998, 1998.

[4] H. Meyer, "Unterrichtsmethoden", Cornelsen, Berlin, 2002.

[5] L. Selvanadurajan, "Interaktive Visualisierung kryptographischer Protokolle mit Concept Keyboards—Testszenarien und Evaluation", Master's thesis, University of Duisburg-Essen, 2007.

[6] N. Baloian, W. Luther, "Cooperative visualization of cryptographic protocols using concept keyboards", to appear in Int. Journal of Engineering Education (IJEE).

[7] A. Kovácová, "Implementierung des Needham-Schroeder Protokolls in einer verteilten Simulationsumgebung für kryptografische Standardverfahren", Master's thesis, University of Duisburg-Essen, 2007.

[8] V. Colella, "Participatory simulations: Building collaborative understanding through immersive dynamic modeling", Journal of the Learning Sciences 9, pp. 471–500, 2002.

[9] G. Cattaneoa, A. De Santisa, U. Ferraro Petrillo, "Visualization of cryptographic protocols with GRACE", Journal of Visual Languages & Computing Vo. 19 (2), 258–290, doi:10.1016/j.jvlc.2007.05.001, 2008.

[10] S. Diehl (ed.), "Software Visualization, State-of-the-Art Survey", LNCS 2269, Springer, 2002.

[11] P. Crescenzi, N. Faltin, R. Fleischer, Ch. Hundhausen, St. Näher, G. Rössling, J. Stasko, E. Sutinen, "The Algorithm Animation Repository, Proceedings of the Second International Program Visualization Workshop", Århus, Denmark, pp. 14–16, 2002.

[12] C. D. Hundhausen, "Toward effective algorithm visualization artifacts: Designing for participation and communication in an undergraduate algorithms course", unpublished Ph.D. dissertation, Department of Computer and Information Science, University of Oregon, 1999.

[13] J. T. Stasko et al., "Software Visualization—Programming as a Multimedia Experience", MIT Press, 1998.

[14] N. Baloian, H. Breuer, W. Luther, "Concept keyboards in the animation of standard algorithms", Journal of Visual Language and Computing, 19 2008, pp. 652–674, 2008.

[15] M. Eisenberg, "The thin glass line: Designing interfaces to algorithms", Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Common ground, ACM Press, pp. 181–188, 1996.

[16] E. Shakshuki, A. Kerren, T. Müldner, "Web-based Structured Hypermedia Algorithm Explanation System", International Journal of Web Information Systems, Vol. 3, pp. 179–197, 2007.

[17] A. Kerren, T. Müldner, E. Shakshuki, "Novel Algorithm Explanation Techniques for Improving Algorithm Teaching", Proceedings of the 2006 ACM symposium on Software visualization, pp. 175–176, 2006.

[18] N. M. Webb, "Peer interaction and learning in small groups", International Journal of Educational Research, 13, pp. 21–39, 1989.

[19] M. C. Bos, "Experimental study of productive collaboration", Acta Psychologica, 3, pp. 315–426, 1937.

[20] M. Azmitia, "Peer interaction and problem solving: When are two heads better than one?" Child Development, 59, pp. 87–96, 1988.

[21] H. Boussairi, "Petrinetz-basierte Implementierung eines verfahrenstechnischen Prozessmodells—SOM-basierte automatische Überwachung der Mensch-Maschine-Interaktion", Master's thesis, University of Duisburg-Essen, 2008.

[22] B. Weyers, "Concept Keyboards zur Steuerung und Visualisierung interaktiver kryptographischer Protokolle CoBo'06", University of Duisburg-Essen, 2006.

[23] B. Weyers, "An error-driven approach for automated user-interface redesign—concepts and architecture", Proceedings of the DAAD summer university Chile, Logos, Berlin, pp. 104-113, 2008.

[24] N. Baloian, H. Breuer, W. Luther, W. Chr. Middleton, "Algorithm visualization using concept keyboards", Proc. ACM SoftVis'05, St. Louis, Missouri, pp. 7–16, 2005.

[25] N. Baloian, W. Luther, Th.. Putzer, "Algorithm Explanation Using Multimodal Interfaces", IEEE Press CS SCCC 2005, Valdivia, pp. 21–29, 2005.

[26] N. Baloian, H. Breuer, W. Luther, "Concept keyboards in the animation of standard algorithms", Journal of Visual Language and Computing, 19, pp. 652–674; doi: 10.1016/j.jvlc.2007.07.005, 2008.

[27] M. Ringel Morris, A. M. Piper, A. Cassanego, T. Winograd, "Supporting Cooperative Language Learning: Issues in Interface Design for an Interactive Table", http://hci.stanford.edu/cstr/reports/2005-08.pdf, 2005.

[28] Ph. Kraft, "Algorithmenvisualisierung mit selbstkonfigurierten Schnittstellen—Implementation und Evaluation", diploma thesis, University of Duisburg-Essen, 2005.

[29] B. Baumgarten, "Petri Netze. Grundlagen und Anwendungen", Spektrum, Heidelberg, 2006.

[30] T. Kotthäuser, A. Kovacova, W. Liu, W. Luther, L. Selvanadurajan, M. Wander, S.Wang, B. Weyers, A. Yapo, K. Zhu, "Concept Keyboards zur Steuerung und Visualisierung interaktiver krypthographischer Algorithmen", Praxisprojekt, University of Duisburg-Essen, 2006.