My Interactive Garden – A Constructionist Approach to Creative Learning with Interactive Installations in Computing Education

Mareen Przybylla, *przybyll@uni-potsdam.de*
Institute of Computer Science, University of Potsdam, Germany

Ralf Romeike, *romeike@cs.uni-potsdam.de*
Institute of Computer Science, University of Potsdam, Germany

Abstract

In this article the question is pursued, how developments in interactive computing systems can be harnessed to strengthen constructionist learning in computing education. Based on an analysis of interactive computing systems, My Interactive Garden provides a concept for motivating principles of computing including a construction kit, activities and examples that may empower students to create meaningful interactive objects.

Keywords

Constructionist learning in Computing Education, interactive computing systems, Arduino

Introduction

With the Logo microworld and the corresponding constructionist approach to learning Papert (1980) introduced a sound concept to learning, especially in mathematics, that in the following years had a tremendous impact on learning in other subject areas as well. Especially teaching and learning of principles of computing in primary and secondary education, e.g. programming, until today very much rely on the ideas of Papert. Microworlds and virtual interactive robots are commonly used in introductory courses. As such, tools like Kara (Hartmann, Nievergelt, & Reichert, 2001) and Karel the Robot (Pattis, 1981) provide early experiences in programming to students in computing education. However, criticism of these tools often refers to a lack of tangibility, a lack of creativity and an outdated understanding of what programming or computing really is about. Even with the further development of Logo, e.g. Lego Mindstorms, creative possibilities are limited. Also, it seems to be difficult to increase female participation in Lego Robotics activities (cp. Resnick, 2007).

Due to the technical developments within the last years computers are no longer considered as machines that merely receive and act on orders, but as interactive and ubiquitous media. This development of interactive computing systems can be used in order to address the above-mentioned issues. Corresponding projects have been used successfully in primary education (e.g. with Pico Cricket, cp. Rusk, Resnick, Berg, & Pezalla-Granlund, 2008). With the concept of *My Interactive Garden* this trend shall be seized and adapted for computing education by pursuing the question how developments in interactive computing systems can be harnessed to strengthen constructionist learning in computing education. The question is addressed with the analysis of recent progress in interactive computing systems and by integrating the findings into a constructionist approach for learning fundamental ideas of computer science. As a consequence, a constructionist learning environment is proposed, which focuses on creative learning, supports moti-

vation and leaves the pure virtual world by offering to design tangible interactive objects for ubiquitous media installations. In this way, a constructionist approach can be applied at all levels. It makes an elementary introduction to physical computing possible, but also the sophisticated application and programming of microcontrollers with advanced tools. Additionally, this approach allows for providing a more appropriate notion of computer science as a multi-facet discipline by motivating questions of theoretical concepts, hardware aspects, applications of computing devices as well as referring to influences of and on society.

In chapter 2 the state of problems in the context of constructionist approaches to learning in computing education will be discussed. Chapter 3 scrutinizes the perspectives on interactive computing systems for their relevance for computing education. The findings are integrated into the concept of *My Interactive Garden*, which is elaborated and illustrated in chapter 4. Finally, the concept and its prospects are discussed in the context of classroom demands and experiences.

Constructionist Approaches to Learning in Computing Education

First comprehensive reports about constructionist learning in primary education with the Logo programming language date back as far as the 1980s (e.g. Hoppe & Löthe, 1984, Ziegenbalg, 1985). The intention was not primarily to teach principles of computing but to allow learners to explore and understand mathematical and geometrical structures in a constructionist way. As Papert argues, "[learning] happens especially felicitously in a context where the learner is consciously engaged in constructing a public entity, whether it's a sand castle on the beach or a theory of the universe" (Papert & Harel, 1991). Hence, the construction of knowledge is based on an active construction process. In such a way a meaningful artifact will be created, which the learner can try out, show around, discuss, analyze and receive praise for. It is the examination of such an artifact that leads to the understanding of a particular phenomenon. Microworlds describe computer assisted learning environments in which such learning can happen unhindered by the complexities of the world (Papert 1996). This approach was adopted for computing education: Various microworlds by now are used in classroom, e.g. for learning programming (Karel the Robot, Pattis, 1981) or for offering a more motivating approach to theoretical concepts, such as finite state automatons (e.g. Kara the ladybug, Hartmann et al., 2001). However, such microworlds receive criticism for fostering an inappropriate and possibly unattractive notion of computing, which is not primarily concerned with moving robots or lady bugs through labyrinths. Additionally, these microworlds seem to violate the underlying ideas of Logo: Empowering the learner to create a personally meaningful product. Instead, the corresponding exercises of Karel, Kara and similar learning environments pose unauthentic and often merely algorithmic problems (cp. Romeike, 2008). In such contexts constructionist learning can barely happen.

Lego Mindstorms transfers the idea of Logo and Microworlds into the tangible world. Due to the possibility of making programs "come to life" it is used widely in computing education (e.g. Wiesner & Brinda, 2007). But criticism of Lego Mindstorms points out that such constructions can easily become complex, have to cope with a variety of mechanical problems and hence are difficult to achieve, especially for younger children. Also, even though it is possible to create a variety of constructions with Lego Mindstorms, mostly robotic vehicles are built (ibid.). Thus, in educational contexts, it is difficult to address a broad range of interests. As a consequence Lego Robotics is mostly used in non-formal educational settings, such as school clubs and workshops. By analyzing the outcomes of a Girls Scout workshop series over several years Guzdial (2010) found that activities with Lego Robotics did not lead to a positive attitude towards computer science significantly. However, interest of the participants and subsequently a positive notion of computer science were determined after activities with Scratch and Pico Cricket. Pico Cricket is a

construction, programming and learning environment that connects constructionist learning with the tangible world. Its creations can be touched, shown to others, be played with and more. Pico Cricket may be a good way to start with in computing education in primary school, but the system is too trivial for secondary education. Also, for a broader dissemination the available sets are too expensive, difficult to order and break too easily. Even though Pico Cricket was already introduced in 2006, until now it is hardly used in computing education (at least in Germany). Instead, more and more approaches focus on using the Arduino platform (see chapter 3). For the above-mentioned reasons the concept of *My Interactive Garden* builds on such a platform, which is customizable, reasonable, but can be programmed in a variety of ways and is adapted in a way that it is as easy to use as Pico Cricket.

Interactive Computing Systems

Nowadays computing systems have integrated into everyday life. They are no longer perceived as mindless machines that receive orders and act accordingly, but as intelligent devices, which ease and enrich people's lives. Interactive computing systems are based on various innovative ideas such as embedded systems, ubiquitous computing, physical computing and interactive installations. This should become apparent in computing education. Recent developments in this field will be analyzed. The findings shall then be integrated into a constructionist approach for learning fundamental ideas of computer science.

Embedded Systems

Embedded systems pervade nowadays life. Many of the technical artifacts that are used every day contain an embedded system in some way. Due to its importance, this phenomenon can also be called pervasive computing (Weiser, 1991). Embedded systems can be described as "a combination of micro(s), sensor(s), and actuator(s) designed for some specific control function and 'embedded' into a specific device, usually requiring little human input. An example would be the air/fuel mixture control system in an automobile engine" (Pardue, 2010). Thus embedded systems perform single, straightforward tasks, in comparison to personal computers, which are expected to interact with humans and perform a broad variety of tasks. Robotics can be classified as intelligent embedded systems, which integrate hard- and software in one system, but are capable of performing tasks that rely on mechanical actions. Hence, robotic systems are often found in industrial environments. Typical constructionist tools for teaching, such as Lego Mindstorms and Pico Cricket can be classified as tools for creating embedded systems. Even though youth gets in touch with embedded systems on a daily basis, only few have the possibility to create their own embedded system and understand the functionality, potential and limits of such devices.

Ubiquitous Computing

In 1993 Weiser already described his vision of new hardware systems that should be developed for something he called ubiquitous computing: "Ubiquitous computing enhances computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user" (Weiser, 1993). The ideas of embedded systems and ubiquitous computing are closely interrelated. Without embedded systems ubiquitous computing would be impossible. The overall idea is to enhance the usability and efficiency of computers and, at the same time, make them invisible by integrating the systems completely into everyday life. If computers do not stand out anymore, comparable to electricity, the vision succeeded (West, 2011). Weiser sees therein an about-face of computer science: "But it is a start down the radical direction, for computer science, away from emphasis on the machine and back on the person and his or her life

in the world of work, play, and home" (Weiser 1993). For computing education this is a very attractive perspective to learn and discuss computer systems, their capabilities and their limits, but also to invent and create computing artifacts that follow this idea. Typical products of ubiquitous computing are keys that send their position when lost or self-regulated lighting, as opposed to notebook or tablet computers, which are portable, but do not "understand" their environment and act appropriately. In educational settings ideas of ubiquitous computing have been especially used in the context of "Wearables" (e.g. Martin, 2003, Resnick, 2007) and in higher education (e.g. Richards & Smith, 2010) where students have created useful devices, such as weather stations, based on their personal interest. The design of such devices may serve as a challenging task if creatively generating ideas and finding solutions to particular problems, where ubiquitous computing is useful.

Physical Computing

Physical computing is an activity that increasingly received attention within the last years, especially by non-computer scientists, such as artists and designers. The idea of physical computing is to use programmable hardware for creating interactive physical systems. Since these systems use sensors (e.g. for noise, light, cp. chapter 4) and actuators (e.g. motors, lamps, cp. chapter 4) they are capable of sensing and responding to the analog world, thus helping to investigate and redefine the relationship between humans and the digital world (cp. O'Sullivan & Igoe, 2004). Due to its creative potential, artists and designers use techniques of physical computing for handmade art or interactive installations. This is an opportunity for computing education to use the interrelations between art and computing in order to increase the attractiveness of the subject. Physical computing promotes prototyping with electronics, which leads to tinkering with ideas of computing (cp. Banzi, 2011). Summarizing, physical computing is the design and creation of interactive objects. This idea perfectly matches with the primary idea of constructionist learning, which has the creation of personal meaningful artifacts in its core. These artifacts may now become interactive.

Arduino

Physical computing requires a microcontroller, which can be programmed to control a variety of sensors and actuators. The hardware with the largest prevalence and the most active community at the moment is *Arduino*. Arduino boards are microcontrollers, which exist in different shapes for different purposes. They all have in common that in addition to in- and output pins they consist of microprocessors and flash memory and offer the possibility for external power supply. Arduinos do not require a permanent connection to the computer and are therefore, but also because of their small size, suitable for embedding into interactive objects. Extending Arduino boards with particular shields offers the possibility of customizing the board and e.g. adding WiFi or Bluetooth functionality to the microcontrollers. This way, changes in programs can easily be made without deconstructing the whole object. This also allows controlling an object with mobile devices such as smart phones or tablet computers. The use of Arduino offers nearly unlimited possibilities concerning the choice of sensors and actuators. The large variety of components also allows to easily and inexpensively construct spare parts and upgrade an Arduino construction kit.

Interactive Objects in Interactive Installations

Interactive installations may form a constructionist approach to computing education that combines the goals of teaching principles of computing with the ideas of embedded systems, ubiquitous computing and physical computing. Students are encouraged to follow their own interest by realizing various ideas, whether they are stemming from arts, music, technology, or everyday life.

Rusk et al. (2008) describe interactive installations in the context of robotics, which comprise "all types of programmable machines that perform actions based on inputs from sensors – everything from a home security system that sounds an alarm when it detects motion to a greenhouse that regulates its temperature and humidity." In this context they suggest the following strategies in order to raise participation:

1. Focus on Themes (Not Just Challenges)
2. Combine Art and Engineering
3. Encourage Storytelling
4. Organize Exhibitions (Rather than Competitions)

In such a way learners are encouraged to not just copy or rebuild systems which they already know but to use their imagination and creativity in order to develop personally relevant interactive objects that can be used in interactive installations.

Interactive Objects are integrated systems containing a miniature computer (microcontroller) that is invisible to the outside world. They perceive their environment with sensors, which in turn deliver data to be processed by the microcontroller. According to the configuration of the systems these data are processed and passed on to the actuators. In this way, interactive objects communicate with their environment. They are created with crafts, art and design material. They fulfill a particular purpose, which may be purely artistic. Interactive objects can be part of networks of interactive installations.

My Interactive Garden

People sometimes dream about the Garden of Eden – a place where things happen right as you desire. The idea of *My Interactive Garden* is to demonstrate that engaging with principles of computing and creating interactive objects can get one closer to realizing a dream. *My Interactive Garden* is the concept of realizing cooperative exhibitions of interactive installations. Such installations consist of embedded artifacts that implement the idea of ubiquitous computing by applying concepts of physical computing with Arduino microcontrollers. Yet, the application of such a concept in educational settings has been difficult due to the complexity of microcontrollers and the limited technical capabilities of existing construction sets for children. With *My Interactive Garden* a construction kit is developed, which reduces the complexity of using and programming Arduino. It allows for immediate tinkering and trial without the need of elaborated skills in physics or principles of electrical engineering, such as soldering. Also, programming of interactive objects in *My Interactive Garden* can be approached in different levels (cp. fig. 2). Hence, staying in the same context while progressing with programming or computing skills or conducting cross-level projects is possible. With Scratch for Arduino (S4A, 2012) in connection with Arduino microcontroller boards already at a low age level immediate tinkering and constructing of interactive objects is possible even for novices. *My Interactive Garden* hence is based on three pillars: constructionist learning, interactive computing systems and thinkering[1]. Students are encouraged to creatively use their imagination in a challenging constructionist learning environment where there is no predetermined way of doing things. They create interactive installations by ex-

[1] Thinkering: A portmanteau of "thinking" and "tinkering". Means to think about something by tinkering with objects relating to the problem under consideration. Usually unguided, exploratory and individual, often a very good way to explore aspects of difficult problems or to find solutions where none are obvious. (www.urbandictionary.com/define.php?term=Thinkering)

perimenting, analyzing and improving their work and methods. Additionally, students are motivated to make use of and find out about computer science concepts they happen to need for their projects.

The Construction Kit

It is the idea of *My Interactive Garden* to provide a construction kit based on Arduino that overcomes the barrier of technical complexity and encourages immediate tinkering. Until now the complexity of the Arduino hardware prevented from such an approach in elementary instruction. Unlike the Scratchboard, Arduino components are not easy to handle, e.g. you need to apply electrotechnical knowledge if adding a button or light because an additional resistor needs to be added to the circuit. It is therefore necessary to develop components that do not require students to solder or to work with breadboards, which quickly gets confusing. Hence, the items of the construction kit are built considering the following principles.

Simplicity: The items are provided with easy to use plugs so that students do not have to handle tiny wires that easily break or slip off the pins on the Arduino.

Flexibility and Extensibility: Sensors and Actuators can easily be added, removed or exchanged. Extension cords are provided to allow students to mount their parts in a distance to the boards.

Black/whiteboxing: Those components that contain circuits of subordinate relevance (e.g. the assembly of a brightness sensor) are hidden in a black box. However, if intended the black box can be “opened” by recreating such sensors, actuators or boards or by examining the corresponding data sheet.

Emphasize computing principles: Underlying computing principles are visualized, e.g. the IPO model is demonstrated by using separate boards for in- and output.

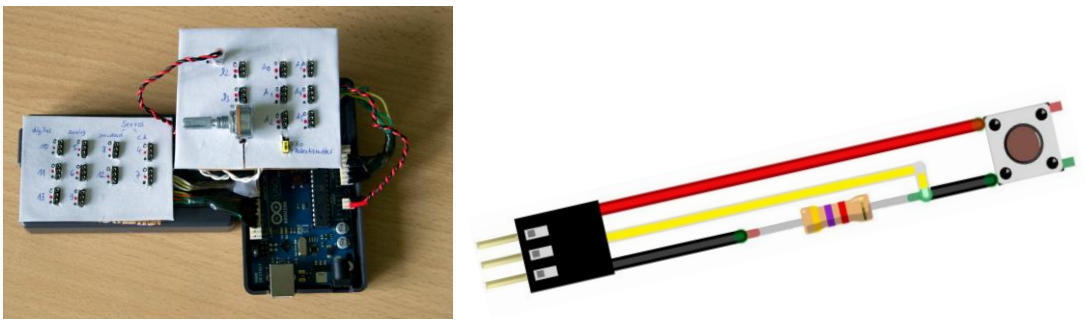

Fig. 1: Left: Boards for sensors and actuators with the Arduino. Right: Circuit of a button.

The construction kit contains the following items:

The Sensor and Actuator Boards

In order to simplify the handling of sensors and actuators on the Arduino and to separate inputs and outputs visually, two separate boards will be used (cp. fig. 1). These can be plugged into a single Arduino microcontroller. The input pins of the sensors and actuators are grouped according to their use in S4A, which is also in line with the use in other programming environments (e.g. ModKit). Arduino boards are not optimized for plugging in pre-assembled sensors and actuators, which makes experimenting more complicated than necessary. With the sensor and actuator boards produced for classroom use, the connectors, which consist of voltage, ground and data pins, are placed next to each other for every in-/output, so that sensors and actuators can easily be plugged in. Color-code labels will prevent users from plugging in components incorrectly.

The Sensors

The sensors are prepared to allow students to plug them in and read their sensing values immediately. Everything but the head of the sensor and the plug will be a black box to the students, e.g. the resistor belonging to the button shown in figure 1 is invisible. As follows, the sensors of the construction kit are described including their technical background and possible uses.

Brightness sensor: A brightness sensor is composed of a light dependent resistor and a pull-down-resistor, which makes it possible to send data to the microcontroller. The resistance changes with the intensity of the ambient light. The darker the environment, the higher the resistance. Threshold values that correspond with particular brightness levels can be used to control actions and actuators.

Temperature sensor: Temperature sensors are very similar to light sensors. The resistance of the sensor changes according to the ambient temperature. When 0°C is measured a value of 0 will be read on the Arduino pin (which means that the resistance is as high as it can be). When 100°C are measured a value of 1023 will be read on the Arduino pin (which means that there is no resistance at all). It is recommended to use average values when calculating temperatures in order to compensate variations in voltage.

Sound sensor: A sound sensor basically consists of a microphone and a preamplifier. To build a good sound sensor many components such as different capacitors are needed. Hence, there are two options: First, a pre-assembled sound sensor can be used. Second, the sensor can be assembled by oneself. Using those sensors depends on what shall be measured. The easiest way is to use them as noise detectors, which means to measure differences to normal ambient noise. This would again imply to determine particular threshold values. In more advanced settings (requiring good sound sensors), particular sounds will have particular patterns. These patterns can be detected, analyzed and used.

Switch and button: The switches used in the construction kit are actually buttons, which have been modified with pull-down-resistors to allow for reading their current state (pushed / not pushed) with the Arduino. Thus - controlled by software - they can be used as switches. To do so, the corresponding pin has to be watched. When the button is pushed, a switch-variable changes its value. A delay of a few hundred microseconds has to be implemented in order to prevent the variable to change its value too quickly (unless intended). Alternatively, switches that mechanically keep their state could be used. The buttons in the kit can of course still be used as simple push buttons.

Potentiometer: Potentiometers are changeable resistors. They are perfect for manually and continuously controlling actuators such as the brightness of lights, the volume of speakers or the speed of a motor. Depending on the resistance a value between 0 and 1023 will be read on the matching pin.

Proximity sensor (IR): In this kit an infrared proximity detector is used. The infrared diode sends infrared light, which is only received by the sensor, when something reflects it. White objects are detected in a distance of up to 6 cm; black objects will only be detected when they are as close as 1 cm. If an object is detected, “low” will be read on the corresponding input pin. The proximity detector can be used as a very basic motion sensor and for contact-free switching.

An expansion of the construction kit could include the following additional components: hall-effect-sensors to detect magnetic fields, touch and pressure sensors, a light barrier construction, vibration sensors, ultrasonic sensors and many more.

The Actuators

For the actuators the same procedure is applied, as has been described for the sensors already. Standard and continuous rotation servos, LEDs and piezo sounders are part of the construction kit. Further components in a future version of the kit could include vibration motors, different types of displays (e.g. dot matrix, seven-segment, LCD) and many more.

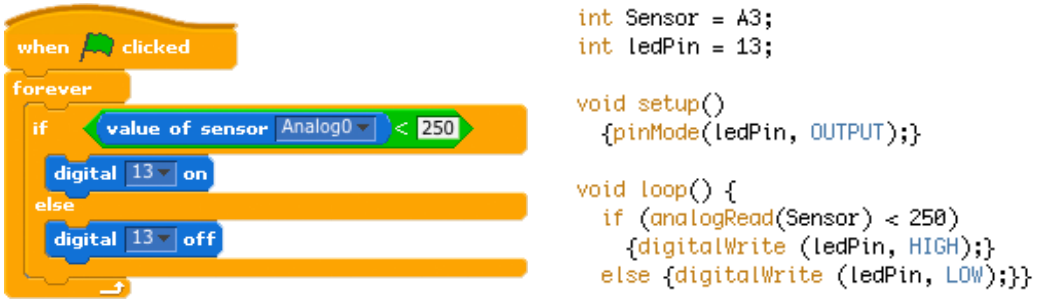

Fig. 2: S4A-program for controlling a lamp in comparison to its code-equivalent.

Activities and Examples

In accordance with the strategies for broadening participation, topics need be broad enough to encourage a variety of different projects. At the same time they should be specific enough to call for ideas and give students the possibility for meaningful discourse. Furthermore, topics should stimulate the students' imagination and even allow the construction of purely artistic projects that do not necessarily represent meaningful devices in reality. With *My Interactive Garden* all this is possible: such a garden can contain many familiar objects of everyday life, as well as futuristic objects that do not yet exist. This ensures that the students' perspective is not limited to devices they know from their environment. To give an impression of the various options, only a few possible projects are listed: lights that glow in different colors depending on the current weather conditions, automatically opening doors, automatic watering systems, alarm systems for house and garden, a balance bridge over the pond that opens for ducks when they come close, magical flowers that interact with people, a swing that starts to move when someone sits on it, solar lanterns, a sun screen that automatically opens when sunlight is detected, lamps that light when movement is detected, a rabbit hutch with an automated feeding system and many more.

In the following, two prototypical interactive objects are described; a simple and a more complex project, which demonstrate the idea of *My Interactive Garden*.

Magical Flower

A relatively simple and easy to realize project is the construction of a magical gleaming flower - a lamp, which is controlled by the intensity of ambient light (fig. 3). The value of a brightness sensor is processed and used to control one or more LEDs. When a measured value falls below a particular threshold value, the LEDs are turned on. The resistance of the sensor is the higher, the darker the environment. This means that in absolute darkness, the resistance is so high that no voltage (and therefore a value of 0) can be read on the corresponding analog input pin. If the ambient brightness is very high, the resistance is very low, and the maximum voltage is detected on the pin, which means that a value of 1023 will be read. A possible extension of the program is to control the LEDs with pulse width modulation (PWM) and thus to adjust the LEDs' brightness according to the ambient brightness. For this purpose output values need to be calculated in correspondence to the input signals. An even more complex task is to additionally change the color of the light in specified time intervals.

This example makes use of several computer science concepts. Loops and decisions are needed to turn the lights on and off. Moreover, students will learn about the representation of information and analog and digital data when using pulse width modulation to achieve differences in brightness. In classroom several analogical objects that require the same programming concepts can be created by different students, according to their interest and intended contribution to the interactive garden: e.g. automatic lanterns, house lights, disco lights and many more.

Fig. 3: Magical flower and bonfire

Bonfire

Depending on ambient light and temperature the bonfire "inflames" itself (fig. 3). For this purpose a light sensor and a temperature sensor are used. When the measured values exceed or fall below particular threshold values, three LEDs (red and yellow, diffused) are turned on with PWM. Brightness as well as the time of illumination are controlled by random values to create the effect of flickering. The bonfire is extinguished when the input signals exceed the specified threshold values or when it is blown out. For this purpose, a sound sensor is calibrated accordingly. To make sure the bonfire does not enlighten itself immediately after extinguishing it, a delay needs to be added to the algorithm. A possible extension of the project is to build a construction for roasting food over the fire. A servomotor can be used to rotate a rod to which the food is attached. In this example many programming concepts are needed: in addition to loops and decisions, variables, comparisons and arithmetic operations are relevant. Random numbers are used and messages exchanged to call subroutines in the program.

Discussion

My Interactive Garden picks up the ideas of Papert and transports them into the 21st century setting. Students can now be empowered to learn principles of computing by constructing meaningful interactive objects. Through feedback from computer science teachers, we know about the interest in applying the idea of creating interactive objects in the classroom, but also about trouble teachers and students have with the raw Arduino system. The construction kit provides them with pre-assembled sensors and actuators as well as an extension for the Arduino board containing connecters for input and output of data via sensors and actuators. The accompanying documentation delivers ideas for projects, which can be exhibited as part of the interactive garden. This way, developments in interactive computing systems are used authentically and form a motivating context for computing education, which in this way can be driven by challenges posed by students themselves.

At the moment only a prototypical version of *My Interactive Garden* exists. However, it is our intention, after a first test run in schools, to prepare the construction kit in a way that it can be provided to schools in larger quantities. Furthermore, the data sheets and construction plans for the kit are published allowing the recreation of every part.

The increasing pervasiveness of interactive objects will lead to a growing demand for educational material, which addresses these ubiquitous media both as tools, as well as subjects of computing education. Elaborating this thought, we are convinced that in the near future, children will not only take home from school a hand made vase made in pottery class but also interactive objects they themselves have created and programmed in computer science class.

References

Banzi, M. (2011). *Getting Started with Arduino* (2nd Edition ed.). Sebastopol, CA: O'Reilly Media.

Guzdial, M. (2010). *Dancing and singing humans, even more than robots*. Retrieved 10.04.2012, from http://computinged.wordpress.com/2010/12/31/

Hartmann, W., Nievergelt, J., & Reichert, R. (2001). *Kara, finite state machines, and the case for programming as part of general education*. Paper presented at the 2001 IEEE Symposia on Human-Centric Computing Languages and Environments

Hoppe, H. U., & Löthe, H. (1984). Problemlösen und Programmieren mit Logo: Ausgew. Beispiele aus Mathematik u. Informatik: Teubner.

Martin, T. (2003). Wearable and Ubiquitous Computing. *Pervasive Computing, IEEE , II* (3), 8-12.

O'Sullivan, D., & Igoe, T. (2004). Physical computing: sensing and controlling the physical world with computers: Course Technology.

Papert, S. (1980). Mindstorms: children, computers, and powerful ideas. New York: Basic Books.

Papert, S. (1996). The connected family: Building the digital general gap. Atlanta, GA: Long St. Pr.

Pardue, J. (2010). *An Arduino Workshop.* Knoxville: Smiley Micros.

Papert, S., & Harel, I. (1991). Situating Constructionism. In S. Papert & I. Harel (Eds.), Constructionism. Norwood, N.J.: Ablex Publishing.

Pattis, R. E. (1981). *Karel the robot: a gentle introduction to the art of programming*: John Wiley.

Resnick, M. (2007). *Sowing the Seeds for a More Creative Society*. Paper presented at the Learning & Leading with Technology, International Society for Technology in Education (ISTE).

Richards, M., & Smith, N. (2010). *Teaching UbiComp with Sense*. Proceedings of the 6th Nordic Conf. on Human-Computer Interaction: Extending Boundaries (S. 765-768). New York: ACM.

Romeike, R. (2008). *Where's my Challenge? The Forgotten Part of Problem Solving in Computer Science Education*. Paper presented at the 3rd ISSEP Intern. Conf. on Informatics in Secondary Schools - Evolution and perspectives, Torun, Polen 2008.

Rusk, N., Resnick, M., Berg, R., & Pezalla-Granlund, M. (2008). *New Pathways into Robotics: Strategies for Broadening Participation.* Journal of Science Educ. and Technology, 17 (1), 59-69.

S4A (2012). Scratch for Arduino. Retreived 10.04.2012 from http://seaside.citilab.eu/scratch/arduino

Weiser, M. (1991). *The computer for the 21st century*, 1991. Scientific American, 256(3), 66-75.

Weiser, M. (1993). Some Computer Science Issues in Ubiquitous Computing. *Communications of the ACM , XXXVI* (7), 75-84.

West, M. (2011). Ubiquitous Computing. SIGUCCS '11 Proceedings of the 39th ACM annual conference on SIGUCCS (S. 175-182). New York: ACM.

Wiesner, B., & Brinda, T. (2007). Erfahrungen bei der Vermittlung algorithmischer Grundstrukturen im Informatikunt. der Realschule mit einem Robotersystem. Did. der Informatik in Theorie und Praxis, 113.

Ziegenbalg, J. (1985). *Programmieren lernen mit Logo*: Hanser München.