Chapter 5
When Students Get Stuck: Adaptive Remote Labs as a Way to Support Students in Practical Engineering Education

Anja Hawlitschek, Till Krenz, and Sebastian Zug

1 Introduction

The field of computer science has to deal with a relatively high number (over 40%) of dropouts at German universities (Heublein, 2014). However, dropout in computer science is not only a problem at German universities but also in other European countries (Kori et al., 2015) or in the USA (Talton et al., 2006). The dropout rate of female students is often even higher than that of their male fellow students (Talton et al., 2006), which might be a result of being underrepresented in the discipline (Cox & Fisher, 2008). The reasons for dropout are complex. Most often the students have false expectations about the contents of study, which lead to motivational problems, or they are frustrated due to high performance requirements. At the same time, the increasing heterogeneity of students leads to dropouts, in particular due to problems with different prior knowledge but also because of sociodemographic factors, e.g., an increasing number of students who have to balance study, work, and/or parenting (Isleib & Heublein, 2017). Especially, prior knowledge and academic preparedness of students are correlated with retention in computer science programs (Horton & Craig, 2015; Kori et al., 2015; Talton et al., 2006). Also motivation and interest of the students play an important role. The higher the motivation and interest in the content, the lower the probability of dropout (Kori et al., 2015, 2016).

The situation at course level is similar. Within a meta-analysis in 161 introductory programming courses in 15 countries worldwide, Watson and Li (2014) revealed a dropout rate of approximately 32%. The percentage of students who

A. Hawlitschek (✉)
Magdeburg-Stendal University of Applied Sciences, Magdeburg, Germany
e-mail: anja.hawlitschek@hs-magdeburg.de

T. Krenz · S. Zug
Otto-von-Guericke-University Magdeburg, Magdeburg, Germany

D. Ifenthaler et al. (eds.), *Utilizing Learning Analytics to Support Study Success*,
https://doi.org/10.1007/978-3-319-64792-0_5

did not pass the introductory programming course remained nearly constant between 1980 and 2013. There were no significant differences in dropout rates with regard to programming language taught. Furthermore, while the authors found significant differences between the dropout rate in the different countries (Portugal and Germany had the highest dropout rates with over 50%, whereas Canada and Taiwan, e.g., had noticeable lower rates of about 20%), because of small sample sizes, these results should not be overestimated or generalized. If reasons for dropout are already reflected on the course level, this could be a starting point for providing individual support to students who have a higher probability of dropping out. With the help of learning analytics, it becomes possible to detect students at risk automatically (Papamitsiou & Economides, 2014). Learning analytics is the collection, storage, analysis, and evaluation of learner data to optimize learning and learning environments (Ferguson, 2012). A growing number of universities all over the world already use the data generated by their students for the evaluation of teaching, the provision of adapted content, and as an early warning system. The latter, for example, filters out students at risk of dropping out on the basis of their activities in the learning management system, e.g., time spent in exercises or quizzes (Arnold & Pistilli, 2012). There are different options to support these students: lecturers probably offer additional material or repeat the basics for the course or individual students. The additional effort addresses the specific needs of the learners, for example, concerning the sequence, difficulty, or scope of content (Leutner, 2002; Melis et al., 2001; van Seters et al., 2012). The goal of implementing adaptivity is to facilitate individualized learning environments to support efficient and effective learning and avert high dropout rates. If it is possible to identify the needs of users on the basis of patterns of user behavior, it is also possible to implement a more fine-grained form of adaptivity without the usage of assessment tests and questionnaires. The challenge here is that knowledge about user behavior, which reveals students at risk might not be sufficient for helping these students. To give an example, on the basis of user behavior, it is not directly evident whether a user spends little time on an exercise in the learning management system and has a result below average in an accompanying quiz because (1) he is demotivated because the task is to difficult or (2) he had too little time because he had to work to finance his study or (3) he is frustrated due to low usability of the learning management system or (4) for any other reasons. Different reasons for an undesirable user behavior require a different reaction of the learning system or the lecturer. This is only possible if the underlying causes are known. While user behavior alone can provide evidence that there are problems in the learning process and that intervention might be necessary, the choice of what type of intervention is needed will usually not be based solely on user behavior. Therefore, in this study, we will start at an earlier point of the analytics and begin by examining which learner characteristics are relevant for dropout in a blended-learning course in computer science. In a second step, we examine whether user behavior is related to such factors and/or to dropout rates.

2 Dropout in Blended Learning

In computer science, as in other STEM subjects, studying in laboratories is especially important. In these laboratories theory and practice are combined, and students acquire practical skills for their professional career. Blended learning is a promising approach for a laboratory learning setting. Blended learning is the attempt to combine the time in the course on-campus, which is highly relevant for the learning performance (Schneider & Preckel, 2017; Schulmeister, 2017), with the advantages of online learning, such as greater local and temporal flexibility of the students. In comparison with courses that take place only online, the dropout rates in blended learning are lower, presumably due to the regular face-to-face time with the lecturer and other students (Park & Choi, 2009). Results from studies suggest that blended learning might also be superior to courses without any online learning, i.e., which only take place on-campus (Al-Qahtani & Higgins, 2013; Bernard, Borokhovski, Schmid, Tamim, & Abrami, 2014; López-Pérez, Pérez-López, & Rodríguez-Ariza, 2011). The remote control of a laboratory (via web interface) provides students with experiences and competencies they will need in a digitized workplace. In addition, there are the advantages already mentioned: Students can access the learning environment regardless of location and time and are not bound to limited laboratory hours. They can work in the laboratory as often and as long as necessary for their individual learning processes. However, despite the advantage of blended learning, to combine the best of e-learning and face-to-face-learning, the online phase is still a challenge because there is no direct contact between the lecturer and the students. Thus, the probability of problems (e.g., if code is not doing what it is supposed to do) leading to frustration and in the long run to dropout is much more likely to occur than in face-to-face time on campus with the possibility of direct feedback and help. In the scientific literature, different factors in the use of digital learning environments are examined with regard to the dropout rate. Park and Choi (2009) distinguish factors that affect the decision to drop out in those that occur prior to the course and those that are relevant during the course. Factors prior to the course are sociodemographic variables. Often, studies hereby focus on age and gender (Marks, Sibley, & Arbaugh, 2005). Factors which affect the possibility to drop out during the course can be distinguished in external factors resulting from influences from outside the course, e.g., family time constraints and job working hours. Internal factors arise from the student's engagement with the learning setting and the digital learning environment. Learners are not a homogenous mass. There are differences in cognitive and affective variables (Narciss, Proske, & Koerndle, 2007), affecting the perception and the effects of a learning environment, for example, whether the instructional design fits the needs of the learner or whether usability issues might result in a lack of motivation. In this study we focus on the internal factors because these are especially important for gaining insight into the learning processes and related factors which are relevant for the decision to dropout (see also the results from Park & Choi, 2009).

With regard to the internal factors, we can distinguish approaches that have a focus on motivational components of learning and approaches with a focus on cognitive processing.

2.1 Motivation and Dropout

Motivation is a basis for learning. Motivation determines whether and how learners (1) deal with the content and (2) use a digital learning environment. Some studies target learners' satisfaction, which in fact appears to have a relevant impact on the dropout rate (Fredericksen, Pickett, Shea, Pelz, & Swan, 2000; Park & Choi, 2009). The more satisfied learners are with the learning environment, the lower the likelihood of dropouts (Levy, 2007). However, satisfaction is a very broad concept that can be influenced by different underlying factors. This is also reflected in questionnaires used in some of the studies, which integrate items for ease of use, usefulness, intrinsic motivation, and social interaction (Levy, 2007). In this study we want to analyze different facets of motivation in order to adapt interventions more precisely to the learners needs. Therefore, we focus on the technology acceptance model which highlights the relevance of user evaluations of learning environments against the background of a cost-benefit model of motivation. Relevant questions for the user therefore are: Is the digital learning environment useful for me? Is the effort I have to invest justified in the light of the benefits? The Technology Acceptance Model (TAM) and the further developments, like TAM2 and UTAUT, have gained particular influence concerning studies on the behavioral intentions to use and the actual usage of software (Davis, Bagozzi, & Warshaw, 1989; Venkatesh & Davis, 2000; Venkatesh, Morris, Davis, & Davis, 2003). Furthermore TAM is also used to analyze and explain the effectiveness of digital learning environments (Legris, Ingham, & Collerette, 2003; Liaw, 2008). Perceived usefulness and perceived ease of use are the most influential factors in the model. The more satisfied a learner is with the usefulness and ease of use of a digital learning environment, the higher the persistence of the learner and the lower the dropout rate (Joo, Lim, & Kim, 2011; Park & Choi, 2009). The self-efficacy of learners in dealing with the learning environment or requirements of the content seems to be a crucial intervening variable (Liaw, 2008; Schneider & Preckel, 2017; Wu, Tennyson, & Hsia, 2010). Additionally learners can also be highly motivated when dealing with a digital learning environment because they are interested in the content and/or they enjoy working on the tasks, i.e., they have intrinsic motivation. The benefits that intrinsically motivated learners derive from engaging with the remote lab are thusly less focused on outcomes, but more on intrinsic incentives of the activity as such. The assumption that learners with more intrinsic motivation drop out less frequently and have a higher learning performance is obvious (Giesbers, Rienties, Tempelaar, & Gijselaers, 2013; Law, Lee, & Yu, 2010).

Accordingly our first research question is as follows: Do persistent learner and dropouts show differences concerning motivational variables as perceived usefulness, ease of use, intrinsic motivation, and self-efficacy?

2.2 Cognitive Load and Dropout

Based on the assumption of a limited cognitive capacity in working memory, research on cognitive load theory (CLT) tries to identify instructional designs which make the usage of cognitive resources for dealing with information as efficient as possible (Plass, Moreno, & Brünken, 2010; Schnotz & Kürschner, 2007; Sweller, Ayres, & Kalyuga, 2011). CLT differentiates between different kinds of cognitive load (Kalyuga, 2011). Extraneous cognitive load (ECL) is caused through suboptimal design of an instruction. An inefficient design requires cognitive capacity that is not due to learning but due to other cognitive activities. During learning extraneous cognitive load should be as low as possible, ensuring that more cognitive capacity is available for the learning processes. Intrinsic cognitive load (ICL) on the other hand is caused by complexity of task and information, especially by the number of interrelated elements that have to be processed simultaneously for understanding the content (element interactivity). However, ICL depends also on the prior knowledge of the learner. More experienced learners have knowledge structures stored in long-term memory, which help them to process and organize novel information in working memory. Therefore, they are able to treat single elements of a task as a whole element (or schema) which in fact leads to decreased element interactivity (Chen, Kalyuga, & Sweller, 2017). Research consistently reveals that to take the domain-specific prior knowledge into account is of high relevance for efficient instructional design (e.g., Chen, Kalyuga, & Sweller, 2017; Kalyuga, 2007; Schneider & Preckel, 2017). Depending on prior knowledge, the learner needs more or less support to process the learning content and to avoid cognitive overload or boredom. Additionally, prior knowledge seems to have a compensation effect: learners with low prior knowledge highly depend on appropriate instructional design to reach an optimal learning performance, while learners with higher prior knowledge could also deal with poor instructional design, e.g., an instructional design which causes a high amount of extraneous cognitive load (Kalyuga, 2007).

In CLT some researchers assume a third type of cognitive load, namely, germane cognitive load, which is caused through schema acquisition; however, there is an ongoing discussion about the necessity to distinguish between intrinsic and germane cognitive load. A reconceptualization of germane cognitive load as germane processing, e.g., the amount of mental effort invested dealing with intrinsic cognitive load goes hand in hand (Kalyuga, 2011; Leppink, Paas, van Gog, van der Vleuten, & van Merriënboer, 2014; Sweller, Ayres, & Kalyuga, 2011). The mental effort learners invest in the cognitive processing of learning content is on the other hand a question of motivation (Bures, Abrami, & Amundsen, 2000). Although the influence of motivation on the amount of invested mental effort was considered in research on cognitive load early on (Paas, Tuovinen, van Merriënboer, & Darabi, 2005; Moreno, 2006), there is still a research gap (Leutner, 2014; Mayer, 2014; Park, Plass, & Brünken, 2014). Leppink et al. (2014) examine an interesting approach by operationalizing germane cognitive load (or rather germane processing) with items that apparently measure the perceived usefulness of the content for the learning process. This way they implicitly implement a factor which is highly relevant for motivation as is already mentioned in the context of TAM. However, in their study they found no significant correlation between germane cognitive load (or usefulness respectively) and the learning performance.

Whereas it seems plausible that the amount of extraneous cognitive load and germane processing is crucial for students dropping out or persisting, there are no empirical results yet. The potential effects of prior knowledge seem to be especially important. In computer science, there are students in the first semester that have been programming for years, attending hackathons, and using GitHub, while others are just beginning with their first "Hello World." Since the remote lab is a complex learning environment in which students actively solve problems and thereby explore and construct knowledge, it is cognitively very demanding in particular for novice learners. Results of a study on a remote lab indicate that the learning performance of the students at least partially depend on their prior knowledge (Zug, Hawlitschek, & Krenz, 2017). Students with lower prior knowledge have lower grades in the exam. However, it is not clear if this effect also is transferrable on dropout rates.

So our second research question is: Do persistent learner and dropouts show differences in cognitive variables like extraneous, intrinsic, and germane cognitive load and their prior knowledge?

2.3 User Behavior and Dropout

Programming is an iterative process, in which the functionalities are implemented as features, step by step. It is common to write a part of a program, for example, a function or a class, with its basic components first and check if the execution of the program with the inclusion of the new code works. If the execution or compilation fails, the code needs to be revisited and amended. As soon as the program compiles with the new code, the complexity of the function or class can be extended, or new features can be implemented. Rinse and repeat.

An experienced programmer will add several lines of code before checking its correctness by trying to compile the code, while a novice might only add a few lines or commands before compilation, since it is easier to isolate the cause of an error with the latter strategy. It could be expected that an experienced programmer's code revisions would grow faster and have fewer failing builds, the time spent between builds would tend to be longer, and the amount of added lines per revision would be higher, than it would be expected for an inexperienced programmer. Especially situations where the code compilation fails several times in succession, we consider to be of high relevance. This could be an indication for an inexperienced programmer, who fails to interpret the error messages in a way that would allow them to get the code working. The complex process of writing program code could thusly be reduced to the occurrence of such error streaks, in order to classify persons as experienced and inexperienced programmers on a macro level. On a micro level, a system that is aware of the error streak concept could provide assistance to students that are currently stuck.

Therefore, the third research question is: Do dropouts and persistent learners show differences concerning the probability of an error streak? Is the probability of an error streak related with prior knowledge?

In the following sections, we will examine how students who have successfully completed the entire course (i.e., got a participation certificate) differ from students who left the course at any point in time. On the basis of the findings, an adaptation to the needs of specific target groups can take place.

3 Study

3.1 Description of the Course

The subject of the study is a course at the Faculty of Computer Science of a German university. The investigated course started with 70 students in the first lecture, 22 of them dropped out prematurely. So, the dropout rate in this course was about 31%. This is slightly better than the general dropout in computer science, but there is still much room for improvement.

The course conveys the fundamentals of embedded systems in theory and practice. In addition to a lecture and weekly appointments with tutors, the students had to program real robots located in the laboratory via remote access in five exercises. These practical exercises are built on each other. Whereas in the first exercise the students only had to establish a connection to the robots, in the last exercise they had to program the robots to escape from a maze. The program code has to be developed in C++ for Atmel microcontrollers.

For the exercises we provided a digital learning environment with task description and literature on the one hand and a programming interface with livestream from the robots on the other hand. The students prepared their code, compiled it, and sent the executable to one of the robots. Based on outputs and by the video stream, the students evaluated the correctness. At the end of each exercise season (2–3 weeks), the program code and the results are checked by a tutor.

3.2 Methods and Instrumentation

The study was conducted in the winter term 2017/2018 (see Fig. 5.1). During the first lecture, the students filled out a quantitative questionnaire concerning their prior knowledge and sociodemographic variables. The prior knowledge test consisted of two parts. The first part was a multiple-choice test based on the content of the course. The test was supplemented by two code snippets in the programming language Java, whose functionality the students had to evaluate. In the second part, the students had to self-esteem their prior knowledge concerning different thematic fields of computer science as well as their general programming skills in comparison with their fellow students (Siegmund, Kästner, Liebig, Apel, & Hanenberg, 2014).

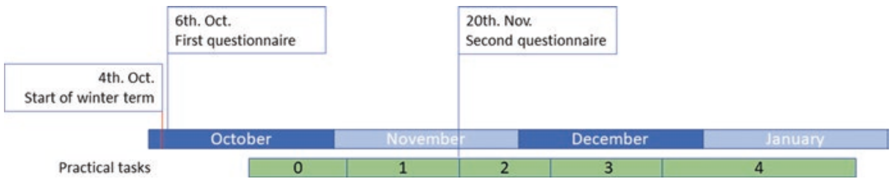

Fig. 5.1 Procedure of course and study

The second questionnaire was submitted after the second exercise. In this questionnaire the students had to rate their intrinsic motivation while working on the exercises in the remote laboratory (based on Isen & Reeve, 2005) and the ease of use of the learning environment (Legris, Ingham, & Collerette, 2003). The extraneous and intrinsic cognitive load as well as the germane cognitive load was examined with an instrument by Leppink et al. (2014). For the measurement of ECL and ICL, we used the original questionnaire. For the measurement of GCL, we used one item to measure perceived mental effort in understanding the content ("I invested a very high mental effort in enhancing my knowledge and understanding."; see Leppink et al., 2014, study 2). We applied the remaining items to operationalize the perceived usefulness of the learning environment. While usefulness in TAM studies is usually operationalized in terms of software efficiency measures (Legris, Ingham, & Collerette, 2003), concerning genuine learning environments and in the context of our thematic focus on dropout, this operationalization seems more appropriate to us. We used a Likert-type rating scale ranging from 1 (very low) to 5 (very high).

The remote system used in our project stores the whole programming code, whenever the user starts the compilation process, alongside the messages the compiler returned: error messages, warnings, and compiling reports. For the analysis presented in this article, we transformed these detailed information into a vector of consecutive build statuses, classifying each compilation attempt as failing or successful. As a next step, we calculated the probabilities of one status turning into the other or staying the same. These probabilities can be visualized as a simple network plot.

3.3 Sample

In the first questionnaire 58 students (f, 8; m, 49; missing, 1) with a relatively homogeneous age (*M* = 23.6; SD = 4.2) took part. The second questionnaire was accomplished by 37 students (f, 4; m, 28; missing, 5). The participants were students of the 3rd to 5th semester. The majority were undergraduate students from computer science (80.7%); additionally, there were 10.5% students from computer systems in engineering (B.A.) and some from other computer science-related bachelor programs.

4 Results

With analyses of variance (ANOVA), we examined the differences between students who dropped out and students who persisted. The results of the prior knowledge test revealed higher means for the persistent students ($M = 8.45$; SD = 3.89) in comparison with the dropouts ($M = 6.76$; SD = 3.65), but no significant differences between both groups ($F(1.55) = 2.60$, $p = 0.11$, $\eta^2 = 0.05$). The self-estimation of their prior knowledge on different thematic fields in the context of the course also showed no significant differences (see Table 5.1).

Concerning the self-estimation of the programming skills in comparison to the fellow students, the means were nearly the same in both groups. There was no significant difference ($F(1.54) = 0.00$, $p = 0.95$, $\eta^2 = 0.00$) between dropouts ($M = 2.95$; SD = 0.89) and persistent learners ($M = 2.94$; SD = 0.95).

We applied a principal component analysis (with oblimin rotation) to analyze the items we used for measuring ease of use. Two components were extracted, which could be interpreted as actual ease of use (e.g., "The remote lab is easy to use.") and technical reliability (e.g., "The remote laboratory has worked reliable."). The results of the group comparisons on the motivational variables showed higher means for the persistent students in intrinsic motivation and ease of use. However the ANOVA yielded no significant difference between the groups concerning motivational variables (Table 5.2).

Table 5.1 Prior knowledge group comparison

Variables	Dropout learners ($N = 22$)		Persistent learners ($N = 35$)				
"Please rate your prior knowledge concerning …"	M	SD	M	SD	F	p	η^2
Roboter applications	2.14	1.24	1.83	1.24	0.82	0.36	0.02
Embedded controller/boards	2.05	1.04	1.91	1.17	0.18	0.67	0.00
Embedded operating systems	1.45	0.96	1.37	0.64	0.15	0.69	0.00
Smartphone apps	2.00	1.19	2.31	1.07	1.05	0.30	0.02
Web front end	2.86	0.99	2.43	1.19	2.03	0.16	0.04

Table 5.2 Motivation group comparison

Variables	Dropout learners		Persistent learners				
	M	SD	M	SD	F	p	η^2
	$N = 21$		$N = 34$				
Self-efficacy (Cronbach's alpha, 0.86)	3.27	0.84	3.39	0.79	0.29	0.59	0.00
	$N = 9$		$N = 27$				
Intrinsic motivation (Cronbach's alpha, 0.90)	3.39	0.70	3.74	0.78	1.29	0.26	0.04
Ease of use (Cronbach's alpha, 0.89)	4.00	1.02	4.55	0.78	2.61	0.11	0.07
Technical reliability (Cronbach's alpha, 0.86)	2.62	1.18	2.67	1.03	0.01	0.91	0.00
Perceived usefulness (Cronbach's alpha, 0.84)	3.34	0.32	3.67	0.88	1.07	0.30	0.03

Table 5.3 Cognitive load group comparisons

Variables	Dropout learners ($N = 9$)		Persistent learners ($N = 26$)				
	M	SD	*M*	SD	*F*	*p*	η^2
Intrinsic cognitive load	3.60	0.82	3.05	0.82	2.73	0.10	0.08
Extraneous cognitive load**	3.87	0.81	2.54	1.05	10.75	0.00	0.25

$^{**}p < 0.01$

We applied a principal component analysis (with oblimin rotation) to analyze the items for measuring extraneous, intrinsic, and germane cognitive load. Against our expectations, the analysis only yielded two components—intrinsic cognitive load (Cronbach's alpha, 0.86) and extraneous cognitive load (Cronbach's alpha, 0.87). The item for measuring germane cognitive load actually loaded on the intrinsic cognitive load component.

The group comparison yielded higher means for the dropout learners for both load types (Table 5.3). However, the results of the ANOVA revealed a significant difference between the groups only concerning extraneous cognitive load.

We analyzed the differences between both groups concerning the probability of error streaks with ANOVA. Indeed the means for the dropout learners ($N = 14$, $M = 0.41$, SD = 0.14) were significantly higher $F(1.53) = 8.14$, $p = 0.00$, $\eta^2 = 0.14$) than for the persistent learners ($N = 40$, $M = 0.24$, SD = 0.17). With a regression analysis, we checked whether prior knowledge had a significant effect on the probability of error streaks. Indeed our finding indicate that students with lower prior knowledge had a higher probability of error streaks ($b = -0.29$, $t = -1.96$, $p = 0.05$, $R^2 = 0.06$).

5 Discussion

In our study we tried to identify learner characteristics which are relevant for dropout rates in computer science courses. We therefore focused on a course with a combination of face-to-face instruction and online study. Such a blended learning approach gives students the possibility to learn at their own pace and in their individual learning spaces, at their chosen time, while at the same time give them the opportunity of direct interaction with the teacher and fellow students in the lecture on-campus. This configuration offers manifold methods of additional support for dropout candidates.

To ensure a specific assistance, we analyzed whether we could identify differences between motivational as well as cognitive variables between students who drop out and students who persist in the course. We assume that finding such differences is the first step for making our remote lab adaptive. An adaptive learning environment should automatically detect whether a student is at risk of dropping out and give adequate support. To know why a student is about to dropout is a precondition to provide a suitable intervention. It is a difference if a student has motivational or cognitive problems because of a lack of usability or because her programming skills are way too low to deal with the challenges of an exercise or because of other problems.

Unlike in previous studies (e.g., Kori et al., 2015, 2016), we could not find significant differences on the motivational variables between students who dropped out and persisting students (our first research question). Neither the intrinsic motivation or the self-efficacy nor the usefulness of the content or the ease of use of the learning environment were different between both groups. Motivation is a complex theoretical construct, with a lot of influencing variables. Hence, we can only guess why we have results which not support previous research. The ratings of the ease of use of the remote lab were relatively high on average so we might conclude that given a sufficient usability, the effects of that variable are not as relevant as in a poorly working system. This should be a target of further research. Given the fact that attendance and learning in the course is not entirely self-determined but also driven by external goals (e.g., a need of a participation certificate), it might be useful to include items in the questionnaire, measuring not only intrinsic but also extrinsic motivation (see also Kori et al., 2016). However, since a limitation of our study is the small size of participants and especially of dropouts in our sample, the results have to be interpreted with care. This also holds true for the results on prior knowledge which were in contrast to earlier research as well (e.g., Horton & Craig, 2015; Talton et al., 2006). Again we could not find statistically significant differences between the groups, though the mean of the prior knowledge test was rather lower for the dropouts.

The cognitive variables on the other hand revealed an interesting pattern (our second research question). While we could not find a significant difference between both groups concerning intrinsic cognitive load, this was different for extraneous cognitive load. The dropout group rated the cognitive load which was irrelevant for learning significantly higher than the persistent group. That result goes hand in hand with earlier results from cognitive load theory concerning the high relevance of eliminating extraneous cognitive load (Sweller, Ayres, & Kalyuga, 2011). Our results indicate that extraneous cognitive load not only affect learning outcomes but also persistence in a course. Students who drop out had problems that mainly arise from the design of the instruction and not necessarily from the difficulty of the exercises itself. For them, it was not always clear, what they should do in an exercise and what the next steps should be. Apparently they got stuck in the instruction rather than in the programming of the code.

However, there were also students who had the latter problem: from our results, we consider the detection of error streaks as a promising approach for learning analytics in computer science (our third research question). There was a significant difference between students who dropout and students who persist in the probability of error streaks. The former had a significantly higher probability of error streaks in the process of programming. The less prior knowledge the students had (according to prior knowledge test) the higher was the probability of error streaks. Although it seems likely that the probability of error streaks and extraneous cognitive load might correlate, there is no statistic correlation ($r = 0.08$, $p = 0.64$). So in our study, students at risk had two different problems which we have to deal with differently.

5.1 *Extraneous Cognitive Load: Practical Implications and Future Work*

Concerning extraneous cognitive load, there are two approaches how to proceed. The first one is a learning analytic approach. Because we know that there are students that got stuck in the instruction, in the following semester we can explicitly search for a pattern of user behavior this learner might show. Since we know that these students have difficulties to understand the task and the further steps to go on, we could explicitly look for user behavior which might correlate with disorientation, uncertainty, and help-seeking behavior, i.e., extensive clicks or time in the task section or a high proportion of switching between task section and editor. The second approach is to improve the design of the instruction to avoid extraneous cognitive load. Empirical research on instructional design of remote labs, for example, suggests different forms of guidance, e.g., prompts, process constraints or scaffolds to help students to keep extraneous cognitive load as low as possible, and manage intrinsic cognitive load as well (de Jong & Lazonder, 2014). Learners with lower prior knowledge highly benefit from guidance, while for a learner with higher prior knowledge guidance often is redundant or even annoying, this should be a case for adaptivity as well (Kalyuga, 2007).

5.2 *Error Streaks: Practical Implications and Future Work*

Apart from the ad hoc and postmortem detection of error streaks, the aim of this endeavor is to administer assistance to students in situations where they are stuck and unable to help themselves, in order to reduce the time students spent on a certain problem and ultimately prevent students from dropping out of the course. The detection of error streaks would allow the lecturer and trainers to intervene in person or to make the system pull up appropriate instructions to guide the students out of their error valley. In person interventions could be triggered by the system, which would flag the user and notify the lecturer about the occurrence of an error streak. The trainers could then sit down with the student, analyze the problem, and help to solve misconceptions or understandings the student might have. Of course, the trainers could point the students to resources, which cover the problematic topic. An alternative in-person intervention could be to invite other students for a common debugging session. They would then proceed to solve a similar task using the method of pair programming. In such a process the experienced students would be enabled to make the knowledge behind their capabilities explicit, thusly helping the less experienced student to confront their knowledge deficits with appropriate strategies. In system interventions could administer, whenever an error streak occurs in a manner that has been observed and solved several times before and certain resources proved to be key in their solution.

One method of implementing adaptive support is directly related to the error messages. Compilers or interpreters of programming languages encode the error in "cryptic" expressions. The correct interpretation of these messages in some cases requires years of experience. Students without the necessary background knowledge might apply trial and error programming strategies instead of evaluating the compiler outputs systematically. In a future implementation of our framework, we intend to support the students at this step on different levels. There exist some databases providing examples and additional information for specific error messages. Hence, the students are able to earn experiences in a realistic but augmented environment, where the error class is explained by isolated examples, possible solutions are sketched out, and links to further resources are provided (Czaplicki, 2015).

In order to improve the detection of error streaks in further research, we will define and detect more differentiated statuses that allow employing more sophisticated and individually tailored assistance (see also Berges et al., 2016). Those statuses could include the duration of an error streak, the amount of repeated errors, and the meaning of specific errors. Another important part is the counter part of an error streak: success streaks. Whenever the code compiles without errors, several times in succession, we can assume that the user is not satisfied with the way the program is acting. Syntax errors might be absent, but logical and semantic errors are still present. Especially when programming embedded systems, which interact with their surroundings, the process of finding the configuration and values for sensors and actors that are needed to accomplish the given task can be a time and energy consuming part of the whole process. Automatically detecting such situations would expand the scope of application for these methods. While the current state, presented in this paper, allows to help students with little experience that struggle with the basics of programming, the extension of detecting logic and semantic errors would enable the lecturers to offer helpful assistance to more experienced students and advanced students projects, which focus the specific set of skills useful in the context of programming embedded systems (Table 5.4).

Table 5.4 Practical statuses, detection strategies, and error classes

Practical steps and statuses	Representations in logs	Measurements	Corresponding error class
Code compiles without errors	Built success message	Count of successful builds	–
Code compiles with errors	Error messages; describing the error	Count of failed builds	Syntax errors
Code that compiled without errors before, now fails	Error message following a built success message	Probability of a code revision that worked before turning into non-compiling code	Syntax errors
Code that failed before, now compiles	Built success message following an error message	Probability of a code revision that didn't work before turning into compiling code	–
Code compiles; features are functional	Indication: built without errors	–	–
Code compiles; features are not functional	Indication: successive builds without errors	Size and time differences between code versions	Logic and/or semantic errors

This study is a first step to an adaptive remote lab tailored to the needs of the learner. We could show that the perception of extraneous cognitive load as well as the probability of error streaks is relevant for dropout rate. On the basis of our finding, we can now automatically detect learner that got stuck (in either way) and apply interventions suited for the different needs of these learners. We assume a combination of both, explorative analysis of variables which affect the decision to drop out as well as detection of related patterns of user behavior as a promising way for defining and implementing rules for adaptivity in a digital learning environment.

Acknowledgments This work was partially supported by the German Federal Ministry of Education and Research (BMBF, Funding number: 16DHL1033).

References

Al-Qahtani, A. A., & Higgins, S. (2013). Effects of traditional, blended and e-learning on students' achievement in higher education. *Journal of Computer Assisted Learning, 29*(3), 220–234. https://doi.org/10.1111/j.1365-2729.2012.00490.x

Arnold, K. E. & Pistilli, M. (2012). Course Signals at Purdue: Using learning analytics to increase student success. In *Proceedings of the LAK12: 2nd International Conference on Learning Analytics and Knowledge* (pp. 267–270). https://doi.org/10.1145/2330601.2330666

Berges, M., Striewe, M., Shah, P., Goedicke, M., & Hubwieser, P. (2016). Towards deriving programming competencies from student errors. In *Proceedings of the 4th International Conference on Learning and Teaching in Computing and Engineering (LaTiCE'16)* (pp. 19–23). Los Alamitos, CA: IEEE Xplore Digital Library.

Bernard, R. M., Borokhovski, E., Schmid, R. F., Tamim, R. M., & Abrami, P. C. (2014). A meta-analysis of blended learning and technology use in higher education: From the general to the applied. *Journal of Computing in Higher Education, 26*(1), 87–122. https://doi.org/10.1007/s12528-013-9077-3

Bures, E. M., Abrami, P. C., & Amundsen, C. (2000). Student motivation to learn via computer conferencing. *Research in Higher Education, 41*(5), 593–621. https://doi.org/10.1023/A:1007071415363

Chen, O., Kalyuga, S., & Sweller, J. (2017). The expertise reversal effect is a variant of the more general element interactivity effect. *Educational Psychology Review, 29*(2), 393–405. https://doi.org/10.1007/s10648-016-9359-1

Cox, A., & Fisher, M. (2008). A qualitative investigation of an all-female group in a software engineering course project. *Journal of Information Technology Education, 7*, 1–20.

Czaplicki, E. (2015). *Compiler errors for humans*. Rethinking the terminal UX in Elm 0.15.1, http://elm-lang.org/blog/compiler-errors-for-humans

Davis, F., Bagozzi, P., & Warshaw, P. (1989). User acceptance of computer technology - a comparison of two theoretical models. *Management Science, 35*(8), 982–1003. https://doi.org/10.1287/mnsc.35.8.982

de Jong, T., & Lazonder, A. W. (2014). The guided discovery principle in multimedia learning. In R. E. Mayer (Ed.), *The Cambridge handbook of multimedia learning* (2nd ed., pp. 371–390). Cambridge, MA: Cambridge University Press. https://doi.org/10.1017/CBO9781139547369.019

Ferguson, R. (2012). Learning analytics: Drivers, developments and challenges. *International Journal of Technology Enhanced Learning, 4*(5/6), 304–317. https://doi.org/10.1504/IJTEL.2012.051816

Fredericksen, E., Pickett, A., Shea, P., Pelz, W., & Swan, K. (2000). Student satisfaction and perceived learning with on-line courses: Principles and examples from the SUNY learning network. *Journal of Asynchronous Learning Networks, 4*(2), 7–41.

Giesbers, B., Rienties, B., Tempelaar, D., & Gijselaers, W. (2013). Investigating the relations between motivation, tool use, participation, and performance in an e-learning course using web-videoconferencing. *Computers in Human Behavior, 29*(1), 285–292. https://doi.org/10.1016/j.chb.2012.09.005

Heublein, U. (2014). Student drop-out from German higher education institutions. *European Journal of Education, 49*(4), 497–513. https://doi.org/10.1111/ejed.12097

Horton, D., & Craig, M. (2015). Drop, fail, pass, continue: Persistence in CS1 and beyond in traditional and inverted delivery. SIGCSE'15. In *Proceedings of the 46th ACM Technical Symposium on Computer Science Education* (pp. 235–240) Kansas City, MO. https://doi.org/10.1145/2676723.2677273

Isen, A. M., & Reeve, J. (2005). The influence of positive affect on intrinsic and extrinsic motivation: Facilitating enjoyment of play, responsible work behavior, and self-control. *Motivation and Emotion, 29*, 295–323. https://doi.org/10.1007/s11031-006-9019-8

Isleib, S., & Heublein, U. (2017). Ursachen des Studienabbruchs und Anforderungen an die Prävention. Empirische Pädagogik, 30. Jahrgang (Heft 3/4), 513–530. Landau in der Pfalz, Germany: Verlag Empirische Pädagogik.

Joo, Y. J., Lim, K. Y., & Kim, E. K. (2011). Online university students' satisfaction and persistence: Examining perceived level of presence, usefulness and ease of use as predictors in a structural model. *Computers & Education, 57*(2), 1654–1664. https://doi.org/10.1016/j.compedu.2011.02.008

Kalyuga, S. (2007). Expertise reversal effect and its implications for learner-tailored instruction. *Educational Psychology Review, 19*, 509–539. https://doi.org/10.1007/s10648-007-9054-3

Kalyuga, S. (2011). Cognitive load theory: How many types of load does it really need? *Educational Psychology Review, 23*(1), 1–19. https://doi.org/10.1007/s10648-010-9150-7

Kori, K., Pedaste, M., Leijen, Ä., & Tõnisson, E. (2016). The role of programming experience in ICT students' learning motivation and academic achievement. *International Journal of Information and Education Technology, 6*(5), 331–337. https://doi.org/10.7763/IJIET.2016.V6.709

Kori, K., Pedaste, M., Tõnisson, E, Palts, T., Altin, H., Rantsus, R., … Rüütmann, T. (2015). First-year dropout in ICT studies. In *Proceedings of IEEE Global Engineering Education Conference* (pp. 444–452). https://doi.org/10.1109/EDUCON.2015.7096008

Law, K. M. Y., Lee, V. C. S., & Yu, Y. T. (2010). Learning motivation in e-learning facilitated computer programming courses. *Computers & Education, 55*(1), 218–228. https://doi.org/10.1016/j.compedu.2010.01.007

Legris, P., Ingham, J., & Collerette, P. (2003). Why do people use information technology? A critical review of the technology acceptance model. *Information & Management, 40*, 191–204. https://doi.org/10.1016/S0378-7206(01)00143-4

Leppink, J., Paas, F., van Gog, T., van der Vleuten, C. P. M., & van Merriënboer, J. J. G. (2014). Effects of pairs of problems and examples on task performance and different types of cognitive load. *Learning and Instruction, 30*, 32–42. https://doi.org/10.1016/j.learninstruc.2013.12.001

Leutner, D. (2002). Adaptivität und Adaptierbarkeit multimedialer Lehr- und Informationssysteme. In L. Issing & P. Klimsa (Eds.), *Information und Lernen mit Multimedia und Internet* (pp. 115–125). Weinheim, Germany: Beltz.

Leutner, D. (2014). Motivation and emotion as mediators in multimedia learning. *Learning and Instruction, 29*, 174–175. https://doi.org/10.1016/j.learninstruc.2013.05.004

Levy, Y. (2007). Comparing dropouts and persistence in e-learning courses. *Computers & Education, 48*, 185–204. https://doi.org/10.1016/j.compedu.2004.12.004

Liaw, S.-S. (2008). Investigating students' perceived satisfaction, behavioral intention, and effectiveness of e-learning: A case study of the Blackboard system. *Computers & Education, 51*, 864–873. https://doi.org/10.1016/j.compedu.2007.09.005

López-Pérez, M. V., Pérez-López, M. C., & Rodríguez-Ariza, L. (2011). Blended learning in higher education: Students' perceptions and their relation to outcomes. *Computers & Education, 56*, 818–826.

Marks, R. B., Sibley, S. D., & Arbaugh, J. B. (2005). A structural equation model of predictors for effective online learning. *Journal of Management Education, 29*(4), 531–563. https://doi.org/10.1177/1052562904271199

Mayer, R. E. (2014). Incorporating motivation into multimedia learning. *Learning and Instruction, 29*, 171–173. https://doi.org/10.1016/j.learninstruc.2013.04.003

Melis, E., Andrès, E., Büdenbender, J., Frischauf, A., Goguadze, G., Libbrecht, P., … Ullrich, C. (2001). ActiveMath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education, 12*, 385–407.

Moreno, R. (2006). When worked examples don't work: Is cognitive load theory at an impasse? *Learning and Instruction, 16*(2), 170–181.

Narciss, S., Proske, A., & Koerndle, H. (2007). Promoting self-regulated learning in web-based learning environments. *Computers in Human Behavior, 23*(3), 1126–1144. https://doi.org/10.1016/j.chb.2006.10.006

Paas, F., Tuovinen, J. E., van Merriënboer, J. J., & Darabi, A. A. (2005). A motivational perspective on the relation between mental effort and performance. Optimizing learner involvement in instruction. *Educational Technology Research and Development, 53*(3), 25–34.

Papamitsiou, Z., & Economides, A. (2014). Learning analytics and educational data mining in practice: A systematic literature review of empirical evidence. *Educational Technology & Society, 17*(4), 49–64.

Park, B., Plass, J. L., & Brünken, R. (2014). Cognitive and affective processes in multimedia learning. *Learning and Instruction, 29*, 125–127. https://doi.org/10.1016/j.learninstruc.2013.05.005

Park, J.-H., & Choi, H. J. (2009). Factors influencing adult learners' decision to drop out or persist in online learning. *Educational Technology & Society, 12*(4), 207–217.

Plass, J. L., Moreno, R., & Brünken, R. (2010). *Cognitive load theory*. Cambridge, MA: Cambridge University Press. https://doi.org/10.1017/CBO9780511844744

Schneider, M., & Preckel, F. (2017). Variables associated with achievement in higher education: A systematic review of meta-analyses. *Psychological Bulletin, 143*(6), 565–600. https://doi.org/10.1037/bul0000098

Schnotz, W., & Kürschner, C. (2007). A reconsideration of cognitive load theory. *Educational Psychology Review, 19*(4), 469–508. https://doi.org/10.1007/s10648-007-9053-4

Schulmeister, R. (2017). *Presence and self-study in blended learning*. eLeed, 12, urn:nbn:de:0009-5-45027. https://eleed.campussource.de/archive/12/4502

Siegmund, J., Kästner, C., Liebig, J., Apel, S., & Hanenberg, S. (2014). Measuring and modeling programming experience. *Empirical Software Engineering, 19*(5), 1299–1334. https://doi.org/10.1007/s10664-013-9286-4

Sweller, J., Ayres, P., & Kalyuga, S. (2011). *Cognitive load theory*. New York, NY: Springer. https://doi.org/10.1007/978-1-4419-8126-4

Talton, J. O., Peterson, D. L., Kamin, S., Israel, D., & Al-Muhtadi, J. (2006). Scavenger hunt: Computer science retention through orientation. In *Proceedings of the 37th SIGCSE Technical Symposium on Computer Science Education* (pp. 443–447).

van Seters, J. R., Ossevoort, M. A., Tramper, J., & Goedhart, M. J. (2012). The influence of student characteristics on the use of adaptive e-learning material. *Computers & Education, 58*, 942–952. https://doi.org/10.1016/j.compedu.2011.11.002

Venkatesh, V., & Davis, F. D. (2000). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science, 46*, 186–204. https://doi.org/10.1287/mnsc.46.2.186.11926

Venkatesh, V., Morris, M. G., Davis, G. B., & Davis, F. D. (2003). User acceptance of information technology: Toward a unified view. *MIS Quarterly, 27*, 425–478. https://doi.org/10.2307/30036540

Watson, C., & Li, F. W. B. (2014). Failure rates in introductory programming revisited. In *Proceedings of the 2014 Conference on Innovation Technology in computer science education (ITiCSE'14)* (pp. 39–44). New York, NY: Association for Computing Machinery (ACM).

Wu, J.-H., Tennyson, R. D., & Hsia, T.-L. (2010). A study of student satisfaction in a blended e-learning system environment. *Computers & Education, 55*(1), 155–164. https://doi.org/10.1016/j.compedu.2009.12.012

Zug, S., Hawlitschek, A., & Krenz, T. (2017). *What are the key features of future Remote Labs? A critical evaluation of an existing one*. In FDIBA Conference Proceedings, http://www.elab.ovgu.de/elab_media/_users/hawlitsc/Paper_FDIBA_eLab/FDIBA_eLab-p-90.pdf