The Theory Behind Theory - Computer Science Education Research Through the Lenses of Situated Learning

Maria Knobelsdorf

Universität Hamburg, Computer Science Department
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
knobelsdorf@informatik.uni-hamburg.de

Abstract. This paper introduces key characteristics of the situated learning approach and discusses from that perspective questions of pedagogy and educational research in Theory of Computation. This discussion exemplifies how a change in learning theories alters the unit of analysis, thus reframing research questions and potential answers. In its conclusion, this paper provides an outlook on potential research questions in secondary Computer Science Education.

1 Introduction

In the research community of Computer Science Education (CS Ed), the awareness for discussing and explicitly incorporating theoretical frameworks into research is constantly rising [17], [6]. Theories and concepts of how individuals learn play an important role in educational research because they not only affect which research questions are posed and what kind of data collection and analysis methods are chosen, but more importantly influence the development of pedagogical concepts and interventions. While cognitivist and constructivist concepts of learning are established frameworks in CS Ed, recent theories and related discourses from educational psychological research are still being introduced and discussed. Theories that go under the names *situated learning* [17], *sociocultural learning* [23], *situated cognition theory* [3], *distributed intelligence* [21], or *activity theory* [11] have been developed over the last three decades and started to become more important in field of CS Ed research, see for example [2], [10], [14], [16], [22], [26]. Likewise, in other educational disciplines the situated cognition theory became influential [19], which also inspired the development of comparable approaches for secondary CS Ed [7], [13]. These new approaches extend and challenge established beliefs and understanding of learning and therefore the corresponding research programs and their achievements in related pedagogies and didactics.

In this paper, I summarize key characteristics of the situated learning approach and discuss from that perspective questions of pedagogy and educational research in CS Ed. For this matter, I rely on previous work, i.e., especially [26], a review of sociocultural cognition theory, as well as [15], [13], [14], where parts of concepts and arguments presented in this paper have been already introduced and discussed.

A. Brodnik and J. Vahrenhold (Eds.): ISSEP 2015, LNCS 9378, pp. 12–21, 2015.
DOI: 10.1007/978-3-319-25396-1_2

In particular, I summarize modifications to the pedagogy of an undergraduate Theory course held at the University of Potsdam, Germany. Here, I contributed to by taking into account the pedagogical approach of cognitive apprenticeship which led to a significant reduction of the course's failure rates in the final exam [15]. I will reflect these changes from the perspective of situated learning and draw conclusions for further research investigations in the educational scope of Theory of Computation.

The situated learning perspective represents a paradigm change from many other kinds of psychological frameworks [25]. Some of the concepts do not have straightforward mappings to established psychological theories, and must be understood as part of a larger, but different, theoretical whole. Such a paradigm shift in psychological theory may engender the kinds of cognitive dissonance for the readers that are also felt by an experienced imperative programmer on first encountering an object-oriented language. Therefore, readers with a strong background in CS and cognitive theories may find it challenging to adopt and appreciate this way of thinking. However, the useful new lenses that the approach offers is worth the endeavor because it significantly broadens our understanding of learning on which sustained innovation research for CS Ed can unfold.

2 Theoretical Framework

2.1 The Situated Learning Perspective

The cognitive view on learning has its roots in cognitive psychology and artificial intelligence research. Learning is conceptualized as a process in which individuals create a mental representation of a specific knowledge entity in their minds. A person's cognitive processes operate on such mental models, are based on logic-like rules of inference, and are understood to happen solely in the person's mind. Hence, the approach focuses on the question how specific knowledge is acquired and represented in the mind of an individual ([26], p. 5-6). The cognitive approach was criticized for being too much focused only on the single student and solely on his or her cognitive process while neglecting the social and cultural environment in which students' learning takes place [1], [3], [7].

The situated learning approach has its roots in Russian cultural-historical psychology developed by Vygotsky [27] and was strongly influenced by insights from artificial intelligence, as well as developmental psychological research ([26], p. 5-6). From a situated perspective, learning is conceptualized as a process of enculturation into a domain-specific community. The latter is often called "Community of Practice" (CoP), a term coined by Lave and Wenger [17], in order to emphasize "practices in which individuals have learned to participate, rather than on knowledge that they have acquired" ([7], p. 8). From this perspective, the goal of learning is to become a full member of a CoP.

In the trajectory of participation, individuals of a CoP learn to interact with each other mediated by material and representational systems, which are often metaphorically referred to as *tools*. In this context, the term "tool" denotes material objects such as pencils, hammers, automobiles, or coffee machines as well as representational systems such as "language; various systems for counting; mnemonic techniques;

algebraic symbol systems; works of art; writing; schemes, diagrams, maps, and mechanical drawings" and similar ([28], p. 137). Both aspects of a tool affect the material and mental activity of an individual in a CoP.

Tools do not simply arise de novo in the hands and minds of individual actors. Rather, they are provided to individuals by the surrounding culture of a CoP, accreting over time and, passed from one generation to the next. As Pea points out, tools "represent some individual's or some community's decision that the means thus offered should be reified, made stable, as a quasi-experiment form, for use by others. In terms of cultural history, these tools and the practices of the user community that accompany them are major carriers of patterns of previous reasoning" ([21], p. 53). Cultural practices of tool use evolve in tandem with the evolution of the tool. Therefore, tools represent socially distributed cultural entities that implicitly embed collective knowledge of their use ([26], p. 8-10).

2.2 Implications for Research and Formal Education

One of the most important implications for educational research is that the understanding and specific skills that students develop during an activity depend strongly on the tools used to carry out the activity. In consequence, mental processes, tool use, and interaction with the world are regarded to be tightly bound together: "This has the important implication that when understanding learning, we have to consider that the unit that we are studying is people in action using tools of some kind (see Wertsch, 1991, 1998; Säljö 1996). The learning is not only inside the person, but in his or her ability to use a particular set of tools in productive ways and for particular purposes." ([24], p. 147). In addition, Greeno points out that "[w]hen an analysis of an individual's knowing is proposed, the analysis should be an account of the ways that the person interacts with other systems in the situation. Just presenting hypotheses about the knowledge someone has acquired, considered as structures in the person's mind, is unacceptably incomplete, because it does not specify how the other systems in the environment (including other people) contribute to the interaction." ([7], p. 8).

Regarding formal education, a major critique is that parts of practices and tool use of a CoP are singled out, formalized, and organized around a curriculum with courses, assignments, and tests. This is depriving a community's practice of the coherent whole it represents within the community making it difficult for students to adopt the practice and become engaged. Lave and Wenger argue that learning in conventional schools "is predicated on claims that knowledge can be decontextualized" (p. 40) and "suggest that learning occurs through centripetal participation in the learning curriculum of the ambient community. Because the place of knowledge is within a community of practice, questions of learning must be addressed within the development cycles of that community" ([17], p. 100). Therefore, educational settings in school need to embed curriculum topics in authentic contexts in which students can better recognize the full meaning and reasons of specific practices and tools of a CoP.

Ben-Ari among others has pointed out that formal education of a specific domain has slightly different goals as well as constraints than traditional apprenticeship-like settings of education. Besides economical reasons, secondary but also tertiary education is designed to enable students broad participation in different domains or subfields of a domain and preclude a premature determination of future occupation ([2], p. 88). Students are supposed to enculturate into different domains enough to get acquainted with a broad range of practices from different disciplines or subfields and later on choose a specialization in order to fully grow into the CoP of their choice.

On the other hand, a school also represents a CoP with a specific culture, practices, tools, members, etc. and Greeno points out that "[m]ethods of instruction are not only instruments for acquiring skills; they also are practices in which students learn to participate. In these practices, students develop patterns of participation that contribute to their identities as learners, which include the ways in which they take initiative and responsibility for their learning and function actively in the formulation of goals and criteria for their success." ([7], p. 9). In consequence, when discussing educational settings and related research more attention needs to be paid how student enculturation takes place and how the curriculum, pedagogical approaches, teacher education, etc. are supporting students in this process.

Collins et al. argue that in traditional apprenticeship tasks or activities required to be accomplished by the students make sense for them because it is part of a coherent whole and arises from the demands of a specific workplace [5]. For this matter, educational settings need to put forward all aspects of practices and tools of a community and teach them with regard to students' enculturation process. Especially in higher education, where students are supposed to adopt the expertise of a particular scientific community, it is not enough paid attention "to the reasoning and strategies that experts employ when they acquire knowledge or put it to work to solve complex or real-life tasks. […] To make real differences in students' skill, we need both to understand the nature of expert practice and to devise methods that are appropriate to learning that practice" ([5], p. 38-39).

In the next section, the situated learning approach and these implications will be discussed within the context of Theory of Computation.

3 Enculturation into Theory of Computation Community

At German universities, Theory of Computation is considered one of the fundaments of academic CS education and introductory courses to Theory of Computation are an integral part of undergraduate CS Ed programs. This is also the case at the department of CS at the University of Potsdam, Germany. The Theory of Computation courses cover the foundations of automata, programming languages, computability, and computational complexity. The corresponding concepts, theories, and algorithms are strongly mathematical in nature, regarding formalized inscriptions used for the discourse as well as a strong focus on mathematical proofing of presented theories and approaches (for better overview see for example [9]). By introducing idealized mathematical models of the computer and discussing methods for designing and analyzing them, students are supposed to develop the ability of thinking abstractly about computational processes and models. However, before the introduced modifications in the course's pedagogy many students had problems in achieving these goals and failure rates in final exams were very high (usually between 30-60%).

From the situative perspective, Theory of Computation is a specific CoP within the bigger community of CS. The Theory community includes members which mostly work at academic institutions like universities and therefore the discourse of the community is strongly based on academic research practices. In addition, the community's culture is strongly affected by mathematical practices and inscriptions. Academic education in this field can be regarded as the first step towards enculturation into this specific CoP. Since members of this community teach Theory courses at the university, consequently the pedagogy of these courses implicitly represents the community's beliefs and practices of how to enculturate newcomers. This includes the topics covered in these courses as well as goals and assumptions of how students are supposed to learn and work successfully. This needs to be taken into account when arguing for changes or improvements of established pedagogies in this field.

3.1 Teaching Practices: Theory of Computation

Until the course setup and its pedagogical approach were modified, the Theory course consisted of the following components, which are typical for an introductory CS course in Germany:

- Approximately 90-120 minutes of lectures per week given by a faculty member who presents the course topics, central concepts, algorithms, and their proofs and illustrates them with examples.
- Weekly homework assignments based on the current lecture topics, which students are expected to solve individually and submit in writing for reviewing and grading by tutors (usually senior students). Handing in homework can but doesn't need to be mandatory.
- Approximately 90 minutes student session per week attended by approximately 25-30 students and chaired by teaching assistans, during which students are expected to present their solutions to last weeks' homework assignment. The objective of these sessions is to give them an opportunity to check the correctness of their solutions and discuss them with the group.
- Summative assessment by the end of the course including several assignments to be solved in written form.

Within this pedagogical approach it was implicitly assumed that during the lectures students can follow and understand the presented concepts, theorems, and corresponding proofs, and are also able to deduce from the presented practices how to handle and work on the weekly homework assignments by themselves. In addition, student sessions were supposed to serve students as verification and improvement of their self-directed development of practices in Theory of Computation. However, teaching assistants reported that students had difficulties coping with the contents of the course due to its abstract and theoretical nature and that students seemed not to know how to tackle weekly homework assignments.

The situated learning theory drew attention to the Theory practices and how they were taught in the course. It was observed that these were not addressed and exposed sufficiently. Since all lecture topics were prepared in advance for a smooth presentation, students did not experience the enormous effort it took to create them, especially since the historical dimension of this field was not part of the curriculum. In consequence, students did not experience on a regular basis how the teacher or professor (who represents the expert member of the Theory of Computation CoP) is approaching a new problem, trying out different approaches, making mistakes, taking notes, etc. before creating a solution.

Altering the pedagogy of the course, the main idea was to use student sessions to demonstrate and discuss relevant techniques of how to solve specific problems so that students are better prepared to solve their homework by themselves (for more details see *scaffolding & fading* method [5]). Also, the textbook used in the course was chosen with regard to a strong emphasis on explaining and reflecting presented approaches [9].

Furthermore, important tools in Theory of Computation are mathematical inscriptions and visualizations both created with pen and paper or chalk and blackboard. These tools are used to describe, specify, and reason about ideas, approaches, examples and potential solutions. We found it important to demonstrate and explain students how these tools are meant to be used in this field of discourse. For this reason, we offered specific preparatory exercises that were solved jointly during the student sessions and served as a preparation for the homework assignments. These preparatory exercises included detailed written solutions with extensive reflections and explanations of the solution, providing insights about the used technique, method or strategy. Also, the solutions explicated how mathematical inscriptions are used in this field and are expected to be used by the students when they turn in their homework.

Redesigning the student sessions, the major change was to align the preparatory exercises with homework assignment in respect to structure and content. This means that in each session the same type and amount of problems was used for the exercises as well as for the homework assignments. In addition, it corresponds to students' activities during final exam, where students have to formulate a written solution for an assignment, which then will be graded by tutors under supervision of the instructors. In order to understand the expectations, especially with respect to the very strict and formal character of solutions to assignments, students need to train this skill and to receive a weekly feedback about their efforts. Since the final exam represents a situated activity particularly relevant of the overall pedagogy of the course, we believed that students must be prepared for this as well, especially when this is their very first university exam. For this reason, we also started to offer a *pre-exam* after the first half of the course. The pre-exam gave students an opportunity to practice the assessment situation and explicates what will be expected from them during the finals.

Students acknowledged the described pedagogical changes. In a survey conducted during the course, they reported to feel comfortable with the weekly homework and overall expectations of the course since they would know how to work on their assignments. As a result, the failure rate decreased 60% to below 10% while keeping the requirements for final exams comparable to those of the previous years. However, other CS colleagues, with whom we talked about this approach, were concerned that students are not fully acquainted with what they called “real” Theory of Computation. They argued that students are just trained to understand and apply presented solutions and are not learning to develop solutions to given computational problems on their own. This argument is very important because it emphasizes 1) how relevant the ability of *solving* computational problems is in the community of Theory of Computation in comparison to understanding given solutions and 2) the expectation that students should develop this ability without any scaffolding and from the very beginning of attending a Theory course. We argued that in order to be able to develop solutions to theoretical computational problems students should first understand and apply given solutions and the success of our students proved this approach to be right. However, there is no substantial empirical evidence showing how students best develop this important ability in Theory of Computation and what kind of pedagogy is required for supporting this adequately. Investigating this research question would require revealing the community’s implicit beliefs about educational goals and pedagogical practices in this subfield of CS.

3.2 CS Students’ Engagement in Theory of Computation

Since the majority of CS graduates do not pursue a research career in Theory of Computation, it can be concluded that except for a temporary enculturation, CS students do not intend to become full members of the Theory of Computation CoP. In addition, in the survey conducted during the Theory course at the University of Potsdam, most CS students reported that they did not find the course topics to be particularly interesting and only attended the course because it was mandatory. When asked for detailed reasons, most students explained that the purpose of studying Theory of Computation was unclear to them. Therefore, it seems to beimportant to explicitly provide students with reasonable rationales for studying this field of CS in order to foster their interest in becoming more engaged in Theory practices.

One line of reasoning for including Theory of Computation into the CS curriculum, which is voiced regularly in the community, is that prospective computer scientists should be familiar with the theoretical underpinning of computing. Students who intend to become members of the software engineering community need to be familiar with topics like complexity or design and analysis of algorithms. For this matter, short examples of a topic’s possible applicability are usually provided during the lecture and this was also the case in the course at the University of Potsdam. The challenge is that CS students really start to understand this argument much later in their education and therefore require additional reasons to spark their interest when being newcomers to the Theory of Computation CoP.

In line with the purpose of enculturating students, presenting students with computational problems and teaching them practices of how to develop solutions and proof their correctness both emphasize the CoP’s focus on these activates. In addition, it can be helpful to explain how theoretical computer scientists are motivated to work on computational problems. Addressing this goal, Chesñevar et al. [4] introduced “biographical notes, videos and articles associated with the historical context in which the theory of computing emerged as a new discipline" (p. 8). The authors reported that students responded very well to this historical perspective of theoretical CS, developed a deeper understanding, and became therefore more engaged. It seems that the historical perspective is a good approach not only to help students in creating meaning but also the teachers of the course to reflect and emphasize on the reasoning and strategies of creating this body of knowledge. Nevertheless, it is not yet clear if understanding the CoP's motivation will actually make it meaningful for students as well. In this context, another research question is how practices of Theory can contribute to students' practices of other CS fields, e.g., modeling or programming and – if there is evidence for such transfer – how this can be used to engage students in addition.

4 Conclusion and Outlook

This paper introduced key characteristics of the situated learning theory and discussed questions of pedagogy and educational research in the field of Theory of Computation from this perspective. This discussion exemplifies how the change in learning theories alters the unit of analysis, thus reframing research questions and potential answers. Situated learning theories focus on a unit formed by individuals, interacting with each other and with tools emphasizing patterns of participation and the culture of learning in this unit of analysis.

Discussing questions of learning and formal education in the field of Theory of Computation is exemplary because tertiary education in this field is usually implemented by the members of the related (scientific) Community of Practice (CoP). The trajectory of enculturation is intended to start with introductory courses to Theory of Computation and leads to advanced seminars with open problems from the field. On the other hand, introductory courses to programming are more complex regarding the questions of community belonging and enculturation. The latter is meant towards a CS community, but this is a shared roof of different subcommunities of CS with in parts very different belief systems, practices, tools, or traditions and therefore different cultures of learning and practices of enculturation. These can be for example: web development in the field of e-commerce, the development of embedded aviation systems, or distributed development of operating systems in open source projects. The relation to these different CoPs becomes particularly challenging when creating meaningful learning situations and teaching students programming practices beyond presentations of factual knowledge of programming syntax and exemplifying simple algorithms. What seems to be complex regarding tertiary CS education is even more so in secondary CS education. For instance, in Germany the primary goal of secondary CS Ed is not to engage and prepare students for tertiary CS Ed, but rather to enable students to interact consciously and well informed with information technology and use modeling skills "for understanding and solving problems in other fields of inquiry but also as tools for exploring and producing cultural artifacts" ([12], p. 6). Programming or coding is explicitly not regarded to be the most important practice students are supposed to adopt in CS class. From the situative perspective, this notion of CS Ed can be questioned as follows: What is the related CoP of these taught practices? What kinds of practices and tools determine the activities of this community and are they incorporated in the educational standards? What is the culture of learning and what are the practices of enculturation in CS class? What kind of pedagogy supports this enculturation and how does it need to be implemented regarding CS teacher education and lesson design? However, all these questions just refer to the notion of secondary CS Ed envisioned by the CS Ed community.

Student engagement is yet another facet to be taken into account: What kind of enculturation did students encounter before taking a CS class in high school? What are their expectations and belief systems regarding information technology and CS and is this aligned with the notion of CS Ed we are offering in secondary schooling? How is offered CS Ed contributing to their evolving identity as learners? Using a theoretical framework based on situated learning, Kolikant and Ben-Ari (2008) investigated how students and their teachers in a concurrent and distributed computing course in high school were interacting with each other. They observed different cultures of students and teachers that lead to a "clash of culture" and in consequence to learning difficulties in the CS classroom [16], see also ([26], p. 16). In order to overcome the cultural clash, Kolikant and Ben-Ari created a *fertile zone of cultural encounter*, a pedagogical innovation that bridges between student beliefs and expectations of CS and the language and formalisms of the professional culture presented by the CS teacher. This study exemplifies how the situated learning approach leads to questions of educational research and pedagogy beyond knowledge acquisition.

References

1. Anderson, J.R., Reder, L.M., Simon, H.A.: Situative versus cognitive perspectives: Form versus substance. Educational Researcher 26(1), 18–21 (1997)
2. Ben-Ari, M.: Situated learning in computer science education. Computer Science Education 14(2), 85–100 (2004)
3. Brown, J.S., Collins, A., Duguid, P.: Situated Cognition and the Culture of Learning. Educational Researcher 18(1), 32–42 (1989)
4. Chesñevar, C.I., González, M.P., Maguitman, A.G.: Didactic strategies for promoting significant learning in formal languages and automata theory. In: Proceedings of the 9th Annual SIGCSE Conference on Innovation and Technology in Computer Science Education (ITiCSE 2004), pp. 7–11. ACM (2004)
5. Collins, A., Brown, J.S., Holum, A.: Cognitive apprenticeship: Making thinking visible. American Educator 6(11), 38–46 (1991)
6. Daniels, M., Pears, A.: Models and methods for computing education research. In: Proc. Australasian Computing Education Conference (ACE). CRPIT, vol. 123, pp. 95–102 (2012)
7. Gramm, A., Hornung, M., Witten, H.: Email for you (only?): design and implementation of a context-based learning process on internetworking and cryptography. In: Proceedings of the 7th Workshop in Primary and Secondary Computing Education (WiPSCE), pp. 116–124. ACM, New York (2012)
8. Greeno, J.G.: On claims that answer the wrong questions. Educational Researcher 26(1), 5–17 (1997)
9. Hopcroft, J.E., Motwani, R., Ullman, J.D.: Introduction to Automata Theory, Languages, and Computation, 3rd edn. Pearson (2006)
10. Hundhausen, C.D.: Toward Effective Algorithm Visualization Artifacts: Designing for Participation and Communication in an Undergraduate Algorithms Course. Doctoral Thesis, University of Oregon, US (1999)

11. Kaptelinin, V., Nardi, B.: Acting with Technology –Activity Theory and Interaction Design. MIT Press, Cambridge (2004)
12. Knobelsdorf, M., Magenheim, J., Brinda, T., Engbring, D., Humbert, L., Pasternak, A., Schroeder, U., Thomas, M., Vahrenhold, J.: Computer Science Education in North-Rhine Westphalia, Germany – A Case Study. ACM Transactions on Computing Education 15(2) (2015)
13. Knobelsdorf, M., Tenenberg, J.: The context-based approach iniK in light of situated and constructive learning theories. In: Diethelm, I., Mittermeir, R.T. (eds.) ISSEP 2013. LNCS, vol. 7780, pp. 103–114. Springer, Heidelberg (2013)
14. Knobelsdorf, M., Isohanni, E., Tenenberg, J.: The reasons might be different – why students and teachers do not use visualization tools. In: Proceedings of the 12th Annual Finnish/Baltic Sea Conference on Computer Science Education (Koli), pp. 1–10. ACM, New York (2012)
15. Knobelsdorf, M., Kreitz, C., Böhne, S.: Teaching theoretical computer science using a cognitive apprenticeship approach. In: Proceedings of the 45th ACM Technical Symposium on Computer Science Education (SIGCSE), pp. 67–72. ACM, New York (2014)
16. Kolikant, Y.B.-D., Ben-Ari, M.: Fertile Zones of Cultural Encounter in Computer Science Education. Journal of the Learning Sciences 17(1), 1–32 (2008)
17. Lave, J., Wenger, E.: Situated Learning: Legitimate Peripheral Participation. Cambridge University Press (1991)
18. Malmi, L., Sheard, J., Simon, B.R., Helminen, J., Kinnunen, P., Korhonen, A., Myller, N., Sorva, J., Taherkhani, A.: Theoretical underpinnings of computing education research: what is the evidence? In: Proceedings of the International Workshop of Computing Education Research (ICER), pp. 27–34. ACM, New York (2014)
19. Nentwig, P.M., Demuth, R., Parchmann, I., Gräsel, I., Ralle, B.: Chemie im Kontext: Situated Learning in Relevant Contexts while Systematically Developing Basic Chemical Concepts. Journal of Chemical Education 84(9), 1439–1444 (2007)
20. Nunes, T.: What organizes our problem solving activities? In: Resnick, L., Saljo, R., Pontecorvo, C., Burge, B. (eds.) Discourse, Tools, and Reasoning: Essays in Situated Cognition, pp. 288–311. Springer (1997)
21. Pea, R.: Practices of distributed intelligence and designs for education. In: Salomon, G. (ed.) Distributed Cognition: Psychological and Educational Considerations, pp. 47–87. Cambridge University Press (1993)
22. Peters, A.-K., Rick, D.: Identity development in computing education: theoretical perspectives and an implementation in the classroom. In: Proceedings of the 9th Workshop in Primary and Secondary Computing Education (WiPSCE), pp. 70–79. ACM, New York (2014)
23. Rogoff, B.: The Cultural Nature of Human Development. Oxford University Press (2003)
24. Säljö, R.: Learning as the use of tools: a sociocultural perspective on the human-technology link. In: Littleton, K., Light, P. (eds.) Learning with Computers, pp. 144–161. Routledge, New York (1998)
25. Sfard, A.: On Two Metaphors for Learning and the Dangers of Choosing Just One. Educational Researcher 27(2), 4–13 (1998)
26. Tenenberg, J., Knobelsdorf, M.: Out of Our Minds: A Review of Sociocultural Cognition Theory. Computer Science Education 24(1), 1–24 (2014)
27. Vygotsky, L.S.: Mind in Society: The Development of Higher Psychological Processes. Harvard University Press (1978)
28. Vygotsky, L.S.: The instrumental method in psychology. In: Wertsch, J.V., Sharpe, M.E. (eds.) The Concept of Activity in Soviet Psychology (1981)

http://www.springer.com/978-3-319-25395-4

Informatics in Schools. Curricula, Competences, and Competitions
8th International Conference on Informatics in Schools: Situation, Evolution, and Perspectives, ISSEP 2015, Ljubljana, Slovenia, September 28 - October 1, 2015, Proceedings
Brodnik, A.; Vahrenhold, J. (Eds.)
2015, XVI, 183 p. 35 illus. in color., Softcover
ISBN: 978-3-319-25395-4