Flow of Electrons: An Augmented Workspace for Learning Physical Computing Experientially

Bettina Conradi, Verena Lerch, Martin Hommer, Robert Kowalski, Ioanna Vletsou, Heinrich Hussmann
Ludwig-Maximilians-Universität München
Amalienstr.17
80333 Munich
{bettina.conradi, hussmann}@ifi.lmu.de, {lerch, hommer, kowalski, vletsou}@cip.ifi.lmu.de

ABSTRACT
Physical computing empowers people to design and customize electronic hardware tailored to their individual needs. This often involves ”tinkering” with components connections, but due to the intangible nature of electricity, this can be difficult, especially for novices. We use a multi-stage design process to design, build and evaluate a physical prototyping workspace for novices to learn about real physical computing hardware. The workspace consists of a horizontal surface that tracks physical components like sensors, actuators, and microcontroller boards and augments them with additional digital information in situ. By digitally exploring various means of connecting components, users can experientially learn how to build a functioning circuit and then transition directly to building it physically. In a user study, we found that this system motivates learners by encouraging them and building a sense of competence, while also providing a stimulating experience.

ACM Classification: H5.1 [**Information interfaces and presentation**]: Multimedia Information Systems. - Artificial, augmented, and virtual realities

General terms: Design, Experimentation, Human Factors

Keywords: Physical user interfaces, prototyping, learning, interactive workspaces

MOTIVATION
Physical user interfaces allow people to control digital information using physical sensors and actuators. Today only enthusiastic end users with substantial background knowledge can build these interfaces. They customize textiles with electronics [2], create physical extensions for social software [9] [21], or even build devices that reflect their need for sustainability, by for example appropriating electronic waste [12]. In order to build such interfaces, a basic understanding of electrical principles is still needed (for example, knowing the meaning of different pins, or when and how to use resistors). Books and web-based tutorials can provide this information in principle; however, since the information is not in situ and is physically distant from the learner’s focus, mapping it to the actual task is difficult. In addition the intangible nature of electricity makes it hard to experience what is actually happening inside an electronic circuit. We see an opportunity for tangible interfaces on tabletops to make electronics more graspable, therefore offering a new experience that brings physical components closer to their behavior.

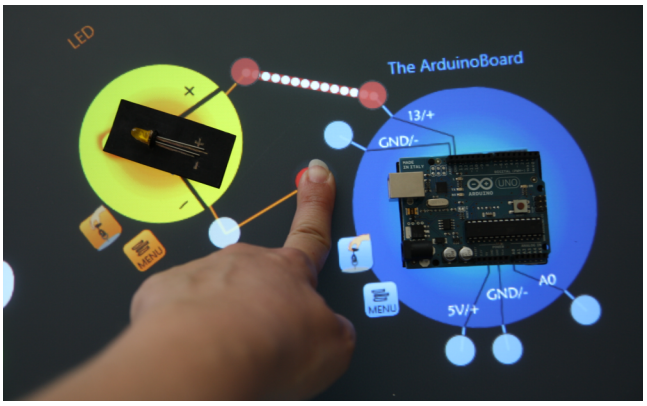

Figure 1: Digitally experimenting with how to correctly wire physical components. Electronics can be experienced with in situ visualizations.

We therefore argue for a better integration of physical hardware components and digital information in educational toolkits. In order to support novices in understanding electrical principles, augmented workspaces can provide a safe test environment for digital experimentation (see Figure 1). They can provide digital information in situ, combined with a tangible experience of electronics, and they can guide learners in progressing step-by-step from digitally augmented hardware components to fully wired prototypes.

After categorizing toolkits for building physical interfaces and understanding their underlying electrical principles, we detail how augmented workspaces can serve as a useful medium for experiential learning of physical computing. Initial requirements for an educational toolkit are then refined through observations, a diary study, and evaluations of paper prototypes. Transferring our refined requirements to an implementation and then evaluating it revealed that users had a stimulating, inspiring experience and felt encouraged to prototype on their own in the future.

ITS 2011, November 13-16, Kobe, Japan.

RELATED WORK

Related work draws from two domains: First, the domain of physical computing toolkits that simplify the building process of prototypes or the understanding of electricity. The second domain is that of interactive workspaces that examines how ordinary workspaces can be enhanced and augmented with digital information.

Physical Computing Toolkits: Build and Understand

Physical user interfaces consist of a hardware and software part. Development of both can be simplified for novices with adequate toolkits. Software development can be made easier for non-programmers with visual programming like Scratch [23] or Splish [10] by letting users visually arrange program blocks, loops, conditions, or other statements. BrickLayer [4] combines this approach with a learning outcome: Users can program an Arduino application visually, while the resulting code is simultaneously displayed for modification. We want to extend this approach to physically constructing a project while at the same time understanding the underlying electrical principles.

In order to classify related work in the field of hardware prototyping for novices, we introduce a categorization (see Figure 2). It consists of four categories that are characterized by two aspects: 1) The **purpose** of a particular toolkit (vertical axis) determines whether it supports the actual *building* process of physical interfaces and has a project-driven outcome, or whether it leads to *understanding* electricity and how to build an electric circuit correctly. 2) The **type** (horizontal axis) determines whether it is a *software* or *hardware* approach.

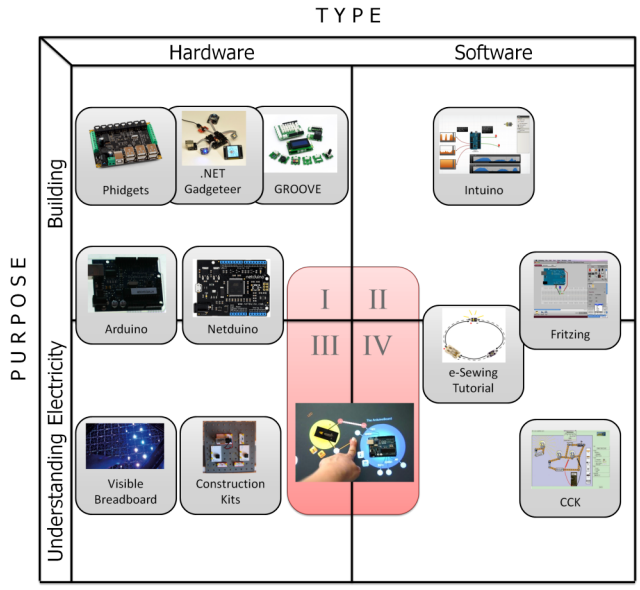

Figure 2: Toolkits for hardware prototyping categorized according to their *purpose* (building vs. understanding electricity) and *type* (hardware vs. software), (images from website or papers of projects).

Category I: Hardware Toolkits for Building Physical User Interfaces Inspired by educational construction sets for children like the MIT Cricket [18] or Lego Mindstorms[1], toolkits like Phidgets [7], .NET Gadgeteer[2], or GROVE[3] add a layer of abstraction over complex electricity and integrate "off the shelf" components (required resistors/ transistors/ capacitors are already included on the component). They tackle the problem of limited complexity of programs or available libraries [3] with a more flexible pin configuration and programming abilities. However, the components available are limited. There is no support for acquiring the knowledge needed for the inclusion of more complex or uncommon components (e.g. how to read datasheets or differentiate a component's pins). More advanced microcontroller toolkits for novices, like Arduino[4] or Netduino[5] expand the design space, but require to gain a basic understanding of electricity (and therefore extend into Cat. III).

Category II: Software Toolkits for Building Physical User Interfaces Software applications can support novices in building a physical interface. In contrast to professional software for constructing hardware layouts like EAGLE[6] and SPICE[7], Fritzing [13] is a simple tool for digitally documenting and sharing circuits in different views (breadboard, schematic, and PCB). Therefore, it serves as a good starting point for novices to rebuild a shared project. It also introduces the schematic layout notation, which enables users to understand notations of electric components and read simple datasheets, extending toward Cat. IV. Intuino [28] helps users test a first prototype with software. It visualizes the current state of components connected to an Arduino board, and is a valuable tool for debugging and quick experimentation.

Category III: Hardware Toolkits for Understanding Electricity Educational construction kits teach children electrical basics (for example, how to regulate a battery and LED with a switch). Basic components are included in the kit, and booklets provide further instructions. However, what is happening electronically remains invisible, which makes the experimentation process difficult. The Visible Breadboard [19] tries to tackle this problem by visualizing current flow inside a breadboard. Although this helps beginners understand a breadboard's functionality, they use a custom-made board with integrated lighting, and so it is not applicable to a wide range of different hardware components. These toolkits provide a good mapping of learned concepts to actual hardware components. However, visualizing immaterial information with hardware alone has until now been realized only with expensive or impractical extensions of components.

---

[1] http://mindstorms.lego.com

[2] http://research.microsoft.com/en-us/projects/gadgeteer/

[3] http://garden.seeedstudio.com/index.php?title=GROVE_System

[4] http://www.arduino.cc/

[5] http://netduino.com/

[6] http://www.cadsoft.de/

[7] http://bwrc.eecs.berkeley.edu/classes/icbook/spice/

Category IV: Software Toolkits for Understanding Electricity
Software can document or visualize immaterial information like electricity. For crafting electronic textiles with LilyPad Arduino, an online tutorial provides detailed explanations on what is happening inside an electric circuit [17] by providing a diversity of representations (illustrations, photographs, and textual descriptions). The Circuit Construction Kit (CCK) [31] is an online application that allows users to experiment with electronic circuits in order to discover how current flows can be modified. The authors found that students learn more through experimenting with simulations rather than through ordinary classroom lessons. Software applications can make electronics experiential; however, this experience needs to have a direct connection to real-life objects, which is also argued by the authors - a criterion that is more easily achieved when interacting with real components instead of their digital counterparts.

As we have seen so far, hardware and software toolkits simplify the construction of prototypes by lowering the entry barrier for novices. However, beginners often lack the ability and background knowledge required for building a more complex prototype and not simply copying existing circuit designs. The tangibility of educational construction sets supports their understanding of physical computing, but experiencing electricity is only possible when specialized, modified hardware is provided. Software applications that provide simulations are a vivid way to convey electronic basics, but they for the most part lack a direct and meaningful connection between virtual information and the physical objects they are referring to, and cannot provide practical experience that is only gained through tinkering with hardware components. We therefore want to bridge the gap between physical and digital in educational toolkits for understanding physical computing.

Augmented Workspaces

This gap between physical component and digital information can be tackled with augmented workspaces that track physical objects and provide information in situ. Research in this domain shows that the possibilities for making workspaces more interactive are manifold: screen and interaction space can be extended to the horizontal office desk around the keyboard [1]. Interactive tabletops can sometimes be used as a replacement for regular vertical screens [32], and even new kinds of workspaces that combine horizontal and vertical interactive surfaces have emerged (e.g. Curve [33] or BendDesk [29]).

Applications can make use of these available technologies to augment physical objects with digital information in situ. Wellner's DigitalDesk showed how to efficiently combine physical paper (e.g. bills) with superimposed digital information, and how to transition from physical to digital [30]. Visualizing and explaining invisible information such as light reflection on physical objects is shown in the luminous room [27]. Urp provides digital wind simulation data for physical model houses in situ [26]. With Sensetable, learners can explore chemical reactions by using pucks as physical handles on atoms [20]. Domains like architecture or chemistry require using proxy objects as tangible interface. In contrast, a physical interface consists of real physical components (microcontroller, sensors, actuators) that can be used as physical handles. We have proposed initial requirements for an educational toolkit [5], which are detailed, implemented and evaluated in this paper.

Because there is a gap in educational toolkits for physical computing with respect to the combination of physical components with digital information in situ, our approach combines the benefits of augmented workspaces with teaching electronic basics to novices.

INITIAL CONCEPT

In order to develop an educational toolkit for physical computing that better integrates digital information with physical hardware components, we utilize the experiential learning cycle - a learning theory that supports learning through experimentation. Based on related work and this theory, we derived initial requirements for our system.

Experiential Learning

Kolb argued that learning from experience is an adequate way to acquire knowledge: "*Learning is the process whereby knowledge is created through the transformation of experience*" [14][15]. His experiential learning cycle (see Figure 3) details how learners refine their knowledge through experimentation. After having a concrete experience, one can reflect on observations, conceptualize abstractly how it might work, and test these newly formed hypotheses through active experimentation. If the resulting experience and reflection do not fit the conceptualizations, they are adapted and tested with new experimentations.

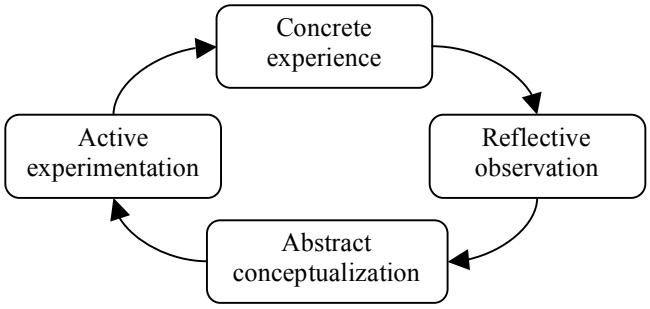

Figure 3: Kolb's experiential learning cycle [14] serves as framework for our educational workspace for learning physical computing

Kolb's learning cycle has earned much criticism due to its over- simplification of the learning process: for example, it is argued that learning phases cannot be separated that distinctly (see [11] for a summary of critics). Nevertheless, it can guide us in developing an educational toolkit for physical computing that integrates these phases and allows for a fluent transition between them. Augmented workspaces can support experimentation by providing a safe digital test environment, they can provide a tangible experience of electronics, and they support reflection and conceptualization with digital information in situ.

Initial Requirements

Based on related work and the experiential learning cycle, we derived the following a priori requirements:

- **Provide a safe test environment:** Experimenting with hardware can be intimidating to novices: Selecting and connecting components may be beyond their abilities. If components are connected the wrong way, they may be destroyed electrically, leading to the learner's frustration. Augmented workspaces can provide a safe test environment: Learners can first experiment digitally on how to wire components correctly without worrying about faults (Figure 4, right).
- **Provide a tangible experience of electronics:** As electricity is immaterial and intangible, it is difficult for learners to experience what is happening inside an electronic circuit. In order to make electronics experiential, workspaces can augment hardware components with digital information. Components can be connected with digital wires that simulate flowing electrons, and learners can gain an impression about how electricity functions (see Figure 4 right).
- **Progress from digital to physical**: Concepts learned from digital experimentation should be transferable to hardware components and their assembly. Learners can be guided in a step-by-step refinement from single digital augmented prototypes to a working physical prototype.
- **Provide theoretical input:** Learners should be able to conceptualize their observations of experiments. Therefore, information about electrical laws has to be provided to prove their conceptualizations. An augmented workspace can offer this digital information on demand, depending upon the component placed on it. Therefore, our workspace should provide background information during the learning process, and a book of facts that can be consulted in order to read more about a topic in detail.
- **Provide immediate feedback:** In order to reflect on their observations, learners can be supported with immediate feedback on their experiments that tells them if they have done something right or wrong. A rapid feedback loop speeds up learning and encourages the active exploration of different approaches.
- **Provide information in situ:** In order to mentally map digital information to actual hardware components, augmented workspaces can provide in situ illustrations of the real hardware setup with an orientation corresponding to the physical component, which spares learners from mental mapping tasks (e.g. by explaining pin functionality right next to the component, see Figure 4, left).
- **Provide a framework for incremental, experiential learning:** Concentrating their experiments on a single aspect and leaving out distracting or misleading peripheral aspects can be difficult for learners. Modular tutorials should guide users in step-by-step experimentation that always keeps the focus of an experiment on a single aspect.

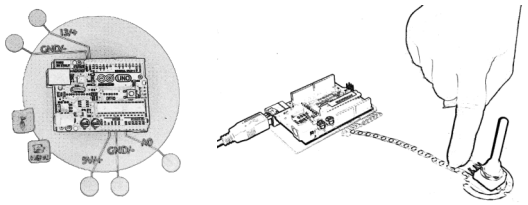

Figure 4: (Left) Physical components are augmented with digital information explaining pin functionality, which provides information in situ. (Right) Augmented workspaces can depict intangible electronics and provide a safe test environment for digital experimentation.

DESIGN PROCESS

To further substantiate our initial requirements, we conducted various pre-tests: (1) observing novices while they build a physical prototype to understand their approaches to information-gathering and problems with web-based tutorials, (2) diary study in a physical prototyping workshop to gain insights on how students experiment with hardware, (3) guided discussions with paper prototypes on the validity of our initial requirements, especially on how to best support learners in experiential learning.

Observations

To understand strategies and difficulties of novices when they are building a simple physical prototype for the first time, we conducted an informal observation. Two people were assigned the task of connecting an LED and potentiometer to an Arduino. They had access to a computer with an open browser to retrieve information for their current task. Both found a comprehensive tutorial and tried to follow it thoroughly (browser history had been cleared before each new participant). Each step was illustrated with detailed pictures, which provided the most and best help to the participants (as also stated in [16]. Although both were able to complete the tutorial, we recognized many *attention shifts* between computer display and hardware components and *mental mapping problems* between the illustrations and hardware. One held the Arduino board next to the screen, orienting it next to the illustration in order to identify the pin equivalents. This informal observation assured us that providing digital information on hardware components in place and in time is helpful for novices in obtaining a direct mapping.

Diary study

To gain insights on how students experiment with hardware over a longer period of time while working on a project, we let students log their progress, tasks, and difficulties in a diary study, adopting the approach developed in [6] during a one-week physical prototyping workshop at our lab. We were interested in how students progress from idea to prototype, and what kind of development phases they traverse. As physical computing consists not only of hardware development but also of software development to a large degree (programming the microcontroller), we examined how these two processes are interleaved when building a proto-

type, and whether they can be separated when structuring tutorials. Our hypothesis was that hardware and software development are done in separate cycles, each one building on the results of prior development, forming an iterative and incremental process.

Prior to this workshop, the 12 students (students of computer science, who had solid programming skills and little or no previous experience with hardware) were given a two-day practical introduction to the basics of physical computing. In the observation week, they had to brainstorm a project idea, develop and test it, and present it on the last day. The course consisted of 12 students who formed groups in pairs. Every student received a diary that consisted of tables with 11 different activities related to concept development, building hardware, and programming software (see Table 1). These activities were derived from our own work practices as well as the observations mentioned in the previous chapter. Students were introduced to the meaning of each of the 11 phases and how and when they should record something in their diary. Every three hours they had to checkmark which of the 11 activities they had completed during that time slot. Nine students logged their tasks to completion and delivered their diary. From these diaries, we gathered 236 records that represented one or more tasks completed in a three-hour interval (total of 451 activities logged). We evaluated this data with a two-tailed bivariate analysis of Pearson's correlation coefficient (see Table 1).

At first sight, the results confirm our hypothesis that concept, hardware, and software development are done in separate cycles. However, a closer look, reveals two hybrid or transitional phases: Concept development correlates most positively with other concept development phases (r≥.363), except the phase "pick and shop hw" which also correlates positively with concept development, especially with "design the exterior" (r=.157). This shows that selecting hardware also belongs to the creative process and influences the concept design of the prototype. Software and hardware development cannot be strictly separated. Although the highest correlation (r≥.5) is found between activities in the same domain, correlations between most hardware and software development phases are above zero. In particular, "testing hardware with software" is correlated to both hardware and software development: As soon as new hardware is integrated, the accuracy of its wiring and capabilities are often tested with software (r=.33). Conversely, if a new software functionality is integrated, its effects on hardware are immediately tested (r=.303).

Since students tested their hardware prototypes primarily with software in order to experience its full capabilities, our interactive workspace needs to enable users to easily "*test hardware with software*". Learners should be enabled to scan and modify software parameters for single sensors and actuators, without needing to write test programs.

Paper prototype

Before implementing our system, we developed ideas with sketches and transferred these into a paper prototype (see Figure 5, right) in order to explore guidance mechanisms and participants' progress and interest in tutorials. "Breadboard tutorial" lets learners digitally explore the connection mechanism of a breadboard, a frequently used tool in physical computing. The "Arduino Blink tutorial" guides learners in connecting an LED to Arduino and changing the blinking interval with software. "Electrical basics tutorial"

Table 1: Correlation between concept/hardware and software development in a sketching with hardware workshop. While concept development is mainly done in a separate cycle, hardware and software development (especially testing) are intertwined.

		Concept			Hardware					Software		
		create or refine concept idea	draw sketches	design of the exterior	pick and shop hw	collect infos: specifications, examples	put hw together	test hw with multimeter, etc.	test hw with sw	pick sw: libraries, APIs etc.	collect infos: example code	program sw functionality
Concept	create or refine concept idea	1	.546	.363	.017	-.027	-.222	-.178	-.217	-.068	-.104	-.146
	draw sketches	.546	1	.446	.058	-.079	-.175	-.170	-.179	-.045	-.079	-.079
	design of the exterior	.363	.446	1	.157	.109	-.117	-.101	-.172	-.058	-.094	.005
Hardware	pick and shop hw	.017	.058	.157	1	.283	.049	.194	-.020	.025	.012	-.047
	collect infos: specifications, examples	-.027	-.079	.109	.283	1	.278	.239	.083	.067	.045	.013
	put hw together	-.222	-.175	-.117	.049	.278	1	.501	.330	-.001	.012	.058
	test hw with multimeter, etc.	-.178	-.170	-.101	.194	.239	.501	1	.366	.146	.126	.109
	test hw with sw	-.217	-.179	-.172	-.020	.083	.330	.366	1	.132	.249	.303
Software	pick sw: libraries, APIs etc.	-.068	-.045	-.058	.025	.067	-.001	.146	.132	1	.523	.440
	collect infos: example code	-.104	-.079	-.094	.012	.045	.012	.126	.249	.523	1	.634
	program sw functionality	-.146	-.079	.005	-.047	.013	.058	.109	.303	.440	.634	1

introduces learners to resistors and serial/parallel circuits (the last tutorial was not available as a full tutorial, only as preview). Five participants (three female, age 23-28) took part in a study based on the paper prototype. Three of them were students of computer science, one was student of business administration, and one was an analytical chemist. They all described their experience with electronics as none or very minor. An introductory video of the Microsoft Surface[8] showed participants the capabilities of augmented workspaces and how to interact with them (multi-touch and physical objects). Participants were standing at a (actually interactive) table; however, they did not interact with software, but only with the paper prototype that fully covered the screen (see Figure 5, left). Hardware components were placed at the side of the table within reach of participants. When they placed a component on the table, the instructor attached a paper augmentation representing virtual information. Whenever participants were interacting with the component, the instructor moved or changed the "digital" augmentation accordingly. Subsequently, they were assigned the task to find out what a potentiometer is and how it works (place it on the table and look at a book of facts), explore a breadboard digitally, and finally, open the "Arduino Blink tutorial". They explored and built the connection between LED and Arduino digitally, and afterwards rebuilt it physically. We recorded all sessions on video for later analysis and told participants to think aloud about their current task and problems with the interface.

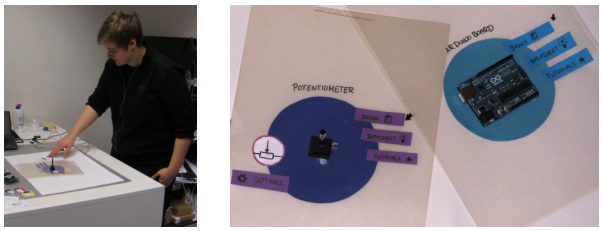

Figure 5: (Left) Paper prototype study setup. (Right) If a physical component is placed on the paper surface, the instructor attaches "digital information" in the form of paper to it.

The results were very promising: All participants were able to complete the tutorials without assistance. Most of them did not even need an image of the actual setup after experimenting with the digital version. Participants were excited about the blinking LED and felt quite competent, leading to the following statement: *"This tutorial is good [points to Arduino Blink tutorial][…], because I know that I will be able to blink an LED."* However, we also learned that tutorials about electrical basics do not have enough concrete outcomes to be attractive to learners: *"With these tutorials [points to breadboard, serial/parallel circuit and resistor tutorial], I thought: Well, this is something dry/uninspiring, where I only learn some fundamentals, but do not know why [I need it]. I imagine that tutorials should let me do something cool while learning something at the same time."* Tutorials should therefore introduce the learner to a new concept, and at the same time lead to a working physical prototype when completed.

Participants also mentioned difficulties with interaction: The level of guidance during the progression from digital to physical was not sufficient. Participants did not know the goal or understood when they had successfully completed a task. As a result, we redesigned the guiding process toward a sub-goal-driven system. This featured smaller and more specific sub-goals for each step, while leaving enough room for free experimentation (e.g. they can digitally experiment how to correctly wire components; as soon as they find the right connection, they are led to the next step).

Summarizing the findings from our design process, we were assured that our approach is feasible and tackles problems with current web-based tutorials (mapping problems found during observations). We refined our requirements with a one-week diary study and an evaluation of paper prototypes. With the one-week diary study, we saw that we also need to integrate software development insofar as that *hardware behavior needs to be modifiable through software coding* in order to fully experience its capabilities. Paper prototyping showed that *tutorials should be guided in a clear way and have a concrete outcome*.

IMPLEMENTATION

This chapter summarizes the technical implementation, followed by a description of one introductory tutorial, the "Arduino Blink" tutorial. We thereby detail how we implemented our refined requirements.

Technology

We used the Microsoft Surface v1.0 (see Figure 6, left) with the Surface SDK 1.0 SP1. The system was developed in C# utilizing the Windows Presentation Foundation. Byte tags (dimension: 1.9 x 1.9 cm) are attached to the physical components to detect type and orientation of hardware (see Figure 6, right). We used an Arduino Uno board that has 14 digital I/O pins and 6 analog input pins and can be connected to a computer via USB. Arduino is preconfigured with Firmata – a library for establishing a communication protocol between a microcontroller and host software [25].

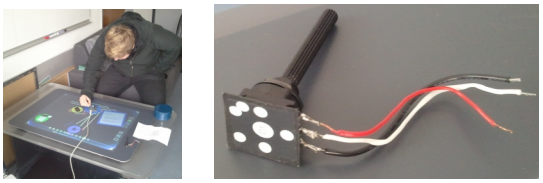

Figure 6: (Left) System runs on Microsoft Surface v1.0. (Right) Byte tags are attached to components to identify type and orientation.

A tour through the system

We structure the presentation according to the requirements formulated above. Users start their learning experience by placing components on the table (see Figure 7, left). The resulting menu items "Basics", "Datasheet" and "Tutorials" serve as entrance into theoretical and experiential learning of physical computing concepts.

[8] http://www.youtube.com/watch?v=6VfpVYYQzHs

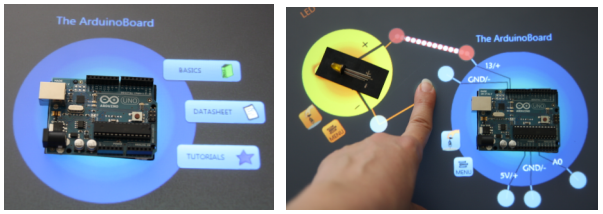

Figure 7: Digital experimentation phase: (Left) Menu for hardware: (1) Basics book to retrieve theoretical input, (2) Datasheet to retrieve a schematic diagram in situ, (3) Hardware pins are explained and can be wired digitally. Animated electrons simulate a closed circuit.

Provide theoretical input: The first option "Basics" presents a book of facts that can be consulted throughout the entire learning experience. It provides information about the component itself as well as common uses.

Provide information in situ: The second choice "Datasheet" is targeted to advanced users and provides more in-depth information via a datasheet, which utilizes a schematic diagram to provide information on how this component can be connected to other electric parts in the right orientation and scale. A simplified pin explanation can be found in tutorials, where hardware pins are augmented with digital information (e.g. + or -) (see Figure 7, right).

Provide a framework for incremental, experiential learning: To build a device under digital guidance and experiment with hardware, learners can choose the third option "Tutorials". We implemented the "Arduino Blink" tutorial and added "Analog Input", which introduces a potentiometer that controls the blinking interval of a LED (meets the refined design guideline to provide tutorials with concrete outcomes). Tutorials serve as a framework for progressing step-by-step from digital experimentation to a wired physical prototype that can be manipulated with software.

Provide a safe test environment: The "Arduino Blink" tutorial lets learners experiment with digital counterparts of LED and Arduino pins. They serve as a starting point for digital experiments on how to correctly wire an electric circuit. This digital information layer provides a "safety net" for learners, giving them more confidence in their experiments, by alleviating the fear of breaking hardware in the case of incorrect wiring.

Provide immediate feedback: In case a digital wire was connected to the wrong pin, Bread the Board (a small cartoon figure that provides guidance throughout the tutorial) immediately alerts the learner and, after more mistakes are corrected, helps them to achieve the task.

Provide tangible experience of electronics: Learners can try to connect components digitally. They can immediately experience and observe the results with animations of digital wires confirming a correct wiring. As soon as the virtual circuit is closed, animated "electrons" start to flow (see Figure 7, right).

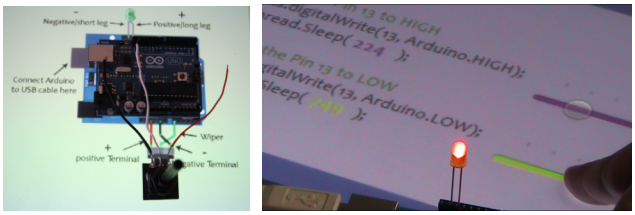

Figure 8: Construction phase: (Left) Illustrations provide in situ information on how to wire hardware physically. (Right) Hardware is tested with code fragments that can be manipulated with sliders.

Progress from digital to physical: After a digital lesson on how to connect components to the Arduino, learners rebuilt the electric circuit physically to transfer their knowledge from theory to practice with actual components. Learners are provided with images of the setup, enabling them to easily map where each wire is placed (see Figure 8, left). They construct the setup directly on the tabletop or remove single components from the surface, leaving a digital copy.

Test hardware with software: To further experience the hardware's capabilities, learners can manipulate and test the programmed behavior of a component's software. Code fragments and variables of the (future) program code are displayed and can be influenced via simple UI elements, for example sliders, that affect the LED blinking interval (see Figure 8, right). Consequently, this stage not only supports the learner in validating and evaluating the various parts of the setup, it also provides first insights into an Arduino program and its interactions with the hardware.

EVALUATION

A hypothesis and goal of the system was to raise students' enthusiasm for hardware prototyping, relieve their aversion to the complex topic of electronics and to increase their knowledge. We therefore evaluated motivational aspects as well as learning outcomes of our system.

Study setup

For the evaluation, 13 students of computer science were invited (seven females, age 20-28). To get an impression of physical computing, the participants watched a video[9] of a prototype created during the one-week physical prototyping workshop mentioned above. The second video was the same as in the paper prototype evaluation, and showed the interaction possibilities of Microsoft Surface. After explaining the purpose of the system, we handed out a questionnaire to determine previous knowledge and the participants' interest in the topic, before asking them to use the system. They were not given a particular task, and could freely explore components and tutorials. However, if participants stopped the system before having fulfilled both tutorials, they were asked to continue. A final questionnaire completed the study session. Participants were instructed to use the think-aloud method and we videotaped each session for later analysis.

[9] http://www.youtube.com/watch?v=UAlCT870grE

Study Goals and Questionnaires

According to Prenzel et al. [22], learners can be motivated in self-guided learning by the following factors: *autonomy, competence, quality, social inclusion, relevance, and interest of teacher* (we do not evaluate the last factor due to the fact there is no teacher in that context). To evaluate these factors, we used the following questionnaires:

We used the AttrakDiff [8] standardized questionnaire that evaluates the pragmatic (usable and useful) as well as hedonic (identifying and stimulating) qualities of a system. Two opposite adjectives describing the system, like "complex" and "simple" are rated with seven-point Likert scales. For analysis, all question pairs were switched so that the negative adjective always has the lowest score.

Another part of the questionnaire is aligned to Sheldon's need-satisfying items [24]. They provide answers about fulfillment of psychological needs such as autonomy, competence, and self-esteem, thus directly relating to Prenzel's factors. We customized questions to focus on our application, and let students rate statements on a seven-point Likert scale ranging from "strongly disagree" to "strongly agree".

Finally, we included questions regarding the participant's interest and attitude toward physical computing before and after the study. This questionnaire used a seven-point Likert scale that was analyzed with a dependent t-test to reveal significant differences in attitude changes.

Providing a motivational learning environment

This chapter details how our application was perceived by participants, according to Prenzel's factors for self-guided motivation:

Autonomy In sum, autonomy was perceived as medium (M=3.86, SD=1.88) (see Figure 10). Participants felt strongly guided by our application (M=5.25, SD=1.71). On average, they were neutral toward experimenting with components more freely (M=4.08, SD=1.93). As both statements can have a negative effect on autonomy, we use the inverse scale of this statement. However, most of the time they felt like they could act according to their own interests (M= 4.92, SD=1.44).

Competence In sum, participants felt quite competent after finishing the tutorials (M=5.19, SD=1.47) (see Figure 10). In particular, they felt like they were able to master demanding tasks (M=5.67, SD=0.89) and felt challenged in an enjoyable way (M=5.83, SD=1.53). Sheldon proposes self-esteem as another need-satisfying item that we will subsume here (see Figure 10), as it contributes to feelings of competence. Participants were very satisfied with their skills (M=5.75, SD=0.87). Their confidence in accomplishing a similar project to those seen in the hardware hacking workshop increased after accomplishing tutorials (M_{before}=4.32, M_{after}=4.92 p=.055). Digital experimentation "*simply removed the fear of doing something wrong. You see that it is correct and electrons are flowing, and I simply rebuild it: it is that easy*".

Quality To assess the perceived quality of our application, we looked at pragmatic quality and attractiveness from the AttrakDiff questionnaire (see Figure 9). Overall pragmatic quality was rated very well (M=5.38, SD=1.32). Participants perceived our application as more easy than complex (M=5.77, SD=1.09), more practical than unpractical (M=6.31, SD=0.85), and more manageable than unmanageable (M=5.77, SD=1.36). Overall attractiveness was rated even higher (M=6.20, SD=0.81): It was perceived as very drawing (M=6.00, SD=0.91) in contrast to repellant and as very encouraging (M=6.31, SD=0.75). One participant stated "*This is damn cool! It works fantastic, I am delighted!*" and another said "*It's so beautiful and welcoming.*"

Social Inclusion As we did not design a multi-user environment but rather one that supports individual learning, it is not surprising that the pair "separates me vs. brings me closer to people" was evaluated rather neutrally (M=4.15, SD=0.90). However, it was perceived as more connecting than isolating (M=5.92, SD=1.04) and more inclusive than exclusive (M=5.85, SD=0.99), and therefore it at least does not negatively affect motivation. These results were drawn from AttrakDiff and its section on the hedonic quality "identity" (see Figure 9), that in addition to aspects of social inclusion also covers aspects like value and worthiness for an individual.

Relevance In order to evaluate how relevant the subject appears to participants, we asked them questions about their interest in this topic before and after the study. Two of them had already tried prototyping with Arduino, and another two had already heard about its functionalities. Participants first showed neutral to slight interest in building something electronically on their own soon, but were more assured after the tutorials (M_{before}=4.77 M_{after}=5.62 p=.059).

Additional: Stimulation In addition to these factors, our questionnaires also included questions regarding (pleasure-) stimulation (see Figure 9 and Figure 10). In particular, the participants perceived the application as more creative than unimaginative (M=6.46, SD=0.78) and as more innovative than conservative (M=6.31, SD=0.75). Questions regarding "pleasure-stimulation" (as proposed by Sheldon) were also answered very positively (M=5.89, SD=1.30): Participants in particular felt that they had tried new activities and gained new experiences (M=6.17, SD=0.94). A quote from the study also reflects the stimulating effect of our system: "*You can learn things in a playful way*". Participants' assessment of working with hardware as being fun before and after the study also significantly increased (M_{before}=4.85 M_{after} =6.77 p=.00).

Summarizing our evaluation of motivational factors according to Prenzel [22], we found that our system provides an attractive, stimulating environment that gives users a feeling of competence. However, participants' need for autonomy was not addressed well enough to gain high ratings. The motivational aspect of social inclusion was also not supported by our system, but neither was this factor described as missing by participants.

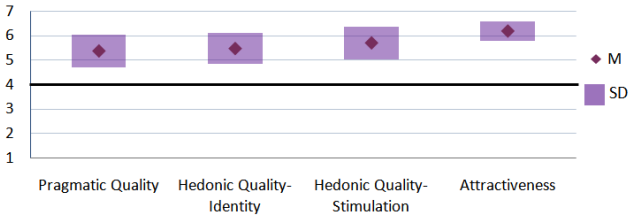

Figure 9: Subsumption of AttrakDiff questionnaire on pragmatic and hedonic quality.

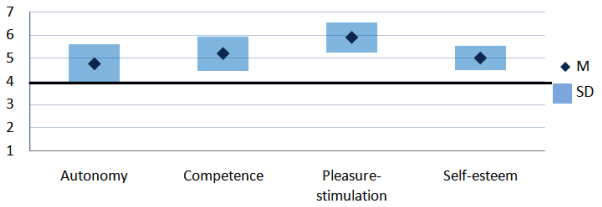

Figure 10: Subsumption of questionnaire regarding need-satisfying items according to Sheldon.

Knowledge test

Because we are providing an educational toolkit for understanding physical computing, we interviewed and tested participants about their learning outcome. When requested to differentiate the positive and negative leg of an LED, 11 out of 13 gave a correct answer. When asked how a potentiometer has to be connected to the Arduino, five participants out of 13 were not able to tell the right connection. We do not see this result as overly disturbing, since remembering pin layouts in detail is not an indicator that one has learned something. However, only four were able to explain the functionalities of Arduino and its pins appropriately. Analyzing video material revealed that the students who had a high learning effect were interested in increasing their knowledge, and most of them consulted the "basics" book and read Bread the Board's instructions while the others put minimum effort into achieving the tasks of the tutorials and did not read the additional information offered. *"It was puzzling, I only replicated what I saw on the screen, but I didn't get what was behind all this"*, a participant stated in explanation. Overall, low scores of the knowledge test could be traced back to the aforementioned lack of autonomous experience. Participants were guided too much by our system, and could also follow instructions instead of experimenting freely. Theoretical input was seen as an extra feature that can be consulted if desired, but should in fact be more tightly coupled to experimentation, e.g. with videos explaining electrical laws, as proposed by one participant. However, the participants' own assessment of their learning outcome was rated more positively (M=5.54, SD=1.39), especially due to the combination of digital AND physical components (M=6.08, SD=1.04). "*I think I have learned a lot*", a motivated participant stated. A more in-depth evaluation that compares our application to ordinary tutorials and investigates differences in interaction between digital and physical is a part of our future work.

CONCLUSION

As hardware appropriation and physical computing has gained interest among novices, they need to be provided with workspaces that help them in their early learning stages. Current educational toolkits only reside in the physical or digital realm, which either does not let learners experience electronics adequately or leaves a gap between digital information and actual hardware components. We therefore argue for a better integration of digital and physical in educational toolkits for learning physical computing.

In order to inform the design of our application, we observed novices as they tried to build their first prototypes. We focused especially on what phases they traverse and how hardware and software development are interleaved. We found that experimenting with hardware relies heavily on software to test the component's full capabilities. Testing hardware with software therefore needs to be integrated in educational toolkits. Evaluations of our implementation showed that our system provides a motivational environment for experiential learning, offering information in situ and a tangible experience of electronics.

FUTURE WORK

During both development and evaluation, we were collecting ideas and suggestions for further improvements and additional features. An obvious extension is to also provide tutorials for digital input (like buttons and switches) as well as analog output (like RGB LEDs and motors).

Another direction is the exploration of different hardware setups that overcome the need for a specialized, expensive device like the Microsoft Surface. Smaller tablet devices like the iPad can be better integrated into a hardware prototyping environment. With technologies like PixelSense, ordinary displays can become sensors themselves that are able to track objects. Another idea is to alter an ordinary workspace from above: a high-resolution camera tracks components and wires from above and a projector displays information directly on the components. For example, a breadboard could be altered with digital information that highlights current flow in situ. This setup also enables the user to check for correct wiring of physical components, a feature our current system lacks because we track components only from beneath.

We presented our current prototype to more experienced hardware prototypers, and they were also attracted to it, but they thought of it more as a workspace for advanced users. Our requirements can also be transferred to a more advanced workspace: Datasheets can be provided in situ, schematic diagrams can be specified with actual components, and the resulting circuit can be cauterized and printed using rapid manufacturing.

ACKNOWLEDGEMENTS

We wish to acknowledge the funding support for this project from Deutsche Forschungsgemeinschaft (DFG) under the PREMIUM project (Professional Engineering Methods for Interactive Ubiquitous Machinery) (BU-1402/2).

REFERENCES

1. Bi, X., Grossman, T., Matejka, J., and Fitzmaurice, G. Magic desk. *CHI '11*, ACM Press (2011), 2511-2520.
2. Buechley, L. and Hill, B.M. LilyPad in the Wild: how hardware's long tail is supporting new engineering and design. *DIS '10*, ACM Press (2010), 199-207.
3. Blikstein, P., Sipitakiat, A. QWERTY and the art of designing microcontrollers for children. *IDC '11*, (2011), 234-237.
4. Cheung, J.C.Y., Ngai, G., Chan, S.C.F., and Lau, W.W.Y. Filling the gap in programming instruction: a text-enhanced graphical programming environment for junior high students. *ACM SIGCSE Bulletin 41*, 1 (2009), 276-280.
5. Conradi, B., Hommer, M., and Kowalski, R. From Digital to Physical: Learning Physical Computing on Interactive Surfaces. *ITS EA '10*, (2010), 249-250.
6. Czerwinski, M., Horvitz, E., and Wilhite, S. A diary study of task switching and interruptions. *CHI '04*, ACM Press (2004), 175-182.
7. Greenberg, S. and Fitchett, C. Phidgets: easy development of physical interfaces through physical widgets. *UIST '01*, (2001), 209-218.
8. Hassenzahl, M., Burmester, M., and Koller, F. AttrakDiff: Ein Fragebogen zur Messung wahrgenommener hedonischer und pragmatischer Qualität. *Mensch & Computer*, (2003), 187-196.
9. Kalanithi, J.J. and Jr., V.M.B. Connectibles: tangible social networks. *TEI '08*, (2008), 199-206.
10. Kato, Y. Splish: A Visual Programming Environment for Arduino to Accelerate Physical Computing Experiences. *C5 '10*, (2010), 3-10.
11. Kayes, D.C. Experiential learning and its critics: Preserving the role of experience in management learning and education. *Academy of Management Learning and Education*, (2002), 1, 2, 137-149.
12. Kim, S. and Paulos, E. Practices in the creative reuse of e-waste. *CHI '11*, ACM Press (2011), 2395-2404.
13. Knörig, A., Wettach, R., and Cohen, J. Fritzing: a tool for advancing electronic prototyping for designers. *TEI '09*, (2009), 351-358.
14. Kolb, D. Experiential learning: Experience as the source of learning and development. Prentice-Hall, 1984.
15. Kolb., D., Boyatzis, R.E., and Mainemelis, C. Experiential Learning Theory: Previous Research and New Directions. *Perspectives on cognitive, learning, and thinking styles*. Lawrence Erlbaum, (2000), 227-247.
16. Kuznetsov, S. and Paulos, E. Rise of the expert amateur. *NordiCHI '10*, ACM Press (2010), 295-304.
17. Lovell, E. and Buechley, L. An e-sewing tutorial for DIY learning. *IDC '10*, (2010), 230-233.
18. Martin, F., Mikhak, B., Resnick, M., Silverman, B., and Berg, R. To mindstorms and beyond: evolution of a construction kit for magical machines. *Robots for kids: exploring new technologies for learning*, (2000).
19. Ochiai, Y. The visible electricity device. *SIGGRAPH '10 - Poster*, ACM Press (2010), 1.
20. Patten, J., Ishii, H., Hines, J., and Pangaro, G. Sensetable: a wireless object tracking platform for tangible user interfaces. *CHI '01*, ACM Press (2001), 253-260.
21. Peek, N., Pitman, D., and The, R. Hangsters: tangible peripheral interactive avatars for instant messaging. *TEI '09*, (2009), 25-26.
22. Prenzel, M., Drechsel, B., and Kramer, K. Lernmotivation im kaufmännischen Unterricht: die Sicht von Auszubildenden und Lehrkräften. In *Kompetenzentwicklung in der Berufserziehung*, (1998), 169-187.
23. Rosenbaum, E., Eastmond, E., and Mellis, D. Empowering programmability for tangibles. *TEI '10*, (2010).
24. Sheldon, K.M., Elliot, A.J., Kim, Y., and Kasser, T. What is satisfying about satisfying events? Testing 10 candidate psychological needs. *Journal of Personality and Social Psychology 80*, 2 (2001), 325-339.
25. Steiner, H.-C. Firmata: Towards making microcontrollers act like extensions of the computer. *NIME '09*, (2009), 125-130.
26. Underkoffler, J. and Ishii, H. Urp: a luminous-tangible workbench for urban planning and design. *CHI '99*, ACM Press (1999), 386-393.
27. Underkoffler, J., Ullmer, B., and Ishii, H. Emancipated pixels: real-world graphics in the luminous room. *SIGGRAPH '99*, (1999), 385-392.
28. Wakita, A. and Anezaki, Y. Intuino: An Authoring Tool for Supporting the Prototyping of Organic Interfaces. *DIS '10*, (2010), 179-188.
29. Weiss, M., Voelker, S., Sutter, C., and Borchers, J. BendDesk: Dragging Across the Curve. *ITS '10*, (2010).
30. Wellner, P. Interacting with paper on the DigitalDesk. *CACM 36*, 7 (1993), 87-96.
31. Wieman, C.E. and Perkins, K. K. A powerful tool for teaching science. *Nature Physics 2*, 5 (2006), 290-292.
32. Wigdor, D., Penn, G., Ryall, K., Esenther, A., and Shen, C. Living with a Tabletop: Analysis and Observations of Long Term Office Use of a Multi-Touch Table. *TABLETOP '07*, (2007), 60-67.
33. Wimmer, R., Hennecke, F., Schulz, F., Boring, S., Butz, A., and Hussmann, H. Curve: Revisiting the Digital Desk. *NordiCHI '10*, (2010), 561-570.