HCI in K12 Computer Science Education – Using HCI as a Topic and a Didactic Tool

ANDREAS BOLLIN, MAX KESSELBACHER, NINA LOBNIG, STEFAN PASTERK, ELISA RECI, and MARKUS WIESER, Alpen-Adria-Universität Klagenfurt, Austria

The role of human-computer interaction in school lessons is usually limited, if at all, to the usability factor of applications. In this paper, we report on a set of teaching principles and a project in which students between the ages of 15 and 18 playfully work through 3D, Virtual Reality, and usability issues without any particular prior knowledge in this field. In this work, we show that the suggested approach is founded on general didactic principles. Furthermore, by following neurodidactic findings, it is highly appealing to students, and essential computing science skills can be taught quasi incidentally. By systematically mapping our setting and decisions to teaching principles, we also show that HCI is not only one topic to be taught – it becomes a valuable didactic tool.

CCS Concepts: • **Human-centered computing → Human computer interaction (HCI)**; • **Social and professional topics → Computing education**.

Additional Key Words and Phrases: K12 Education, HCI-supported Teaching, Brain-based Teaching

ACM Reference Format:
Andreas Bollin, Max Kesselbacher, Nina Lobnig, Stefan Pasterk, Elisa Reci, and Markus Wieser. 2021. HCI in K12 Computer Science Education – Using HCI as a Topic and a Didactic Tool. In *CHItaly 2021: 14th Biannual Conference of the Italian SIGCHI Chapter (CHItaly '21), July 11–13, 2021, Bolzano, Italy.* ACM, New York, NY, USA, 14 pages. https://doi.org/10.1145/3464385.3464717

1 INTRODUCTION

In recent years, the topic of competence orientation [9] has increasingly become the focus of modern computer science teaching. A competence is understood as a combination of skills and knowledge. Still, one point is usually neglected: the acquisition of competencies is an individual process rather than a product and requires a lot of time and effort from the learner's perspective. This has at least two implications: on the one hand, as students are individuals, the learning brain also needs differentiated stimuli, and on the other hand, the learning progress is only recognized when the new skills are also demonstrated.

Since the acquisition of competencies in computer science is often based on and related to "language interpretation machines" (our computers), and since computer science is essentially part of socio-technical systems, it is obvious to grant the interface a particular weight in teaching, even to use it explicitly. The subject of Human-Computer-Interaction (HCI) is thus both facilitator and tool. As can be seen from the literature chapter, however, the topic of HCI plays a rather subordinate role in the classroom. It is most likely to be found in the areas of user experience or design principles.

However, we think that HCI-topics can add intrinsic motivation to our teaching. Thus we defined our **research objective** to investigate if it is possible to use HCI as a topic and didactic tool to initiate learning of computing science (CS) topics grounded on (neuro-)didactical principles. In this paper, we introduce some interventions which show that teaching with a focus on HCI helps to fulfill the key-ideas behind brain-based teaching [28] and many of the general didactic principles according to Rüdeger Baumann, founder of the so-called system-oriented didactics of computer science [3]. We also describe the structure of the interventions to be used as a stimulus for one's teaching.

Please note that the paper does not aim to present statistically significant results as the number of participants is still too small to do so, but (apart from COVID-19 restrictions), our interventions are going on and will be part of future research of our department. Instead, we present an analysis of the intervention from a didactical and methodological view and aim at demonstrating its benefit to the HCI and education community.

The remainder of the paper is structured as follows: Chapter 2 describes the background and related work, Chapter 3 explains the projects and setting in detail, Chapter 4 reflects on the results, and Chapter 5 presents a summary and our plans for future work.

2 BACKGROUND

2.1 Teaching HCI

When we take a closer look at the history of computing science education, we find that there was first a focus on hardware (at the beginning of the 1970s). In 1976, the focus was laid on algorithms, and only the user orientation in the 1980s put the interfaces in the spotlight [15, p.52]. However, this was then sidelined again in the early 2000s and replaced by fundamental ideas [37], great principles [8], and competency orientation [33]. Since then, be it implicitly or explicitly, HCI is an issue in computer science didactics. Denning's very influential work refers to HCI in his design principles, practices, and core technologies [8].

Recently, lecturers and practitioners all over the world took a closer look at HCI-related topics in education. Oleson et al. [21] provide a broad overview of design and HCI's role in K-12 education and look at several pedagogical challenges on edge to computer science education. They also mention that an agreed-upon pedagogy seems to be missing [18], but there are also success stories to be found in the literature. Open and playful cooperation and a multidisciplinary HCI education seem to be very successful [1], and the focus on graphical interfaces (VRML, Blender) highlighted that there is an enormous potential for stimulating individuals' motivation [10].

However, the number of success stories should not obscure the fact that HCI is neglected in many curricula. Thus, the following section takes a closer look at HCI implementation and the necessary competencies described in curricula worldwide before looking closer at the essential didactic principles that should steer our classroom interventions.

2.2 Curricula

Topics concerning HCI are frequently part of higher education. In lower education, primary or secondary school levels, this area is also rarely included in computer science-related subjects. An analysis of seven international curricula, educational standards, and competency models for these levels from Australia (national curriculum for technologies, AU) [22], Austria (digikomp - competency model for digital education, AT) [24], England (national curriculum for computing, GB) [27], Germany (German Informatics Society - educational standards for informatics, DE) [23], Switzerland (national curriculum for media and informatics, CH) [26], and the USA (Computer Science Teachers Association K-12 Computer Science Standards from 2011 and 2017, US11 and US17) [38][25] illustrates, that in the minority of them topics connected to the area of HCI appear repeatedly. The curriculum from England contains one learning outcome that mentions "usability" in programming [27]. In the version from 2011 of the standards suggested by the Computer Science Teachers Association (CSTA), one learning outcome refers to HCI by evaluating the usability of existing software [38]. Both models introduce the secondary level topic at the age of 11 (GB) respectively 15 (US11). In this case, the national curriculum for technologies from Australia and the revised K-12 Computer Science Standards from the CSTA in the USA is of particular interest because they start to introduce HCI-related topics at the primary level.

In the Australian curriculum, a focus on design and the evaluation of design is recognizable. This includes the design of user interfaces as it is combined with learning programming. The following examples illustrate that HCI topics repeatedly appear in several levels [22]:

- *By the end of Year 2, students list the features of technologies that influence design decisions and identify how digital systems are used.* (Technologies achievement, Foundation and Year 2)
- *By the end of Year 6, students incorporate decision-making, repetition, and user interface design into their designs and implement their digital solutions, including a visual program.* (Digital Techn. achievement, Years 5 and 6)
- *By the end of Year 8, students independently and safely plan, design, test, modify and create a range of digital solutions [...], including user interfaces and the use of a programming language.* (Techn. achievement, Years 7 and 8)

Also, the revised version of the CSTA standards includes topics from HCI, starting with the identification of user needs and preferences [25]:

- *By the end of Grade 2, students will be able to select and operate appropriate software to perform various tasks and recognize that users have different needs and preferences for their technology.* (Kindergarten and Year 1)

In higher levels also the usability gets into the focus of HCI-related topics. It is mentioned in several learning outcomes as the following examples show [25]:

- *By the end of Grade 5, students will brainstorm ways to improve the accessibility and usability of technology products for users' diverse needs and wants.* (Years 3, 4, and 5)
- *By the end of Grade 10, students will be able to compare various security measures, considering trade-offs between a computing system's usability and security.* (Years 9 and 10)

The competency model from Austria [24], the standards from the German Informatics Society [23] as well as the national curriculum from Switzerland [26] include no topics from HCI in their learning outcomes.

2.3 Didactic Principles

Rüdeger Baumann founded the system-oriented didactics of computer science in 1993. He formulated didactic guidelines and the following general didactic principles as a subject-specific basic concept for teaching fundamental ideas and achieving general educational goals:

P1 Principle of the life proximity and topicality (references to the current life world of the pupils)
P2 Principle of the factual-structural structure (structuring in subject-systematically related units)
P3 Principle of goal setting (communication of the respective teaching goal)
P4 Principle of individual learning speed (differentiation of lessons)
P5 Operative principle (mental operation with learning contents)
P6 Principle of stage appropriateness (contents according to the respective intellectual development stage)
P7 Spiral principle (relevant topics are to be treated at different levels)
P8 Genetic principle (alignment with epistemological principles of "creation and application" of the subject)

With imaging techniques in cognitive neuroscience, like, e.g., imitation learning [34], insights into the brain's functioning are now being gained, finding their way into school practice. This scientific direction is called "neurodidactics" [2][6][11], and already in the 1980s, brain-based pedagogy became popular in the Anglo-American world [16].

Brain-based pedagogy attempts to translate neuroscientific findings into didactically concrete recommendations for instructional design. Since many factors contribute to good teaching [31], and since the learning brain is a complex organ, the field of brain-based teaching is also not without controversy. However, insights into the learning brain's capabilities and constraints help explain why some learning environments support learning and others do not. Caine and Caine [6] summarize a lot of neurodidactic principles. The most important of them are:

N1 The learner needs opportunities to have concrete experiences.
N2 Learning processes involved in social situations are more effective, as is the consideration of interests and ideas.
N3 Connecting with prior knowledge is central to the learning process.
N4 Positive emotions lead to more effective learning.
N5 The learner needs to understand the connection between individual details and the whole, which helps her remember details better.
N6 Time for reflection improves learning (consolidation).
N7 A person learns better by combining information and experiences.
N8 Recognizing and integrating individual differences enhances learning.
N9 One learns better through a supportive, challenging environment. Attention to individual competencies is crucial.

Our department has developed a teaching approach called "COOL Informatics" [28], based on these neurodidactic principles. The acronym can be translated as "Cooperative Open Learning" or "Computer-Supported Open Learning" and is derived from an Austrian teaching model [14] that offers thematic, methodological, and institutional openness as well as cooperation on different levels and between other subjects.

The COOL-Principles are based on the following four cornerstones:

(1) *Discovery.* This includes solution-based learning; [32] observational learning; step-by-step instructions and tasks; video tutorials, hands-on, minds-on; learning with all senses.
The related neurodidactic basis is: pattern recognition; mirror neurons; individual learning rhythm; modality and multimedia effect.
(2) *Cooperation.* This includes team and group work; [35][12]; peer tutoring and peer teaching; [19][29] pair programming; [36][17] cross-curricular learning; project-based learning.
The related neurodidactic basis is: ”A joy (= knowledge) shared is a joy (= knowledge) doubled”; recall = re-storage in long-term memory; integrating individual needs, talents and competencies as well as practical relevance.
(3) *Individuality.* This includes competence-based learning; questioning; [13][7] self-organized learning with compulsory and optional tasks.
The related neurodidactic basis is: connecting new information to previous knowledge; considering individual interests, needs, tasks, methods, and learning rhythm.
(4) *Activity.* This includes learning by animation, simulation, and playing; [5][4] hands-on, minds-on; learning by doing; [13][30] learning by playing and designing games (creative learning).
The related neurodidactic basis is: knowledge must be newly created by each learner (= constructivism); learning is an active process (= progressive education).

Later, in Section 4, the relationship between these principles and the interventions is explained in more detail. The didactic concept serves as a background for several projects in our department. Since 2015, we have had more than 14,000 participants in our workshops and events and provide a Creative-Commons-based and rich set of materials (including associated didactic concepts) to all users of our workshops and partners, in addition to scientific monitoring.

Compared (and contrary) to existing literature, our approach as well as our considerations here are put in the light of the COOL approach. To the best of our knowledge, this didactic aspect is new and can be seen as a contribution to a refined HCI didactics.

3 PROJECTS' SETTING

3.1 Project Topics

Since 2015, our department is taking part in a funded, university-organized summer internship program for pupils aged 15 to 18 to promote technical studies in our region. Every year, pupils from the Carinthian State (and beyond) apply to this program, and we typically invite 10 to 20 pupils from different school types (with and without computer science education) for one month to learn more about computing science. They work for 30 hours a week and are accompanied by members of the department. Here, we report on the projects and interventions of two successive years (2018 and 2019) wherein a total of 44 students (17 female, 27 male) applied for seven different projects. Out of these, we selected 26 students of age 15 to 18, 69% coming from a secondary and vocational school with no computing-science background, and 31% coming from a technical vocational school with a bit of background knowledge in programming. Our selection is based on interviews to assess the pupils, making it possible for us to arrange project groups with excellent prospects of learning about computing science while also including some background knowledge.

To attract girls and boys equally, we decided to avoid gender cliches and gender-specific formulations. We came up, after several strategy meetings, with the following project topics (the projects marked by asterisks had an explicit focus on HCI in the projects descriptions):

[18-1] *Computer languages at your fingertips* (3 male, 1 female applications). The project was about finite automata, formal languages, the Turing machine. The students had to produce learning materials about these topics, program a robot, and implement a Turing machine program.

[18-2*] *Language in 3 dimensions - possibilities and limits* (2 male, 2 female applications). The project was about analyzing and improving an existing 3D/VR-Interface of an escape-room game and implementing 3D/VR-based learning materials about hardware components and the stack data-structure.

[18-3] *Computer Science - A Child's Play for Everyone?!* (4 male, 1 female applications). The project was about producing materials for and with kids, touching modeling, logic, and 3D printing.

[19-1*] *Playful learning in virtual reality* (14 male, 5 female applications). The project was again about analyzing the escape-room interface, extending an existing 3D/VR escape-room game, touching topics like graph representation, and creating a 3D-model of the Enigma.

[19-2*] *Secure communication through encryption* (12 male, 4 female applications). The project was about interface design with Blender and Unity for teaching encryption and producing short videos units about it.

[19-3] *Developing robotics for kids* (3 male, 2 female applications). In this project, the students had to test different educational robot systems and produce stop-motion videos about cubelets.

[19-4] *The beaver is out in computer science* (1 female applications). In that project, Bebras tasks had to be broken down to pupils age 4 to 8. Materials were developed and tested in small units.

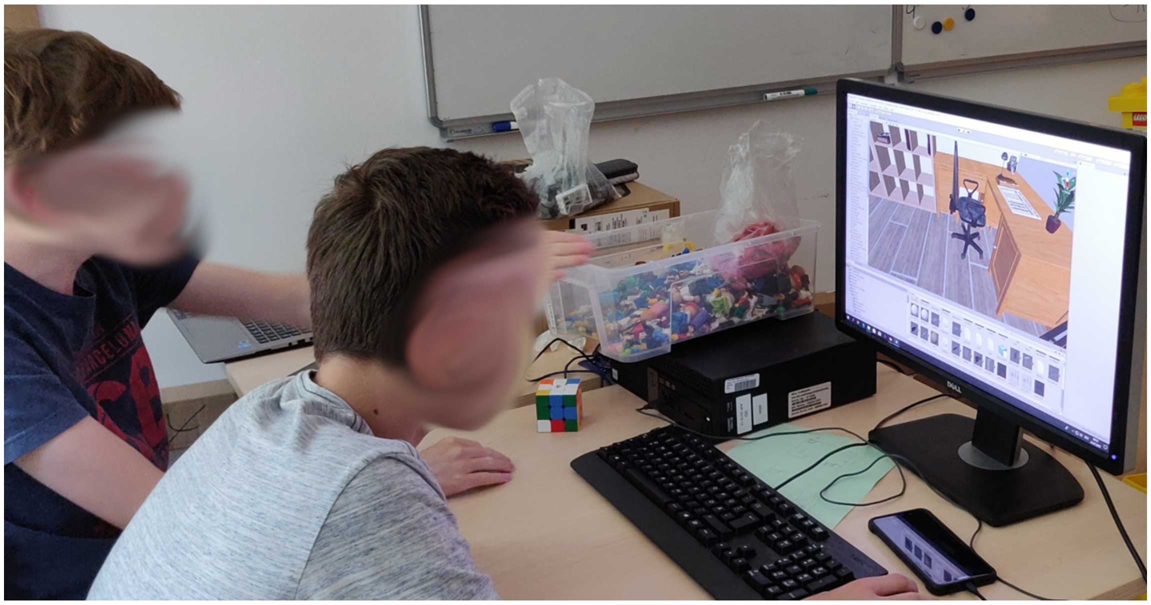

Fig. 1. Two students working together and creating one level of the escape-room game.

The project calls competed with dozens of other projects being advertised at our university (maths, geography, computer science). We learned a lot from the project call in 2018 (and the interviews with the applicants), and for 2019 we put even more focus on principle P1 (life proximity and topicality), stressing the communication with computers, the design of computer technology, and the interaction between users and computers. As can be seen, projects with explicitly focusing on communication and 3D/VR topics have attracted most of the students (summing up to 85% for HCI-related topics compared to 15% for the other topics in 2019).

3.2 Learning Objectives and Setting

Apart from core topics from computer science that can be seen in the project descriptions, in the projects [18-2], [19-1], and [19-2], we tried to incorporate the HCI topics mentioned in the international curricula in Section 2.2. That is, looking closer at *design decisions*, stimulating *user interface design*, thinking about issues of *accessibility* and the *user's needs*. Additionally, every student should be able to produce user interfaces considering various input and output devices. The general work order was as follows:

(1) Analyze an existing learning environment (e.g., our escape room game) and develop suggestions for improvement.
(2) Familiarize with the environment and make improvements.
(3) Incorporate a new exciting area of computer science to be taught through the game (a stack, encryption, or hardware components were given as idea generators.).
(4) Implement the competency as a different level in the learning environment.
(5) Let other teams test the implementation (and also test an implementation).
(6) Reflect on the usability and implement improvements.

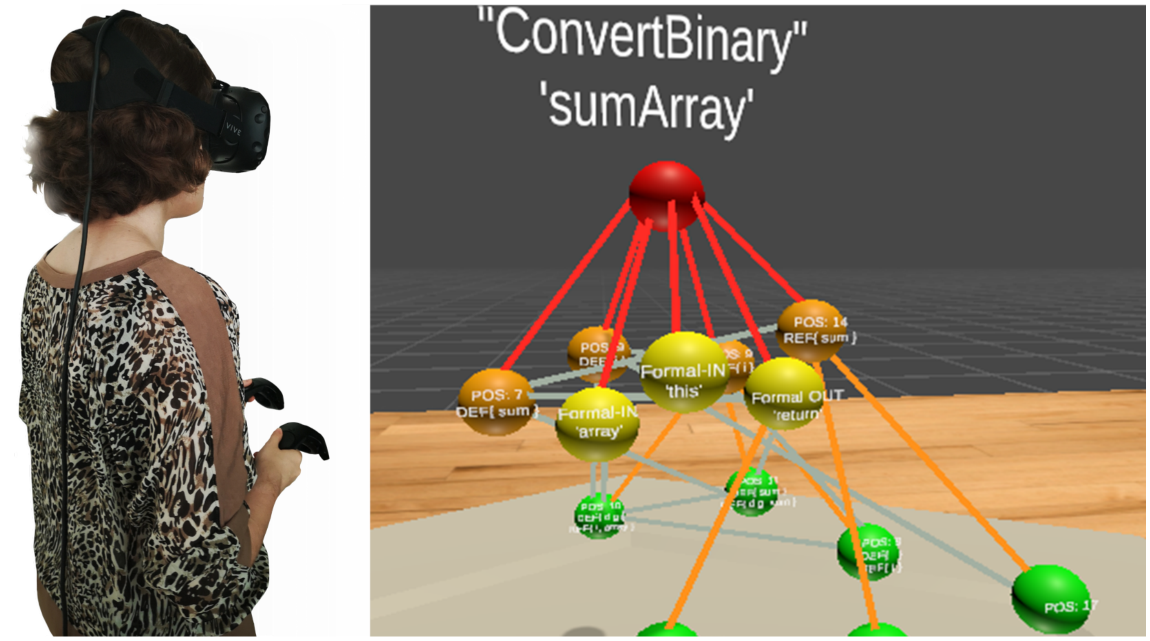

Fig. 2. Student using our HTC Vive to visualize control and data flow graphs for Java programs.

Along with the tasks, all teams had to cooperate with other teams and share results and products using state-of-the-art development practices.

To achieve the above goals, we provided the following environment for every team: a room equipped with laptops and workstations (Blender, the Unity game engine, and 3D printing software installed) suitable for working in pairs. The equipment was necessary, as about one-third of the students did not have own laptops. They also had access to flip charts/whiteboards/piles of paper/colored pencils, an interactive panel, 3D printers, a video projector, an HTC Vive, a Sony VR system, several educational robots, terminal blocks, and recreational equipment (Dartboard, sound system, and our library). Figure 1 shows an example of a team working on the setting of one of the escape rooms. As one can see on their table, they also use terminal blocks to test scenes and conditions in advance and only then implement them.

For software development (and any other artifact being created), we demanded compliance with standard development techniques. That implied using a git system, technical documentation using a CMS system, and, to analyze their learning progress, the students had to keep a learning diary.

For the development process, we were following an adopted SCRUM model. Every project had a primary mentor acting as SCRUM master (and one backup mentor). There were daily stand-up-meetings, accompanied by training sessions on-demand, but, as we only had four weeks (minus two vacation days and recreational activities), at most, two sprints were conducted. At the end of the project, the results were presented at a university's final major event.

For knowledge transfer, students were also used as peers across teams, and each team member knew that all materials developed would be used for instructional purposes in the future. All the projects started with analyzing existing environments and interfaces and a reflection phase where the requirements were defined (yielding a back-log). The mentors then directed the students to generate the products in a spiral fashion (in complexity and scope) within 3-4 weeks of work.

It is essential to mention that every team consisted of if at all, just one student from a technical vocational school (knowing a programming language). The rest of the team members were students from secondary schools without technical background (so they did not have computing science except one year in information and communication technologies). In 2019, just one student had a bit of knowledge of using Blender, but nobody had prior experience in HCI or Interface design. But, most important: none dropped out of the internship, and all projects were completed. Moreover, even complex tasks like the visualization of data structures and graphs were motivating enough to be completed and to end up in appealing protoypes (see Figure 2).

4 DISCUSSION

Even when an agreed-upon pedagogy seems to be missing [18], this contribution presents a way to link and use computer science and HCI topics to increase interest or build competencies in these two areas. However, is it a useful approach that is more than just anecdotal evidence? For an evaluation, let us look at the proper selection of the content, the didactic principles, and the students' acquisition of competencies. The discussion is put into the context of our **research objective**: investigating whether HCI can be used as a topic and didactic tool to initiate learning of computing science topics grounded on (neuro-)didactical principles.

4.1 Content

Modrow describes several criteria that have to be met when introducing new content in teaching [20]. Among others, the following criteria should be met:

- Students can work independently on the content.
- The content is suitable for lessons based on the division of labor.

In our case, all three topics were designed so that every student could design his/her level in the escape room game. Moreover, as a prototype of the escape-room (one room designed in 2016 that was used for language learning) was available, the entry barrier was assumed to be very low.

- The topic allows specializations/extensions.

As the possibilities and also the interests varied, totally different rooms were allowed to be created. One team focused on sound (as so far no sound was available), one team included self-designed 3D models (see Fig. 3), whereas others used objects from libraries. Moreover, another team put the gamer into the role of a person helping the (stack-based) computer to escape from the space station (see Fig. 4). So, we had many extensions of our prototype. The new skills were also generalized and stimulated for private projects (e.g., own YouTube videos, own music recordings).

- The content does not represent only a standard solution.
- There is a longer-term relevance of the topic.
- The experience gained can be generalized.

Working with Blender and the Unity game-engine implied developing several skills useful for future life (apart from learning the tools). The students learned about 3D and VR interfaces, different graphics formats, sound, navigating, assessing/criticizing solutions. They also learned a programming language and modern software engineering techniques. Furthermore, as mentioned above, all the covered content is also part of international curricula.

Fig. 3. Enigma modelled by team [19-02] in Blender for understanding en- and decryption processes and implemented in the project of team [19-01] in one room.

Concerning HCI, the projects focused on two mentioned learning outcomes: *"students will brainstorm ways to improve the accessibility and usability of technology products for users' diverse needs and wants"* [25], and *"students independently and safely plan, design, test, modify and create a range of digital solutions that meet intended purposes, including user interfaces and the use of a programming language"* [22]. Both could be achieved up to a certain level, as the executable products show, by giving the students the freedom to be creative and to discuss and implement their own ideas.

- The time required to learn user knowledge is reasonable.

All projects achieved their goal. However, this did not come for free. The team at the department invested a bit more than a month for preparation. Materials (instructional videos, checklists, web-links) were collected. However, when done, the students could learn the new skills quickly - also using their preferred media. There was only one team that complained a bit that they had to learn a lot on their own, but they also implemented more than necessary.

Summarizing, in relation to our research objective, we identify HCI as a suitable topic to initiate learning of computing science topics.

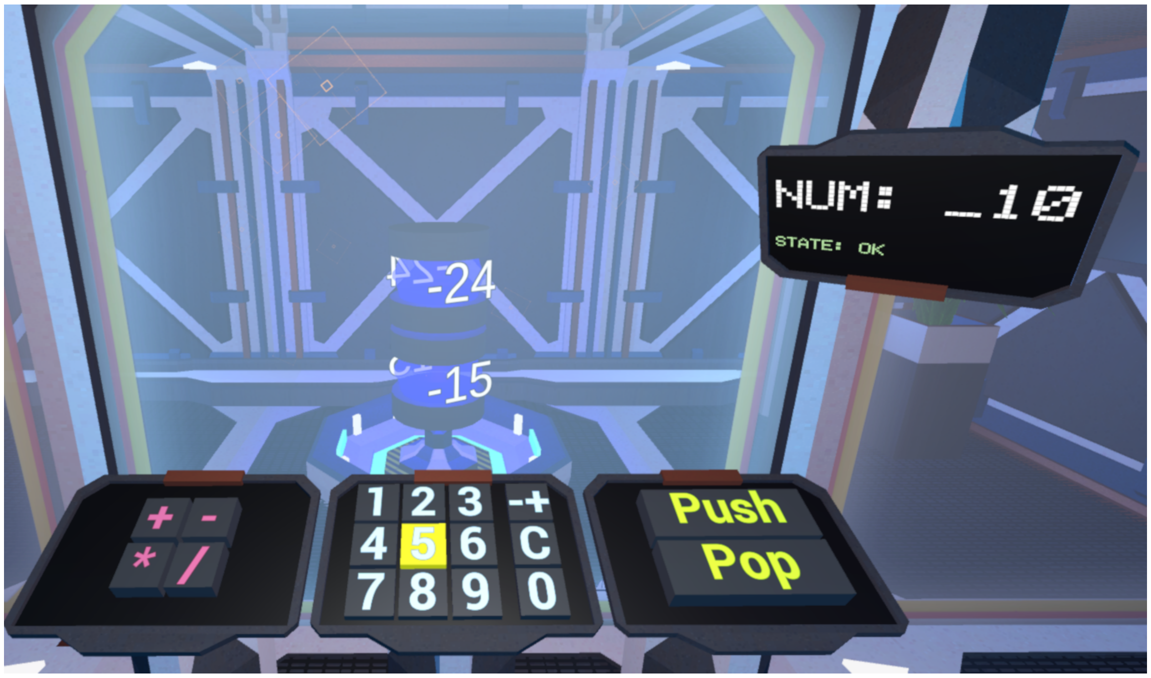

Fig. 4. Stack-simulator in the escape-room game implemented for HTC Vive.

4.2 Appropriateness of Didactic Principles

All projects followed the above-mentioned COOL Informatics approach, and this section now takes a closer look at the appropriateness of the tasks and the neurodidactic and general teaching principles.

Discovery Learning took place in multiple dimensions. We provided video tutorials (some of them produced by other students) at the beginning, including step-by-step instructions, which facilitate hands-on learning with multiple media and with an individual learning rhythm. Solution-based learning was facilitated by providing existing projects. The use of VR technology additionally stimulated the multimedia effect of *Discovery Learning*.

Cooperation was encouraged by our preparation of the projects, teams, and working environment. We encouraged pair programming and teamwork supported by peers and faculty members. Each project was clearly defined and followed a clear vision, shaped by the project supervisors and the students. The agile development process SCRUM also encourages team and group work.

Individuality was supported with open-ended project work, facilitating self-organized learning. Each student could use their strengths to progress the project and learn new and exciting tools, techniques, and computing science topics. We, therefore, regard *Individuality* in our project work as an added value. The learning journals additionally confirm the fulfillment of this principle.

Finally, *Activity* was naturally supported by a lot of hands-on and learning by doing activities. This includes designing games (incorporating animation, simulation, and playing). Especially playing and designing games was naturally given by choice of the project topics. The students were empowered to construct their knowledge, with learning as an active process, by designing and implementing learning environments and accompanying HCI elements in 3D and VR.

When taking a look at the principles as defined by Baumann (a "P" and the number indicating the coverage) and Caine and Caine (an "N" and the number indicating the coverage), we also fulfill all the requirements.

- We have chosen the topic of an escape-room game with riddles as computer games are naturally part of pupils' current life-world. Also, they very likely knew some interfaces in that area. It also connects to prior knowledge and, at least it was assumed, also positive emotions (P1, N3, N4).
- Different escape-rooms allowed for a subject-related structure (P2). The students were allowed to look at existing rooms and tasks and to collect their own concrete experiences with the possibility also to fail (N1).
- At the beginning of the project, it was clear that all the work contributes to future learning materials for students in schools and at universities ... and future internships (P3, N2).
- Every team member focused on their level and ideas, so individual learning speed was not a problem when team-formation was appropriately done. According to one's own goals, he or she got much feedback from team members and also the mentors (P4, N7, N8, N9)
- Every team member reviewed others' work and followed and reflected on others' ideas mentally. They understood the whole and their role and tasks better and had to coordinate their work (P5, N2, N5, N6).
- Not everyone started with the same set of competencies. So, tasks of different difficulty levels were prepared to be adjustable to existing skills (P6). This was necessary to ensure that everyone in the project had a sense of achievement (N4).
- The topics were prepared in such a way that first existing work had to be improved, then computing science content had to be worked out, and finally, it hat do be integrated to the new solution ... allowing for a spiral principle in working through the topics (P7)
- Between all the steps, a reflection step had to take place, which meant trying everything out and thinking about it. It meant working together with other teams and members of the department (P8).

Summarizing the coverage of both neuro-didactical COOL Informatics principles and the mentioned didactic principles, concerning our research objective, we identify HCI as a suitable didactic tool to initiate learning of CS topics.

4.3 Learning Diaries

All students had to keep a learning diary and hand it in at the end. Besides, the mentors observed them. The students produced a poster, which was presented publicly on the last day of the internship at the university campus to parents and other guests.

One very positive statement we collected was the following (translated to English): *There was much freedom in the internship, which promoted our self- and time management. From a computer-science point of view, Unity and C# were new experiences for most of the team. VRTK (Virtual Reality Tool Kit), on the other hand, was new for all of us, so there were some complications in this area. We versioned with Git in the project and gained new experience or deepened existing ones.* It confirms that, even though the technical requirements were very high, it was doable by the students at the age of 15 to 18.

However, another statement can be seen quite critically (translated to English): *We had to learn the Blender program ourselves. This was very fun, but sometimes it would have been easier if someone had known their way around. It was an exciting internship where we learned a lot of new things. An essential point of the internship is especially creativity. We were given a task and could do it the way we wanted to.* This is something that we experience quite often in our workshops and interventions. Due to the predominance of frontal teaching (in our region?), students are less accustomed to independent work. It always takes some time to get used to it and then to enjoy it.

Finally, again a resume that shows that at least from the motivation point of view, the internships have been a real success: *The internship allowed us to take away some knowledge of Unity and C# and become more familiar with the version control system Git. We got to know the informatics content in a new way. However, the most fascinating for all of us was the first experience with virtual reality.*

In all, combining neurodidactic principles with appealing topics seems to have enabled a high learning outcome, a positive feeling about computing science in general, and, even more important, interested and hopefully future students at our University.

5 CONCLUSION

HCI is rarely used in didactic considerations at the secondary level. In this paper, we, therefore, took a closer look at didactic and neurodidactic principles and related them to projects (summer internships) at our university where students between the ages of 15 and 18 worked through 3D, Virtual Reality, and usability issues. As we competed with other departments at our university for students, it turned out that HCI topics are a perfect driver for attracting students to our projects.

By systematically reviewing our project settings and tasks for students in terms of 8 widespread general and current teaching principles, we can also show that our approach, combined with nine neurodidactic findings, is very well suited for use in an educational setting. Although our study is not based on a large group of students, the results show that the approach also helps students reach peak performance through play. With that, HCI is more than a just topic in the computer science curriculum. It is also a tool and should be used as such.

So far, the results are only anecdotal evidence, but the didactic reflection leads to a solid foundation. As future work, we plan to collect more data about the students' personalities, interests, self-concept (especially those of young women), and the increase in knowledge and competencies to understand better which interventions affect those attitudes.

ACKNOWLEDGMENTS

To all the staff of our department, the Pedagogical University and the Directorate of Education, who help us develop and test new materials. Without this cooperation, the projects described here could not have been carried out. Thank you.

REFERENCES

[1] P. D. Adamczyk and M. B. Twidale. 2007. Supporting Multidisciplinary Collaboration: Requirements from Novel HCI Education. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems 2007, CHI 2007.* Association for Computing Machinery, New York, NY, USA, 1073–1076.

[2] P. A. Arndt and M. Sambanis. 2017. *Didaktik und Neurowissenschaften. Dialog zwischen Wissenschaft und Praxis.* Narr Studienbücher, Tübingen, Deutschland.

[3] R. Baumann. 1993. Ziele und Inhalte des Informatikunterrichts. *Zentralblatt für Didaktik der Mathematik* 25 (1993), 9–19.

[4] T. Bell and L. Lambert D. Marghitu. 2012. CS unplugged, outreach and CS kinesthetic activities. In *PProceedings of the 43rd ACM technical symposium on Computer Science Education.* ACM, Raleigh, North Carolina, USA, 676.

[5] A. Bollin, E. Hochmüller, and R. Mittermeirand L. Samuelis. 2012. Experiences with Integrating Simulation into a Software Engineering Curriculum. In *Proceedings of 25th IEEE Conference on Software Engineering Education and Training CSEE&T 2012.* IEEE Computer Society Press, Los Alamitos (CA), 62–75.

[6] R. N. Caine and G. Caine. 1990. Understanding a brain-based approach to learning and teaching. *Educational Leadership* 48, 2 (1990), 16–70.

[7] S. D. Craig, J. Sullins, A. Witherspoon, and B. Gholson. 2006. The deep-level-reasoning-question effect: The role of dialogue and deep-level-reasoning questions during vicarious learning. *Cognition and Instruction* 24, 4 (2006), 565–591.

[8] P. Denning. 2003. Great Principles of Computing. *Commun. ACM* 46 (2003), 15–20.

[9] Christina Dörge. 2014. A Methodological Approach to Key Competencies in Informatics. In *Proceedings of the 2014 Conference on Innovation & Technology in Computer Science Education* (Uppsala, Sweden) *(ITiCSE '14)*. Association for Computing Machinery, New York, NY, USA, 201–206. https://doi.org/10.1145/2591708.2591742

[10] Jorge Ferreira Franco and Roseli de Deus Lopes. 2009. Three-Dimensional Digital Enviroments and Computer Graphics Influencing K-12 Individuals' Digital Literacy Development and Interdisciplinary Lifelong Learning. In *ACM SIGGRAPH ASIA 2009 Educators Program* (Yokohama, Japan) *(SIGGRAPH ASIA '09)*. Association for Computing Machinery, New York, NY, USA, Article 15, 8 pages. https://doi.org/10.1145/1666611.1666626

[11] G. Friedrich. 2005. *Allgemeine Didaktik und Neurodidaktik - Eine Untersuchung zur Bedeutung von Theorien und Konzepten des Lernens, besonders neurobiologischer, für die allgemeindidaktische Theoriebildung.* Peter Lang, Frankfurt am Main, Deutschland.

[12] B. Gokkurt, S. Dundar, Y. Soylu, and L. Akgun. 2012. The Effects of Learning Together Technique Which is based on Cooperative Learning on Students' Achievement in Mathematics Class. *Procedia-Social and Behavioral Science* 46 (2012), 3431–3434.

[13] J. Hattie. 2009. *Visible Learning: A Synthesis of Over 800 Meta-Analyses Relating to Achievement.* Taylor & Francis, London, New York.

[14] (Online) R. Hölbling, H. Wittwer, and G. Neuhauser. 2021. *COOL - Cooperatives Offenes Lernen.* Impulszentrum für Cooperatives Offenes Lernen. Retrieved February 26, 2021 from http://www.cooltrainers.at/fileadmin/impulszentrum

[15] L. Humbert. 2005. *Didaktik der Informatik mit praxiserprobtem Unterrichtsmaterial.* Teubner, Wiesbaden, Deutschland.

[16] E. Jensen. 2008. *Brain-based learning: the new paradigm of teaching (2nd ed.).* Corwin Press, California, USA.

[17] C. M. Lewis. 2011. Is pair programming more effective than other forms of collaboration for young students? *Computer Science Education* 21, 2 (2011), 105–134.

[18] S. Lewthwaite and D. Sloan. 2016. Exploring pedagogical culture for accessibility education in computing science. In *Proceedings of the 13th Web for All Conference (W4A'16)*. Association for Computing Machinery, New York, NY, USA, 3:1–3:4. https://doi.org/10.1145/2899475.2899490

[19] V. Miller, E. Oldfield, and M. Bulmer. 2012. Peer Assisted Study Sessions (PASS) in first year chemistry and statistics courses: insights and evaluations. In *Proceedings of The Australian Conference on Science and Mathematics Education (formerly UniServe Science Conference)*, Vol. 10. UniServe Science, Sydney,Australia, 30–35.

[20] E. Modrow. 2003. *Pragmatischer Konstruktivismus und Fundamentale Ideen als Leitlinien der Curriculumentwicklungn.* Ph.D. Dissertation. Martin-Luther-Universität Halle-Wittenbergy. http://sundoc.bibliothek.uni-halle.de/diss-online/03/03H066/prom.pdf

[21] A. Oleson, B. Wortzman, and A. J. Ko. 2020. On the Role of Design in K-12 Computing Education. *ACM Transactions on Computing Education* 21, 1 (2020), 1–34. https://doi.org/10.1145/3427594

[22] (Online). 2021. *Australian Curriculum for Technologies.* Australian Curriculum, Assessment and Reporting Authority. Retrieved February 26, 2021 from https://www.australiancurriculum.edu.au/f-10-curriculum/technologies/

[23] (Online). 2021. *Bildungsstandards Informatik SI und SII.* Gesellschaft für Informatik. Retrieved February 26, 2021 from https://informatikstandards.de/

[24] (Online). 2021. *Digitale Kompetenzen Informatische Bildung.* National Competence Center eEducation Austria. Retrieved February 26, 2021 from https://digikomp.at/

[25] (Online). 2021. *K-12 Computer Science Standards.* Computer Science Teacher Association. Retrieved February 26, 2021 from https://www.csteachers.org/page/standards

[26] (Online). 2021. *Lehrplan 21: Modul Medien und Informatik.* Deutschschweizer Erziehungsdirektoren-Konferenz (D-EDK). Retrieved February 26, 2021 from https://v-fe.lehrplan.ch/index.php?code=b|10|0&la=yes

[27] (Online). 2021. *National curriculum in England: computing programmes of study.* UK Department for Education. Retrieved February 26, 2021 from https://www.gov.uk/government/publications/national-curriculum-in-england-computing-programmes-of-study

[28] S. Pasterk, B. Sabitzer, H. Demarle-Meusel, and A. Bollin. 2016. Informatics-Lab: Attracting Primary School Pupils for Computer Science. In *Proceedings of the 14th LACCEI International Multi-Conference for Engineering, Education, and Technology: Engineering Innovations for Global Sustainability.* LACCEI, San Josè, Costa Rica, 7.

[29] L. Porter, C. Bailey Lee, and B. Simon. 2013. Halving fail rates using peer instruction: a study of four computer science courses. In *Proceeding of the 44th ACM technical symposium on Computer science education*, Vol. 10. ACM, Denver, Colorado, USA, 177–182.

[30] H. W. Reese. 2011. The Learning-by-Doing Principle. *Behavioural Development Bulletin* 17, 1 (2011), 1–19.

[31] E. Reçi and A. Bollin. 2018. A Teaching Process Oriented Model for Quality Assurance in Education - Usability and Acceptability. In *IFIP TC 3 – Open Conference on Computers in Education.* Springer Verlag, Linz, Austria, 128–137.

[32] A. Renkl and R. K. Atkinson. 2002. Learning from examples: Fostering self-explanations in computer-based learning environments. *Interactive learning environments* 10, 2 (2002), 105–119.

[33] G. Röhner, T. Brinda, V. Denke, L. Hellmig, T. Heußer, A. Pasternak, A. Schwill, and M. Seiffert. 2016. Bildungsstandards Informatikfür die Sekundarstufe II (in German). In *LOG IN 36. Jg. (2016), Heft Nr. 183/184.* LOG IN Verlag GmbH, Bonn/Berlin, Deutschland, 1–80.

[34] Giacomo Rizzolatti and Laila Craighero. 2004. The Mirror-Neuron System. *Annual review of neuroscience* 27 (02 2004), 169–92. https://doi.org/10.1146/annurev.neuro.27.070203.144230

[35] C. J. Roseth and D. W. Johnson amd R. T. Johnson. 2008. Promoting early adolescents' achievement and peer relationships: The effects of cooperative, competitive, and individualistic goal structures. *Psychological bulletin* 134, 2 (2008), 223.

[36] N. Salleh, E. Mendes, and J. Grundy. 2011. Empirical studies of pair programming for CS/SE teaching in higher education: A systematic literature review. *IEEE Transactions on Software Engineering* 37, 4 (2011), 509–525.

[37] A. Schwill. 1994. Fundamental Ideas of Computer Science. *EATCS-Bulletin* 53 (1994), 274–295.

[38] D. Seehorn, S. Carey, B. Fuschettoand I. Leeand D. Moix, D. O'Grady-Cunniff, B. B. Owens, and C. Stephensonand A. Verno. 2011. *CSTA K–12 Computer Science Standards: Revised 2011.* CSTA, New York, NY, USA.