Agile Introduction to Microcontroller Programming

A Course Concept on Empowerment by Making

Matthias Ehlenz
Lehr- und Forschungsgebiet i9 RWTH Aachen
Aachen, Germany
matthias.ehlenz@rwth-aachen.de

Hans-Peter Kühn
Lehr- und Forschungsgebiet i9 RWTH Aachen
Aachen, Germany
hans-peter.kuehn@rwth-aachen.de

ABSTRACT

Physical Computing makes it possible to create everything a student could dream of. What is missing today is not the tools, but the mind set. Self-efficacy lacks in many students. This work presents a course concept for highly gifted students, which focuses on self-empowerment by self-driven product development far beyond academic challenges provided by other programs. We describe a successful course iterated four times now and show how proper introduction to physical computing and adapted Scrum lead to highly identifiable results.

CCS CONCEPTS

• **Social and professional topics → Project management techniques**; **Computing education programs**; *Informal education*; • **Applied computing → Collaborative learning**;

KEYWORDS

maker movement, scrum, agile education, Craft-based projects, Arduino programming, self-empowerment, STEAM

Reference Format:
Matthias Ehlenz and Hans-Peter Kühn. 2018. Agile Introduction to Microcontroller Programming: A Course Concept on Empowerment by Making. Presented at *Craft- and Project-based Making for STEAM Learning - eCraft2Learn Workshop,* Koli Calling 2018. Available online at https://www.kolicalling.fi/index.php/ecraft-workshop.

1 INTRODUCTION

Supportive programs for highly gifted children are still on the rise and rare to find, especially in lower secondary education. Public schooling is on the verge of installing mechanisms for individual support for all kinds of special needs. For those striving for new challenges the JuniorAkademie is a public funded program designed to fill this gap. This paper describes a course concept which is created to empower children to realize own ideas in the sense of the Maker Movement. It offers the possibility to develop, design and implement own projects in an intense contextualization of STEAM learning, leaving room for following own interests and strengths, emphasizing the A in STEAM learning. After a rapid but nonetheless thorough introduction in both Arduino and Scrum, the participants organize themselves in a week of agile, time-constraint development of their own visions, using the whole spectrum of STEAM and a deep grip into the fascinating toolbox of the Maker Movement.

eCraft18, November 2018, Joensuu, Finland
2018.

2 DEUTSCHE JUNIORAKADEMIEN

The program "Deutsche JuniorAkademien" (translatable to German Junior Academies, in the following DJA) is a state-specific promotional program for highly gifted students of lower secondary education which is by now implemented all over Germany. The DJA committed themselves to rigid standards to ensure highest pedagogical quality, approved by the Standing Conference of the Ministers of Education and Cultural Affairs in 2006.

2.1 General Concept

In North Rhine-Westphalia the DJAs are organized by the ministry of education. It's supported by partners from industry and research institutions and offers nine courses for 18 students each, distributed over three locations. The students commit ten days of their summer break and have to pay a comparable low fee. Primary aim is to increase and sustain performance motivation, encourage networking with like-minded and raise interest in challenge-seeking.

The long-term sustainability of such programs has been scientifically shown in [4]. The courses are intended to reach levels comparable to introductory university courses and leaves the comfort zone of lower secondary school contents by far. This offers the possibility to explore new areas of interest under the guidance of young scientists and highly motivated teachers in a way not achievable by schools. The experience after overcoming unusual challenges leads to increased self-perception, resulting in increased long-term learning motivation. [4] also shows increased interest in the course topics (83% of participants), 51% even reported significant influence on their later career choices. The courses are accompanied by supplementing activities including sports, choir, orchestra and self-organized work shops. The overall impact on group dynamics is paramount, usually resulting in deep emotional reactions after the parting ceremony.

2.2 Student body

The students are all in lower secondary education, therefore range in age from 13 to 15 years. Recommendation by their schools or a similar institution is mandatory as well as a letter of motivation by themselves. Mental maturity is comparable to higher education students. An independent committee sifts all application and selects candidates, resulting in an acceptance rate of roughly 30 percent.

2.3 Courses and Educators

The main idea of this program is to bring together students with high intrinsic motivation and offer them the possibility to immerse themselves into an extracurricular topic of their choice for ten days straight. Courses offered include forensics, nano-technology and

the here presented micro-controller programming course. Courses offer 18 places each with strict parity of gender, including 9 girls and 9 boys. The courses are held by scientists and educators from the courses domain, usually following a team teaching tandem pattern.

3 COURSE CONCEPT - MICRO-CONTROLLING

Organizational constrains limit the number of courses per location to three. Traditionally those have been nano-technology, forensics and cryptography in Jülich. To supplement the highly theoretical, somehow limited spectrum with a more practical approach, a micro controlling course was piloted in 2015 and raised a high demand within the applying student body. Intention of the presented course is to raise self-empowerment in the participating students and to allow for utmost creative freedom, focusing on Arduino but touching various aspects of the maker movement, including design, app development, usage of tools, woodworking, 3D printing, laser cutting and soldering. Center of the course is a project chosen by the students and self-organized work flow throughout the project to achieve a maximum identification with the product. This way, we want to convey the creativity innate to computer science which is less recognized by the general public, but nonetheless a very important part to bridge the gap between STEM subjects and arts. [6, p. 27, 36-42] has shown the importance of creativity in computer science education and described the multiple connections between arts and computer science.

3.1 Why Scrum?

Scrum is currently one of the most employed strategies in software development. Universities worldwide teach their computer science students methodologies derived from the agile manifesto. Scrum unfolds all it's strength in development processes on a tight schedule, values short-iteration prototypes and encourages product- and feature-oriented development. Communication is valued higher than documentation and team work is in the center of it. Most important, when talking about a target audience of young students, it makes progress visible, even graspable using methods like cardboard on pin board back logs. The DJA program offers a strictly constrained time window of ten days, which makes a modified Scrum the perfect fit for this course. To achieve maximum success our goal is to maximize students identification with the product, which can be solely obtained by indirect guidance. The necessary modifications will be described in a following section. Nonetheless, each course still rises unforeseen challenges, so experience and a little talent for improvisation is mandatory for the supervisors.

3.2 Course structure

The course starts out with getting to know each other. During this phase it is important to obtain knowledge about personal interests, experiences and expectations of the students. This is all resolved in the first evening session. Day one starts out with an introduction to Arduino and ends with a structured overview of all available resources. Day two begins with an introduction to Scrum and will lead seamlessly into the working phase, all of which will be described in this section.

3.3 Phase 1: Informatik Enlightened

In the first phase the course starts by giving a structured introduction into micro-controller programming by employing the workshop materials "Informatik Enlightened" of the InfoSphere school lab of RWTH Aachen University. Informatik Enlightened was developed in a federal funded project "Personal Photonics", aiming to bring a basic introduction of textual programming to schools while using the context of light and photonics. The workshop material focuses on self-paced learning in teams of two students. Starting with a "Station 0" after a really short educator's input, students learn using the Arduino IDE, basic program structure (setup and loop) and electronic circuits fundamentals. In the end, all students will have programmed their own blink sketch, as well as read the state of a button. Following station 0, students can choose between four further tasks: One station builds up a light-sensitive sunflower rotating to the brightest source of light, a parking assistant beeping according to a distance measured with IR-light, measuring and calculating speed using photoelectric barriers and mixing light in a RGB LED according to measured temperature. All stations are created with gender-neutrality, achievability of satisfying results and openness for individual extensions in mind. Furthermore, combined together the stations cover a wide range of the Arduino universe: libraries, sensors, analog inputs, PWM output, servo motors, reference resistors and a basic idea of state machines. The workshop material is designed for early secondary education and usually takes up five hours for students to complete station 0 and one follow-up of their choice. The material is available in the spirit of open educational resources on the school labs website in both German and English language[5]. Considering the special nature of the DJA's audience, most students complete the introduction and two of the following stations on the first day, usually even organizing themselves to distribute the different stations and thereby knowledge over the group.

3.4 Phase 2: Scrum

In the second phase, we introduce Scrum. We lay out the idea behind agile software development by explaining the up- and downsides of waterfall project management and then discussing the agile manifesto [3]. Afterwards, we explain the core ideas of Scrum (empirical, incremental, iterative), clarify that we're using a simplified version and then explain our implementation of the Scrum framework, which is based directly on the Scrum Primer v2.0 [1, p. 10]. We touch on the differences to pure Scrum, but focus on presenting the upcoming project management method to the students.

3.4.1 Roles. The three introduced roles are: Development Teams, Product Owner, Scrum Master Team.

Like original Scrum, *Development Teams* are cross-functional and self-organizing. They have three tasks: 1. design, construct and implement the agreed upon functionality. 2. work autonomously and self-organizing. 3. uphold the quality standards priorly agreed upon. In order to achieve this, they have to become specialists as much as generalists.

The *Product Owner* is a single external person. He doesn't necessarily have a technical background but organizes any outside influences, e.g. the communication with stakeholders. He has to change user stories according to the stakeholders wishes, prioritize

user stories, to decide upon the general working environment and should not command or impede. Traditionally the students choose the sports teacher to fulfill this role.

Scrum Master Team While the Scrum Master is usually a single person taking the responsibility of the project organization and method, we deemed it unreasonable to place such a responsibility on a single student. Therefore, this role is implemented as a team effort with distributed responsibilities. The team consists of three to four students and has multiple duties:

- to facilitate communication between all roles and teams,
- to monitor the compliance with Scrum rules,
- to remove impediments and solve conflicts,
- to not instruct or command, but to organize and help wherever necessary.

In order to have a trustworthy contact person for every student and enforce a more harmonic group process, we require the team to include students of each gender.

Furthermore it is the responsibility of the Scrum Master Team to manage resources efficiently. They control time and task prioritization of the "work force" and manage the small budget for procurement of additional sensors, construction material or electronics.

3.4.2 Artifacts. After presenting the roles, we introduce the students to the typical Scrum artifacts: *Tasks, User Stories, Backlogs.*
User Stories are a description of a single functionality of the product, written from the user's point of view without any technical details, e.g. "The user can steer the vehicle with a joystick." The Product Owner specifies and prioritizes user stories, although Development Teams can suggest and discuss user stories with the Product Owner during activities.
Tasks are technical assignments that are necessary to implement specific user stories. The Development Teams deduce tasks from user stories, prioritize, schedule and distribute them. In order to facilitate planning and organization, we introduce the SMART framework [2, p. 35-36], which is an acronym for Specific, Measurable, Achievable, Reasonable, Time-bound. As a *Definition of Done* we enforce the dual control principle.
We implement this by providing the students with a formalized layout for task slips, that include a task description, signatures of the student in charge and the examiner, estimated person-hours and a field for actual person-hours.
We introduce the students to three organizational charts: *Product Backlog, Sprint Backlog, Product Increment.* The Product Backlog contains and organizes all user stories. They can have multiple states, e.g. New, Ready for Sprint, In Sprint or Done. The Sprint Backlog follows the same principle, but contains tasks. It discerns the states: New, Ready, In Progress, Done. The Product Increment is introduced just a progress board showcasing all implemented user stories. But beyond the Sprint Backlog, we leave it to the students to decide which charts they want to use during the development process.

3.4.3 Order Slips. To remain accessible as advisers whenever necessary after the reversion of control, we implement one non-Scrum artifact: order slips. Workshop assignments, procurement tasks, advisory tasks and expert discussions are all issued by filling a specific form, that includes task description, issuer for queries and most importantly priority. In the role of the workshop team, we only accept assignments in exchange for a correctly filled order slip.

3.4.4 Meetings. Because the academy is constrained to 10 days in total, we implement Scrum meetings heavily adapted. One sprint usually lasts one or two days. As a result, there is no practical distinction between Sprint Planning and Daily Scrum. Technically are no real Sprint Reviews, but we enforce one Sprint Retrospective in the middle of the project. The *Product Planning* is done once after the decision on a project. The Product Owner is invited and the class acts as a kind of start-up company trying to convince a business angel to invest in their product. The PO is briefed by the supervisors and demands a broad but feasible feature set and sets reasonable priorities. *Sprint Plannings* are organized by the Scrum Master team. Due to the reduction to one day sprints they are held in combination with the *Daily Scrums* as stand-up meetings in the beginning of each work day. *Sprint Reviews* conclude the day and monitor progress. A *Sprint Retrospective* is held once after the first half of the course. It acts as a brief period of methodological reflection and helps to resolve group conflicts and encourages a redistribution of work load.

3.5 Phase 3: Self-Organization

After the previous phases the active role of the supervisors is nearly over. The students are self-organized at this point, know all the programming basics and have a brief overview of the available resources. In the following time, the students have to actively pull any support from their teachers to increase their recognition of their role as makers and creators.

3.5.1 Deciding on a project. This is the most suspenseful phase for the teachers. The students are left alone to discuss possible projects for the upcoming week of work. The limitations given are simple: Nothing edible (hygiene), nothing aerial (legal constraints) and no high voltage. They have to narrow it down to three projects they'd like to implement and present it afterwards. After a short reflection period the teachers give objective statements to each project regarding feasibility with the available resources (time, material, tools and knowledge), leaving the final decision to the students.

3.5.2 Organizing teams. Teams start out organized by core tasks, e.g. programming actors, sensors, defining a communication protocol, planning construction & design and, if necessary, app programming. Reasonable division is presented as last years student self-organization as a starting point, open for project-specific adaption. Over the course of the week, students typically start moving from one team to another to support where help is needed.

3.5.3 The teachers role. After the decision is made the teachers switch into a different role: Previously somehow comparable to school teachers, the supervisors are now subordinates of the development teams, announced as blue collar workers responsible for the usage of heavy machinery as well as specialist devices (laser cutters and 3D printers). Each "order" to the workshop has to be exact and accurately specified and handed in on a form signed and prioritized by the project management team. Apart from those order sheets there are additional forms to request procurement of material (i.e.

from the hardware store), technical support (problems with code) or expert discussions, the latter including a preparation time of at least one hour to help with conceptual, strategical problems. It is strictly in the responsibility of the students to manage the time of their "employees".

3.5.4 Means of influence. The reversion of control does not include the reversion of responsibility. Success and outcome of the course are still in the responsibility of the educators and therefore means of influence have been developed to maintain the identification with the product while still ensuring a reasonable degree of control. Even while leaving it to the group to self-organize, at least one educator attends the Daily Scrums in character to supervise the progress. If this demands interference, the following measures can be employed. The first one is the *Guidance for Scrum Master Team*, which acts as kind of single point of contact and hinge between developers and workshop. Taking them aside and briefing them on certain matters at hand does not affect the overall group dynamics and usually is enough if problems are detected in an early state. The second mean is a *Briefing of Product Owner*. Acting as technical inexperienced investor, the Product Owner is usually in the loop of development and invited to meetings at least every second day. Features getting out of hand or come out as too complex can be skipped and replaced that way. Example: "I don't need a remote control! Market research shows a blinking rainbow effect of LED's will set us paramount over all competitors!".

In unique cases, the usage of *In-Character Actions* has also shown great effect. As the Product Owner was not kept in-loop on that occasion due to deep immersion of the Scrum Master Team in their role of additional hands in development teams, the group was left headless and in a state of confusion. The solution was simply doing nothing. After two hours of coffee consumption and laying back, the students recognized the lack of output from the workshop. Demanding an explanation, the "management" was confronted with their workforce being on strike due to unpaid wages after the investor cut of funds for not been included in the development process. Self-reorganization happened in less than one hour.

A lack of pulled support or sub-teams losing focus is compensated by introducing daily lectures as course starter in the morning. This consists of brief presentations on topics at hand like power management, protocols like I2C and introduction of 3D printing software.

"Santas little helpers" should also be mentioned here. Initially the idea was for the workshop to transfer any bread board circuits to copper via order slips, but each course so far demanded introduction to soldering to solder themselves. The results are of mixed quality, so a lot of after course hours is usually spent secretly checking and redoing wiring as well as any kind of (adhesive) bonding.

4 CONCLUSION AND OUTLOOK

So far the course concept has been iterated four times with impressive results. Many students stayed in contact with each other and the staff and continue to show interest in micro controllers and the maker movement in general as inquires for new learning resources and accidental meetings on maker faires show. The 2016 project has been shown on the "Mensch und Computer" conference, the leading conference on HCI in Germany and won the peoples choice award for best demo.

4.1 Projects

Projects implemented so far include a juice cocktail mixer, a remote controlled hovercraft, a cloud-connected smart greenhouse and a four-factor-authentication locker with alarm system. All products have been built from scratch without any pre-constructed kits.

4.2 Lessons learned

Most lessons learned have already been mentioned, but to summarize the most important ones: Using beverages/edibles will result in mold, nearly unavoidable in a student project, rendering the result useless for demonstration after a few days. With our method, the learning progress is highly individual and difficult to monitor. Division of tasks will cover the whole spectrum of STEAM, leaving space for each student to follow their interests and accounting for any level of prior knowledge. That said, every student will learn and refine valuable soft skills by cooperating in an agile development team. For Self-organization to work well, high identification with the product is mandatory.

All groups worked very differently with Scrum artifacts and essentially implemented their own adaption, but developing a functional product increment has been an issue every year, with only one group producing a functional product prototype before the retrospective. Working components are ready and tested well before, but integration effort is usually underestimated and leads to frantic last days. Some groups finish their tasks sooner than others which is not problematic as those students put their time into the product presentation during the closing ceremony. Documentation is easy enough in this concept: All produced artifacts can be used as progress documentation and task slips help to monitor overall activity and show, to some degree, self-sufficiency and learning progress. Nonetheless, the concept is product focused and due to the self-organization leaves a lot of freedom. Development Teams divide tasks as they see fit and team output is what matters. This works well with this specific audience and we encourage to transfer our concept to different extra-curricular learning environments, but as individual contribution is hard to monitor, transfer to grading scenarios like schools requires further adaptions. Gender-parity in the Scrum Master team will lead to a more harmonic experience for everyone. Indirect guidance works well as long as progress, group dynamics and motivation are monitored closely. A technically unsavvy Product Owner has facilitated this process, because he requires detailed explanations by the students and can ask questions seriously, which, uttered by the teachers, would seem implausible.

Last but not least: Both duct tape and silicone will be considered a load-bearing component. By the students starting from day 1, by the educators at the latest at 2 a.m. in the night before the final presentation.

4.3 Outlook

There has been no quantitative evaluation yet. The concept went through major changes during it's development, effectively rendering all evaluation attempts void. Since the concept now reached a stage where we consider it stable, an evaluation concept is planned

for upcoming iterations. Results will be published, but the small numbers of participants and the special audience won't make it possible to derive universally valid results. We plan to evaluate applications for and adaptions to other student bodies, especially upper secondary education, but this course concept will only be applicable in an extracurricular learning contexts, because grading and performance evaluation seem highly problematic.

ACKNOWLEDGMENTS

First of all, we want to thank Michael Funke, state commissioner in charge of the DJA program for NRW for giving us both the opportunity and freedom to develop the concept we dreamt of. Thank you for your trust. Our thanks also belong to Christian Taraschewski, who went through the first iteration as instructors with one of the authors. Without your knowledge of agile development the course would not be as it is. We want to send thanks to the teams of InfoSphere and LuFG i9 of RWTH Aachen University, especially Dr. Nadine Bergner, for developing such a great introduction to Arduino and providing us with all kinds of resources! Last but not least we thank our colleagues of the JA team. There can't be any better people to spend your 'holidays' with. Worth every minute.

REFERENCES

[1] Pete Deemer, Gabrielle Benefield, Craig Larman, and Bas Vodde. 2012. The Scrum Primer. Retrieved October 8, 2018 from http://scrumprimer.org/scrumprimer20_small.pdf

[2] G. Doran. 1981. There's a S.M.A.R.T. way to write management's goals and objectives. *Management Review* 70 (1981).

[3] Beck et. al. 2001. Agile Manifesto. Retrieved October 8, 2018 from http://agilemanifesto.org/

[4] E. Hany. 2007. Hochbegabtenförderung auf dem Prüfstand: Evaluationsbefunde und Desiderata. In *Begabt sein in Deutschland*. LIT Verlag Münster.

[5] InfoSphere School Lab. [n. d.]. Computer Science Enlightened. Retrieved October 8, 2018 from https://schuelerlabor.informatik.rwth-aachen.de/en/modulmaterialien/informatik-enlightened

[6] URL = http://ddi.cs.uni-potsdam.de/Forschung/Schriften/RomeikeDiss2008.pdf Year = 2008 Ralf Romeike, School = Universität Potsdam. [n. d.]. *Kreativität im Informatikunterricht.* dissertation.