The Reasons Might be Different – Why Students and Teachers Don't Use Visualization Tools

Maria Knobelsdorf
University of Oldenburg
Department of Computer Science
26111 Oldenburg, Germany
+49 441 798 2990
knobelsdorf@uni-oldenburg.de

Essi Isohanni
Tampere University of Technology
Insinöörinkatu 42 B 37
33720 Tampere, Finnland
+358 40 8490717
essi.isohanni@tut.fi

Josh Tenenberg
University of Washington, Tacoma
Computer Science and Systems
Tacoma, WA 9802-3100, USA
+1 253 692 5800
jtenenberg@uw.edu

ABSTRACT

In this paper, we address the problem that most teachers and students tend not to use existing visualization tools for teaching and learning programming, respectively, although visualization tools are one of the most investigated research fields in Computer Science Education. We discuss possible reasons of the problem mentioned above as well as directions for future research based on Activity Theory, a theoretical framework from developmental psychology. Therefore, this is a philosophical paper, with the purpose of briefly presenting those aspects of Activity Theory that are most relevant to the development of program visualization tools, and pursuing the implications of this theory for deepening our understanding of how these tools impact teaching and learning.

Categories and Subject Descriptors

K.3.2 **[Computer and Information Science Education]**: Computers and Education - Computer and Information Science Education

General Terms

Human Factors

Keywords

Program Visualization Tools, Activity Theory, Theoretical Framework, Educational Effectiveness

1. INTRODUCTION

When teaching programming, most teachers use visualizations in order to illustrate and specify concepts and ideas. In general, this seems to be an obvious and sense-making educational approach, since programming concepts are abstract constructions that mostly lack matching objects from everyday life and therefore cause students to have many difficulties in understanding. In consequence, an immense amount of software development and empirical research has been done over the past 30 years in the field of educational software tools for displaying and visualizing algorithms and programming [47]. As a result, the range of visualization tools is impressive, with tools available for learning most of the languages used to teach introductory programming as well as data structures and algorithms [30][36][40]. Although being one of the most investigated research fields in Computer Science Education, the field has been facing the problem that teachers and students don't use the visualization tools when teaching and learning programming, respectively. Several research approaches have been suggested and accomplished to address this problem, and among these student engagement was particularly focused. Still, the problem remains unsolved.

In this paper, we propose that in order to solve this problem a deeper understanding is required. For this reason, we suggest to use a *theoretical framework* which helps to analyze and reflect the role of visualization tools. With theoretical framework we mean an ontological and epistemological characterization of a domain of discourse. In a research field, the elements of discourse and how they are understood and interrelated determine which research questions are posed and in consequence how data collection and analysis are structured [9].

The awareness for discussing and explicitly incorporating theoretical frameworks was already advocated by Hundhausen more than a decade ago when discussing the problem described above with regard to algorithm visualizations (AV): "The solution, I contend, is to address the problem not at the surface, but at its roots. In other words, instead of tweaking our current design, pedagogy, and evaluation methods, we need to rethink the theory of effectiveness in which they are rooted. Only by proceeding from an alternative theoretical position - one that sheds new light on why AV technology is pedagogically valuable - do we have any hope of overcoming the obstacles that have stood in the way of AV technology's becoming a viable pedagogical aid" ([12] , p. 5ff). In a recent publication, Sorva discussed visual program simulation tools with regard to cognitivist and constructivist learning theories and demonstrated the high amount of issues both theoretical approaches bring into question ([45], p. 212ff). But this kind of epistemological and ontological characterizations of the field remains unquestioned in most of recently conducted studies and a theoretical debate is still missing. If we want to understand the problem mentioned above better, we need to focus on the theoretical assumptions research in this field stands on.

As an alternative to a cognitivist understanding of visualization tools, Hundhausen ([12], p. 33ff) suggested the situated learning approach by Lave and Wenger [23]. In this paper, we propose Activity Theory as another possible theoretical framework for understanding cognitive processes involved in using tools such as program visualizations. Stemming from the work of Vygotsky in the 1920's, Activity Theory has been successfully used as a theoretical framework in the field of Human-Computer-Interaction [17]. By promoting its use within the field of program visualization tools for educational purposes, we hope to not only deepen our insight into the design, understanding, and use of visualization tools, but to reconsider many of the questions that Computer Science Education researchers ask about their development and use. This paper, then, is neither a research study nor an experience report. It is explicitly a philosophical paper, with the purpose of briefly presenting those aspects of Activity Theory most relevant to the development of program visualization tools, and pursuing the implications of this theory for deepening our understanding of how these tools impact teaching and learning.

We begin the argumentation by giving in Section 2 an overview of the current problems in the research field of program visualization tools. Then, we introduce Activity Theory and its main concepts in Section 3. Drawing upon the framework of Activity Theory, we interpret in Section 4 visualizations tools and examine the previously introduced problems. The paper concludes in Section 5.

2. Problems in the Field of Visualization Tools

The field of educational software tools for displaying and visualizing algorithms and programming is a subfield of software visualization ([45], p. 140ff). This field evolved in the early 1980's, starting with development and research of algorithm visualizations. The focus then shifted in the mid 1990's towards program visualization tools [47]. In this paper, we will talk about *visualization tools* as referring to those educational software tools that display, simulate, and visualize *programming concepts.* Because algorithm and program visualization tools are regarded to be part of the same visualizations research family, we will refer to related work from both fields.

In this section, we discuss each of the two subproblems mentioned in the introduction of this paper, specially focusing on what has been suggested and done to solve these problems.

2.1 Teachers Do not Use the Visualizations

The first problem is that most of programming teachers do not tend to use the existing visualization tools in teaching programming [1][48]. Hundhausen et al. claim that visualizations are mostly used by those teachers who were also involved in the development of the respective tool [13]. According to Naps et al., the most common reasons that teachers report for this situation are related to the practical aspects of the visualization technology [32]. According to that, it produces too much overhead for the teachers to incorporate the visualization tool in their work, for instance, searching for good examples, learning to use the tools, and adapting the materials to one's own teaching approach. In addition to these practical reasons, teachers doubt if visualizations are educationally effective [13] [32]. In the following subsections we will discuss work related to these reasons.

2.1.1 The Practical Problems

Visualization tool developers have charted teachers' needs to tackle the practical problems. Naps et al. identified teachers as the key persons to enhance the use of visualizations in class rooms and thus provide an extensive list of instructions on how to address teachers' needs when developing visualization tools [32]. For example, they suggested that visualization tools should enable platform independence, capture larger programming concepts, and be mapped to existing teaching and learning resources. For disseminating visualization tools they suggested an outline of a standard web site, whereas for sample items they recommend to included them in evaluation instruments intended to measure teachers' and students' satisfaction with the tool. Shaffer et al. analyzed a collection of over 500 visualizations and concluded also that it should be improved how material is disseminated, propagating known best practices, and informing developers about what kinds of materials are needed [43]. To make this improvement happen, Shaffer et al. promote an online educational community whose purpose is to better focus the future directions of tool development and use[42] [44].

Another aspect of how teachers perceive visualization tools is related to their attitudes. In a phenomenographic study, Levy et al. described teachers' attitudes towards a visualization tool using four categories [26]. Two of them described a positive attitude towards the tool and the other two a negative attitude. In addition to the simple negative orientation, the other negative category described that the teachers experienced the tool in a conflicting manner: they were enthusiastic about the tool but still reluctant to use it. Levy et al. investigated the reasons for the teachers' attitudes in a further quantitative survey using a theory of planned behavior [25]. They concluded that teachers feel they are not in control when using a visualization tool and they are not confident with this situation.

2.1.2 The Educational Effectiveness of Visualizations

There are many studies on the educational effectiveness of visualizations. Gurka and Citrin summarized the results of these studies to be "markedly mixed" [11]. With this they mean that some of the studies demonstrated that visualizations were pedagogically advantageous for learning, some that they were not, and some that the advantage was partial. Apparently, such results are not convincing teachers to stop doubting the tools' educational effectiveness.

Hundhausen et al. took a closer look below the surface of the "markedly mixed" results and conducted a meta-study comprising 24 empirical studies about visualization tools' educational effectiveness [13]. The main result of this meta-study was the insight that students' utilization patterns of visualization tools have a much greater impact on their learning success – and therefore on the tool's educational effectiveness – than the quality of the visualizations. As a consequence, Hundhausen et al. suggested investigating the educational effectiveness of visualization with a focus on students' engagement. Also Stasko and Hundhausen suggested that research on visualizations should be more student-oriented, i.e., researchers should study how students use visualizations in order to develop tools and materials according to discovered needs [47].

2.2 Students Do not Use the Visualizations Either

The second problem is that even if visualization tools are introduced to students in lecture and they grasp their importance as a learning tool, they still do not seek to use them to study and explore concepts outside class ([4], p. 393). A survey conducted by Naps et al. pointed out that one of the key obstacles in the adoption of visualizations is that from the learners' perspective the visualization technology may not be educationally beneficial [32].

There are not many studies reporting how often students use the visualizations provided to them. One reason for this can be that in many cases visualizations are used in class as a compulsory material so there is no data about how many students would use it in a self-directed way. An international survey handling different kinds of program visualizations revealed numbers of times students used program visualizations during a programming course when it was voluntary for them ([21], see Figure 2): out of 335 respondents, approximately a fourth had not used them at all. Most of the students used them only a couple of times and less than a third of the respondents had used them for more than 5 times. This gives some perspective on what the user rate of visualizations might be in general.

Stasko and Hundhausen present the history of the visualization field noting that both, the development of visualization tools and research on their usage, have been technically oriented and focusing on the tools rather than the users, especially in the infancy of this area [47]. Later on, there has been a change of direction towards a more student-oriented approach in both of them. However, the technically oriented opening can be seen as a reason for not making a hit with the students. In the following two subsections, we will summarize what has been done in both of these fields to move towards a user-oriented direction.

2.2.1 Visualization Tool Development

Many of the visualization tools were developed by expert programmers or teachers of programming [47]. For novice programmers this can lead to difficulties in using them. To gain an understanding on how fundamental problems this can create, we take an eye-tracking study by Bednarik et. al. as an example [5]. This study revealed that expert and novice programmers use different visual attention strategies when using a visualization tool. Thus, it can be difficult for an expert to understand how the tool should be designed in order to support novice programmers' way of using it.

Stasko and Hundhausen requested that in the future visualization tools should be developed using a learner-centered design process and usability specialists as designers instead of CS teachers [47]. In addition, instead of developing tools and materials according to the technical visions of the developers', the field should study how students use visualizations and develop tools and materials according to their needs. Shaffer et al. also demanded more fundamental research on how to develop and use visualizations [24]. In consequence, usability studies have been conducted for some visualization tools to overcome the mentioned problem [20]. This certainly improves the usability of the tools, but still keeps the expert perspective in the tool development.

2.2.2 Research on Visualizations

The shift to students' perspective in the research on program visualizations goes back to the meta-study by Hundhausen et al. [13] whose main result was the insight that students' utilization patterns of visualization tools have a much greater impact on their learning success – and therefore, on the tool's educational effectiveness – than the quality of the visualizations. As a consequence, Hundhausen et al. suggested investigating the educational effectiveness of visualizations focusing specifically on student engagement.

Following the meta-study, Naps et al. explored the role of visualizations and the corresponding student engagement in CSE and proposed an Engagement Taxonomy (ET) with a general research framework for further inquiry [33]. The ET defines different levels of engagement, for instance, the level responding means answering questions concerning the visualization presented by the system; meanwhile, the level viewing describes non-active involvement. The different engagement levels describe single situations or activities. Their research framework proposes hypotheses contrasting these engagement levels (e.g. "Responding results in significantly better learning outcomes than viewing"), indicates these hypotheses can be tested, and recommends a classical experimental study that is based on three steps: pre-test, use of materials, and post-test. A great number of controlled studies have been conducted following this research framework; summarizing them Urquiza-Fuentes and Velázquez-Iturbide presented a review where they analyze 33 evaluation studies of visualization tools with regard to the different levels of the ET [49]. All these studies emerge from the general question how to better engage students with visualizations testing the use of visualization on different levels of the ET.

The ET was developed mostly normatively in order to describe possible engagement levels that can be tested in experiment studies but this leaves possible other forms of student engagement outside the scope. The experimental framework also limits to a single use session whereas learning programming is a long-term process. Thus, it does not capture the students' perspective of using visualizations completely. It focuses on student engagement but not on the student perspective.

There are also studies addressing the students' perspective on visualizations that use other approaches than the ET, for example [15][16][22][29][27]. These studies have been conducted e.g. interviewing or observing students and using qualitative methods to analyze this data. Some of these studies reported students' behavior when using visualization tools with a category system and some were more concentrated on the usability of a visualization tool. In summary, this work gives interesting insights into students' use of visualizations and widens the students' perspective on the use of visualizations. However, the ultimate reasons for the low usage rate of visualizations have apparently not been found since the state of affairs has not changed.

2.3 Concluding Remarks

All this work aiming to solve the mentioned problems has certainly improved the quality of visualizations, extended possible practices of using them in class, and the support given to teachers' regarding their usage. However, these efforts seem to be not helpful enough since visualization tools are still not used widely by teachers and students.

We suggest that before trying to solve the mentioned problems we need to analyze further their possible reasons. For this matter, a theoretical perspective is reasonable because it questions the ontological and epistemological characterization of the domain of discourse, see also ([12], p. 5ff).

3. ACTIVITY THEORY

Activity Theory is a psychological theory about the relationships between human beings and their goal-directed activities. The theory has its roots in Russian cultural-historical psychology from the 1920's by Vygotsky [50], and was developed further by Leontiev [24]. There are many similar theories that have been elaborated over the last three decades and that have built on these historical foundations that go by the names situated learning [23], sociocultural learning [37][53], distributed intelligence [35], among others. For our purposes in this paper, we focus on the primary concepts from Activity Theory as originally developed by Vygotsky and Leontiev and some of the above named extensions for a better explanation.

Activity Theory has been applied to different fields like for example Education and Technology, among which the interpretation by Engeström became quite known [10]. Activity Theory has been also very influential in Human-Computer-

Interaction (HCI) [8][17] as the focus in this field of research has shifted towards activities of people using technology over the past twenty years. Here, it was "recognized that technology use is not a mechanical input-output relation between a person and a machine; a much richer depiction of the user's situation is needed for design and evaluation. However, it is unclear how to formulate that depiction in a way that is not purely ad hoc. Here is where activity theory helps: by providing orienting concepts and perspectives." ([34], p. 8). Based mostly on these both depictions, Berglund introduced this framework to Computer Science Education ([7], p. 45ff).

Although, there are many differences between HCI and the field of EVs, we find the same distinct relationship between an artifact and persons who interact with them and these are respectively: the visualization and students using it for programming, teachers using it in instruction, and tool-developers building it for former's uses. We will introduce Activity Theory for the same purpose as it was applied in HCI. As a theoretical framework it will help us to conceptualize this relationship and to reflect the problems discussed in section 2.

As an account of human activity and psychological development, Activity Theory represents a paradigm change from many other kinds of psychological frameworks, such as constructivism [14] or cognitivism [52]. Some of the concepts that will be introduced next do not have straightforward mappings to these other 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 reader as those felt by an experienced imperative programmer on first encountering an object-oriented language. Therefore, it might be difficult for readers with a strong background in CS and cognitivism to adopt and appreciate this way of thinking. But, the sophisticated lens that the Activity Theory approach offers is worth the endeavor because it helps us to develop a richer conception of what it means to teach and learn programming and algorithms with visualizations.

3.1 Activities and Tools

Activity, the main concept in Activity Theory, denotes the interaction of a person with the world he or she lives in. Activity Theory emerges from the assumption that people are goal-directed, and that they carry out activity as a means to achieve their goals. The term *goal* refers to a desired state of the world, and the term *activity* denotes sequences of action—both mental and physical—that people carry out by their own volition. Activity Theory differentiates between internal and external activities, but emphasizes that both are highly connected to each other and to the subject of activity, as well.

According to Activity Theory, activities involve the use of *tools* [37]. The term *tool* denotes not only material objects used to affect the material world, such as pencils, hammers, automobiles, and steam shovels. It also denotes symbolic objects used to affect the mental world of the self and others, such as "language; various systems for counting; mnemonic techniques; algebraic symbol systems; works of art; writing; schemes, diagrams, maps, and mechanical drawings" and similar ([51], p. 137). Although physical and symbolic tools are distinguished here with respect to their domains of use, physical tools can also come to have symbolic importance beyond the purely functional.

Activity Theory posits that almost all human activity is said to be *mediated* by tools and that people rarely act directly on the world, see [53], [51] and ([17], p. 42ff). Saying that activity is mediated makes the assumption that action cannot be separated from the milieu in which it is carried out ([53], p. 18). For example, pencil and paper, an abacus, and an electronic calculator all are different tools for summing a set of numbers. The related external activities with these tools are different as are the internal activities which are inextricably bound to the particular tools a person chooses to use for summing numbers. Because of their role in human goal-directed activity, tools are sometimes referred to as *mediational means* [53], i.e. they are not incidental to activity, nor do they simply enable it. Rather, they are inseparable from activity, serving as the point of contact between person and world.

3.2 Internalization and Externalization

Tools mediate activities because they embody a certain meaning for how to use them and which kind of purpose or goal can be achieved with them. Using them, a person adopts this implicitly embodied knowledge within the activity that he or she carries out with the tool. For example, the specific form of a hammer embeds knowledge about the ergonomic properties of the human body as well as physical properties of the external world, such as force and momentum. In consequence, nailing activities that are mediated by a hammer will be shaped through the tool's incorporated knowledge. This issue becomes crucial in the process of internalization, a concept that is introduced next.

3.2.1 Internal–External Dimension

Internalization is a concept in Activity Theory that refers to a process by which the tool's embodied knowledge is internalized through a person's activities, being first external and then internal activities. More precisely, internal activities are derived from external activities, and both are mediated by the specific tool in use ([17], p. 56). Let's consider for example the task of finding the way in a new city using a map. In the beginning, the mediation is highly visible because it incorporates first the external activity of looking at the map, reading its symbols and pictures, and connecting this information to the environment of the city. This is what Vygotsky distinguishes as external mediation (and the tool's function as external mediator): the map mediates an individual's external and internal activities. After a while when a person internalized the map's concepts, he or she will stop to use the map and does not practice the external activities with this tool anymore. He or she will be able to move through the city due to the internalized concepts and understanding developed through external mediation by the map. But his or her internal activities will still remain mediated because they have been mediated by the map and through the external activities. This is what Vygotsky distinguishes as internal mediation and the tool's function as internal mediator ([17], p. 43ff).

As Kaptelinin and Nardi point out internalization is not a carbon copy or a simple transfer from previously external to internal activities. Intenalization must be understood as "redistribution between external and internal components of activity". In consequence "internal activities cannot be understood if they are analyzed in isolation from external activities, because there are mutual transformations between the two kinds of activities." ([17], p. 69)

Externalization on the other hand means the process of transforming internal activities into external ones by creating tools that can be used for further mediation. What starts as ideas inside the mind of a person can thus become part of the surround that the person uses for subsequent activity or for collaboration between several people. Externalizing thought in a perceivable form (a sketch, a model, a prototype, an outline, a draft) is therefore much more than simple cognitive offloading. This is because these externalized artifacts are available to the perceptual system, thus giving rise to iterated perceptual-cognitive loops that are not possible with purely (internal) mental representations.

3.2.2 Individual–Social Dimension
Mediation includes not only the introduced internal-external dimension. 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, 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" ([35], p. 53). Cultural practices of tool use evolve in tandem with the evolution of the tool. For example, just as the materials and form of hammers have evolved over time [3], so have hammer-mediated activities changed; if the tool changes, so must its use. Therefore, tools represent socially distributed cultural entities that implicitly embed collective knowledge of their use in context.

According to Vygotsky, an individual first performs a particular tool-mediated activity in collaboration with or guided by others that already have certain tool-using skills. With gained experience, the individual transforms activities from what was initially social to one that are performed individually. The same way, they contribute to the social activity in the process of externalization [50]. This is an active transaction between the social and the individual dimension of activity.

3.3 Concluding Remarks
One of the most important aspects of the principle of mediation and one of the main ways that Activity Theory departs from cognitivism, is that what is internalized during an activity depends crucially on the mediational means used to carry out the activity. Mental processes, tool use, and interaction with the world are tightly bound together and this is especially true for activities of learning: "A fundamental assumption in a sociocultural understanding of human learning is precisely this: learning is always learning to do something with cultural tools (be they intellectual and/or theoretical). 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." ([41], p. 147).

4. VISUALIZATION TOOLS IN LIGHT OF ACTIVITY THEORY
In this section, we will interpret the role of visualization tools for learning and teaching programming with the introduced concepts of Activity Theory. For this matter, we will consider in the next subsections first the activity of learning programming with visualization tools, and second the process of creating visualizations for the purpose of teaching programming. Drawing from this interpretation, we discuss the two problems introduced in Section 2 and give implications for future research directions.

4.1 Visualizations as Mediational Means for Learning Programming
With regard to Activity Theory, we can conceptualize students' process of learning programming to be activities in which students interact with a specific world in order to achieve certain goals. Being set in an introductory programming course, this world will include not only the physical environment like classrooms, labs, or the library, but specifically the interaction with other individuals that are met in this world like professors, tutors, and other students. The specified goals will be for example attending regularly class, doing programming assignments or homework, and passing tests and exams.

According to Activity Theory, every tool in use, being physical like a pen and paper or non-physical like a timetable, will mediate the students' activities of learning in a certain way. From this point of view, we can understand visualizations to be symbolic tools that embody an implicit understanding of programming concepts for the purpose of mediating students' programming-related activities. This means that visualizations can be seen as an external mediator for learning programming. In consequence, using a visualization tool represents a dynamic process of internalizing programming concepts; as students get better in programming, the external activities of using the visualizations transform into a mental process of internal activities.

4.2 Visualizations as Mediational Means for Teaching Programming
Let's consider now the process in which a teacher creates program visualizations for a pedagogical purpose, no matter if this happens on the blackboard during class or is supported by a specific visualization tool. This activity requires understanding of the visualized programming concepts as well as the ability to create a specific visual model that represents them. With regard to Activity Theory, this can be understood as an internal activity that is externalized by creating an external mediator. From this point of view, program visualizations represent for teachers externalized programming concepts. We can extend this interpretation and state that teaching activities in general are mediated by external mediators like language, visualizations, among others, as well as physical tools like blackboard, programming environments, slides, and many others.

While tools are used by students for internalizing programming concepts, teachers use tools to externalize the latter. During their own learning process, teachers internalized a rich understanding of programming concepts. Teaching, they choose tools that serve them as external mediators. Concerning the social-individual dimension, while introducing a visualization tool and its usage during class a teacher distributes tools that represent to students socially shaped and preserved cultural entities. After class, when students first start to interact with the tool they perform the social and external activity as it had been introduced to them.

4.3 Why Students Do not Use Visualizations Tools
In section 2.2 we introduced the problem that students don't use visualization tools when learning programming. In the next two subsections, we will discuss two possible reasons for the problem and suggest implications for future research directions.

4.3.1 Internalization is not Stimulated Enough
One reason why the majority of students don't use visualizations regularly when learning programming although their benefit was

proven in research studies may be that the students' internalization process with the tool has not been stimulated enough and students tend to use the tool only from time to time, not knowing how to support their programming activities with it. This argument can be supported with the research studies that proved the tools' educational effectiveness.

The evaluation of a visualization tool is mostly conducted in purposefully designed learning situations for controlled experimentation. Interpreting learning programming with visualization tools being an internalization process, this means that the social distribution of the visualization tool might be much higher and more intensive than in a regular programming course. Depending on the amount of students that attend a regular programming course, the social dimension of activities between students and teachers is less intensive: students are expected to work individually on their computers, having contact with the instructor or teacher once or twice per week. In addition, when teachers don't promote the visualization tool as intensively as during a research study, it might be even less accepted among students. In summary, a significant difference between research study and regular class situation can be assumed. The consequence of all this is that the internalization process with the tool is not stimulated enough like it happens during a research study.

This interpretation suggests different directions in the course setup and instructions on how visualizations should be used in order to stimulate the internalization process with the tool. First of all, the social distribution must be more emphasized; just presenting the tool once or twice during class or lecture is probably not enough to demonstrate the tools function in the process of internalization. Efforts suggested by Naps et. al. [32] and described in subsection 2.1.1 are plausible with regard to the social-individual dimension of internalization. Furthermore, it appears to be not enough to develop a visualization tool and to test its educational effectiveness as well as its usability. It is also important to investigate how students' internalization process with the tool can be stimulated appropriately. But this requires a research approach that would focus on possible different stages of the internalization process students might undergo when using visualization tools for learning programing.

Such research approach would be a very different than the one taken with the Engagement Taxonomy introduced in subsection 2.2.2. Instead of testing possible approaches of how students can be stimulated to more engagement with the visualization tool according to the different engagement levels, the purpose would be to detect the different stages in which student engagement with the tool rises and declines with the regard to the tool's role as external and internal mediator.

4.3.2 Internalization Already Happened

Another reason why students don't use visualizations although their benefit was proven in research studies might be that the internalization already happened. Activity Theory lends support to the view that the mediator might be advantageous for its user mostly during the process of internalization. Finding it helpful during internalization, the student does not necessarily need the visualizations any longer when proceeding, because external conceptualizations of programming transform into internal activities. Then the tool becomes an internal mediator and is physically not needed anymore.

Research studies investigate the educational benefit of a visualization tool in a single use session as part of a controlled experiment. From the perspective of Activity Theory, this means that only different stages of the internalization process are evaluated instead of the internalization itself. In addition, studies that investigate if students are using and benefiting from visualizations usually inquire students directly about their usage patterns and habits and very often by the end of the course in order to let students reflect their learning process. But when the benefit of visualizations is to become internalized and not used anymore such direct inquiry might be misleading, especially when students are not aware of the principles of internalization. Therefore, in order to investigate if students were using and benefiting from the tool, it would be more appropriate to investigate their programming activities during the whole course in order to grasp all the different stages of internalization. By the end of the course it would be also important to inquire their internal mental models of programming concepts by asking them to externalize their internal activities and check how much this resembles the visualizations. This could be done using language by asking students to describe how they understand a programming concept or using visualizations by asking them to draw pictures. But other kind of externalization would be surly possible, as well. This kind of student inquiry would reveal a more differentiated picture about students' real usage and benefit of visualization tools.

4.4 Why Teachers Do not Use Visualizations

In section 2.1 we introduced the problem that teachers don't use visualization tools when teaching programming. In the next three subsections, we will discuss three possible reasons for the problem and suggest implications for future research directions.

4.4.1 Changing Meditational Means for Assessment

Let's assume that teachers are right and the same visualization tools that were beneficial for students in experimental studies are not beneficial for students in common programming courses. Beside possible reasons already discussed in the previous subsection, how can this be explained? The introduced concepts of Activity Theory suggest an explanation of this biased situation.

In Sec. 2.1.2 we reported on the meta-study by Hundhausen et. al. which revealed that students' utilization patterns of visualization tools have a much greater impact on their learning success – and therefore on the tool's educational effectiveness – than the quality of the visualizations. The fact that students' activities with a specific mediator are more relevant than just the mediator seems obvious from the perspective of Activity Theory, where person, tool and activity are regarded to be an inseparable unit. In addition, the internalization of what novice students are learning is adapted to the mediating tools involved. This implies that changing a specific mediator requires the ability to transfer what was learnt from one context of activities to others. This is specifically important to the way students are tested or assessed by the end of a programming course.

In the experimental study the assessment might be conducted with the visualization tool that was also used for stimulating students learning activities which means that the activity's mediator remains the same. The assessment in a regular programming course instead might use different mediational tools (e.g. pen and paper) than those students used normally when doing programming (e.g. visualization tool, specific, programming environment with debugger). It is obvious that keeping the same mediator for learning and assessment makes it much easier for students to accomplish a programming task. A change of mediators for stimulating learning activities and for assessing learning outcome can be the reason why teachers don't observe visualization tools to be educational effective. This implies that for further investigations it will be very important to consider this change of mediators when testing the visualization tool's benefit as well as assessing students' outcome.

4.4.2 Visualization Tools Represent an Unwanted Standardization

In order to introduce it in a programming course and promote it to the students, teachers must be engage with the visualization tool as well. From this point of view, teachers' engagement with a visualization tool must be regarded quite differently: Teachers have to adopt it as a supportive educational tool for their teaching and acknowledge the externalized concepts by promoting them to their students. It is clear, that a teacher who was also involved in developing a visualization tool is highly motivated in using it as an educational tool because, among other factors, the visualized concepts are an externalization of his or her understanding and he or she has a clear understanding of the mediational mean the tool is supposed to fulfill.

In general, a visualization tool represents an embodiment of externalized programming concepts. Proposed for teaching it can be therefore understood as a form of social agreement about how programming concepts are understood and being therefore a form of standardization. However, there are many different kinds of visualizations and there is no consensus among teachers, students, and developers about what a standardized representation should look like. Therefore, we can assume that beside organizational aspects like lack of time, among others, teachers might not become engaged with a visualization tool because the externalized programming concepts of the tool's designers do not match their own internalized concepts, which they developed when learning to program themselves. This might be the reason why teachers feel they are not in control when using a visualization tool and that they are not confident with this situation, see [25] and section 2.1.1. When asked, teachers might report concrete organizational problems with visualization tools, but the reasons for overcoming them are related to such profound problems as not acknowledge standardization. From this perspective, promoting an online educational community as suggested by Shaffer et al.[42] [44] and reported in sec. 2.1.1 is more than just a solution to overcome teachers' practical problems with visualization tools. An online educational community creates a social group that has the potential to negotiate what is acknowledged to be a standardized externalization of programming concepts.

4.4.3 Creating and Using Visualizations with a specific Tool is a Divided Activity

Naps et. al. reported that teachers find using visualization tools to be very time consuming [32], although these tools were developed with the argument of being time savers for teachers. Usually, teachers create a specific visualization during class, for instance drawing a simple picture on the blackboard and using language as a further mediator to externalize their understanding. This teaching activity can be accomplished spontaneously without extra preparation and it is directly connected to the overall learning and teaching context in which teacher and students meet. In order to externalize the same with a visualization tool, teachers must do this in advance and doing it for the first time, probably use much more time. Here, they can't use language as a mediator to externalize quickly what they mean. Instead they are bound to what the tool developers provided as possible externalization features.

In contrast to the directly experienced context, a visualization tool designer must make assumptions about such possible future activities with the visualization tool. No matter how profound and elaborated the tool developers' knowledge is about learning and teaching programming, the assumptions address future activities that will take place without them. The direct chain between a teacher and his or her own created visualization in a specific teaching activity is divided between different people, places, and time. It is rather very difficult for an expert to foresee the needs of novice programmers and how it will support the internalization as well as the teaching process involved. It is therefore very important to study how students and programming teachers use visualizations and develop tools and materials according to their needs. Stasko and Hundhausen [47] request that visualization tools should be developed using a learner-centered design process and usability specialists as designers. Beside this general research focus, the question remains if for teachers visualization tools are better external mediators than other ones like for example pictures combined with oral explanations.

4.4.4 Concluding Remarks

It might be that visualization tools are better external mediators than for example pictures combined with oral explanations and that it is just a question of social negotiation to persuade teachers using them. But, being just used without a deeper teaching context, their advantage may never come to its full potential. That is, for the most visualization tools a fully approved pedagogical approach is missing that the tool is supporting and accomplishing. For example, the programming visualization environments BlueJ and Greenfoot are part of the *objects-first* pedagogical approach, see [2] [19] The pedagogical approach gives answer and evidence to how the specific tool is supposed to be used during class to support students learning activities and is much more specific in advocating the tool's use and benefit than a general claimed educational effectiveness.

5. CONCLUSION

In this paper, we discussed the problem that teachers and students don't use regularly visualization tools for the purpose of teaching and learning programming. In order to reflect and analyze possible reasons for this problem, we introduced Activity Theory including the concepts of activity, tool mediation, and internalization-externalization as a possible theoretical framework. We interpreted visualization tools to be mediators of programming concepts that are supposed to be internalized by the students and used by teachers as externalizations. Here, we argued that possible reasons why students don't use visualization tools are:

- The internalization process is not stimulated enough and students tend to use the tool only from time to time, not really knowing how to mediate their programming activities with it.
- The internalization process already happened and therefore the tool became an internal mediator and is physically not needed anymore.

In addition, we argued that possible reasons why teachers don't use visualization tools are:

- The educational benefit proven in research studies is not replicated in a regular programming course because the mediators used in the course as well as for assessing students are changing meanwhile remain the same in the research experiment.

- Visualization tools are socially not accepted because they represent a standardized externalization without being acknowledged by teachers and the rest of the programming experts.
- Creating and using visualizations with a specific tool is a divided activity between people, places, and time.

Based on our argumentation, an important direction for future research in this field is to investigate how students use a visualization tool regularly for their programming assignments and how they interact with the tool in the process of internalization. Furthermore, supporting teachers to include visualization tools into their teaching activities depends on if they acknowledge the tools' to be more useful external mediators than the one used before.

Activity Theory is a foundation for further development that took place over the last decade of research in the field of developmental psychology. Continuing the theoretical reflection of visualization tools and their impact for learning programming is therefore not limited to this theoretical approach only. Other theories and approaches surly will reveal further aspects to discuss that might even lead to different implications than the one proposed and argued for in this paper. We acknowledge such differentiation of discussion and theory building as it is deepening our understanding of how these tools impact teaching and learning.

6. ACKNOWLEDGEMENT

We thank Georg Wittenburg for a careful proofreading of the first version of this paper and Randy Connolly for providing helpful comments that improved this paper further.

7. REFERENCES

[1] Baecker, R. 1998. Sorting out sorting: a case study of software visualization for teaching computer science. In *Software Visualization: Programming as a Multimedia Experience,* Brown, M., Domingue, J., Price B., and Stasko, J. Ed. The MIT Press, Cambridge, 369-381.

[2] Barnes, D. J. and Kölling, M. 2012. *Objects First with Java A Practical Introduction using BlueJ.* Fifth edition, Prentice Hall / Pearson Education.

[3] Basalla, G. 1989. *The Evolution of technology*. Cambridge University Press.

[4] Bazik, J., Tamassia, R., Reiss, S. P., and Dam, A. v. 1998. Software Visualization in Teaching at Brown University, In *Software Visualization*, Stasko, J. T., Domingue, J. B., Brown, M. H., and Price, B. A. Ed. The MIT Press, Cambridge, 383-398.

[5] Bednarik, R. 2007. *Methods to Analyze Visual Attention Strategies: Applications in the Studies of Programming.* Doctoral Thesis. Joensuun yliopisto, University of Joensuu, Finland.

[6] Bednarik, R., Moreno, A., Myller, N. 2006. Various Utilizations of an Open-Source Program Visualization Tool, Jeliot 3. *Informatics in Education* 5, 2, 195-206.

[7] Berglund, A. 2005. *Learning computer systems in a distributed project course: The what, why, how and where.* Doctoral thesis, Uppsala Dissertations from the Faculty of Science and Technology nr. 62, Acta Universitatis Upsaliensis, Uppsala University, Sweden.

[8] Boedker, S. 1989. A Human Activity Approach to User Interfaces. *Human-Computer-Interaction*, 4, 171-195.

[9] Daniels, M. and Pears, A. 2012. Models and Methods for Computing Education Research. In *Proc. Australasian Computing Education Conference.* (Melbourne, Australia, 2012). ACE '12. ACS, CRPIT, 123, 95-102.

[10] Engeström, Y. 1990. *Learning, working and imagining: twelve studies in activity theory*. Orienta-konsultit, Helsinki.

[11] Gurka, J. S. and Citrin, W. 1996. Testing effectiveness of algorithm animation. In *Proceedings of the 1996 IEEE Symposium on Visual Languages*, IEEE Computer Society Press, Los Alamitos, 182-189.

[12] Hundhausen, C. D. 1999. *Toward Effective Algorithm Visualization Artifacts: Designing for Participation and Communication in an Undergraduate Algorithms Course*. Doctoral Thesis, University of Oregon.

[13] Hundhausen, C. D., Douglas, S. A. , and Stasko, J. T. 2002. A meta-study of algorithm visualization effectiveness. *Journal of Visual Languages & Computing*, 13, 3, 259–290.

[14] Illeris, K. 2002. *The Three Dimensions of Learning.* Krieger Publishing Company.

[15] Isohanni, E. and Knobelsdorf, M. 2010. Behind the curtain: students' use of VIP after class. In *Proceedings of the Sixth international workshop on Computing education research*, ICER '10, ACM Press, New York, 87-96.

[16] Kannusmäki, O., Moreno, A., Myller, N., and Sutinen, E. 2004. What a Novice Wants: Students Using Program Visualization in Distance Programming Course. In *Proceedings of the 3rd Program Visualization Workshop*, Report CS-RR-407, 126–133.

[17] Kaptelinin, V. and Nardi, B. 2004. *Acting with Technology – Activity Theory and Interaction Design*. MIT Press, Cambridge.

[18] Karavirta, V., Korhonen, A., Malmi, L., and Stalnacke, K. 2004. MatrixPro - A Tool for On-The-Fly Demonstration of Data Structures and Algorithms. In *Proceedings of the Third Program Visualization Workshop*, 26-33.

[19] Kölling, M. 2009. *Introduction to Programming with Greenfoot: Object-Oriented Programming in Java with Games and Simulations*, Pearson Education

[20] Laakso, M.-J., Salakoski, T., Grandell, L., Qiu, X., Korhonen, A., and Malmi, L. 2005. Multi-perspective study of novice learners adopting the visual algorithm simulation exercise system Trakla2. *Informatics in Education*, 4, 49-68.

[21] Lahtinen, E., Järvinen, H.-M., and Melakoski-Vistbacka, S. 2007. Targeting Program Visualizations. In *Proccedings of the 12th Annual Conference on Innovation & Technology in Computer Science Education*, ITiCSE '07, 256-260.

[22] Lattu, M., Meisalo, V., and Tarhio, J. 2000. How a visualization tool can be used — evaluating a tool in a research & development project. In *Proceedings of the 12th Annual Conference on the Psychology of Programming Interest Group*, PPIG '12, 19-32.

[23] Lave, J. and Wenger, E. 1991. *Situated Learning: Legitimate Peripheral Participation*. Cambridge University Press.

[24] Leontiev, A. N. 1978. *Activity, Consciousness, Personality.* Englewood Cliffs, NJ. Prentice Hall.

[25] Levy, Ben-Bassat, R. and Ben-Ari, M. 2008. Perceived behavior control and its influence on the adoption of software tools. *SIGCSE Bull.* 40, 3,169-173.

[26] Levy, Ben-Bassat, R., and Ben-Ari, M. 2007. We work so hard and they don't use it: acceptance of software tools by teachers. In *Proceedings of the 12th annual SIGCSE conference on Innovation and technology in computer science education*, ITiCSE '07, June 25-27, 2007, Dundee, Scotland

[27] Lonnberg, J., Malmi, L., and Ben-Ari, M. 2011. Evaluating a visualisation of the execution of a concurrent program. *In Proceedings of the 11th Koli Calling International Conference on Computing Education Research*, Koli Calling '11, ACM Press, New York, 39-48.

[28] Malmi, L., Karavirta, V., Korhonen, A., Nikander, J., Seppälä, O., and Silvasti, P. 2004. Visual algorithm simulation exercise system with automatic assessment: TRAKLA2. *Informatics in Education*, 3, 2, 267–288.

[29] Moreno, A. and Joy, M. S. 2006. Jeliot 3 in a Demanding Educational Setting. In *Proceedings of the Fourth Program Visualization Workshop*, Florence, Italy, 51–59.

[30] Moreno, A., Myller, N., Sutinen, E., and Ben-Ari, M. 2004. Visualizing Programs with Jeliot 3. In *Proceedings of the International Working Conference on Advanced Visual Interfaces*, AVI '04.

[31] Myller, N., Bednarik, R., Sutinen, E., and Ben-Ari, M. 2009. Extending the Engagement Taxonomy: Software Visualization and Collaborative Learning. *Trans. Comput. Educ.*, 1, 9, 1-27.

[32] Naps, T., Cooper, S., Koldehofe, B., Leska, C., Rößling, G., Dann, W., Korhonen, A., Malmi, L., Rantakokko, J., Ross, R. J., Anderson, J., Fleischer, R., Kuittinen, M., and McNally. M. 2003. Evaluating the educational impact of visualization. *SIGCSE Bull.* 35, 4, 124-136.

[33] Naps, T., Rössling, G., Almstrum, V., Dann, W., Fleischer, R., Hundhausen, C., Korhonen, A., Malmi, L., McNally, M., Rodger, S., and Velazquez-Iturbide, J. 2003. Exploring the role of visualization and engagement in computer science education. *SIGCSE Bulletin*, 35, 2, 131–152.

[34] Nardi, B. A. 1996. Activity Theory and Human-Computer-Interaction. In *Context and Consciousness: Activity Theory and Human-Computer Interaction*, Nardi, B. A., Ed. The MIT Press, 8-16.

[35] Pea, R. 1993. Practices of distributed intelligence and designs for education. In *Distributed Cognition: Psychological and Educational Considerations*, Salomon, G, Ed. Cambridge University Press, 47-87.

[36] Rajala, T., Laakso, M. , Kaila, E., and Salakoski, T. 2007. VILLE A Language-Independent Program Visualization Tool. In *Proceedings of the 7th Koli Calling Conference on Computer Science Education*. Koli '07.

[37] Rogoff, B. 2003. *The Cultural Nature of Human Development*. Oxford University Press.

[38] Romero, P., Boulay, B. du, Cox, R., Lutz, R., and R. Bryant, R. 2005. Graphical visualizations and debugging: A detailed process analysis. In *Proceedings of the 17th Annual Workshop of the Psychology of Programming Interest Group*, PPIG '05, 62-76.

[39] Rössling, G. and Freisleben, B. 2002. ANIMAL A system for supporting multiple roles in algorithm animation. *Journal of Visual languages and Computing*, 1, 3, 341-254.

[40] Sajaniemi, J. and Kuittinen, M. 2004. Visualizing roles of variables in program animation. *Information Visualization*, 3, 3, 137–153.

[41] Säljö, R. 1998. Learning as the use of tools: a sociocultural perspective on the human-technology link. In *Learning with computers*, Littleton, K. and Light, P., Ed. Routledge New York, 144-161.

[42] Shaffer, C. A., Akbar, M., Alon, A., Stewart,M., Edwards, S. 2011. Getting algorithm visualizations into the classroom. *In Proceedings of the 42nd ACM technical symposium on Computer science education,* SIGCSE '11. ACM, New York, 129-134.

[43] Shaffer, C. A., Cooper, M. L., Alon, A., Akbar, M., Stewart, M., Ponce, S., and Edwards, S. H. 2010. Algorithm Visualization: The State of the Field. *Trans. Comput. Educ.* 10, 3.

[44] Shaffer, C. A., Naps, T., Rodger, S., and Edwards. S. 2010. Building an online educational community for algorithm visualization. In *Proceedings of the 41st ACM technical symposium on Computer science education,* SIGCSE '10. ACM Press, New York, 475-476.

[45] Sorva, J. 2012. *Visual Program Simulation in Introductory Programming Education*. Doctoral Thesis. Aalto University publication series DOCTORAL DISSERTATIONS 61/2012, Aalto University, Finland.

[46] Stasko, J. T. 1990. TANGO: A Framework and System for Algorithm Animation. *Computer*, 23, 9, 27-39.

[47] Stasko, J. T. and Hundhausen, C. D. 2004. Algorithm Visualization. In *Computer Science Education Research*, Fincher, S. and M. Petre, Ed. Taylor and Francis, 199-228.

[48] Stasko, J. T. 1997. Using student-built animations as learning aids. In *Proceedings of the ACM Technical Symposiumon Computer Science Education*, ACM Press, 25-29.

[49] Urquiza-Fuentes, J. and Velazquez-Iturbide, J. A. 2009. A survey of successful evaluations of program visualization and algorithm animation systems. *Trans. Comput. Educ.*, 9, 1-21.

[50] Vygotsky, L. S. 1978. *Mind in Society: The Development of Higher Psychological Processes*: Harvard University Press.

[51] Vygotsky, L. S. 1981. The instrumental method in psychology. In *The concept of activity in Soviet psychology*, Wertsch, J. V., Ed. M. E. Sharpe.

[52] Wallace, B., Ross, A., Davies, J. B., and Anderson, T. 2007. *The Mind, the Body, and the World: Psychology after Cognitivism?* Imprint Academic, Exeter.

[53] Wertsch, J. V. 1993. *Voices of the Mind: Sociocultural Approach to Mediated Action*. Harvard University Press.