Universität Bremen

Fachbereich 03: Informatik & Mathematik

Masterarbeit

im Studiengang Informatik

An adaptive E-Learning System featuring an individualized difficulty adjustment process for mathematical word problems

zur Erlangung des akademischen Grades
Master of Science

Autor: Mathias Eggerichs
MatNr. 2788839

Version vom: June 6, 2019

1. Betreuer: Prof. Dr. Andreas Breiter
2. Betreuer: Prof. Dr. Karsten Detlef Wolf

Acknowledgements

Throughout the process of writing this thesis and during the development phase of this E-learning Software, I have received a great amount of support and assistance. I want to thank Prof. Dr. Andreas Breiter for being my main adviser, whose dedication to consulting me was an integral part in helping me successfully complete this thesis. The same praise goes to Irina Zakharova, who helped me a great deal with the participatory methods that were used. Additionally, the Information Management working group of the University of Bremen was always open and happy to give valuable feedback. I also want to thank Prof. Dr. Karsten Detlef Wolf for being my secondary adviser. Lastly I want to express gratitude to the University of Bremen, as well as to the Faculty 3 - Computer Science and Mathematics for allowing me to work on such an exciting topic.

I would also like to thank my friends and family, who gave me great mental support and were always very supportive.

Abstract

In this master thesis participatory software development methods were used to develop an adaptive learning platform for pupils that offers the creation of mathematical word problems of which the task difficulty is individually adjusted to the pupil that is using the software. Freely available structured information in the semantic web is used to curate information that are relevant for the creation of realistic mathematical word problems. The software offers a manual and automatic approach to determine the individual difficulty settings for a student. The generation of a task is being supported by a flexible component based solving system so that the difficulty settings and preferences can be optimally utilized by having a task that is easily split into sub-tasks, each having a variable degree of difficulty. The displayed user interface separates the information of a given task with the actual wording of the word problem allowing the student to choose what paths to take to solve the questions, which furthermore prevents unwanted hints being provided to the student. The behaviour of the student is then analysed and the resulting data is used to update the student model that leads to the individual difficulty settings. The expert in the evaluation assessed that the software would be valuable to use as a supplementary learning experience to the already established catalogue of E-Learning Softwares. Especially the manual adjustment of difficulty parameters was seen as valuable. The evaluation revealed problems in the initial comprehensibility of how to use the solving system.

Contents

1 Introduction

In a rapidly growing E-Learning industry, schools are slowly starting to use E-learning supported tools to accommodate their classroom experience. Especially Intelligent Tutoring Systems are being used by students to either learn or practice new concepts that they have learned in school. According to Mohammadi, Ghorbani, and Hamidi (2011) E-learning increases the engagement and motivation of a student. It is important to note that the positive effect of E-Learning depends on certain Critical Success Factors like *Content Completeness* and *Easy Navigation* (Naveh, Tubin, & Pliskin, 2012).

In an Intelligent Tutoring System, an artificial tutor takes the place of a teacher. It models a classroom experience, including the student model, the domain model and the tutoring model, which represents a teacher. Information about the student's behaviour is logged in order for the intelligent tutoring to make decisions about the problems that are being presented and in which degree the tutor has to intervene. There are numerous ITS system that are used to support the learning experience of students, such as *Bettermarks* (bettermarks GmbH, 2018), *ActiveMath* (Melis, Andrès, & et al., 2001), *DiSNeT* (Rosic, Glavinic, & Stankov, 2005), *eTeacher* (Schiaffino, Garcia, & Amandi, 2008) and many more. Bettermarks will be frequently mentioned in this thesis, because the software is used in a project lead by the University of Bremen in connection with a school in Bremen. It offers students and teachers the possibility of practising concepts that were taught in school. Based on the performance of the student, further practice in areas where the student did not perform well are recommended. In addition to that the offered tasks are based on the current curriculum in schools. Furthermore ActiveMath will be talked about in more detail, because it features a modular user interface, which will be expanded upon on in the proposed new E-Learning System.

The amount of available practice tasks is limited by the manual work a teacher or an expert has to invest into creating them. Even then the tasks are not individualized for each and every student because it would demand too much effort and time to do so. To tackle these problems an E-Learning Software was developed that creates tasks with individualized difficulty settings and a modular component-based solving system.

1.1 Problem

Three core issues with the currently available Intelligent Tutor Systems were identified that this thesis aims to address. The first problem lies with the availability of varied practice tasks for students. The second problem is about the difficulty scaling of math tasks that are being presented to the students. Lastly the problem of students not being able to fully express their conceptual knowledge because they have to follow a certain way of solving a task will be addressed.

Availability of varied practice tasks: While some Intelligent Tutoring Systems (ITS) like Bettermarks support the generation of varied tasks by using a template-based task generation and by mixing up the used values in these tasks, most ITS require previously manually created tasks that the students can solve. Naturally the amount of available practice tasks are limited by manual labour. By varying up the structure and content of a task to a major degree, it encourages the student to not plainly rely on a strict procedure in solving the task or by just remembering an error they made in the past in a similarly stated math question, but to conceptually understand the problem at hand and find a specific solution to solve it. A similar remark was made by an expert teacher that I concluded an interview with to get an understanding of the environment in which the software could be used eventually as well as to get a picture of the needs of the teachers using E-Learning Softwares (see chapter 3.7).

Doing this in an automated way without the need of manual input would lead to an essentially infinite amount of available practice tasks. The book *Theory of authentic task situations* (Palm, 2009) also shows that questions that are related to real world objects and feature realistic outcomes are beneficial for the user's learning process. An interesting side effect is also that real world information is being learned as a by-product. The expert interview revealed that teachers try to incorporate the student's hobbies into the tasks given, so that the students can better relate to the task. Furthermore the National Research Council and Institute of Medicine (2004) shows that making mathematics personally relevant led to significant gains in mathematics achievements.

This thesis aims to address the problem of a finite pool of available tasks by creating an E-Learning Software that automatically generates math tasks that are varied in structure and content while referencing real world objects that students can relate to on a personal level (see section 3.4.3).

Difficulty scaling of math tasks: There is an obvious need for personalisation of tasks that are being given to students for practice purposes. Students have differing needs when it comes to the difficulty of the task, how the task is structured and how it is to be solved. The difficulty of a task can either be found in the conceptual difficulty or in the procedural difficulty. The concepts that a student learns are tied to the current curriculum and school class the student is enrolled in. Even within these concepts there are obvious differences in task difficulty that need to be adjusted based on the student. WolframAlpha (wolframAlpha LLC, 2018) for example allows three different difficulty settings for their automatic creation of math tasks. However these settings only influence the procedural difficulty.

Results of the expert interview, which are laid out in more detail in section 5, show the desire of teachers to accommodate to each and every pupil to their best ability, which is not possible because of time constraints. Furthermore, categorizing pupils into

predefined categories of skill like "bad", "average" and "good" leads to a stigmatizing effect. Another result of this categorization is that there are outliers that do not fit into these predefined categories. Additionally, tasks are being assigned to these groups of skill, which results in the task being constructed in a way that every part of it fits the desired difficulty level. It is certainly possible that a pupil excels at one particular skill that is needed to solve the task at hand but has troubles with another one. This process leads to either an overburden for the student or an underchallenge in parts of the task.

This thesis tries to address this problem by creating an E-Learning Software that dynamically adjusts difficulty settings for both conceptual and procedural difficulties. The difficulty of every sub-task will be able to be independently adjusted to accommodate to the skill level of the student (see section 3.4.4).

A constrained way of solving a task: In ITS, such as bettermarks (bettermarks GmbH, 2018), the student is not given absolute freedom in which way the user likes to approach the math task. The student is being supplied with the incremental steps that are needed to be completed so that the task can be solved. in the case of CTAT (Carnegie-Mellon-University, 2018) a teacher is able to modularly choose a rough outline the student has to follow. **This thesis addresses the problem of a student not being able to show his or her conceptual knowledge skills by creating an E-Learning Software that gives freedom in the way the student wants to solve the task at hand.** (see 4.6)

1.2 Goal

The motivation behind writing this thesis is the goal to combine the knowledge that has been researched in the field of *Natural Language Question Generation* with an Intelligent Tutoring System that takes advantage of the automatic creation of math tasks. Using an automated system enables the creation of adaptive tasks that are dynamically adjusted based on the skill and learning preferences of the student or teacher and to offer an optimal challenge catered to the individual student thus increasing the learning success (Yerkes & Dodson, 1908). Additionally, creating the tasks automatically results in a very detailed model of the task in the system, which furthermore leads to a finely grained question solving system for the student.

The goal of this master thesis is to develop a fully functional working prototype of a learning system that shows the proposed techniques of solving the laid out problems adequately in a way that a usability test can be conducted successfully to reveal the degree of suitability for a realistic school environment of secondary schools, as well as room for improvements.

Because of the limited scope and time constraints of the thesis, the focus was on solving the identified problems with a limitation on supported mathematical concepts that are taught in the 5th grade up to the 8th grade of the *Sekundarstufe 1*, namely *General Number Handling*, *Area Measurement* and *Percent Calculation*. Those concepts were chosen because they represent a wide range of skills that are needed to solve mathematical word problems that include them. Furthermore these concepts allow to be combined together in a way that shows off the features that were planned to be implemented in the learning software. Further elaboration behind the reasons of choosing these concepts will be given in section 3.4.4.

Even though these concepts only represent a small part of the whole curriculum, they suffice to show what the software is aiming to accomplish. Further work will be required to offer the student a full classroom experience by implementing the whole curriculum and also by adjusting the difficulty parameters for each new concept that is introduced. An expansion to include the whole curriculum is not a trivial matter, however doable.

At this point it is important to point out what this thesis is not. The aim of this project was not to develop a production ready software that can be used in a realistic environment once the development is done, but rather to implement a fully functional prototype that best shows the new ideas that were implemented to tackle the problems that were listed. The goal was to show the possibilities and positive impacts the new features can deliver for a teacher to better be able to provide a good learning experience for students. It was to be expected that the platform shows usability errors in the way it is presented, as well as missing features such as no compatibility with existing softwares like itslearning. However an explanation will be given on how a transition of this software into a production ready software could be carried out.

1.3 Research questions

- How can freely available information in the semantic web be categorized and modelled so that it can be used to generate word problems that connect multiple math concepts together while referencing real world objects?
- How can skill representations of students be utilized to determine each individual level of difficulty of all sub-tasks that together make up the generated word problem?
- Will the generated word problems resemble currently available practice pool tasks from a validity standpoint and show variety on long-term use?
- How can a question solving process be designed in a way that it supports the use of automatic generated word problems while allowing the student to solve the tasks in an unrestricted manner?

- What design choices of the question solving system are being enabled by the use of an automated generation of questions?

1.4 Overall Approach

The overall approach to the development of the prototype was to utilize participatory methods to gain an understanding of the environment the software would operate in. For that an interview with a teacher in the city of Bremen was conducted to gather valuable information pertaining the use of E-Learning Software in a school environment, as well as the operational needs of the teacher to be able to use such a software. The results of the guided interview were then incorporated into the development process of the software. Individual difficulty adjustments being a major focus of the thesis, a compendium of literature researching task difficulty was gathered. The information was then used to create a system that automatically adjusts the difficulty levels of each and every sub-task in a mathematical word problem that is being generated. To achieve question variety the software mixes up the generation of tasks by choosing from a pool of suitable task-templates to iteratively construct the word problem, as well as using real world objects as content of the generated task. To solve the task, a component based solving system was developed that enables the student to have freedom in how the task is to be solved. To connect the skill model of the student with the generation of individualized tasks, the software analyses the behaviour of the student whenever an action can be observed to update the model. Lastly the prototype was tested by the mentioned expert in an effort to judge the usability of the prototype, the results of which were analysed with an heuristical approach.

2 Theoretical Background

2.1 Intelligent Tutoring Systems

The fundamental purpose of an Intelligent Tutoring System is to provide the learner with customized instructions and feedback (Psotka, Massey, & Mutter, 1988). The traditional architecture of an ITS consists of the four components or models *domain*, *student*, *tutor* and a *user interface*. The domain model describes the concepts, rules and problem solving strategies that are relevant to a specific problem or task. The student model tries to accurately model the student that uses the system by assessing the students mastery or other relevant properties. The remaining part of an ITS, the tutoring model, supports the learner via hints or other helpful advice. Besides that, problem sequencing and courseware planning is also part of an ITS (Nkambou, Bourdeau, & Mizoguchi, 2010, pp. 3–5).

2.1.1 Domain model

A popular approach to modeling the domain is to use a rule-based cognitive modeling process in order to understand the student's thinking and problem solving capabilities and approaches (Nkambou, Bourdeau, & Mizoguchi, 2010, pp. 6–7). It accompanies the learner while solving a task by giving personalized support. The support can for example lie in error correction and detection or in helpful hints to help the student solve the task at hand. Another characteristic of a *Cognitive Tutor* is the logging and monitoring of the user's behaviour.

Constraint-Based Modeling(CBM) is another approach that aims to overcome some existing problems of the *Rule-based Cognitive Modeling* process. It gives the student freedom to solve tasks however he or she wants to by only providing constraints on how the solution of a problem shall look like, without constraining the student on how to solve the problem (Mitrovic, 2010). A CBM represents the solution space in terms of abstractions with solution states that require the same reaction from the tutor grouped in equivalence classes (Nkambou, Mizoguchi, & Bourdeau, 2010). An example of a tool that implements this kind of tutor is CTAT (Carnegie-Mellon-University, 2018). It allows a teacher to programmatically create math tasks and a supporting tutor that uses a defined ruleset to help the user depending on which path the user takes.

2.1.2 Student model

The goal of modeling a student is to gather as much accurate information about the user of the ITS which includes cognitive and affective states and the evolution of their learning process (Nkambou, Bourdeau, & Mizoguchi, 2010, pp. 3–4). The information of the student model should then influence strategical decisions on the tutoring path (Wenger, 1987). A popular approach to update the mastery of a student in certain skills is the *Bayesian Knowledge Tracing* which will further be explained in section 2.3.

2.1.3 Tutoring model

The tutoring model describes how and to which degree the system interacts with the students. It has to decide when to intervene and how to help the student (Nkambou, Bourdeau, & Mizoguchi, 2010, pp. 4–5). It is also called the *Teacher model* because it represents the functions that usually a teacher has (Dašić, Dašić, Crvenković, & Šerifi, 2016).

2.1.4 Adaptive ITS

Adaptive Intelligent Tutoring Systems offer content that is adapted to the user's knowledge, goals and preferences by maintaining a model of the user (Phobun & Vicheanpanya, 2010). In this thesis an Adaptive Intelligent Tutoring System was created in a way

that tasks are created based on certain parameters in an automatic fashion. Therefore the following lists adaptive ITS and their characteristics and how they differ from the proposed system in this thesis.

DiSNeT (Rosic et al., 2005) The Distributed Semantic Networks Tutor's goal is to create personal agents for specific users that share a distributed knowledge space. The system describes three types of users for the system: students that are using the ITS to learn concepts by using the knowledge base, experts that create the expert knowledge bases and tutors who select and structure courseware. The DiSNet assists every type of users in their intended goal. The system for example helps experts to publish and register new knowledge bases or by notifiying tutors if new knowledge bases of interest have been added. Courseware is being created by tutors with parameterized metadata tagging of the specific tasks.

eTeacher (Schiaffino et al., 2008) eTeacher uses a Bayesian model to detect the learning style of a student by observing how the user interacts with the ITS, such as what kind of learning material and excercises are preferred and how the student interacts in a forum or chats. To model the student they use the model proposed by Felder (1988), that categorizes the students as intuitive or sensitive, global or sequential, visual or verbal and active or reflective. This leads to a distinction between visual, active and sequential learners which results in different learning material in the course and how the intelligent tutor helps the student while solving a task. The system recommends coursework for the student based on the information it has on the student, which includes recent exam results, the current stage of the curriculum the student is at and the preferred learning style of the student.

ActiveMath (Melis et al., 2001) ActiveMath is a generic web-based learning system that dynamically generates coursework for the student based on the student's goals, preferences, capabilities and knowledge. Initially the user has to complete a self assessment asking about the current student's knowledge level. A user can choose goal concepts and scenarios for the course generator. The system then queries a knowledge base to find out what mathematical concepts are needed to be mastered in order to master the goal concept. ActiveMath bases their modeling of a student on the **Bloom Taxonomy Levels** such as *Knowledge*, *Comprehension* and *Application* (Melis et al., 2001, page 399). Behaviour of the user is logged and stored in the respective categories. Mastery level of each concept with each sub-concept that depend on it are stored. The system also presents learning material for the student to use. Using these learning materials furthermore triggers an update to the likelyhood percentage of the student mastering the respective concept.

Andres, Heeren, and Jeuring (2013) implemented a modular solving system in ActiveMath that shows a simple prototype of a highly interactive and component based interface that enables an individualised way of adding and subtracting fractions.

The Smart Tutor (Gamalel-Din, 2018) The Smart Tutor introduces a web-based Intelligent Tutoring System that uses *Case-based Reasoning* to determine a specific approach for the student to learn a particular concept. The course strategy tries to find coursework that is both suitable for the student and also suits the teachers ability to teach a certain concept. The system features concepts that are broken down into smaller concepts that are linked with each other serving as prerequisites. The student model is categorized in *Background*, *Knowledge* and *Skills*, which are updated based on the students behaviour. A course is designed by either the student or an instructor by defining goal concepts. Taking the background and skills of the student into account, the system then identifies an optimal coursework by finding prerequisite concepts that the student needs to learn in order to master the goal concept.

A unique feature of The Smart Tutor is that not only the student is being modeled but it also features a *Teacher Capability Model*.

Bettermarks (bettermarks GmbH, 2018) Besides offering learning material for the supported concepts that are linked to the schools curriculum, Bettermarks offers the student the possibility to practice concepts that he or she have learned in school. It divides procedural knowledge into subcategories. For example the addition of numbers include multiple different task structures and skills the user can show a different amount of expertise in. Adding two positive values together or finding the sum of one negative and one positive value could lead to a different assessment and are therefore different skill categories. Doing this allows the system to recommend learning areas that the user lacks proficiency in. Bettermarks also gives the teacher the ability to see the proficiency of each student in each category of expertise.

itslearning (itslearning, n.d.) itslearning differs from the other listed Intelligent Tutoring Systems in the way that it does not directly supply the student with tasks but rather acts as a *Learning Management System*. A main feature is the possibility of connecting multiple third party applications into it, supporting the interconnection of various tools like planner, courses and assessment for curriculum management. It also features reporting and analytics to track the student's behaviour. itslearning also features a vast network of communication within a school environment, be it between teachers, students or parents. In 2013 the city of Bremen and Bremerhaven initiated a call for bids for an appropriate E-Learning Software that the schools in the respective cities can start to use, in which itslearning was eventually accepted. Hence why this particular software is very relevant to the creation of the ITS proposed in this thesis

because it would co-exist in a school environment together with itslearning.

The tasks in these systems have profound methods to recommend the user learning areas he or she lacks proficiency in. Especially Bettermarks features a thorough categorization of arithmetic concepts broken down into smaller concepts. Students solving tasks in said system are being given hints in the form of how the answer has to be structured. All these systems use finite problem pools that they use to construct their questions, which will lead to repetition of questions given to the user. The difficulty settings are also mostly static in a way that the tasks are categorized based on a few difficulty settings. The proposed ITS in this thesis will differ from these systems by providing an automatic creation of tasks with individual difficulty settings on a conceptual and procedural level while expanding on the idea of ActiveMath to provide a component based solving of the tasks by not giving the user structural hints on how the task has to be solved.

2.2 Critically assessing E-Learning

The rise of E-Learning necessitates a comparison of the up- and downsides of using E-Learning Systems for educational purposes. Arkorful and Abaidoo (2014) analysed the advantages and disadvantages of E-Learning being used in higher education. There are certain undeniable upsides, especially when it comes to a higher degree of individualization by adjusting the content dynamically based on the learners interests. Additionally it is cost effective, as it helps to compensate for a possible lack of staff. E-Learning Systems also allow the student to work on their educational deficiencies on their own and at their own desired pace. Disadvantages include the lack of interaction on a personal level. Vrasidas (2004, p. 911) emphasizes that even though technology is moving to the centre of educational transactions, that does not mean that the personal interaction between students and teachers should disappear. Furthermore according to Arkorful and Abaidoo (2014) the effective use of E-Learning Systems require the softwares to offer good and adequate support and guidance for the student. This requirement is more so important when the student uses the software remotely. Another problem could be the possibility of the student cheating or in general not engaging with the software as they are supposed to. Even if the E-Learning Software is used in a classroom with teachers available to give oversight, it still is a detriment by requiring the teacher to manually interfere. Vrasidas (2004, p. 912) offers the advice that problems with using E-Learning Systems could be combated by keeping in mind in which environment and context the software works in and that it is the job of the developer to create a software that is catered to the diverse audience. Especially when it offers poor usability, customizability and reusability, the positive effects of E-Learning diminishes.

2.3 Assessments and student model

A functional E-Learning System that aims to offer highly individualized word problems to students requires a representation of the student within the system. The model itself requires data to first initialize the model and afterwards to continuously update it. The field of *Learning Analytics* (LA) deals with these problems. It can be defined as "Learning analytics is the measurement, collection, analysis, and reporting of data about learners and their contexts, for the purposes of understanding and optimizing learning and the environments in which it occurs."(LAK'11, 2010)

There are multiple avenues for analysis and data mining that are relevant in an educational setting that can span from individual classroom experiences to data of an international level (Perrotta & Williamson, 2018).

Johnson, Smith, Willis, Levine, and Haywood (2011) describes LA as being a tool to tailor education to individual students more effectively by using new advancements in data mining, interpretation and modelling.

Bienkowski, Feng, and Means (2012) show how LA can be used in adaptive Learning Systems. There are multiple components that are necessary to work together in order for LA to work effectively, such as a content management system that delivers the student individualized subject content, a student learning database that stores the input and behaviour of the student and a predictive model that makes predictions about future performances, such as how the student might fare in an upcoming task. Furthermore an adaption and intervention engine is needed to regulate the content delivery based on what predictions were made and to allow teachers to interfere.

They also show various different LA applications that can be used to gather information about the student such as:

- User knowledge/behaviour/experience modelling
- User profiling
- Domain modelling
- Trend analysis
- Learning component analysis
- Adaptation and Personalization

Modelling the user is done in order to create a model of the student's mastery, be it specific skills or low-level expertise. The student operates in a domain, hence it is important to also model the domain by analysing in which environment the student is working in.

The data that is needed for an adaptive and personalized experience depends on the actual current situation a student might find themselves in while using the E-Learning System.

Brusilovsky (1994) showed various approaches on how specific subject knowledge can be represented. The *Scalar Model* simply assigns a value to how good a student is at a certain skill. The *Overlay model* assumes that the relevant course material can be subdivided in finer subjects that the student can show mastery in. Another way of storing relevant information about the student is to use an *Error Model*, storing information about misconceptions or mistakes by the students. Additionally to reflect the domain connection to the student, genetic models represent the process of the evolution of mastery that a student goes through (Brusilovsky, 1994).

After deciding what kind of information needs to be stored about the student and the environment, the actual model of the student has to be created. Multiple approaches have been used to do so. Froschl (2005) gives an overview about various methods, including *Machine Learning Methods*, *Bayesian Methods*, *Overlay Methods*, *Stereotype Methods* and *Plan Recognition*.

Using Machine Learning to create student models has the added benefit of less manual effort being necessary to create these models. An example of that is Li, W. Cohen, Koedinger, and Matsuda (2011). They created an intelligent agent that uses automatic discovery of student models. They showed that the agent achieved better accuracy than human coded student models.

A widely utilized method to model the skills of a student is *Bayesian Knowledge Tracing* as seen in Butz, Hua, and Maguire (2008), Yamna, Mellouli, and Wuillemin (2010), Melis et al. (2001) and Schiaffino et al. (2008). It was introduced for the first time in Corbett and Anderson (1994). It is based on a *Hidden Markov Model* (Rabiner, 1990) with four parameters for every relevant skill of the student. The two learning parameters consist of *Initial Learning*, which describes the probability of the student having mastered a skill and *Acquisition*, which represents the probability of a student transitioning from an unlearned state to a learned one. Furthermore there are two performance parameters *Guess* and *Slip*. Guess describes the probability of the student guessing the correct answer even though the student has not mastered the skill yet. The Slip probability shows the likelihood of a student mastering a skill but still making an error. The mastery parameter is constantly updated based on the observations of the student practising a task. According to R. S. J. d. Baker, Corbett, and Aleven (2008), setting the Guess and Slip parameters based on the context is preferable to just having a setting that is the same for all kinds of skills. Furthermore Yudelson, Koedinger, and Gordon (2013) came to the conclusion that introducing another parameter, the learning speed of a student, leads to an increase in the predictive power of the model. Other models adapted the Bayesian Knowledge Tracing by including time between

attempts (Qiu, Qi, Lu, Pardos, & Heffernan, 2011) and Item Difficulty (Khajah, Huang, González-Brenes, Mozer, & Brusilovsky, 2014).

The Overlay Method was used by (Melis et al., 2001). Their system uses a knowledge base that divides mathematical concepts in smaller concepts, as well as storing dependencies between concepts. The behaviour of the student regarding learning new materials influences the respective mastery level.

(Grubišić, Stankov, & Žitko, 2013) used a Stereotype model approach for an adaptive E-Learning System by using the Bloom's knowledge taxonomy to define stereotypes. Users of the system are being assigned stereotypes like "Novice", "Beginner" and so on for each category of the taxonomy.

These models have to be initialized on first use. Depending on what kind of model is used, there are differing requirements of the data. Machine Learning Methods and Bayesian Networks usually require large amount of data so that predictions can be accurate. Other models that focus more on manual input can be used with less available data. According to Self (1994) there are two ways of initializing a student model, by explicit questioning or by default assumptions. The former is self-explanatory and contains asking questions that seem relevant to the student model such as inquiring about the self-assessment of certain skills. Depending on how much information is available about the student, default assumptions can be made, such as assuming that the student has certain stereotypical hierarchies that describe what kind of knowledge the student already likely has.

Getting the required data to update these models requires information that is gained during the student's use of the E-Learning System. Barr (1979, p. 622) shows that the student himself is the primary source of information that is needed in order to update the student's model. Data can be gathered by observing the student using the system and logging the behaviour and performance. The specific process of updating the model depends on what kind of model is used. A machine learning algorithm can use the new data to adjust its prediction by using newly found parameters. Bayes Networks can adjust the probability of the student mastering a certain skill.

2.4 Automatic creation of math tasks

Automated question generation is a field well researched and is used to generate math tasks. Papasalouros, Kanaris, and Kotis (2008) generates multiple choice questions by using ontologies to extract their data from. Coniam (1997) is using *Corpus Word Frequence Data* to automatically create cloze texts to improve the student's vocabulary knowledge. Mathematics is especially suitable for the automatic generation of tasks. WolframAlpha (wolframAlpha LLC, 2018) creates tasks automatically by changing up the values that are being used in the mathematical operations. Because this thesis

focuses on the generation of math tasks and especially mathematical word problems, the work of Sandra Williams (Williams, 2011) is relevant. She created a prototype to create mathematical word questions by querying the semantic web and ontologies. Furthermore she hypothesizes potential difficulty adjustments that are based on the syntax of the questions (Williams, 2011, page 62).

Personalized Mathematical Word Problem Generation (Polozov et al., 2015) employs an automated system of creating mathematical word problems that are highly individualized regarding the non-mathematical content of the question. Pupils can decide which characters are present and how they relate to each other in the little story that encompasses the math task. The book *Measuring What Counts: A Conceptual Guide for Mathematics Assessment* (National Research Council, 1993) shows diminishing returns in emotional responses to a math task, if the task has less novelty, which results in the need of the creation of nonroutine tasks. Furthermore it explains that good education provides opportunities for students to connect what is being learned to their prior knowledge and that assessments should examine whether students have managed to connect the concepts they have learned.

2.5 Task Difficulty

Before analysing what the current knowledge regarding task difficulty entails, it is important to explore what the purpose of tasks being a certain degree of difficult is. The general goal when an E-Learning System is created that contains a Student Model describing the educationally relevant traits, is to measure as accurate as possible how well developed the trait is. In the following this is called a latent trait (F. B. Baker, 1985, p. 13). The traits of a student and the difficulty of tasks have to be connected in some way. *Item Difficulty* and *Item Response Theory* are two models trying to assign specific tasks with numerical or probabilistic values regarding their difficulty.

Item Difficulty: The difficulty of tasks can simply be inferred by looking at the ratio of students successfully solving the task compared to how many attempts at solving were made overall. The result is a number ranging from 0%-100% (University of Washington, n.d.). There is no fixed desired Item Difficulty however and it depends on the exact task that is being presented. The University of Wisconsin shows a few desired Item Difficulties that differs based on the amount of choices the student has in a multiple choice test. It varies from 75% Item Difficulty when there are only 2 choices to make to 60% Item Difficulty for 5 alternative multiple choices (University of Wisconsin, n.d.-a). The effectiveness of a task can furthermore be analysed by the discrepancy in solving ability by two groups of students of which one group has mastered the coursework while the other has not. The bigger the difference in the so called *Item Discrimination* value is, the better the task is at evaluating the student's ability.

Item Response Theory: The IRT picks up the usage of latent traits that students have to evaluate how good the ability of a student in a certain category is. The basic premise is that the performance of a student in a test can be predicted by certain latent traits that are relevant to the item (Hambleton, 1990). Furthermore the relationship between the ability of the student to solve a certain task and the latent trait that is the engine behind the ability can be described by an *Item Characteristic Curve*. In general students that have a higher score on the traits are more likely to solve the item correctly. Seeing that there is a predictive element, the task that is being assigned can be chosen based on the probability of the student solving the task. If the probability is very high, the task might be too easy and vice versa. Using the theory of Item Difficulty bears the problem that the data that was used to collect the Item Difficulty can theoretically only be used to evaluate the same sample of students (Hambleton, 1990). There is no guarantee that a separate sample of students would also lead to the same Item Difficulty. IRT however is invariant when it comes to the item parameters and latent traits. The ability of the student is independent from the test items. This is the case because information about the items are incorporated into the ability estimation process (Hambleton, 1990). The items itself have characteristics that align with certain latent traits of the student. Instead of using a simple ratio of correct answers and the amount of total answers given, IRT uses the ability scale to predict the probability of the student solving a particular task. In IRT the Item Difficulty value depends on the ability scale of a specific student and their traits that are relevant for the task. As already mentioned it is not set in stone what the optimal Item Difficulty value is. Shernof et al. (2017) shows, in agreement with the flow theory, that students are more engaged when the problems are solvable but still challenging. The Flow Theory describes a state of mind that is highly engaged by tackling a task that is not too easy or too difficult (Csikszentmihalyi, 2008). Furthermore, tests have to be reliable and valid. Reliability is necessary so that the test consistently gives out the same scores if the test itself did not change. Validity is important so that the test leads to meaningful findings (University of Wisconsin, n.d.-b).

Daróczy, Wolska, Meurers, and Nuerk (2015) did a thorough review of the task difficulty in word problems analysing the linguistic and numerical factors of the problem. The difficulty of a mathematical word problem can be categorized in linguistic factors that are comprised of structure and semantics of the task, mathematical factors that include property of numbers, the required operation, mathematical solution strategies and relevance of the information and general factors like skills and social aspects, categorization and solution strategies. Seeing that the E-Learning Software that is proposed in this thesis will deal exclusively with the generation of mathematical word problems, this source is crucial and requires further explanation.

Linguistic factors: The linguistics of a word problem can be summarized as how language is used to construct the question. It is the way in which the actual task presents itself to the student. Daróczy et al. (2015, p.4 Structural Factors) lists the multiple ways of measuring the structural expression of a task by analysing with a more quantitative nature by counting words, sentence length, how many complex words exist and so on. Other factors include the complexity of the sentences that make up the word problem (Cummins, Kintsch, Reusser, & Weimer, 1988). Another important factors are the semantic ones. A task containing irrelevant information and the addition of extra steps the student has to solve decreases the accuracy of the student solving the task (Muth, 1992), (Schley & Fujita, 2014).

Mathematical factors: A student that has to interpret and solve a task represented within a mathematical word problem will need to engage with arithmetic operations. Therefore another difficulty aspect is to look at arithmetic operations that are contained within the problem in respect to the difficulty of them. Daróczy et al. (2015, p.5 Numerical Complexity and Numerical Studies) present some general findings about the difficulty of word problems in regards to their mathematical factors:

- Higher difficulty through higher complex arithmetic structure.
- Carry problems have a higher cognitive load than memory problems.
- Tasks involving multi digit numbers are harder to solve than single digit tasks (Kadosh, Dowker, Nuerk, Moeller, & Willmes, 2014).

Daróczy et al. (2015) also suggests to categorize the numerical difficulty of a word problem into the following categories:

- (i) the property of numbers
- (ii) required operation
- (iii) mathematical solution strategies
- (iv) relevance of the information
- (v) other numerical processes and representations

Butterworth, Marchesini, and Girelli (1999, p.14) shows an interesting effect. Calculation time of students increases with its problem size. It is however the case that multiplications that include the number 5 are calculated faster, which is an important fact regarding the creation of multiplication tasks.

In Lee and Heyworth (2000) they measured problem difficulty and came to the conclusion that besides numerical complexity of the task, perceived number of difficulty steps during the problem solving process, number of operations and the students degree of familiarity with the question are relevant.

2.6 Participatory Design

Participatory design aims to improve the development of a product by including the users and in general stakeholders in the design process (Vines, Clarke, Wright, McCarthy, & Olivier, 2013). The desired product can be of commercial nature or be part of a research project. The first of the three main goals of participatory design is to share control with the users who will eventually use the product in a real environment, therefore creating a social negotiation among all the relevant parties to create a product that responds to the concerns of all stakeholders (Vines et al., 2013).

The second goal is to see the users as a source of valuable information (Vines et al., 2013). Software developers do not have the insight into what challenges and obstacles the potential users of their software face in their working environment. The goal is to extract the information and insight that is relevant for the development process. Qualitative research like interviews, focus groups and the use of stimulus material like storyboarding and prototyping are used to identify the needs of the user and to learn more about the working environment (Design Council, 2007, p. 11)

The third goal of participatory design is to elicit an individual, organizational and technological change (Vines et al., 2013). The collaboration with users and the resulting accumulation of insight and experiences lead to the change of old limiting structures and ideas that lead the way to new ideas and designs (Ehn, 1990).

2.6.1 Usability Testing

Usability Testing is a process to evaluate a product based on specific usability criteria by employing users of the target audience (Knoedler & Wenger, 1995). The main goal is to iteratively eliminate deficiencies of a product by gathering insight and data from the user. It is defined in ISO ISO 9241-210 (2010) that Usability is 'The effectiveness, efficiency and satisfaction with which specified users achieve specified goals in particular environments.'

Nielsen (1993) described the usability of a product with the elements *Efficiency of use*, *Subjective Satisfaction*, *Learnability*, *Memorability* and *Few and Noncatastrophic Errors*. The first element evaluates the performance of the product, how easy and efficient it is to successfully complete a task. Subjective Satisfaction speaks to the pleasure of using the product. Learnability measures the ability of the product to teach how it is to be used while Memorability is about retaining the handling of the product. Lastly Few and Noncatastrophic Errors evaluate the amount of mistakes the user makes and how severe the consequences of said errors are.

There are multiple approaches to testing usability. Often times it includes observing of the user's behaviour and actions in an actual work environment to get an idea of the suitability of the product in a realistic scenario. The testing can be done in a qualitative and quantitative manner and includes the use of interviewing and probing

of the participants (Knoedler & Wenger, 1995, p. 25). Usability Testing can be used in every stage of development and is therefore a technique that can be inserted in the participatory design process at every stage, be it as an explorative method or as validation of an already existing prototype (Knoedler & Wenger, 1995, Usability Testing: An Overview).

3 Methods

The developed E-Learning System is comprised of various parts that need to work together in order to be usable as a learning tool for students. Three distinct components can be identified. First of all a student model is needed in order to have a student representation in the system. Secondly word problems have to be created using certain difficulty parameters that change the make up of the tasks. Lastly the behaviour of the student has to be logged from which information will be extracted that will be used to alter the current iteration of the student model.

The model of the student needs a foundation that classifies the skills the student will need to show competency in. For that reason the guidelines from the Ministry of Culture are used (see section 3.1). These skills furthermore need a numerical value attached to it. The skills are represented with a *Scalar Model* with values ranging from 0-100. Additionally, an *Overlay Model* will be used to represent dependencies between concepts and their sub-concepts. An *Error Model* is used to analyse the errors the student makes while using the E-Learning System. Based on extracted logging data, the student model will be updated by using a probabilistic approach, a Bayesian Network (see section 3.1.2). The values are initialized by allowing the system to scale the impact the student's performance has on the update process.

An expansive logging system was developed in order to gather implicit data of the student at every action by the student pertaining the calculation of sub-tasks as well as an overall data collection process that gathers information about the general performance of the student, including the analysis of help and support functions that were used (see section 3.1)

In order to update the skill model of the student based on the data that was collected, *Bayesian Knowledge Tracing* was used. The competencies of the student have numerical values attached to it in the form of probabilities that indicate if the current level of mastery is reached or not. Each mastery is divided into multiple levels. Bayesian Knowledge Tracing requires parameters to initialize the model. The database of ASSISTments by Heffernan and Heffernan (2014) was used because an own effort of collecting enough data was not feasible (see section 3.3.2)

Math tasks are automatically being created by combining various templates together that are associated with different mathematical concepts. The templates are filled with

objects that were extracted and curated from sources in the semantic web beforehand (see section 4.5).

To determine the difficulty of the tasks, certain parameters are created based on the student model (see section 3.4.4). According to the *Item Response Theory* by Hambleton (1990) the item difficulty is determined by the relevant latent traits a student has. The difficulty of the tasks is set, so that the difficulty of the tasks will remain challenging and engaging. Student's skills are translated into difficulty parameters by analysing the traits that are needed to complete certain tasks (see section 3.4.5).

The actual effect of the difficulty parameters are based on findings from (Daróczy et al., 2015). In order to ensure a high grade of individualization of the desired difficulty, all difficulty parameters also have multiple levels that influence how easy or difficulty a sub-task becomes.

For the flexible question solving system, light inspiration was taken from the work of Andres et al. (2013) in how the students have to decide themselves on how to construct a calculation. In the software that was developed in this thesis project, the student has to find missing information that is relevant to solving the questions that are also being automatically created. The actual word problem text and the containing object information are separated from each other.

The use of E-Learning has downsides, as expressed by Vrasidas (2004) and Arkorful and Abaidoo (2014). A common denominator is that in order to ensure the suitability of the developed software in a real environment, it is important to gain insight from experts in the field in order to fill developer knowledge gaps and implement the software in a way that it is usable by the relevant parties. Participatory methods were used by first conducting an exploratory interview with an expert in the field using the guided interview guidelines by Niebert and Gropengießer (2014). After the development of the software was concluded taking into consideration the initial feedback that was gathered in the initial interview, the expert was contacted again to conduct a usability test. The method of *Discount Usability Testing* by Nielsen (1993) was slightly altered and used to gather empirical data to evaluate the success of the created E-Learning Software, including *Thinking Aloud*, the creation of *Scenarios* and a *User Test*. The usability test was accompanied by another guided interview to seek out feedback about the novelty features that were presented. The transcript was analysed with an heuristic evaluation as outlined in Sarodnick and Brau (2006, p. 157) (see section 3.7.2).

The resulting data of the usability test was then used to analyse the current feasibility of the software and to discuss possible further improvements.

3.1 Assessment and modelling of a student

Assessing the student is done via Bayesian Knowledge Tracing by updating and analysing the mastery of a student in a skill based on observed behaviour of a student showing

a certain level of knowledge in a particular skill. The list of skills is derived by the guiding principles and core competences shown in the competence framework from the German Ministry of Culture (Kultusministerkonferenz, 2012). The behaviour of the student is tracked via a logging system.

3.1.1 Student model

When it comes to choosing a student model, there are three choices to be made. What kind of information are to be stored about the student, how will the information change based on newly extracted data and lastly how the student model is initialized on first use. The information about the student's mastery needs to resemble how good a student is at solving problems that are part of the current curriculum in the relevant subject, in this case math. For this reason it is important to define certain skills that the student is using while solving a task. The competence framework from the Ministry of Culture (Kultusministerkonferenz, 2012) displays the various competences a student should show proficiency in.

3.1.2 Skill representation

There are two major categories of skills, the guiding principles which refer to certain mathematical concepts and the core competences that are more broad describe a more general mathematical intelligence. In this thesis the full list of guiding principles and core competences will not be implemented, in part because of the limited scope of this thesis and secondly because the system will not allow the student to give open ended answers and explanations of visual illustrations. The full list of guiding principles and core competences can be found in the document published by the Ministry of Culture (Kultusministerkonferenz, 2012) (Translation by author).

- G1: Number
  - Computational rules
  - Percent and interest calculation
- G2: Measurement
  - Length, area and volume measurement
  - Identify measurement units in the task
- C2 and C5: Solving mathematical problems
  - Choose correct formulas
  - Choose correct information and dismiss distractions
  - Insert values into formulas

- C6: Communication
  - Steps needed to solve the task vs optimal amount of steps
  - Logical traversal of the task

In addition to these skills, procedural skills mastery for mathematical operations like addition, subtraction, multiplication, and division are further subdivided into the following sub skills:

Decimal Handling: For every arithmetic operation the handling of decimals within said operation is stored for each student. The representation of the level of decimal handling is categorized in four levels, which represent a different amount of decimal places.

Big Number: This skill represents how well a student handles different dimensions of numbers while doing arithmetic calculations.

Numerical Complexity: Numbers have characteristics that determine the difficulty of the calculation they are involved in. The four levels are determined by the property of the numbers being able to be divided by certain divisors without leaving a remainder.

The effect the skills have on the eventually generated math task and how the level of each skill changes the output will be discussed in chapter 3.4.4.

At this point what kind of information the student model needs to store is decided. The next step involves the choice of how the information is being stored. Possible approaches include *Scalar Models*, *Overlay Model Error Model* and the *Genetic Model* (Froschl, 2005). Notably it is not necessary to choose just one of these models exclusively but rather a combination of two or more models. Looking ahead the goal of the student model is to support the creation of individualized word problems in regards to how difficult the tasks are. Single behavioural actions by the student should have an impact on how the student model is representing the skill of the student. A pure overlay method is therefore not suitable because it presents the student showing mastery in a binary fashion. As a simple example a student model using the overlay methods would store if the student knows the concept of calculating geometrical shapes. This skill could be divided into more finely-grained skills like calculating the area or volume of simple or more complex objects. This would depend on the student to know the concept of multiplication. An exclusive overlay method makes sense when the purpose of the E-Learning System is to find the missing concepts that are dependencies on other more advanced concepts in order to assign the student the necessary coursework that is needed to advance. This is however not the goal of this E-Learning System. For that reason a complex overlay model that includes the Scalar Model was chosen

(Brusilovsky, 1994, p. 72). With the Scalar Model, skills like *Measurement* or *Percent and Interest Calculation* can be assigned a certain numeric value, showing the mastery of the student in a non binary way. In the following these values are called levels. Using this method it becomes necessary to define what each level entails. This is although not part of selecting the student model but has to be kept in mind for the eventual process of translating the skill levels into actual difficulty adjustments. In addition to the Scalar Model, an Error Model was used in order to support analysing all possible errors a student can make by dynamically analyzing if the student made a possible error in their calculations. A genetic model was not used because the available concepts that are implemented do not show a lot of dependencies and are limited in their amount.

The information pertaining the student model will therefore be displayed by a mix of the Scalar Model, Overlay Model and Error Model. These values however need to be initialized. There are multiple ways of doing so, mostly differentiating each other from the amount of data that is initially needed. Recognizing the eventual process of updating the skills in the student model, a simple model like this does not suffice. Assuming a certain amount of levels per skill, it would not be advisable to simply make binary decisions of increasing or decreasing said level by a single action by the student, such as completing a task correctly or wrongly. Therefore the student model needs to be further adjusted. Two possible approaches come to mind, the first being Machine Learning Models. These models require a vast amount of available data in order to make predictions about the mastery level of the student. Especially in the field of education, it is not easy to accumulate the necessary amount of data to make accurate predictions. The alternative method of using a Bayesian Network can mend these problems to an extent. A Bayesian Network probability model assigns probabilistic values to a certain proposition, in this case the mastery level of a student (Sison & Shimura, 1998). They also show the potential upside of using a Bayesian Network: "Bayesian networks allow a more convenient representation and manipulation of the entire joint probability distribution of a domain. Specifically, one need only specify the prior probabilities of the root nodes and the conditional probabilities of non-root nodes given their immediate predecessors." (Sison & Shimura, 1998, p. 136) This results in a vast decrease of information that needs to be known, because only certain conditional values are needed. Therefore the final student model includes the three mentioned techniques displaying each skill by a certain numerical value, usually ten. The Bayesian Network enhances the numeric value by applying a probability to it, representing the current progress or probability that the student has mastered the current level of the skill mastery.

(Self, 1994) lists two ways of extracting data to initialize student models. *Explicit Questions* and *Default Assumptions*. Starting off with explicit questions, this method of getting data is simple when it comes to the effort to implement the questions into the system. The potential downside is that the questions that are being given to the

students need to be constructed in a careful way in order to avoid false or invalid information given by the student. Self assessment by the student can lead to the developers interpreting the skill estimation differently than the student. Using Default Assumptions or stereotypes is another method that allows the E-Learning System to get a baseline understanding of the new user. Background information like grade and type of school can be used to infer an approximate skill level of the student.

Another way of initializing the model is to just let the student take exemplary tests. These tests are a great tool to get a baseline understanding of the student that is starting to use the system. It leads to a normalization of the skill representation because it basically presents the student with predefined tasks, that can be constructed to evaluate all the skills that the student model is comprised off. The problem with this method clearly is the effort that is involved with constructing the test in a way that it covers all the necessary skills. It also leads to a certain amount of time investment by the student on an initial use.

Seeing that a Bayesian Network is part of the student model, two kinds of data sets are required. First of all information about the student in the form of scalar values are needed to determine the skill levels of the student. The second data set is needed when it comes to updating the skills, which will be discussed at a later point. At first, a test on initial first use of the software was supposed to set the baseline for the student model, but was rather quickly found to be obsolete by a simple realization. Seeing that eventually the behaviour of the student has to influence the values that are included in the student model, a parameter will be necessary that decides how impactful the actions of the students should be. The approach of implementing a sensitivity slider that simply increases the degree of influence a student's actions has on the student model, leads to the possibility of adjusting the student's skill level in a fast manner, while also allowing the more experienced user's actions to have a lesser impact on the outcome of the update process of their skill mastery. It however can be noted that an initial test would allow the system to cover all the skill estimations, which the normal use of the software might not.

The current prototype shows the beginnings of how this problem can be solved, by asking the student explicit questions to self assess how good he or she is at certain skills 4.15. Additionally, information regarding the type of school and grade are being asked, which could be used to infer how well the student might be at certain skills. This would fall under the category of Stereotyping.

At this point it is important to note that there are certain challenges associated with the use of Learner Analytics in regards to the process of modelling a student model that then gets updated based on the analysis of the behaviour of the student. Using strictly technical means of determining how the student is to be represented in the system has its problems. (Perrotta & Williamson, 2018) mentions that the learning process is a social one that can not exclusively be modelled by technical processes. Therefore it is

important to recognize that human elements should not be eliminated completely. Even if ethical and privacy issues are disregarded, Learning Analytics need to be carefully utilized in order to avoid bad data. Siemens (2013) states that a student model is supposed to accurately represent the mastery level of student, but fails to completely integrate all the intricacies that come when modelling a real human being. The student model can only be regarded as an approximation, not a factual representation of the student. They also mention that once a student model is determined, depending on how the E-Learning System works, it is hard to break out of the environment the system creates based on the assumptions it makes about the student. In light of this Kitchin (2014, p. 5) states that generated data is never free from human bias. Data still has to be interpreted by algorithms that are integrated in a particular scientific approach. A similar reasoning from **challenges** states "There is thus a risk that learning analytic systems may reproduce and entrench existing biases such as class, gender, ethnicity."

(Wilson, Watson, Thompson, Drew, & Doyle, 2017) emphasizes that learning activities and performance measurements vary greatly which needs to be considered and personalized for each avenue where Learning Analytics is used. To insert the necessary element of human interaction, participatory methods were used in this thesis in order to exploratively learn about the environment the software would operate in, which includes characteristics and behaviour of the students that would use the software, as well as environmental factors that relate to the classroom experience as a whole, including the role of the teacher. As shown in section 2.2, using E-Learning Systems can have certain downsides. Moving the technological aspect of education to the forefront can lead to a loss of personal interaction between the students and their teachers. Vrasidas (2004, p. 3) therefore recommends to conduct learner and audience analysis in order to identify the needs of the users, teachers and environments they operate in, that the software needs to consider. Additionally, one of the big upsides of E-Learning is the possible high degree of individualization (Arkorful & Abaidoo, 2014). This further emphasizes the importance of using participatory methods to create an E-Learning System that does not present itself with the mentioned problems and takes advantage of the possible upsides such a system can entail.

The participatory methods that were used in the development process of the developed E-Learning System are explained in section 3.7. Additional ethical considerations can be found in section 3.8.

3.2 Continuous Difficulty Adjustment Process

Now that the student model is defined, the continuous cycle of creating tasks, analysing the behaviour and then using the data to update the skill representations can be explained. In order to create individualized mathematical word problems, this software goes through a cyclic process that is dynamically run while the software is being used.

1. Logging of student's behaviour while solving a task.
2. Analyse the behaviour and create an update packet.
3. Update skills of the student via Bayesian Knowledge Tracing.
4. Map skills of a student to difficulty parameters.
5. Creation of a task based on difficulty parameters.

These are the steps that the system will go through for each attempt at solving a task by a student. Initially a math task is being created with default values that are dependent on the initial settings for the specific student. In order to create the task, difficulty parameters are being derived based on the skill levels of the student. While the student solves the task, the behaviour is being logged and when appropriate used to create update packets. The accumulation of update packets are then used to update the skill levels of the student via Bayesian Knowledge Tracing. Once the student gets presented a newly generated tasks, the success or failure at solving the former task influences the difficulty settings, that are then used to create the new task. This cycle is constantly running and dynamically updating to suit the needs of the student. The following will go into great detail for every one of these steps.

3.3 User Behaviour Data

In order to evaluate the performance of the student and eventually update the student model, information relevant to the student's mastery level of skills have to be extracted. According to Barr (1979, p. 622) there are four ways of determining if the mastery level of a student needs updating. *Implicit data* can be gathered through observation of the student model by logging the action the student makes while using the system. The E-learning System itself has to implement technical ways of gathering this data when it is feasible and wise to do so. *Explicit data* is a more direct approach of gathering data by taking part in a dialogue between the system and the student. Historical data refers to assumptions that can be made about the student that reflect on the student's knowledge and capabilities. Lastly *Structural information* can be gained by comparing the specific concept related to the relevant mastery level that is being looked at with the contextual curriculum, by analysing if dependencies can be found. This E-Learning System mainly focuses on the implicit information that can be gained by an extensive logging system. A system like that offers objective information about the student's behaviour and does not require input from the student or any teacher. Historical information are not used in the current system. Structural information is implicitly used for updating the mastery skill, because the updating of individual skills influence more high-level concepts. For example improving the mastery in solving multiplication tasks involving numbers with

decimal places influences how good of a mastery a student can show in a more high-level concept, such as percent and interest calculations.

In order to evaluate the performance of the student and eventually update the student model, each and every relevant action has to be logged in the system. The results of the logging process are then used to update the skill model of the student via Bayesian Knowledge Tracing.

3.3.1 Logging

The behaviour of the student is constantly being monitored in the background. In order to create a valid calculation to solve the math task, the student has to show sufficiency in a multitude of skills that correlate to the student model in the system. Starting out the student has to choose the correct information that are present in the task and add all the necessary data to construct valid calculations to solve a particular subtask. That includes adding already present information and to combine them with the appropriate operators. The last step is to work out the result of the calculation before submitting it. It is apparent that it is possible for the student to choose the correct information but not calculate the correct result and vice versa. All these information are logged in the system.

To illustrate this point, imagine the following task *Calculate the length of the Zoo*. The student has the information about the area and width of the Zoo to choose from. The first step is to find out what kind of formula is needed, which in this case is area divided by width. The student now has to insert the area value, followed up by a division operator and then finished by the width value. After submitting a result, the task is evaluated by the system.

In this particular task the student had to show proficiency in the following sub-tasks:

1. Choose the correct formula
2. Choose the correct information
3. Dismiss distracting information
4. Calculate a task that includes division

There are multiple scenarios in which the student can successfully complete any amount of these tasks. For example it is possible that the student knows the correct formula, but ends up choosing the wrong information to include in the calculation. Each involvement of skill leads to the generation of a unique update packet that considers the outcome of each skill presentation. It is easy to see that the sub-tasks correlate to the competency framework as follows:

- G2: Length, area and volume measurement.
- G2/C2/C5: Identify the measurement units of the Zoo.
- C2/C5: Choose correct information about the Zoo.
- C2/C5: Choose correct formulas and insert the identified values.
- C6: Steps needed to solve the task.
- Procedural: Division

The direct correlation between the gathered information about the student's behaviour in this example task and the competency framework that makes up the student's model makes it easy and intuitive to update the student's skill levels.

3.3.2 Skill Updates via Bayesian Knowledge Tracing

The choice of a student model as mentioned before was a Bayesian Network, which means that probabilities are assigned to mastery levels. These values need to be updated in order to have an up-to-date representation of the student. There are multiple ways of updating the Bayesian Network. The base component stays the same, but additional features can be added that influence the update process slightly. This is easily done because a Bayesian Network relies on conditional probabilities that each influence the formula of eventually updating the mastery level of a student for a particular skill (Corbett & Anderson, 1994). As mentioned in section 2.3 the most simple approach of updating the network is to only have a positive or negative action. Either the student solves a task correctly or not. The other necessary component is the probability of a transition from not mastering a skill to mastering it. These values could be used to then update the probability of the student mastering the skill. Realizing that a student might try to guess the answer, a probability of this being the case must exist. The same is true for a student possibly mastering a skill but just slipping up and not getting the correct answer. For this a probability must exist as well. Clearly it is not easy to just logically determine what those probabilities are. Hence a collection of data is needed to fit these parameters. Optimally each system fits this kind of data themselves because the tasks that are given are different for each system. However this endeavour would lead to an enormous amount of data collection effort and was not done. In the end fitting these parameters wrongly would not impair the main features that are sought after in this thesis by that much. However there are alternatives to just manually deciding what these values ought to be. For that reason the mean average of the parameters that Pardos and Heffernan (2011) analysed based on a dataset consisting of the behaviour of students using ASSISTments (Heffernan & Heffernan, 2014), which over 500.000 students use, were used to fit the parameters. Clearly the slip and guess parameter

must depend on the type of task that the student solves in the moment. As an example a student is more likely to guess what kind of information is not relevant to the answer and correctly identify it as distracting information if there are only a few to pick from. Therefore it would be advisable to adjust the slip and guess parameter for each and every display of mastery. This however was not done in this thesis because of time and budget constraints. Having more accurate parameters would however lead to a more accurate update process for the student's mastery levels.

In the developed E-Learning System the student was modelled based on the skills and competences that are mentioned in section 3.1.2. For each of these skills a Bayesian Knowledge Tracing was implemented that updates the mastery of the student in the respective skills in accordance to how BKT is portrayed in Corbett and Anderson (1994). The calculation of mastery in Bayesian Knowledge Tracing that was used is as follows:

$$P(L_{n-1}|Correct_n) = \frac{P(L_{n-1}) * (1 - P(S))}{P(L_{n-1}) * \big(1 - P(S)\big) + \ (1 - P(L_{n-1})) * (P(G))}$$

$$P(L_{n-1}|Incorrect_n) = \frac{P(L_{n-1}) * P(S)}{P(L_{n-1}) * P(S) + \ (1 - P(L_{n-1})) * (1 - P(G))}$$

$$P(L_n|Action_n) = \ P(L_{n-1}|Action_n) + \big((1 - P(L_{n-1}|Action_n)\big) * P(T))$$

Figure 1: Bayesian Knowledge Tracing algorithms to update skill mastery (R. Baker, Gowda, & Salamin, 2018).

Depending on if the student showed proficiency in solving a task, either the first or the second equation is used to calculate the probability of the student actually having mastery by incorporating factors like successfully guessing the correct answer or making a slip-up even though the student does in fact master the skill.

Once update packets are being created, the student model is updated based on certain parameters that are being inserted into the calculation of mastery. The mastery of a skill lies between the guess percentage as the minimum and 100% as the maximum. In the student's model this skill mastery is further divided into smaller steps. Each skill is divided into levels that denote the current proficiency of the student in said skill. Dependent on how sensible the update mechanism is set to be in the settings, a high calculated mastery via BKT resolves in the student's level being updated to be one step higher or lower.

3.4 Math task generation

Being the focus of this thesis, the automatic generation of mathematical word problem is a complex process. Therefore an explanation what the task model entails in regards to structure and information presentation is necessary. Every tasks needs information to be displayed such as objects and their values, which in the case of this software are objects extracted from the real world. Those kind of data need to be extracted and curated for it to be usable in the generation of tasks. Afterwards an elaboration about how the difficulty parameters are compiled based on the student's skill levels and then show what effects these parameters have on the creation of the task will be given.

3.4.1 Task model

Logging extensively as earlier explained requires a complex task model that enables the harvest of large amounts of relevant information. The generated word problem itself is a collection of building bricks that represent sub-tasks, which in the following will be called question templates. A task is generated by combining compatible question templates together based on vast amounts of difficulty parameters and other factors. In order to support extensive logging of the user behaviour it is important to be able to assign the behavioural results to the correct skills. Therefore each template stores information about the sub-task that can then later on be extracted if necessary. Visually the template adds a part to the whole question text with a variable component in it. The objects that the template features are variable and are chosen dynamically at the task generation process. These objects represent the attributes of the question, that store information about how to calculate them with other existing information.

If the question template is for example *A zoo is buying food for the animals*, the zoo objects contains information about the length, width, area and volume of the zoo, as well as the amount of animals, income and expenses. These information are separated from the actual question text to avoid confusion. Seeing that it is the student's task to find missing crucial attribute information, every attribute stores its concept affiliation as well as the optimal paths to solve it. The *Area* attribute in the example would be part of the *Measurement* concept. These affiliations are important to later update the correct skills if the student tries to solve a calculation associated with this attribute.

The word problem is iteratively built together by using suitable question templates. In the system there are three types of templates:

Base templates: Every question building process starts off by choosing a base template randomly. The base template describes the overall theme the question will have. An example for this kind of template is *The Zoo buys food for their animals.*

Connected templates: Connected templates are templates that are compatible with other already existing templates such as base templates and other connected templates. An example for this kind of template is *20% of the inhabitants of Lilienthal visit the Zoo every Year. One visitor spends 10 Euro on average.*

Supported templates: Supported templates change the way that a current template is being portrayed. An example of a supported template would be the replacement of an attribute of an already existing object in the question for another attribute in order to create a ratio between two objects that the student has to calculate.

The question creation process starts with choosing a base template and follows up by adding compatible connected templates. However a template can only be chosen if the system evaluates that it is compatible with the current existing templates. This process is only partly random. What template is added depends on multiple difficulty factors such as the determined length of the task and the skills of the student in the respective associated concepts of each template.

Inspiration for the templates was taken from from Haftmann (2014) to ensure the creation of word problems that are valid in their educational value. The validity of these questions were further assessed by the expert user test (see Add. Usability Test U66).

3.4.2 Information Extraction

While creating a word problem, the software constructs a question text by adding compatible templates together. Each of these templates need to be supplied with objects that fit the context and the difficulty requirements. These information are stored in a database ready to be extracted. The process of choosing the information is based on numerous difficulty settings that correlate to the attribute values of the stored objects. For that reason it was important to accumulate objects of different dimensions in order to be able to supply the software with objects that have fitting attributes. This leads to requirements of the information being stored. It has to first of all be a large enough amount and secondly it has to be relevant information that can realistically be used in the question templates.

The third requirement was revealed in the expert interview. Realistic and identifiable scenarios are important in order for the student to identify him or herself with the question better, as well as allowing a beneficial side-effect of learning real facts (Add. Guided Interview I43, I45). For that reason most of the objects included in a question template are not of fictional nature but rather extracted from web sources. Polozov et al. (2015) showed that having personalized content that the student can better identify with leads to a better learning experience.

Wikipedia[1] offers vast amounts of structured information in the form of tables that are easy to extract data from. In this software data about the population count of cities in Germany, the measurement of various objects like tennis courts, busses, skateboard and many more were extracted. Theses attributes are then used to fill the necessary variables of a question template based on difficulty factors. In addition to the extracted information, fictional data was made to fill in the gaps of the object data that is needed for the task creation process. It is however important to note that the made up objects were created with realistic values.

3.4.3 Question Variety

A problem that was identified was the limited amount of tasks that are at disposal for a teacher or student to choose from. The same sentiment was shared in the expert interview. By repeating the same task over and over again, the student might resort to remembering how the specific task was solved the last time instead of actually engaging with the question that is being asked (Add. Guided Interview I51). Additionally, the expert criticizes the amount of tasks that are available in currently used ITS Learning Systems (Add. Guided Interview I14). A detailed analysis of the expert interview can be found in chapter 5.1, especially *Allocation and creation of tasks* and *Automated creation of practice tasks*. (National Research Council, 1993) furthermore shows that the novelty of the task determines the emotional response of the student.

The combination of a template based generation of tasks with the information extraction process leads to a unique experience every time a task is being generated. The variety of the asked questions is achieved through three mechanisms:

1. Template based design
2. Use of great amounts of suitable objects
3. Randomized processes

First of all the construction of a task is based on templates as earlier explained. For each added template, compatibility options are being added from which the next template is being chosen from. The process of choosing the next template is mostly random, which adds to the amount of variation. It is easy to see that the amount of possible templates is crucial, because more templates lead to more possible combinations and therefore more variation in the generated tasks. The more templates that are being added, the more impact each template has on the amount of variation, since the increase in combinations grows with the amount of templates there are to choose from.

Assuming an average task is comprised of three question templates, the following illustration shows the impact of adding additional templates on the variety of possible task compositions. Obviously the ordering of templates only marginally matters,

[1] https://www.wikipedia.org/

because the solving path of the task remains the same, but could still influence the experience of the student.

Question Template Pool	Task Combinations
3	1
4	4
5	10
6	20
7	35
8	56
9	84
10	120

Table 1: Possible amount task combinations that use three templates.

It is apparent that increasing the possible question template pool leads to a great variety in possible tasks. An important thing to consider however is that in order to reach the goal of forcing the student to engage with the task instead of remembering how a similar task was solved in the past, the templates need to influence each other. If that was not the case, then parts of the question would still remain the same and therefore not force the student to come up with unique solving strategies. Another point to consider is that this table assumes that every question template is compatible with every other question template which is not the case. For those reasons it is important to have a large question template pool. Seeing that it is fairly easy for software developers to add additional templates, this issue should not pose any problems if the goal was to further include the student's curriculum.

At the moment the amount of templates is limited but great enough to see that each task is different to the one created before as evaluated by the expert (Add. Guided Interview U70). Additionally, connected templates can be displayed in different forms that are called levels. Each level adds variation and difficulty to the question, which further adds to the question variety.

In addition to that, the question templates contain objects. What objects are chosen for each templates is based on mostly difficulty parameter considerations and partly random processes. Seeing that the student might evolve in their mastery of skills and because of the inclusion of randomness, different tasks will feature different objects.

Furthermore the variety of the questions can be influenced by the teacher, because it is possible to enable and disable certain concepts.

3.4.4 Difficulty of Tasks

Before listing the difficulty parameters it is important to explain the requirements for the parameters that eventually led to the final selection.

In order for the difficulty parameters to be useful in the task creation process, the parameters have to satisfy technical as well as conceptual requirements, such as an intuitive understanding of what the settings entails as well as the requirement to be a meaningful representation of the student's skill that leads to distinctive difficulty changes in the task. Lastly it is important to create difficulty parameters that can realistically be used to influence parts of the task creation process.

Intuitive understanding: Knowing that the difficulty parameters will be visible for the teacher to set, it was important that the parameters are intuitively understandable by name and what setting the parameter low or high would entail. This requirement is only relevant when it comes to manually setting the task, but since it was implemented it is a necessary requirement.

Meaningful representation of difficulty distinctions: The major requirement of the difficulty parameters was that each parameter has a meaningful influence on the task creation process. The difficulty of the task relies exclusively on these parameters so it was important that the whole ensemble of parameters are enough to determine the task difficulty sufficiently. In the end the task adaption process can only be influenced by these difficulty parameters.

Easy to implement: Finally realizing that the software has to eventually implement these parameters, it is necessary that an implementation of those parameters is actually technically possible. Numerous potential difficulty parameters could lead to greater individualization but just are not feasible to be implemented. The choice of parameters has to be well defined so that the influence the difficulty parameter has on the task is discernible and easy to make out.

Besides these characteristics, it is important to note that the difficulty adjustments will be made based on the collection of difficulty parameters. This obviously determines the overall difficulty of the task. Seeing that the task is supposed to be presented as an educationally valuable exercise, it is not only important to fulfil the technical and practical needs that were listed, but also be based in a solid theoretical basis when it comes to reliability and validity.

In the case of reliability, the optimal outcome would be that the exact same configuration of difficulty parameter would always lead to the exact same *Item Difficulty* of the task or rather of all its sub-tasks. Generating new word problems over and over should yield in tasks that present the student an equally difficult task. This would especially be important if future endeavours with the software would include comparisons between the performance of students. As outlined in section 2.5 there are multiple approaches on how to determine the difficulty of tasks, that on some level affect how the tasks are

being put together. Going through the outlined methods in a chronological order, the basic approach of determining Item Difficulty by simply calculating the ratio between the correct answers given by students in comparison to the total amount of answers is not possible. One major point of the software developed in this thesis is the unique generation of new word problems, which leads to no task ever being the same as one before. Therefore it is not possible to analyse how students fare with particular word problems. A more complex approach, the *Item Response Theory*, called IRT in the following, advances on the Item Difficulty Theory by introducing latent traits of the student (Hambleton, 1990). The beforementioned student model stores a vast amount of traits a student has. Following the approach of the *Item Characteristic Curve* the approach that was used to determine the difficulty parameters was the assumption that the Item Difficulty is defined by the ability scale of the particular student (Hambleton, 1990). This leads to an invariance of parameters that is important because each newly generated word problem will not have any previously mined data to base its difficulty evaluation on that regards that word problem specifically. The usual approach of allocating tasks to students is then based on the comparison of the needed ability to solve a task compared with the specific student's ability. In this thesis however, this process is more individualized by directly using the collection of abilities of the student and to translate these values into difficult parameters. This should theoretically lead to a perfect alignment between Item Difficulty and latent traits of the student, as long as the translation and difficulty adjustment processes are accurate.

When it comes to analysing the task creation process in regards to reliability and validity, an obvious problem arises. The task creation process can not be deterministic, because that would lead to a stale and repetitive creation of new tasks. Therefore it is necessary to analyse in which way the created tasks differ from each other assuming the exact same configuration. In order to ensure reliability, random processes are introduced in a way that the specific outcomes do not influence the difficulty of the sub-tasks, but rather just introduce variety. To ensure that there is variety without influencing the overall difficulty, the real world objects that fill up the necessary elements of a template need to be plentiful available so that there are multiple different objects that are assigned to the same overall difficulty. This is also true for the process of creating sub-task calculations. The randomized aspect only influences the variety and uniqueness of the calculation but does not change the wanted difficulty of the task.

An easier case can be made for the validity of the task, since a requirement for the validity is that the task does in fact measure the trait for which the task was created (University of Wisconsin, n.d.-b). The only parameters that the system uses to create tasks are the student's traits, therefore validity is given as long as the translation from the student's traits to difficulty parameters and then to the task itself is correct.

With how the system works, the Item Difficulty does not have a fixed value. The student's traits evaluation are constantly being monitored and updated and used to

create a task that is slightly more challenging than what the student has previously shown success in solving in. In accordance with the flow theory (Csikszentmihalyi, 2008) the tasks will therefore always be in the optimal range of engaging the student, as long as the difficulty settings are accurate.

The findings in section 2.5 in combination with the beforementioned requirements of the difficulty parameters have led to the following difficulty measurements of the tasks that are used within this software:

Numerical Difficulty: The numerical difficulty defines the difficulty of the arithmetic calculations a student has to do. It is separated into each of the arithmetic operations Addition, Subtraction, Multiplication and Division. Each of these are further subdivided into the following difficulty parameters that correlate to the findings of Daróczy et al. (2015, p.6 (i)) regarding the property of the numbers as well as the categorization into the operator that is involved in the calculation (Daróczy et al., 2015, p.6 (ii)).

- Decimal Handling: The bigger the decimal handling difficulty is, the more decimal places are featured in the numbers relevant in a calculation.
- Big Number Handling: This parameter determines the magnitude of the numbers in a calculation.
- Numerical Complexity: This parameter defines the complexity of the numbers in a calculation. Based on the complexity, the division features numbers that can be divided by 2 and 5 without having a remainder.

Any calculation that is created uses these three parameters to create arithmetic operations that best accomodate the wanted characteristics of the task.

Conceptual Difficulty: The conceptual difficulty defines the difficulty of the task and sub-tasks as a whole without regarding the single arithmetic calculations. It is further divided into concept difficulty parameters.

- **Replacement of attributes:** This parameter determines how often attributes of objects are being replaced by other objects' attributes. This creates a task for the student to first solve the replacing attributes value before solving the initially desired one.
- **Length of the task:** This parameter defines how long the generated task will be.
- **Question difficulty:** This parameter determines the difficulty of the generated questions.

- **Amount of distractions:** This parameter determines how many distracting information the task should feature that the student has to dismiss.
- **Concept Difficulty:** The concept difficulty determines the difficulty parameters that are associated with the different concepts *General Number handling*, which is a kind of general concept that revolves around doing arithmetic tasks, *Area Measurement* and *Percent Calculation*.

These parameters were created with the findings of Daróczy et al. (2015) in mind. They created five categories, of which the fourth category '(iv) relevance of the information' directly correlates to the difficulty parameter *Amount of distractions*. The other three parameters can be found in the categories '(iii) mathematical solution strategies [...]' and '(v) other numerical processes and representation' (Daróczy et al., 2015, p.6).

These three particular concepts were chosen to be supported because they show off the features that this prototype implements well. The inclusion of *General Number Handling* is necessary because it encompasses a wide range of arithmetic operations that are present in every word problem.

The concept of Area Measurement allows an easy extraction of real world object data because information about geometrical data is widely available. The Percent Calculation concept demonstrates the handling of calculations with decimal numbers. Additionally, the Area Measurement and Percent Calculation are simple to be combined, seeing that the usage of percentages can be applied in all kinds of areas. It is easy to see scenarios in which the student has to calculate the size of a store, which then determines how many products the store can sell which then organically leads to the inclusion of money and percent calculations. To summarize, the supported templates allow the prototype to express the core features well, such as the difficulty adjustment process and the organic construction of word problems. They also fit into the curriculum of the demography this E-Learning System is made for.

At this point it is important to note that expanding the current ensemble of difficulty parameters is entirely possible as long as new potential difficulty parameters are not infringing on the effect the current ones have.

3.4.5 Compiling of difficulty parameters

The generation of tasks necessitates the availability of difficulty parameters. Every time a new task is being generated, the skill levels of the student are taken and transformed into difficulty parameters. There are two ways of doing the compiling process. First of all the teacher is able to either let the software automatically define the difficulty parameters based on the student's skill level or manually adjust the difficulty sliders. Seeing that for option two there is no compiling needed, because the user is setting all the difficulty parameters themselves, the following focuses on the first option of letting the software compile the parameters.

When comparing the student model with the difficulty settings, it is obvious that a direct mapping can not be done with all the parameters. The difficulty settings for Addition, Subtraction, Multiplication and division are identical to the model of the student for procedural skills. Therefore a direct mapping was done. Additionally, the three concept difficulty parameters representing *General Number Handling*, *Percent and Interest Calculation* and *Area Measurement* are included in both the student's model, as well as in the difficulty parameters. That leaves the conceptual difficulty determination of the task, which includes *Replacement of Attributes*, *Length of the task*, *Question difficulty* and *Amount of distractions*.

Seeing that all the available information gathered about the student is contained in the student's model, it is necessary to use that information to create the mentioned difficulty parameters. The approach that was used is to analyse the four parameters and determine what kind of skills are relevant to it. The important categories that can be found in the utilized competency framework are *C2 and C5: Solving mathematical problems*, which include the student's ability of choosing the correct formulas and correct information while dismissing distracting information, as well as the ability to insert values into the formulas correctly. The other more general category *C6: Communication* is important because it contains the overall ability of the student to logically traverse a task and how efficient the student is able to do so. These skills show the conceptual skill of the student instead of specific skills like knowing how to calculate the area of an object.

Finally it is important to note that the process of the difficulty parameter compilation is essential to ensure the validity of the created tasks. As mentioned in section 3.4.4 per Item Response Theory, tasks have to be created with the latent traits of the student that are needed to solve the tasks in mind. The difficulty parameters have to scale with the student's traits if a high validity wants to be ensured. For some parameters this step is trivial because direct measurements of the student's trait are taken. There are parameters however that combine more complex traits together to create the difficulty parameter such as *Question Difficulty* or *Replacement of Attributes*. The transformation process is based on an analysis about what traits play a relevant role, but can not be viewed as objective. Further research and experimentation is needed to ensure this step is done correctly.

Replacement of attributes: Setting a high value for this parameter results in attributes of objects being replaced by other attributes, which results in a situation where an attribute is presented by an other attribute plus a multiplier. This essentially leads to an extension of a sub-task by adding one more step to it. The crucial skill the student needs to show in order to solve this sub-task is the ability to solve carry tasks, because the value of one calculation has to be found in order to calculate the original sought out value. In addition to that, the student needs to show the proficiency in analyzing

the task to find the correct multiplier and operator to use. These two skill evaluations lead to the value of this difficulty parameter.

Length of the task: A similar argument for the compilation of the parameter regarding the length of the task can be made. The longer the task is, the more relevant is the ability of the student to solve carry tasks, because the result of sub-tasks rely on previously calculated sub-tasks. The distinction however to the *Replacement of attributes* parameter is that the concept of the sub-task might change, which necessitates the ability of the student to be proficient in knowing what formulas to use. It can be argued that the proficiency in the general concepts is also important, however the addition of question templates in the task creation process already considers this.

Question difficulty: The difficulty of the questions regarding a word problem in this system is twofold. Firstly the base difficulty is measured based on the complexity of the question and secondly it is relevant how many steps a student has to go through in order to solve the question. The system keeps track of the count of steps the student needs to solve tasks and later compares it with the optimal amount of steps. This leads to an evaluation of the efficiency of the student in traversing tasks. This skill in combination with the general skill of solving carry tasks makes up the *Question Difficulty* parameter.

Amount of distractions: The system tracks if the student is using distracting information in order to construct calculations. This information eventually leads to the evaluation of the student to dismiss unnecessary and wrong information that can be found in the word problem. The evaluation is then used to determine the amount of distractions that a word problem contains.

At this point it is important to note that further research is needed to confirm the assumptions made for the creation of these four difficulty parameters, especially when it comes to the weighting of different factors.

3.4.6 Adaptive Task Generation

In the expert interview it was revealed that teachers dislike the need for clustering students in certain difficulty categories (Add. Guided Interview I40), which then determine the assignment of exercise tasks. The resulting word problems resemble the difficulty category as a whole including all the sub-tasks a student has to solve. It is easy to imagine that students do not have the same expertise throughout the whole required skillset. An obvious example would be a student that is proficient enough in addition but lacks expertise in doing multiplication exercises. In order to remove the problem of having to cluster tasks to certain fixed difficulties that encompass the task

with all of its sub-tasks, each question template that is added to the word problem has its own unique difficulty. This allows the flexible adaption of sub-tasks to the student's needs.

After finalizing the determination of the difficulty parameters, the process of creating the word problem begins. Each question template that is added has different difficulty parameters it uses to determine the specific sub-task that is being created.

The same individual sub-task treatment is true for the analysation of the student's behaviour. Seeing that each sub-task has different skill requirements for the student, only the relevant skills of the student are updated based on performance measures. These updates are independent on the completeness of the task, even a single attempt at solving the word problem affects the skill determination of the student. As soon as a new task is being generated, the cycle of compiling the difficulty parameters followed up by creating an individualized task begins again, but now with adjusted parameters that were triggered by the student's past behaviour.

3.5 Question Generation

Besides the generation of the word problem, it is necessary to create questions that the student has to answer. There are two important factors that are necessary for this process. First of all the question has to be relevant to the word problem text. Secondly the question needs to respect the determined difficulty for the specific student.

The usual approach of using tasks that are assigned to certain difficulties limits the access of information to metadata that encompass the whole task. Having an internal task creation process instead of using predefined tasks yields complete knowledge about the word problem itself.

Relevance: The task creation process compiles a great amount of information about the task as a whole and its sub-tasks. For each question template that is added, more potential questions are added to the question pool. Two types of questions are enabled. The first kind asks about missing information that the student has to calculate in the form of attributes. Every object that is contained within the word problem has certain attributes, that are sometimes missing in order for the student to calculate them. The second kind of question asks broader questions that are variable in its difficulty depending on the length of the question. An illustrative example is the question that asks about how much profit a company is making. Multiple templates in the word problem can influence the answer by adding scenarios that add or subtract from the profit of the company.

Difficulty: The difficulty of questions is determined in two ways. Firstly every question is assigned a base difficulty based on the obscurity of the question. Asking the student to determine specific geometrical information about an object if all the necessary information are already known is an easier task than to ask how much money was spent to construct a building, since it necessitates the student to make logical connections between parts of the question. The more of these steps are needed to solve the task, the higher the base difficulty is. However the creation process also allows simple calculations like the beforementioned geometrical ones to be of a high difficulty, because attributes can be replaced by others. The non-attribute questions are also of a variable degree which needs to be considered. Therefore a system was created that analyses the amount of steps that are needed to find missing information, which is the second measure of question difficulty.

3.6 Question solving system

The main idea behind the solving system was to allow the student to solve a word problem in a structured but flexible way without revealing helpful information to the student if it is not intended to do so from an educational standpoint.

An E-Learning Software that only requests the end result from the viewer would not generate enough information to adequately adjust the difficulty parameters. For such a system it would be impossible to gather relevant data about the student's expertise about knowing what formula to use or if distracting information were correctly dismissed. It is therefore crucial to gather data about the student's behaviour every time a particular skill is being used. It is important to note that such a system needs to allow the student to create multiple valid calculation routes.
The purpose of the question solving system is to incorporate the individualized and contextualized help and guidance for the student, a characteristic of Rule-based Cognitive Models (Nkambou, Bourdeau, & Mizoguchi, 2010, pp. 6–7), with the Constraint-based modelling approach that allows the student to freely approach the task by only providing a framework and constraints on how the solution has to look like (Mitrovic, 2010).

The software prototype features a component-based question solving system that is roughly based on the prototype made by Andres et al. (2013).

The general approach for the student to solve any task is to first find missing attribute information. Once enough information was calculated, the student can then proceed to solve the actual questions. For each of those steps the student has to construct a calculation. The available tools to do so are standard arithmetic operators, complemented by a list of currently available information the student can add to the calculation.

Initially there will be no guidance in how the student would like to solve the question to encourage conceptual thinking and to avoid situations that were mentioned in the expert interview (Add. Guided Interview I27, I28, I29, I30). Seeing that students tend to quickly use help at the first signs of failure, the expert encouraged contextual small-step help. For that reason helper functions are available for the specific calculation at which the student is currently working on. The constraint for the appearance of help however is that the student first has to fail at solving the task once.

The system features an auto calculation feature for certain parts of the question if particular skills want to be trained or if the student wants to continue with the task even though a sub-task was not successfully solved. This decision was supported by findings in the expert interview (Add. Usability Test U4).

If there are multiple valid calculations to find missing information or to solve a question, all valid calculations are being supported by the system.

3.7 Participatory Design

In the introduction the goals of this thesis were mentioned, as well as that it requires insight of the domain where the software is envisioned to operate in. To gain said insight a teacher for the secondary school (grades five to ten) was sought out to conduct an initial exploratory interview later followed by a usability test of a developed prototype. This section will present the motivation behind the choice of the expert, the domain and the methodology that was used. At this point it is important to mention that only a single expert was approached for conducting the tests.

In order to get an understanding of the school environment and the needs of the teachers to provide a good learning experience for the students, a single interview can suffice in retrieving valuable information. However according to Nielsen (1993) a usability test usually needs at least five evaluators to find a sufficient enough amount of faults in the system. The expert in this case was not an usability expert but rather an expert in the domain of teaching. Since the focus on this thesis is not on technical aspects like fault tolerance or specific user interface elements and rather about the task and process suitability of the software in a realistic school environment and how the new concepts are being perceived by teachers, the results of the user test, despite the mentioned shortcomings, still led to relevant conclusions and findings.

Developing a software in the area of E-learning requires a deep knowledge about the domain. Pupils trying to learn math in a school environment can not be compared with a commercial software aimed at adult customers. For that reason an expert in the field was sought out to conduct an explorative study to gain insight into a realistic environment where E-learning software is being used. Fortunately the University of Bremen is collaborating together with a secondary school in Bremen introducing the Bettermarks Software to be used in daily teaching. Since the created software is aimed

at pupils in the fifth to eighths grade, a teacher at said school was contacted that is involved in using E-learning Softwares as a tuition tool.

For the purpose of finding out what kind of knowledge about the domain and end-user behaviour is needed, four distinct categories were created:

Current practices of using E-Learning Software: For the development of the learning software, it was necessary to gain insight into the previous use of E-learning Softwares. What kind of platforms are currently used, how they are used and for what purpose. It would lead to completely different software requirements if the goal of E-learning Software is to be supportive of current methods of teaching or if they are meant to be substitutions.

Occuring Problems with E-Learning Software: The use of E-Learning Software is still in its earlier stages, but it is still important to not repeat problems of current systems. Therefore analysing the problems the teachers face while using various softwares is essential to know.

Degree of independence and teacher oversight: Another aspect is the degree of support and independence that the software needs to support. The degree of manual input that a teacher needs to provide for the software to serve its function changes the functionality of the software greatly. Especially the user interface and logical traversal when it comes to solving mathematical tasks needs to be adjusted based on the expected level of teacher oversight.

Novelty software proposal: Seeing that the software proposal introduces novelty in the form of complex adjustable difficulty parameters and a low degree of teacher input, it is difficult to consult existing literature to get an understanding of how to create a successful product. So it was important to get initial feedback on the essential parts of the prototype of the software by an expert.

3.7.1 Guided Interview

These reasons led to the scheduling of an initial interview with the expert very early into the development phase, at which point a simple coded prototype was finished.

Seeing that the developed software included novelty areas, standardised questioning was not a possibility. For that reason a guided interview as conducted, which permits an open-minded conversation that allows the discovery of questions that arise mid-interview. In addition to that, predefined closed questions would require an already existing expertise in the field, which was not the case. Open-ended questions let the participant decide what information is worthy of sharing. If certain desired topics were

not covered or if the participant did not cover desired topics, follow-up questions were asked.

The general guidelines laid out in Niebert and Gropengießer (2014) were used to construct the procedure of conducting the interview. Because of the limited scope of this study, this interview was concluded with only one expert. If the developed software was to be created on a big scale, additional experts should be sought out. Presenting the novelty software ideas could also be presented via the use of focus groups or group interviews. The resulting transcript of the interview was then paraphrased. For categorization the already predefined categories were used to separate the concerns of the questions in the guided interview. The results were then used in the implementation of the software when suitable.

The categories were created based on an analysis of the interview paraphrases. The goal here was to find distinct categories that are relevant to implementation decisions later on. The first category *Current use of E-Learning Software* encompassed contextual information that do not necessarily explicitly impact any feature determinations in the final product but rather influence the whole process of designing the software. The needs of the user and in which environment and under which circumstances the software would be used in was included in this category. Following that, the two categories *The problem of clustering students* and *Allocation and creation of tasks* contain the identified problems of the expert regarding the difficulty of categorizing students into a category of expertise and how the tasks can be correctly allocated to the students in those categories. The category *Individualization and difficulty adjustments* is of most importance because it includes the problems the expert faces when trying to give the students an optimal individualized learning experience, when it comes to the choice of suitable tasks to practice on. Furthermore *The structuring of help and support* includes important information about how the student is supposed to be guided along in a task and how students react to facing problems while trying to solve word problems. The category *Automated creation of practice tasks* includes information about what kind of characteristics the generated word problems have to contain. The last category regarding *Gamification and reward systems* was included because the expert explicitly states the desire for gamification in E-Learning Systems.

Furthermore for the most part it is possible to draw a direct line from the category to implementation decisions, such as *The structuring of help and support* influences the help and support features in the system (see section 4.6) and the category about *Individualization and difficulty adjustments* heavily influenced the decisions made when the process of difficulty adjustment was implemented (see section 4.9).

The insight that was gained in this interview was insightful and influenced implementation decisions especially when it came to features concerning help functionalities and guidance for the student. Learning about the context and environment of where the

software would be used in was crucial in order to bridge the gap between the ignorance of the developer about the domain and the needs of the end user.

3.7.2 Usability Testing

After finishing the development phase another meeting with said expert was scheduled to conclude a usability test. The motivation behind this test was to validate the decisions that were made in developing the software based on the initial findings of the first guided interview. The focus on interaction and information design was made because of the limited amount of experts participating in the test. According to Nielsen (1993, p. 156) a single participant in a usability test can on average only find about a third of usability mistakes. Obviously a larger amount of experts is desirable, however even a single expert in this instance is sufficient to achieve insightful results. The focus of this thesis was not to develop a software with a user interface that is production ready, but more so to develop novelty and unproven features and to identify if there are fundamental problems with them as well as to justify further research.

There are numerous variations of how to conduct a usability test. Nielsen (1993) proposed the *Thinking Aloud Protocol* that asks the participant to explain each and every thought that comes to mind while solving the task. *The Coaching Method* (Nielsen, 1993) lets the participants of the test ask questions about how to use the system. These methods are not to be used exclusively but rather in combination. A lower budget version of usability testing called the *Discount Usability Testing* can be used to achieve substantial results without having a high cost for conducting the test, making usability testing a great tool for any product development project (Nielsen, 1993, p. 17)

For that reason a combination of techniques was used, specifically a Usability test with realistic Scenarios combined with Simplified Thinking Aloud. Both of these techniques are part of what Nielsen calls the Discount Usability Engineering (Nielsen, 1993, p. 17). Nielsen (1993, p. 225) also states that if there are only few users available to conduct the testing, a focus on the mentioned techniques is appropriate. I purposefully decided against employing the third component *Heuristic Evaluation*, because the focus was on the suitability of the software in an operational environment and if the ideas add value to the teachers tuition efforts. For that reason it is important to not use an usability expert, but rather an expert in the domain. The heuristic evaluation however was not completely disregarded and instead heuristics were used to qualitatively analyse the results of the user test.

The goal of this inductive test is to identify flaws in the software and to receive suggestions in how to improve the software to better suit the needs of the user (Sarodnick & Brau, 2006, p. 156). For inductive tests it is important to have a functional and realistic prototype, that resembles the final product as best as possible to avoid

upcoming problems that might not even be relevant if the product was further along in the development. (Sarodnick & Brau, 2006, p. 157)

Scenarios: For the usability testing seven use-cases were created that best incorporate standard usages of the E-Learning Software, if it was used in a realistic environment. The use cases are as follows:

1. Solving of an easy task
2. Use of contextual help
3. Analysing of the students skills
4. Manual setting of difficulty parameters
5. Automatic setting of difficulty parameters
6. Solving of an advanced task
7. Registration of a new student

After every use case, questions were asked regarding the functionality that was used in the specific task. The structure of those questions again followed the guided interview principles laid out earlier in this chapter.

The questions that were asked right after a use case was concluded were mostly standard usability questions that were answerable in a short amount of time and were directly targeted to find issues that were relevant to the concluded scenario. Even though this thesis is not focused on the standard usability aspects of the software, it is still important to find glaring usability problems as it influences the question if implementing certain features is possible in the way it was done in the software. The follow-up interview then asked questions that were more focused on the E-Learning aspect as well as the new features that were introduced in this software. This was done so to give the participant a complete impression of the software before asking questions that might be answered differently if the inner workings of the software and how different parts relate to each other, is more clear. Seeing that the impression of the question solving system alone could change if the participant knows how the task creation is being influenced by difficulty parameters is one of those examples. For those reasons this approach was favoured.

Thinking Aloud: While the use cases were carried out, the expert was asked to verbalize their thoughts, which allowed an observation of not only their actions but also their motivations and potential problems they faced while doing the tasks (Nielsen,

1993, p. 18). Receiving feedback while the expert is doing the tasks is important, because a follow-up in the way of an interview could lead the expert to do retroactive rationalizations of the actions they took (Nielsen, 1993, p. 196).

The participant was not videotaped while conducting the user test, however audio and the computer screen were recorded so that a comparison of the experts thinking aloud expressions with the actions that were taken at the same time in the software could be made. (Sarodnick & Brau, 2006, p. 162). Additionally, the *Coaching Method* was employed, allowing the user to ask questions and receive general help if needed (Sarodnick & Brau, 2006, p. 164). For the results to be of significance it was important that the expert understood exactly what the software's features are and how to use them. After the usability test was concluded, another small guided interview was conducted that again focused on the three categories that were explained earlier with a heavy focus on suitability in a realistic tuition environment. That includes identified problems in the handling and presentation of the software.

Qualitative Heuristics Analysation To analyze the results of the user test, an heuristic approach was used. Usually an heuristic evaluation means that an expert is using predefined heuristics to judge a software on (Nielsen, 1993, p. 155), but in this case a list of heuristics was used as categories to sort the input that was given by the expert in the user test. These categories function similar to the ones that are used in the method by Mayring (2000) to conduct qualitative content analysis. In case of difficulties regarding the differentiation which category a text snippet is associated to, appropriate rules are put in place.

Nielsen (1994) created a list of ten heuristics to evaluate usability. However since this thesis has a heavy focus on participatory design, the adjusted list of heuristics created by Sarodnick and Brau (2006) was used, who tried to include the participatory approach into the heuristics and combine them with ISO ISO 9241-210 (2010) Norms. The heuristics are as follows[2]:

1. Task suitability: The platform supports the user to do the task effective and efficiently.

2. Process suitability: The platform is well adjusted to the target audience and respects the qualifications and experiences of the user.

3. Self explanatory power: Every dialogue gives feedback that is immediately able to be comprehended or on request be explained.

4. Controllability: User has the control to start and influence the speed of the dialogue with the platform until the goal is reached. Support of multiple peripherals.

[2]ISO 9241 explanations from: http://www.handbuch-usability.de/iso-9241.html (Access on 10. April 2019)

5. Expectation conformity: Consistent presentation of information. The characteristics of the domain are respected.
6. Fault tolerance: System makes it obvious if and why faults occurred. Minimal effort by the user is needed to correct faulty input or mishandling of the platform.
7. System and data security: Faulty input and mistakes by the user should not affect the security of the data.
8. Customizability: Allowance of customizations to the needs of the user.
9. Promotion of learning: Support and guidance for the user to learn the handling of the platform.
10. Cognitive handling: Focus on minimalistic design. Centre of interest on the task relevant components.
11. Joy of use: The user interface should be appealing and modern.
12. Intercultural aspects: Attunement of the platform to the target audience and their national culture and environment.

The additionally included heuristics like Process suitability is especially relevant for the domain the developed software in this thesis would operate in. Pupils have wildly different requirements than the usual end-user. Therefore it is important to evaluate how well adjusted the software is to the target audience. A high amount of controllability is important to accommodate to each learner's ability. Some pupils might need more time to familiarize themselves with the software than others. Another important heuristic that was added by Sarodnick is the customizability. Each student has different demands and learning preferences that the software should be able to adjust to. A focus on minimalistic design and the guiding of attention to what is relevant in the task in the form of a good cognitive handling is another heuristic. The motivation of pupils is affected by how much gamification the platform has to offer (Buckley & Doyle, 2016) or how easy it is to use the platform (Zaharias, 2009). The interface of the platform should be functional but modern to appeal to a young audience as well as being adjusted to the demography as in intercultural aspects of the pupils.

The transcription of the usability test and the follow-up interview were paraphrased an added to the respective category the paraphrase belongs to. The results then led to implications about further potential development plans that the software needs to undergo in order to be suitable for use in a real environment.

To summarize, the use of participative methods first resulted in a more grounded view about the environment in which the software needs to operate, which led to multiple influenced design decisions. The usability test was necessary to further validate that

the implementation decisions lead to the creation of a software that does not only feature the technical viability of new features such as the difficulty scaling process but it critically evaluated the practical viability of the created E-Learning Software.

3.8 Ethical Considerations

Besides the already mentioned potential issues that using Learner Analytics brings with it regarding problems with the data quality and ethical violations when it comes to modelling an accurate student model (see section 3.1), there are additional potential problems when it comes to generating automatic word problems. One of the goals of automating the process of generating tasks, as well as the task assignment of tasks for the student implies little to no oversight by a teacher. For that reason it is important to assure that the quality of the displayed information in the questions is accurate and suitable for the respective grades of the students that use the software. In the current prototype all the content is neutral, but further expansion could lead to the inclusion of templates that would need to be evaluated on a cultural level. Another thing to consider is that the software should only use real world objects and realistic data, as well as realistic outcomes. Using such a system it can be assumed that the students would build up a confidence or attitude that the information they are seeing are in fact objective facts. Therefore it is important to assure the quality of the questions, the objects and their attributes, as well as the result of the word problem.

Besides the worry about the quality of the displayed word problems, maintaining a student model requires the acquisition of vast amount of data. Hence privacy issues need to be analysed as well. Students are generally not asked for consent in order for the E-Learning System to track data about them that might even be shared with other parties (Slade & Prinsloo, 2013). This E-Learning System uses an extensive logging system that analyses the behaviour of the student in order to adjust the student model. It is important to note that the logging data is dynamically created and used. After the student model update process finishes, the logging data is permanently deleted. For the possibility of integrating other tools with this E-Learning System, the logging data that is being extracted might be valuable to be stored. If suitable, the logged data should not be able to be associated to individual students to preserve their privacy (see Wilson et al. (2017)). This is however doubtful in an education environment because of the heavy focus on individualization, which requires the data to be identifiable.

4 Implementation

This section details the implementation details of the E-Learning Software that was proposed in this thesis. The structure of this chapter roughly follows the same path like

the chapter explaining the methodology in order to explain how the different methods were used to build the software and all its parts.

4.1 Technology

Building an E-Learning Software with the features that were laid out has certain requirements. The technology stack has to support a flexible frontend, a database connection, as well as allowing the improvement of the software in the future to be as easy as possible. For those reasons the decision fell on using Spring Boot as an Java-based backend framework in combination with Angular7 as the frontend Javascript-based framework. MySQL was used as the database.

Spring Boot enables the creation of spring based applications without needing to invest a lot of time into the setup. It embeds Tomcat which is an implementation of Java Servlets. This is needed to be able to use the software on a server.[3]

Angular 7 is a JavaScript-based framework to build frontend applications. It natively supports responsive webdesign and cross platform development. It also offers various productivity enhancements such as templating and Command Line tools to get started.[4] The main requirement besides ease of use was cross platform support and responsive webdesign, because it is important for the software to be able to be correctly used on multiple media devices such as tablets, desktop pcs and laptops, especially because students are increasingly using tablets at school.

MySQL is a widely known relational database management system. It is natively supported within Spring Boot, which is the main reason it was used in this project.[5]

The general approach for the creation of the software using the mentioned technology stack is to use Spring boot to develop a Server version of the E-Learning System that does all the necessary calculations and business logic, as well as offering an Application programming interface (API) for Clients to get the necessary data from. Angular7 is used as the frontend framework that utilizes the API to display all of the platform data to the user. The MySQL database holds all the values that describe the student model, as well as all the object data that is used in the creation of tasks. The intricacies of these technologies will be further explained in the following chapters when appropriate.

The communication between the client and the server is handled by an API. The client can request relevant data as so to minimize loading times and to achieve a

[3]https://spring.io/projects/spring-boot
[4]https://angular.io/
[5]https://www.mysql.com/

separation of concerns between the front- and backend. This leads to the possibility of easily replacing the frontend if the need arises.

4.2 Overview

The general approach of this chapter is to figuratively build the software together in a logical way as follows. First, the implementation of the students models will be explained. Following that the fundamental building blocks of the generated tasks are explained via the task and template model. The templates all hold variable objects depicting real world items that need to be extracted from sources in the internet and then stored in the database. This will be explained in the section 4.5. After explaining what tasks consist of, the question solving system will be explained, followed by the logging mechanisms. After being able to log the behaviour of the student, it is possible to update the skills of the student, which will be elaborated in section 4.8. Afterwards the focus of this thesis, the difficulty adjustment process is explained. To conclude this chapter, the remaining features like user management, statistics and user registration will be discussed. For an easier understanding of the following sections, here is a graphic that shows the software architecture as a whole. The figure 2 below shows an overview of the system.

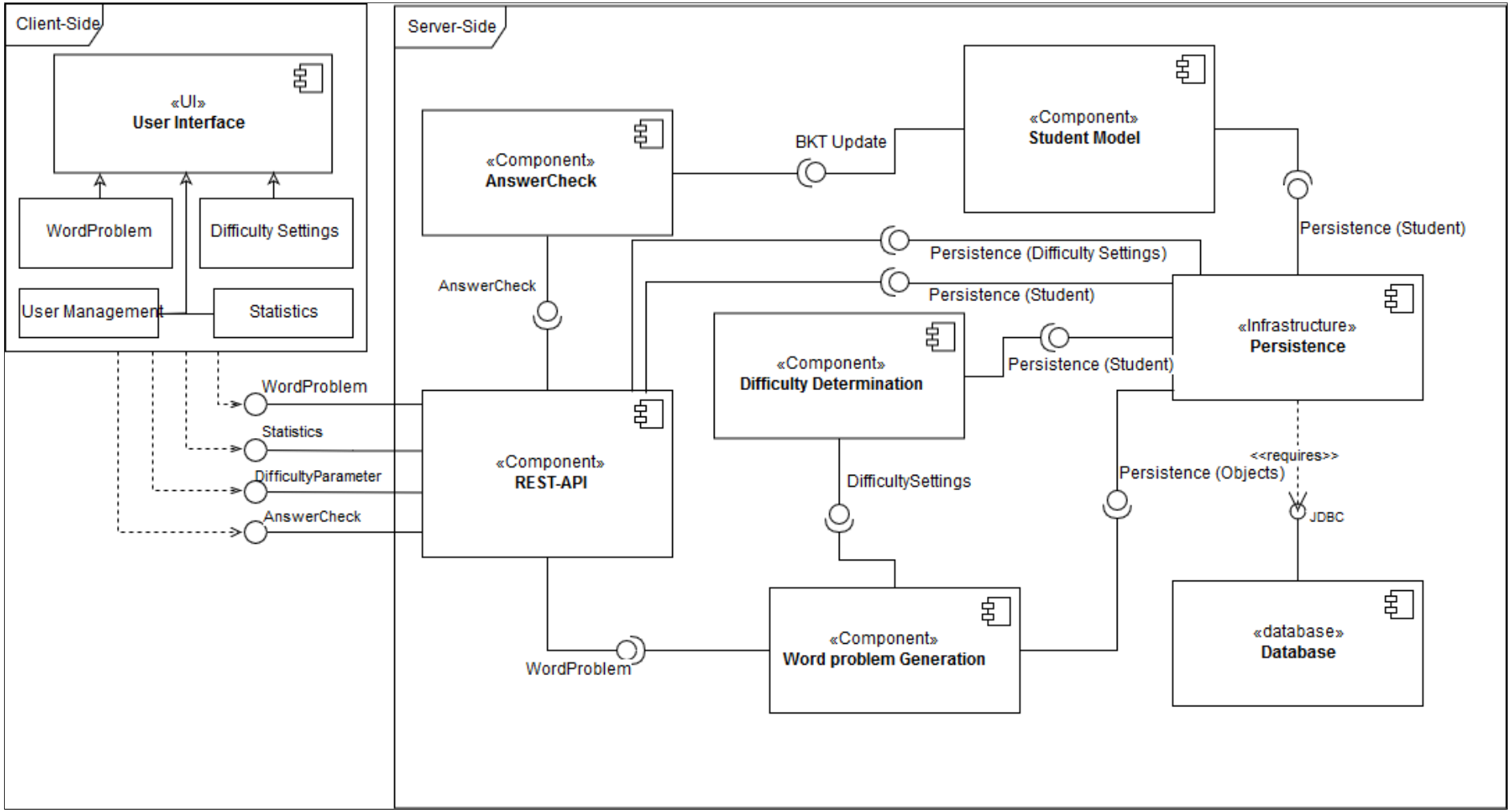


Figure 2: Architecture of the E-Learning System

The architecture is divided into the Client-Side and the Server-Side. The former is only used to display the User Interface to the user, including the actual display of the word problem, the settings page for the difficulty parameters, as well as two pages regarding the user management and an overview about certain statistics. The communication between the client and server takes place exclusively via a *REST-API*. The client can request the generation of word problems, information of students and the current difficulty parameters. Additionally, it can transmit the calculation results of the student which is called *AnswerCheck*. Within the Server-Side the API delegates the requests to various components. A request to generate a word problem gets transferred to the *Word Problem Generation* component. In order to create the task, the *Difficulty Determination* component is being asked to determine all necessary difficulty adjustments that have to be made. For that reason, the Difficulty Determination component requests the *Student Model* from the *Persistence* component and then determines the wanted difficulty settings. While adding *Templates* together, the Word Problem Generation component additionally queries the Persistence component to receive object data that it can use to fill the templates with. Once the word problem is generated, it gets sent back to the Client-Side by utilizing the REST-API. The user interface then displays the generated word problem. A student submitting a calculation result triggers the transfer of the calculation to the *AnswerCheck* component. In this component the calculation result is being evaluated, the results of which are used to create an update package in the Student Model component. The Persistence component supplies the Student Model component with the necessary information regarding the student that is stored. After the update process is finalized, the AnswerCheck component sends the evaluation result of the calculation back to the Client-Side, so that appropriate feedback can be displayed.

To make the process of generating a word probelm more clear, figure 3 shows a sequence diagram of that particular process.

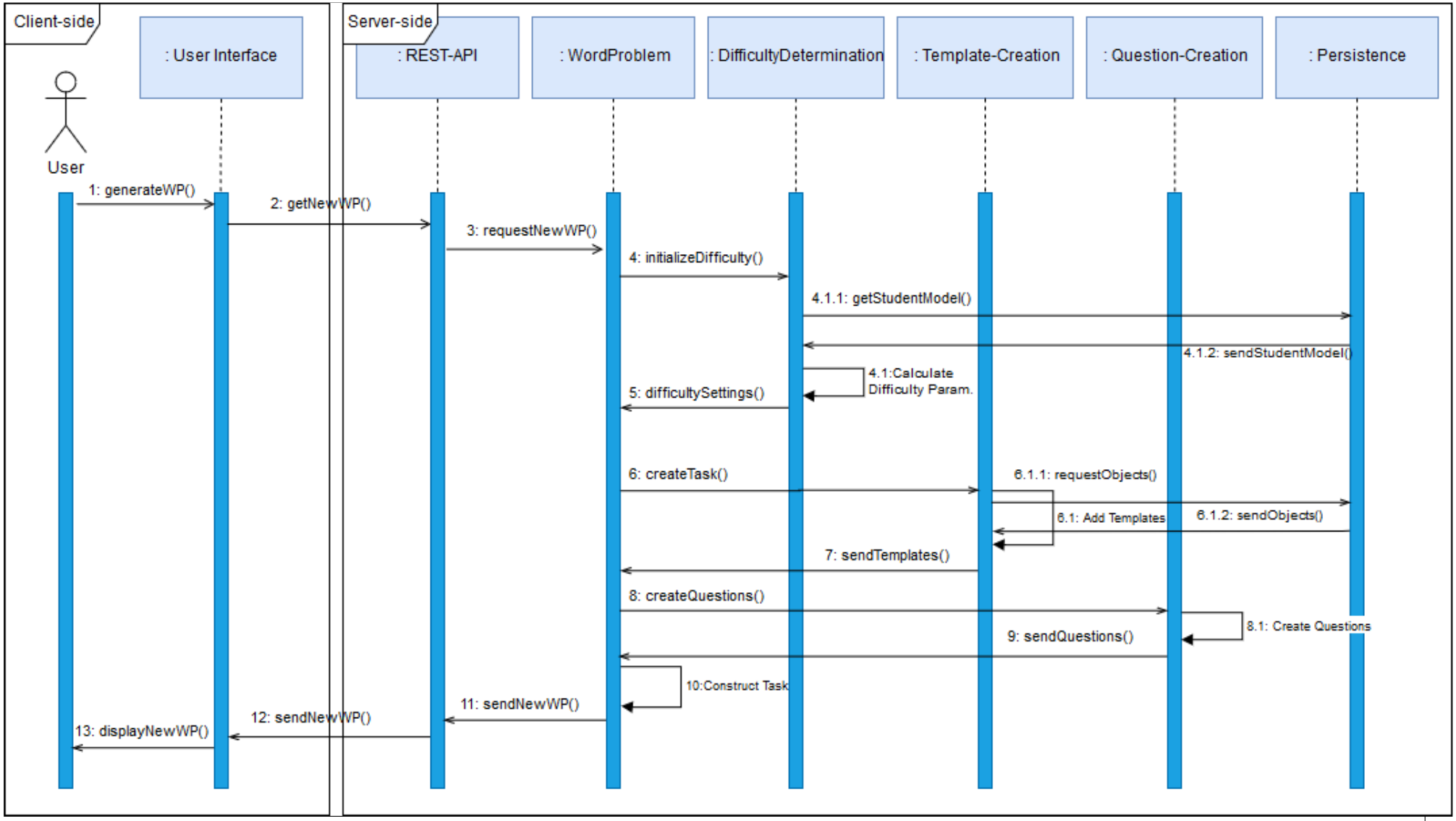


Figure 3: Sequence diagram of a word problem generation process

Once the user decides to generate a new word problem (1), a request is sent to the *REST-API* (2). The API delegates the request further to the *Word Problem* component (3). A requirement for the generation process are current difficulty parameters that are received by requesting the *Difficulty Determination* component (4). In that process, in order to calculate the difficulty parameters (4.1), the component first needs to request the current *Student Model* data from the *Persistence* component (4.1.1). Once the data is present (4.1.2), it can be used to calculate all the necessary difficulty parameters. Afterwards a bundled parameter package gets sent back to the Word Problem component (5), which then starts the process of creating the task (6). The task construction process consists of an iterative appending of compatible templates (6.1). The templates need object data to fill the necessary variables which get requested from the Persistence component (6.1.1). Once the task is created, the template is sent back to the Word Problem component (7). Now that the task is constructed, the component starts to create the questions based on what kind of information is present in the word problem using the *Question-Creation* component (8). Once the task and questions are present, the word problem can be constructed (10) and sent back to the Client-Side via the REST-API (11,12,13). The user is finally being presented with the Word Problem. This process begins again every time the user decides to create a new task.

4.3 Model of the student

The creation of individualized word problems requires a skill representation of the student in the system. For every student registration, a new database entry is being created with all the skills that were shown in 3.1.2. Bayesian Knowledge Tracing Mastery evaluates the probability of a student showing mastery in a specific skill. Measuring the various skills in a binary way, either the student shows proficiency in a certain skill or the student does not, is not sufficient for creating customized difficulty grades. As seen in section 2.5 the difficulty of, in this example addition, can be influenced by numerous factors like number magnitude, number property, decimal places and so on. Therefore a single binary representation of the addition skill is not feasible. Instead each skill was divided into a small-step representation that are defined by levels. With the exception of numerical complexity and decimal handling, every skill is divided in ten levels that each have different implications for the difficulty determination of the created task. A detailed explanation of what the skills and levels entail will be discussed later in the section 4.9.

Since the probability of mastery in BKT is represented with a range of 1% - 100% each skill level has a probability of the student mastering that skill. To illustrate this point, a student can have an Area Measurement skill mastery representation of 457 which translates to level 4 with 57% probability of mastery. Updating the skill via BKT continuously with a successful action would lead the probability to near 100%. It

is apparent that at some confidence level the level has to be increased. This question will be picked up later in section 4.8 because it is an important mechanism in the system.

4.4 Task Model

In order to understand what kind of information are logged and used to update the skills of the student, it is important to fully understand the task model and how the information representation takes place. To reiterate, a task is compromised of sub-tasks that are called question templates. An important distinction is to be made, the question templates should not be confused with the actual questions that will be generated based on the word problem that the student has to solve. The question templates are part of the word problem itself, as they are the building blocks of it. The accumulation of these templates eventually make up the task that is presented to the student.

Template construction: A question template is the most crucial part of the word problem. It contains valuable information that are later needed for further operations. For one it includes a variable text template, that is eventually displayed for the user. There are defined spaces within the template that need to be filled by objects extracted from the database. Additionally, all objects and attributes are saved, as well as the template affiliation to a concept like Area Measurement, Percent Calculation and General Number Handling. This information is important so that an association between the student's action regarding this template and the student's skill can be established. Furthermore each template saves a compatibility map that ascertains with what other templates it can be combined with.

Objects and attributes: The elements of the template that represent the variable content of the template are called objects. Objects represent realistic items such as a Zoo, Animal Shelter, Football Field and many more. These objects have certain characteristics that describe the objects which are called attributes. The objects present the main content the student interacts with. Instead of integrating the attributes within the question text itself, all containing objects with their attributes are displayed separately to avoid the question text becoming too obtuse and large. The choice of which objects are included in the template depend on difficulty parameters. The attributes are comprised of a name, a value and the unit the value brings with it. In addition to that, every attribute stores valid calculations of itself. A Zoo object for example contains attributes regarding the length, width and area. The length attribute stores the valid calculation of area divided by width. This is important for the question solving process as well as displaying help on how to calculate missing information. Figure 4 shows the display of object information.

Figure 4: Overview of the objects contained in the word problem - Missing attribute information is marked with a "?".

Task construction process: The process of generating a task starts with a determination of difficulty parameters, which for now will be ignored because it warrants further detailed explanations, seeing that it is a major focus of this thesis. Once the parameters are set, the desired length of the task is determined. As a reminder, there are three different types of question templates as shown in section 4.4, Base Templates, Connected Templates and Support Templates. It starts of with selecting a random Base Template for which there is currently only one choice. The Base Template itself has a compatibility map enabling the addition of compatible templates. Now an iterative process begins of adding connected templates that are compatible with current existing templates, which themselves again add their compatibility choices. Once the desired length of the task is reached, no further templates will be added and the construction of the task is finished.

Figure 5 shows an example task. In this particular task, each line break represents an additional question template.

Figure 5: Task example constructed by the combination of templates

Template Level : The software supports templates to have varying levels of difficulty, which results in the template adding different amounts of complexity to the word problem. To illustrate the usage of levels, here are two distinct examples of how levels are used. The first example shows that the levels determine the choice of the attribute the student has to calculate, while the second example shows an expanding question instead of adjusting the difficulty of the original sub-task.

Example 1: The software can generate tasks in which the student is asked to calculate how much food an institution like a Zoo has to buy in order to feed their animals. To determine how many animals the Zoo has, the student might get the information that the Zoo can support a certain amount of animals per $1m^2$. In this case the student might need to calculate the missing information of how big the Zoo is, if that information is currently missing. Depending on the level of this particular template, the choice of the area can change based on the levels. If the system creates an easy sub-task, it might formulate the question so that the Zoo supports a certain amount of animals per 1 metre length instead of being measured by area. For a more difficulty task the area could be replaced by volume, increasing the difficulty of the task for the student.

Example 2: The second possible influence of levels lies in the complexity of the questions and in how many steps the sub-task needs in order for it to be solved. The template *IncomeThroughVisitors* describes the income an institution makes because of customers visiting and leaving money there. The following shows the different iterations with various levels of the template:

> (i): xx% of inhabitants of <City> visit the <Institution> every year.
> (ii): xx% of inhabitants of <City> visit the <Institution> every year. A visitor spends xx Euro on average.
> (iii): xx% of inhabitants of <City> visit the <Institution> every year. A visitor spends xx Euro on average. The <Institution> has to pay xx% taxes on their income.
> (iv): xx% of inhabitants of <City> visit the <Institution> every year. xx% of the visitors spend xx Euro on average. The remaining visitors spend xx Euro. The <Institution> has to pay xx% taxes on their income.

The base of the template gives the student information about the amount of visitors the institution has every year (i). The default level, which is the first one, then further adds information about how much money is being spent per visitor (ii). The second level enhances the sub-task by adding complexity in the way of a taxation of the income the student has to account for in the calculation (iii). The final third level changes the original question text by adding complexity to the way on how to calculate the money that was spent at the institution (iv). It is apparent to see that each level adds complexity to the task the student has to solve, while also increasing the variety of the template by adding more sub-tasks and also influencing previous sub-tasks.

4.4.1 Template Overview

The following table shows the existing templates with their concept association, if it is a Base Template (BT), a Connected Template (CT) or a Supported Template (ST) as well as the compatibility with other templates. Notably *Required Template* means that only one of the required templates needs to exist in order for the template to be compatible.

ID	Name	Concept	Type	Required Template
1	StoreProductAnimal	Number Handling	BT	2,3,5,6,7
2	BuildThing	Area Measurement	CT	1,4
3	BuyingFood	Number Handling	CT	1
4	BuyingMaterial	Area Measurement	CT	1
5	CalculateAreaOfStore	Area Measurement	CT	1
6	IncomeThroughVisitors	Percent Calculation	CT	1
7	ProductTimeUntilDepletion	Number Handling	CT	1
8	ReplaceValueByOther	Number Handling	ST	1

Table 2: Concept Association and Compatibility of supported templates

Each implemented template offers the addition of various new scenarios to the word problem influencing it as a whole. Every template has variable parts that depend partly on random processes, as well as on specific difficulty parameters. The table below explains the actual wording of the templates with their variable parts, followed up by an explanation of the contents of every template (Translation of the template wording by the author). The addition of 'L1' displays the differences between levels of the template.

Name	Template Expression
StoreProductAnimal	A *<Institution>* is buying *<Food Product>* for *<Animal>*
BuildThing	Because of an increased offspring, a new building is being build with the dimensions *<Length>* x *<Width>* x *<Height>*
BuyingFood	The reserves of *<Food Product>* have to be restocked to have enough supply for *<Days>*. One package costs *<Cost>* Euro
BuyingMaterial	To build the extension building it is required to buy *<Material Quantity>* of *<Material>* per $1m^2$.
CalculateAreaOfStore	The *<Institution>* houses *<Animal Count>* per *<Dimension>*.
IncomeThroughVisitors L1	*<Ratio of Inhabitants>* of the inhabitants of *<City>* visit the store every year. A visitor spends on average *<Spending>* Euro.
IncomeThroughVisitors L2	*<Ratio of Inhabitants>* of the inhabitants of *<City>* visit the store every year. A visitor spends on average *<Spending>* Euro. Taxes in the amount of *<Taxes Percentage>* have to be paid on the transactions.
IncomeThroughVisitors L3	*<Ratio of Inhabitants>* of the inhabitants of *<City>* visit the store every year. *<Ratio of Visitors>* spend on average *<Spending>* Euro. The remaining visitors spend on average *<Spending>* Euro. Taxes in the amount of *<Taxes Percentage>* have to be paid on the transactions.
ProductTimeUntilDepletion	*<Package Count>* package can provide the *<Animal>* for *<Days>*
ReplaceValueByOther	The *<Attribute>* of *<Object 1>* is *<Multiplier>* as big as *<Object 2>*

Table 3: Template expressions of all supported Templates

- **StoreProductAnimal:** This template lays the foundation of the task. It adds an institution that houses animals and aims to buy food for them. It is compatible with most other templates.
- **BuildThing:** Adds the scenario that the institution needs to build a new building in order to create space for more animals to house.
- **BuyingFood:** The template adds the scenario of the store or institution of the base template needing to stock up their animal food reserves.
- **BuyingMaterial:** Adds the information that the institution needs to buy building material for the new construction project.
- **CalculateAreaOfStore:** This template informs the student of how many animals the institution houses per a specific geometrical size. The variable part of

this template lies in the selection of the geometrical size, be it length, width, area or volume. The selection depends on the chosen level of the template.

- **IncomeThroughVisitors:** Adds the scenario of consumers visiting the store in order to buy goods. The variable amount lies in the city the consumers are from. Supports multiple levels of complexity, adding for example the requirement of taxing the transaction process.
- **ProductTimeUntilDepletion:** Adds the information about how long a single package of food can provide for an animal.
- **ReplaceValueByOther:** This template influences other templates in a way that it replaces certain attributes by replacing them with the attributes of other objects, effectively adding one more step to the calculation of the original attribute of interest.

Even though most templates only need the base template to exist in order for them to be added, the templates still influence each other in various ways that result in different calculations that are needed to solve the questions that are generated based on the resulting word problem.

4.5 Information Extraction

The MySQL database has entries for different kinds of objects that are used while generating tasks. During the process of instantiating templates, queries to the database are made to extract objects that are suitable. To ensure that the various difficulty requirements are met, it is important to have a big enough pool of objects to choose from. In the current version of the software following object types are supported:

- Store: The store type includes the name and geometrical data like length, width and height of four typical institutions that house and sell animals.
- City: The city type includes the name and population count of eleven cities.
- Field: The field type includes the name and geometrical data like length, width and height. The eighteen objects are of various nature including football fields, smaller objects like a billiard table and skateboards.
- Product: The product type includes the name and cost of four food products for animals.
- Animal: The animal type includes the name and food requirement of said five animals.

To further illustrate the representation of an object type in the database, this is a snippet of the stored objects of the Field type:

Name	Area	Length	Width	Height
Tennisfield	1000	40	25	1
Footballfield	3575	65	55	2
Volleyballfield	700	35	20	2
Billiardtable	15	5	3	1
Swimming pool	2500	50	50	3
Desk	1.5	1.5	1	1
Garage	18	6	3	3
Skateboard	0.12	0.6	0.2	0.1

Table 4: Stored information about possible Objects with dimensional data

All the values accurately represent these objects in the real world. The source for these kinds of information was for the most part Wikipedia because it offers structured information in table format that is easy to parse. The City information was extracted from the Wikipedia page featuring a list of cities in Germany[6]. Eleven cities of low to medium population counts were extracted. The choice of only extracting a smaller part of the collection was to minimize the testing effort that results of a large collection of cities. The objective in the process of gathering object data was to get enough variation in objects and their values to demonstrate sufficiently enough how the the difficulty adjustments and therefore choice of objects in the templates entail. Extracting more cities would only further extend question variety, but is not necessary in this prototype. It is however trivially easy to expand the existing corpus of objects by just adding additional entries in the respective tables of object types.

4.6 Question Solving

One of the main premises of the proposed E-Learning Software is to give the student as much freedom as possible in the way the student wants to solve the presented task. Therefore the solving system needs to be as little restrictive as possible while still letting the student focus on the task. Seeing that the interface splits the attention on the question text and the object information separately, it was decided to implement two instances of calculation sections. First, the student can calculate missing attribute information. The second section allows the calculation of the requested question answers.

[6]https://en.wikipedia.org/wiki/List_of_cities_in_Germany_by_population Access on 03.06.2019.

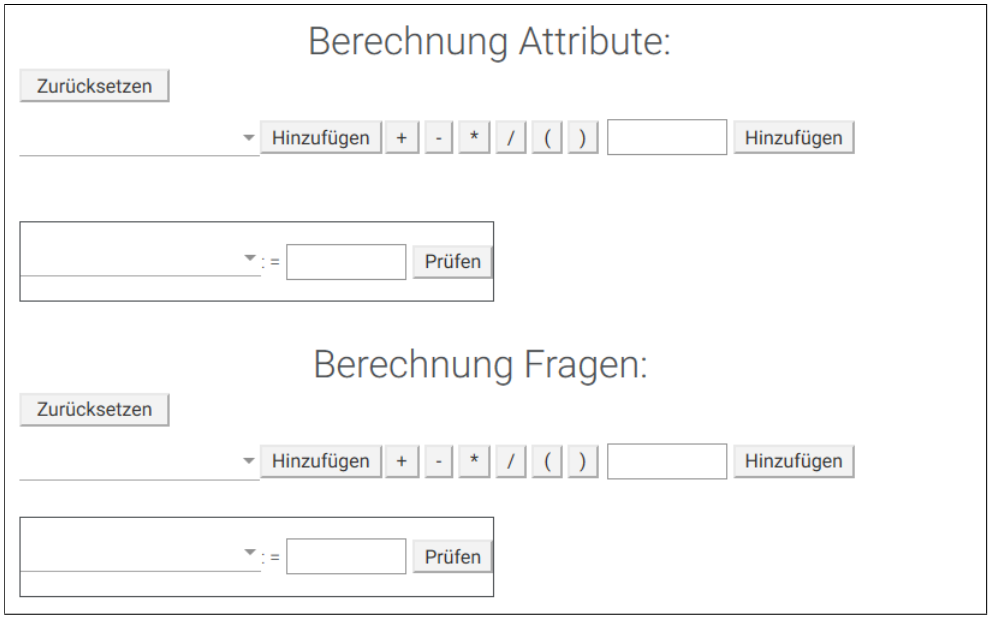


Figure 6: Overview of the Calculation System

Attribute Calculation: As explained in section 3.6, the student has the task to calculate missing attribute information to then eventually be able to solve the questions that are being asked. To make this apparent to the student, the object representation shows all the attributes the object has with its values. If a value is not present, it is displayed as a "?". The student has the task to construct a calculation. All the necessary tools that are needed in order to construct a valid calculation are given, including all the arithmetic operators, as well as the addition of brackets. The collection of these tools in the end resemble a standard calculator. Additionally, so that the student can use the existing calculated information, a drop down menu allows the student to see what kind of information is available that then can be added to the calculation. This is important to make it easier for the student to connect the different steps that have to be taken to solve all the sub-tasks and eventually the overall tasks. Another drop down menu is offered to choose the currently missing attribute information the student wants to calculate.

Question Calculation: The section that allows the calculation of questions is largely similar to the attribute calculation one. The main difference is that instead of choosing what attribute value the student wants to calculate, a drop down menu instead offers to choose one of the asked questions to solve. For all the calculations both mouse and keyboard inputs are supported, as well as easy fault recovery in the case of a mistake in inputs by the student.

Help and support: The decisions made revolving implementing a support and guidance system for the student was largely influenced by the initial guided interview. Contextualized and small-step help was an emphasized demand by the expert (see

Add. Usability Test U17.2). For that reason help is being displayed in two ways. First of all general help about how the system works is being displayed in both of the sections, dividing the question and the object representations. It includes an explanation on how the general approach of the student shall look like. Earlier in this chapter it was mentioned that every attribute stores information about how it can be calculated. Once the student has requested the help for a specific attribute calculation, the optimal solution of the attribute to be calculated is shown to the student. An identical approach was used for contextualized question help in the way that questions itself store the optimal way of solving them. Seeing that sometimes additional steps are necessary to calculate information that is missing, it was initially planned to show all the steps that are needed to solve the task. To illustrate how this works, imagine the following question:

> "How many animals does the Zoo house?" with the word problem text "The Zoo houses 3 animals per $2m^2$" Additional object information about the width and length is available with the area being unknown.

The obvious solution to this question is to calculate 3 times the area of the zoo. This is called the *direct calculation help*. However it is apparent that the area information is not available. The direct calculation help for the area of the zoo is calculated by multiplying the width and the length of the zoo. The *detailed calculation help* is being generated by the system by a search through all objects within a question to check if they are known or unknown. In the case of the information being unknown, the attribute in the direct calculation help is replaced by the underlying direct calculation help. This process is recursively done until all attribute information is known. The final detailed calculation help would look as follows:

> (i) Direct Calculation Help: "3 · Zoo:Area"
> (ii) Detailed Calculation Help: "3 · (Zoo:Width · Zoo:Length)"

This approach however had two problems. The first problem is encountered as soon as the amount of steps needed to calculate the information becomes huge, making the detailed calculation help become bloated and confusing. The second problem arises that it is impossible for the system to know exactly where the student needed help, because the system would just show the complete necessary calculation. For that reason it was not implemented in the final build. The help currently assumes that all the objects involved in the final calculation of the object of interest are known. This could lead to the student needing to backtrack the help for the containing objects if further help is needed. This however is not a problem because it gives valuable logging data, because contrary to the expansive help, the system can see exactly what kind of help was needed by the student.

The guided interview also revealed that students are keen to resort to using help as soon as they encounter a problem. For that reason the contextual help is only displayed once the student fails once. After the student fails a second time the option to automatically solve the sub-task is offered. Once a third failure occurs, the system automatically solves the task, so that the student can continue further. Once the student submits the calculation for review, the logging process is invoked. After all questions are answered, a new task is automatically generated.

Feedback: The student receives feedback based on what mistakes were made during the calculation construction process and the calculation itself. There are three types of feedback that are triggered based on three situations. Adequate feedback is displayed if the student constructs the wrong calculation and if the submitted result is wrong. Obviously it is possible that the construction of the calculation was done correctly but the student could not successfully compute the calculation that was put up (see (i)). The student then gets displayed the error that was made, in this case the construction of the task. If the student both constructed wrongly, as well as doing mistakes in the calculation itself, the feedback will show that (see (ii)). The last case is that the construction of the calculation and insertion of values was done incorrectly, but the submitted result correlates to the calculation construct, then the student will be presented with adequate feedback (see (iii)).

(i): "Wrong calculation but correct values!"
(ii): "Wrong calculation and wrong values!"
(iii):"Correct calculation but wrong values!"

4.7 Logging

At this point the student model is initialized and the student has a representation of skills. Bayesian Knowledge Tracing requires positive or negative actions to update the skill mastery of a skill. For that reason it is important to have relevant tasks the student has to try to solve. The behaviour of the student engaging with these tasks have to be logged. There are three dominant question regarding the logging processes. Firstly it is important when the logging takes place, secondly what data is logged and lastly how the data is logged.

When does logging take place: Every time the student submits a calculation for it to be verified of its correctness, the calculation is being analysed. That includes the calculation of missing attribute information, as well as calculating the result of a question that is being asked. Additional data is logged throughout the whole solving

process regarding metadata that is relevant for the whole assignment and not specific single calculations.

What kind of data is logged: A submitted calculation is split into three distinct parts. First, the attribute or question the student wants to calculate. The second part encompasses the calculation itself, which is further split into single calculations. For illustration, the calculation "5 · 4 + 10 · 7" contains three single calculations namely 5 · 4, 10 · 7 and 20 + 70. The single calculations basically represent all the arithmetic calculations a student has to do, respecting order of operators and usage of brackets. These single calculations are then stored for further analysis. The third part consists of the result the student calculated.

The chosen attribute that is to be calculated is crucial in collecting logging data. The attribute contains data about the affiliation of the attribute to a mathematical concept, the optimal solution for the attribute including what formula to use and which values are relevant to the calculation.

4.7.1 Calculation Analysis

Because of the fact that the student can use whichever approach for the calculation they wish, it is important to also analyse the calculation and not only if the result is correct or incorrect. The calculation itself is parsed and calculated and consequentially compared with the result the student entered. That leads to the determination if the arithmetic operations were done correctly, regardless of if the end result is the actual correct result. Vice versa it is also important to log if the student constructed the correct calculation that would lead to the correct result but failed to complete the arithmetic operations successfully. In order to transform the logging information into data that can be used for updating the skills of the student, the three parts of the calculation, namely the attribute, the calculation and the end result have to go through an analysation process.

Clearly the system has to evaluate if the result the student submitted was calculated correctly or not. There are three steps to that. First, object attributes and questions both store their results that the student is tasked to find out. However due to the usage of objects in a database with fixed values, it is possible for calculations to become too unintentionally difficulty. For that reason in those situations rounded answers are accepted. To illustrate this point, imagine the following arithmetic operation: 92.5 / 20 = 4.625. Depending on the difficulty that was determined by the system to be as best customized to the student's needs, it is possible that 4.6 or even 5 will be accepted as an answer. The third option evaluates if the answer given does not concord with the desired end result, but is the correctly calculated result based on the faulty constructed

calculation the student created.

One of the skills the student has show proficiency in is the correct usage of suitable formulas. In order to do that, each valid calculation that is allowed for the calculation of an attribute or a question is saved within the respective attributes or questions. These valid calculations are affiliated with formulas that are shown as examples in the figure below.

Formula	Required Attributes	Operator	Ordering
Replacement	Variable	Multiplication	None
Area	Width, Length	Multiplication	None
Width	Area, Length	Division	Area > Length
Length	Area, Width	Division	Area > Width
Volume	Length, Width, Height	Multiplication	None
Income Visitors	Population, Customer Spending	Multiplication	None
Expenses	Food Expense, Material Expense	Addition	None
Profit	Expenses, Income	Addition	None
Food Calculation	Animal Count, Food Requirement	Multiplication	None
Buying Materials	Area, Price	Multiplication	None

Table 5: Collection of supported formulas

The system compares the constructed calculation the student created with all the valid calculations that are stored within the attributes or questions. Seeing that there are multiple ways of ordering the values, how to use brackets and other variable factors, it is not feasible to do a straight comparison of the calculations. Rather what is done is the evaluation if the necessary attributes are present, e.g. *Length and Width* for an area calculation, if the correct operators were used and if the ordering in single calculation was done correctly, e.g. *Length = Area / Width* instead of *Length = Width / Area*. If these evaluations all turn out to be correctly handled by the student, the usage of formulas is deemed correct.

In the next step it is analysed if the student used the correct values that have to be inserted into the calculation. This includes attributes and non-attribute values. If every necessary value is present, the correct insertion of values is deemed correct.

Another feat that is similar to the correct choice of values to insert, is the ability to dismiss distracting information that was planted to confuse the student. At the time of task generation, every attribute is marked if it is a distractor or not. The system then simply checks if the calculation contains any one of these attributes. If none are to be found, the dismissal of distracting information is deemed correct.

Following that, another crucial step is to store which concept the calculation is associated too, because it is important to update the skills of the student accurately and only for the concept they currently engaged in.

As mentioned before the arithmetic operations are divided into single calculations in order to analyse every calculation the student encounters. Additional information about the calculation is stored in the form of the arithmetic operation types (Addition, Subtraction, Multiplication or Division), as well as certain properties of the calculation, including the occurrence of decimal numbers, the numerical complexity and the number magnitude. Each of these properties are assigned a certain level which will be explained in section 4.9.

The resulting information package then contains the following data.

- Correct Answer: Yes/No
- Correct Rounded Answer: Yes/No
- Correct User Calculation Result: Yes/No
- Correct Formula Usage: Yes/No
- Correct Value Usage: Yes/No
- Usage of distracting Information: Yes/No
- Concept Affiliation: General Number Handling/Area Measurement/Percentage Calculation
- Single Calculation Data:
  - Decimal Complexity Level 1-4
  - Number Magnitude Level 1-10
  - Numerical Complxity Level 1-10

Calculation Information: It is important to note that the amount of tries the student needs to solve a task, as well as how much help was needed, influences the update packet. Using the automatic solving of the task or the help function leads to a negative assessment of the student's skills. This procedure is the same for the calculation of missing attribute information, as well as solving the final questions. Additionally, there is another step that is being done when all the questions are answered. While the student is in the process of solving the tasks, the software tracks the amount of help, how often automatic solving of the tasks was used and the amount of steps that were needed to finish the word problem. The collection of all information that is mentioned in this section is then transferred to the next step of updating the skills of the student.

4.8 Updating skills

At this point the logging information are transformed in a format that is necessary for Bayesian Knowledge Tracing, boolean values. These information packages are generated every time the student submits a result, be it for an attribute or question calculation. The following figure 7 shows the process of logging certain behaviour from a student that is then used to update the respective student model.

Once the student constructed a calculation and is submitting the results (1), the whole calculation is sent to the server via the REST-API (2). It then gets delegated to the *AnswerCheck* component that is responsible for analysing and evaluating the student's constructed calculation. In this component, a package gets created that contains information about the student's performance regarding the correct choice of values in the calculation, the correct formula being used, if the result is correct and many more (3.1). Once the results are sent back (4) the process of updating the *Student Model* is started (5). The *BKT Evaluation* component is there for creating updates that can be used for *Bayesian Knowledge Tracing*. It is comprised of creating update packages regarding skills that are related to operational performances (5.1) and conceptual performances (5.2). On completion the packages are sent to the *BKT Update* component (5.3) which is responsible for querying the Persistence (5.4) in order to get the current *Student Model* data. The component then compares the evaluation results with the current level of the student's skills and determines if and to what degree updates have to be carried out (5.6). The Persistence component then realizes the database updates. Once the update process is finished, the AnswerCheck component transfers the evaluation results to the client (6,7). The client analyses the evaluation data and determines appropriate feedback to display to the student (8).

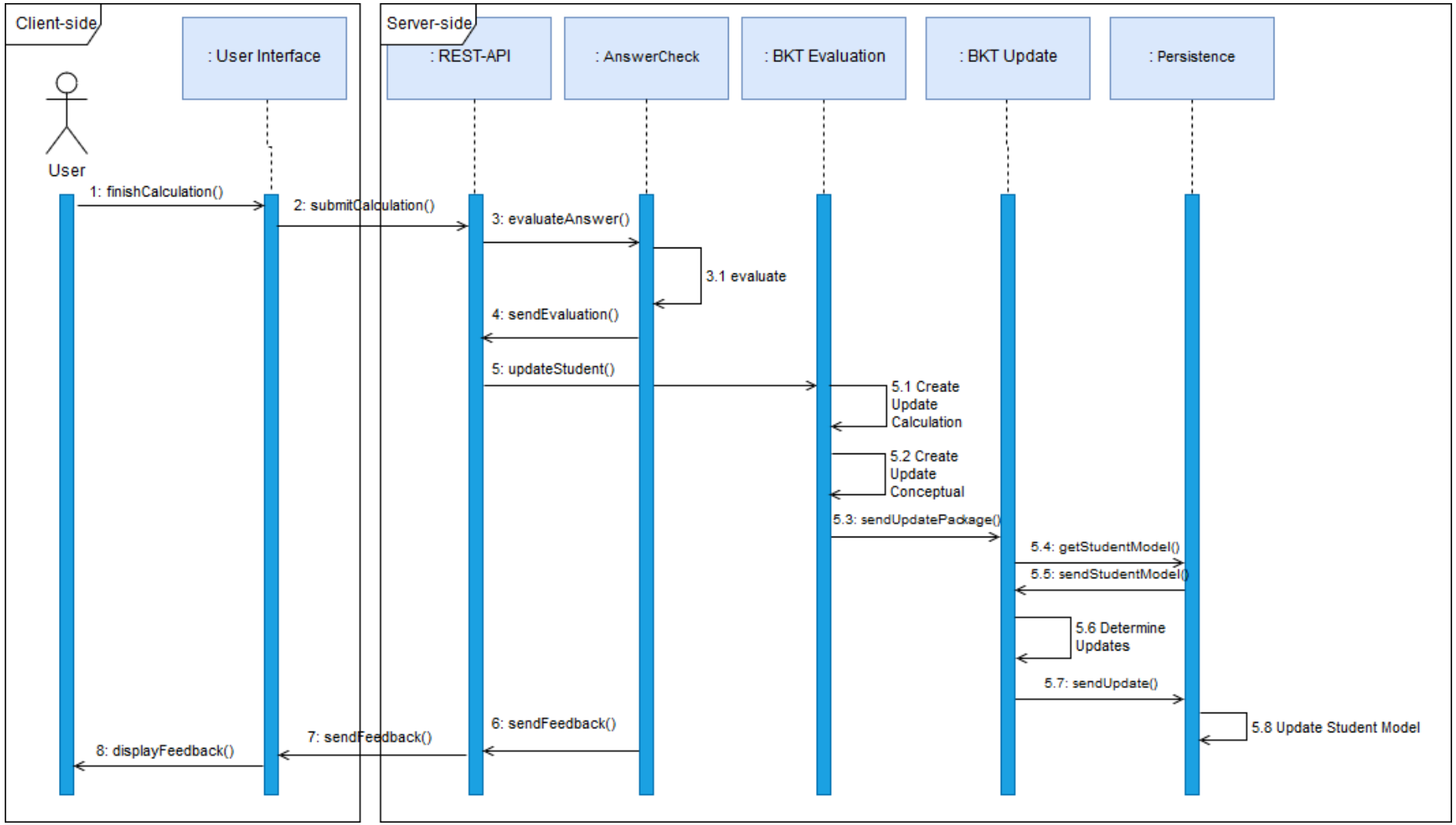


Figure 7: Sequence diagram showing the process of updating a student model

Building the update package: The update process is separated in the update of arithmetic and conceptually related skills. At this point in the process single calculation data was extracted from the calculation. The system now determines the most difficult single calculation of every arithmetic operation, resulting in a maximum of four possible update triggers. This is done as to avoid an overload of the update mechanic. It is more reasonable to only update a skill once per calculation or situations can arise where the difficulty in the next task spikes. A solution to that would be to order the single calculations by descending difficulty and then after the initial update was completed, disregard all remaining single calculations that do not meet the new determined difficulty threshold.

Once the relevant skill target for an update of mastery is identified and all the necessary information is at hand, the system extracts the level of the operational skills namely *Decimal Handling*, *Number Magnitude* and *Numerical Complexity*. As a reminder, if the Number Magnitude skill evaluation of the student is at 741, then the current level is 7, while the progress to the next level is at 41%. The extraction of the current levels is done in order to compare the difficulty the student surpassed in his mastery levels to the difficulty level of the analysed single calculations. The correct handling of a calculation is only regarded as a trigger for an update, if the difficulty level of the task is greater than the current level. For illustration purposes an example calculation is $1000 \cdot 1000$ and the student submitted a correct result. The number magnitude clearly is above average if we assume a lower grade student. The complexity of the number however is low. Assuming an average student, this means that the *Number Magnitude* level is being adjusted while the *Numerical Complexity* level is not. For a more detailed explanation of how the difficulty evaluation is done, see section 4.11.

It is important to note however that failures of the student to solve a task always triggers a negative update event regardless of the levels. This is done because testing showed that more difficult tasks for the student always exist because multiple subtasks in combination lead to more complicated solutions. The task generation process however is designed in a way that only a tiny amount of sub-tasks could be too difficult for the student to solve. For that reason it is to be expected that the student should be able to solve most of task that are presented and a failure of it leads to a decrease in overall difficulty. If this occurs, the length and complexity of the task should be adjusted by the system automatically. Clearly it could also be possible for the system to only trigger a negative event when the student level is higher or equal to the level of the sub-task. This could however lead to the problem of not acquiring a lot of updates within a generated task, because of how carry problems, that a word problem features, work. It is not feasible to only feature tasks that are exactly the same level as the student's level. Therefore this mechanic functions as a safeguard in regards to that problem.

After the update regarding the calculation itself, the update package gets associated with one of the supported mathematical concepts. Based on if the student correctly mastered the task, the skill regarding the overall expertise of said concept is updated positively or negatively. The last update package addition is in regards to the correct usage of formulas, correct value insertion and the dismissal of distracting information.

Calculation of the new mastery: Once the packages are build and binary decisions were made to either positively or negatively update the skill, the mastery level is updated with Bayesian Knowledge Tracing as shown in section 3.3.2. BKT mastery update formulas require a probability of initial mastery, which is extracted from the current skill of the student. For that the level is taken out of the equation, so that the current progress defines the current mastery probability of a skill. Once the respective calculation for applying a skill correctly or wrongly is done, a new mastery probability is determined. Seeing that there are different levels of a skill, usually ten, at some point the decision to "level up" the skill has to be made. For that reason a sensitivity setting, that can also be manually modified by the teacher or even by the student if demanded was implemented, that determines the cutoff for both increasing and decreasing a level. The following limits were chosen:

Sensibility	Threshold High
1	98%
2	90%
3	80%
4	70%

Table 6: Thresholds to reach a higher level

Sensibility	Threshold Low
1	11%
2	15%
3	20%
4	25%

Table 7: Thresholds to reach a lower level

The respective values and levels seem arbitrary but are the result of testing with the goal of getting a linear progression of how many calculations of tasks the student needs to engage with in order to get meaningful difficulty changes. It is recommended to adjust this setting based on if the student is a new or a more experienced user, as so to reach the equilibrium of difficulty faster or slower if the system already had plenty of opportunity to analyse the student's behaviour and therefore adjusted the difficulty well. In that case less swingy difficulty adjustments are preferable.

4.9 Difficulty Settings

The task generation process requires difficulty parameters to create customized word problems. This chapter therefore first explains all the difficulty parameters that are available and how they influence the task creation process.

Concept Difficulty: Each supported mathematical concept in the system has its own difficulty parameter ranging from 1%-100%. The reason for this choice of parameter

presentation is that a more finely tuned adjustment is possible. Since this parameter does not in fact function in defined steps like most of the other parameters, a percentage representation is more suited.

4.10 Conceptual Difficulty Parameters:

There are four distinct difficulty parameters that define the conceptual difficulty of a task: *Replacement of attributes*, *Length of the task*, *Question Difficulty* and *Amount of Distractors*. Each parameter has ten levels of difficulty.

Replacement of attributes: This parameter defines the difficulty of changes to the generated word problem that increase the steps needed to find missing attribute information. These are possible generated tasks that show the effect of this parameter:

> (i) The Zoo houses 3 animals per $1m^2$.
> (ii) The Zoo houses 3 animals per $1m^2$. The Zoo is about 2 times as big as a football field.
> (iii) The Zoo houses 3 animals per $1m^2$. The Zoo is about 2 times as big as a football field. The football field is about 25 times as big as a volleyball field.

Assuming (i) is the whole task, there will be additional information present about how big the Zoo is area wise. This difficulty parameter can change that kind of task construct by replacing the area value with an attribute from another object that represents a multiplier to the original area value resulting in (ii). In this case the student now has to solve an additional step, calculating the area of the Zoo, before the calculation of the amount of animals in the Zoo is even possible. Seeing that this parameter has multiple levels, it increases the amount of replacements within a task. Even replacements of replacements are possible, adding additional steps for the student's calculation. For this particular example another addition could be to state the information that a football field is about 25 times as big as a volleyball field.

Length of the task: This parameter influences the overall length of the generated task. It has ten possible levels. The determined level roughly correlates with the amount of question templates that are added to the word problem, but is influenced by many factors. Starting off, the set level for the length of the task gets multiplied by a factor of 100 which will be called *Length Threshold* from now on. Now an iterative process of adding templates begins, while each addition subtracts from the initial threshold. There are difficulty settings for each supported mathematical concept that influence the subtraction process. The thought process is that if a student is very proficient in a certain concept, a question template associated with that concept should not increase

the difficulty of the task as much as a question template that associates to a concept the student has problems with. Seeing that the difficulty parameters of the concepts have a range of 1-100, 50 is seen as the average value which would lead to the question template that is considered to be added to the task, decrease exactly 100 points from the initial determined Length Threshold.

The following pseudo code shows the calculation of how many points are deducted from the current *Length Points*. These points indicate the current progress towards reaching said threshold.

```
1 if (skill_Proficiency <= 50) {
2         int length_Point_Deduction = 100 + (50 - skill_Proficiency) * 2;
3 }
```

Listing 1: Length Point Deductions with a more difficult concept

```
1 if (skill_Proficiency > 50) {
2         int length_Point_Deduction = 100 - (skill_Proficiency - 50) * 2;
3 }
```

Listing 2: Length Point Deductions with a less difficult concept

If the proficiency of the concept that is relevant in this calculation is lower than 50, more points will be deducted from the Length Points because the addition of the template associated to this concept leads to a high difficulty increase and vice versa if the proficiency is higher than 50. Then less Length Points are deducted, effectively increasing the length of the task because the student has less problems with tasks related to the respective mathematical concept. The multiplier of 2 was added so that a maximum proficiency of a concept leads to no effect on the length of the task meaning that the template is essentially ignored when it comes to the difficulty parameter of *Length of the task*. A minimum proficiency of a concept would lead to a doubling of the effect it has on the length effectively making the template count double and therefore decreasing the overall length of the task.

As explained in section 3.4.3, question templates can have varying levels that correlate to a higher difficulty within the template. The choice of the level depends on the concept proficiency of the student. In general, the higher the proficiency, the higher the chance of using a higher level is. This added randomness is done to avoid a very predictive behaviour. This process follows the earlier mentioned approach of decreasing or increasing the length of the task based on how different the added question template is. Seeing that a higher level leads to a higher difficulty, the choice of an increased level leads to a higher deduction in the earlier determined *Length Threshold*. The specific value that is being subtracted depends on the disparity between the concept skill evaluation of the student and the cutoff percentage for a specific level. The cutoffs are chosen

linearly, as in 10%-34% concept proficiency allows only for a question template level 1, a concept proficiency of 35%-70% allows for template level 1 and 2 while a higher proficiency enables the use of template level 2 and 3. To further illustrate this: student A has a proficiency in Concept X of 100% (level 10). Student B has a proficiency in Concept X of 80% (level 8). Both students get presented a level 3 question template. In the case of student B more points will be deducted from the Length Points than it is the case for student A. To illustrate these mechanics, this is an example of the process of adding question templates with certain assumptions of the student:

- General Number Handling Skill (GNH): 40%
- Area Measurement Skill (AM): 55%
- Percent Calculation Skill (PC): 90%
- Length of the task: Level 3 (300 Length Points)

Event	Length Deductions	Remaining LP
Base Templ.	-	300
Con. Templ. GNH	$100 + ((50 - 40) \cdot 2) = 120$	180
Con. Templ. AM Level 2	$100 - ((55 - 50) \cdot 2) + ((70 - 55) \cdot 2) = 120$	60
Con. Templ. PC Level 3	$100 - ((90 - 50) \cdot 2) + ((100 - 90) \cdot 2) = 40$	20

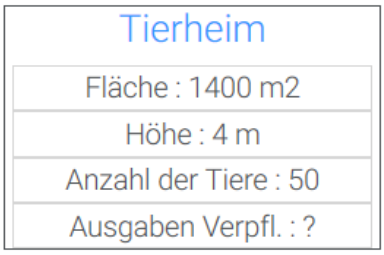
Table 8: Example of the iterative process of deciding the length of the task

Question Difficulty: As explained in section 3.4.5 the question difficulty parameter determines how complex the questions are that the student has to calculate an answer for. As soon as a task is generated, all possible questions are created (see section 4.12). The set difficulty of the questions then define which of those questions eventually end up being asked. All possible questions are sorted starting with the easiest question. The question difficulty parameter is then used to either select a low difficulty question or a high one. In general the higher the question difficulty is set to, the higher base difficulty the question has, as well as require a higher amount of steps to solve. More on how questions generated and how the question difficulty parameters plays into the selection process in section 4.12.

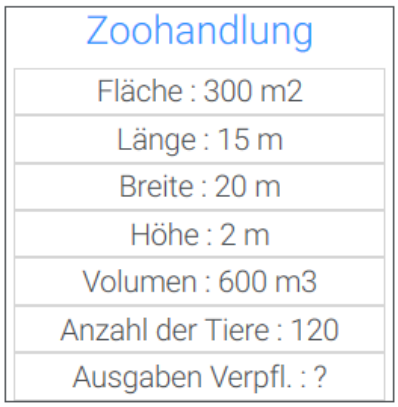
Amount of Distractors: Objects have a fixed minimum of attributes they hold. Others are added dynamically in the task generation process like income and expenses. Some of the fixed attributes are not relevant to the task. The *Amount of Distractors* parameter determines how many of these irrelevant information are still being displayed in order to force the student to dismiss those. The technical implementation of adding distractors is as follows. First, a list of possible distractors is created by searching all, in the task available, attributes. Since every attribute stores a binary value if it is

a distractor or not, the amount of distracting attributes can easily be counted. This difficulty parameter has a range of 1%-100%. Again, a level representation of this setting would not make sense because of the way it influences the distractor amount in a very proportional way. The amount of displayed distractors is linear to the parameter percentage, e.g. 50% would result in exactly half of the distractors being shown.

To illustrate two different settings of this parameter, on the left side of figure 8 the *Amount of Distractors* is set to a low percentage compared to a high percentage setting on the right side of the figure.



(a) Low Distractor Count



(b) High Distractor Count

Figure 8: Varying amounts of distracting information

In this example (a) shows the object of an animal store that only shows four attributes even though information about the *Length*, *Width* and *Volume* is theoretically available but not displayed because the difficulty parameter regarding distracting information is set to a low level. (b) shows a similar example but where a high amount of distractors was chosen. Additional information that is not relevant to the word problem is displayed.

4.11 Arithmetic Difficulty Parameters:

Every arithmetic operation has its own difficulty settings. An overall difficulty can be set from 1%-100%. Additionally, more detailed parameters can be set, namely *Calculation with decimals*, *Numerical Complexity* and *Number Magnitude*, with the latter only having ten levels, while the first two choices each have four levels. These parameters influence the creation of sub-tasks and are actively queried by the instantiation process of a template. Templates have numerous variable spots to fill with either suitable objects or numbers in order to create a calculation that fits the desired difficulty level. There are two major ways the arithmetic difficulty parameters are used. First, templates include numerous arithmetic operations with variable numbers. These numbers are determined by tasking the system to generate suitable numbers that fit in the

calculation. Secondly, these parameters are used to analyse the single calculations the student submits to determine the kind of difficulty the student engaged in.

Generation of Numbers: There are multiple examples for the need to generate numbers while question templates are being added to the word problem. The amount of days the zoo has to buy food for, how many animals the store can house, how much the materials cost to build a new building and many more. The generation of these numbers is not random and deeply ingrained in the difficulty adjustment process. In order to explain the influence of the three arithmetic difficulty parameters, it is important to elaborate how the different levels come into play.

Level	Number Magnitude	Calculation with Decimals	Numerical Complexity
1	1 - 2	0 decimal places	$x = 2$
2	3 - 5	1 decimal places	$x \% 2 = 0$
3	6 - 10	2 decimal places	$x \% 5 = 0$
4	11 - 15	3 decimal places	$x \% 2 \neq 0$, $x \% 5 \neq 0$
5	16 - 20		
6	21 - 30		
7	31 - 40		
8	41 - 50		
9	51 - 99		
10	99+		

Table 9: Breakdown of arithmetic difficulty parameter levels

The values for the *Number Magnitude* parameter are loosely derived from the findings in Daróczy et al. (2015) and Kadosh et al. (2014). It is important to note that the big difficulty increases come from the jump of digits, because it might require the student to use different approaches to solve the task (Kadosh et al., 2014). In this thesis it was experimented to use numbers with much higher magnitudes including 4-digit numbers. That approach has resulted in arguably insurmountable problems. The combination of multiple question template that used numbers with such a high number magnitude led to limits which eventually resulted in the high magnitude numbers never being used. The testing process eventually led to this selection of numbers, which are however not set in stone, but currently deliver a desired progression of difficulty in arithmetic operations without the combination of tasks leading to absurdly big numbers. It can be argued that the cutoffs for these levels are fairly arbitrary. That however should not be a problem, because the system will eventually find a fitting equilibrium for the student for a certain set of magnitude parameters. The actual level denomination the numbers are categorized in, do not effect the task creation process. The only important thing to consider here is that the degree of partitioning of the numbers directly correlate with how good the system can adjust itself to the student.

The level partitioning of the *Calculation with Decimals* parameter is fairly straight forward. The first default level disables any kind of calculation with decimals. The remainder of the levels enable the calculation with decimals. Seeing that a requirement of generating numbers that have exactly the desired amount of decimal places is not trivial and directly combats with the idea of using objects with fixed values, the calculation instead shows the student rounded values either to the first, second or third decimal place, as well as allowing the student to submit a rounded result. To keep the premise of using real world objects with realistic values, the actual values are shown in the object overview instead of the rounded result.

Finding suitable numbers and objects: A question template often times has boundaries when it comes to suitable numbers and objects. Clearly if the template talks about the housing of animals, then the template can not utilize objects that do not fit that requirement such as schools or supermarkets. The boundaries in that case are defined by what object types are allowed. A similar approach is used for finding suitable numbers. The generation of numbers technically is not limited in its magnitude assuming that the student's relevant skill levels are high enough. For the sake of realism and continuity issues of the task, it is important to introduce a boundary for the minimum and maximum of the generated number. To illustrate the two possible issues, imagine the example "A zoo houses x animals per $1m^2$." It could be a valid approach to insert a number in the 3-digit range in order to achieve the desired difficulty level of the multiplication task. That would however not be realistic at all, so therefore a maximum boundary is introduced to achieve realistic outcomes despite the varying difficulty parameters. For that reason there are multiple avenues of finding suitable numbers so that possible highly skilled students do get tasks that are adjusted to their skill level. The second problem that could arise by choosing numbers that are too large, is that the combination of question templates and therefore calculation steps could lead to very big numbers.

The process of choosing the best suitable object is determined by the specific template. The *Base Template* for example paves the way for all the remaining additions to the word problem. Which object is taken influences the general magnitude of numbers that will be used in the following calculations. In this case all available objects are sorted by magnitude. The system then analyses the average concept proficiency of the student expressed in a relative percentage and then picks the store that correlates to said percentage. Another example showing how the requirements for choosing a suitable object can vary, is with the use of the *Support Template: Replacement of attributes*. In order to replace an attribute with another one, a desired multiplier is being determined by the process laid out earlier. Afterwards the ratio of all possible replacements to the

attribute that is to be replaced, is compared. The one closest to the desired multiplier is picked and used as the replacement.

Determination of difficulty levels of single calculations: As explained in section 4.7 the constructed calculations by the student are analysed and partitioned into single calculations. For the process of updating skills (see section 4.8), it is important to analyse the level of difficulty that a single calculation represents. For that reason the system evaluates both values of the calculation and determines the *number magnitude* of the number, how many *decimal places* are being used and how *complex* the numbers are. The highest levels in the respective parameters are then used for the updating process.

Demonstration of the effect of varying difficulty parameters: In order to visualize the effects changing difficulty parameters can have on the generated word problems, the following figures 9, 10 and 11 display three word problems that were generated with increasingly difficult parameters.

Ein Tierheim kauft Futter für Meerschweinchen.
Es muss Futter für 1 Tage nachgekauft werden. Für eine Packung muss 3 Euro ausgegeben werden.
3 Packungen Futter können ein Meerschweinchen 1 Tage lang versorgen.

Figure 9: Generated word problem of easy difficulty

The word problem in figure 9 was generated with the use of difficulty parameters that represent an under-average level of expertise. The length of the task with only three added templates is manageable. Additionally it is clear that the arithmetic operations that result out of the task are easy to calculate because the magnitude of the numbers are in the single digits.

Ein Tierheim kauft Bio-Futter für Schildkröten. Das Tierheim hat eine Länge von ca. 20 Billiardtischen.
15 Packungen Bio-Futter können ein Schildkröten 3 Tage lang versorgen.
Das Tierheim bringt pro m Länge 8 Schildkröten unter.
10% der Einwohner in Varel besuchen das Tierheim jedes Jahr. Ein Besucher gibt im Durchschnitt 14 Euro aus.
Aufgrund starkem Nachwuchs soll nun ein neues Gehege mit den Maßen 14m x 12m x 8m gebaut werden.

Figure 10: Generated word problem of medium difficulty

Figure 10 shows a word problem that assumes an above average expertise of the relevant concepts that are displayed. The amount of templates is now five. In Addition to that, attributes were now replaced with other attributes, increasing the overall steps needed to solve eventual tasks. The word problem also now includes templates with the concept association of all three supported mathematical concepts, namely *General*

Number Handling, *Area Measurement* and *Percent Calculation*. The magnitude of the numbers increased as well, as evidenced by multiple double-digit numbers in the word problem.

Eine Tierhandlung kauft Bio-Futter für Hamster. Die Tierhandlung hat eine Fläche von ca. 12.5 Räumen. Der Raum hat eine Fläche von ca. 20 Billiardtischen.

Es muss Bio-Futter für 11 Tage nachgekauft werden. Für eine Packung muss 5 Euro ausgegeben werden.

Aufgrund starkem Nachwuchs soll nun ein neues Gehege mit den Maßen 13m x 11m x 9m gebaut werden.

23% der Einwohner in Leer besuchen die Tierhandlung jedes Jahr. Ein Besucher gibt im Durchschnitt 13 Euro aus. Auf die Einnahmen müssen 23 % Steuern gezahlt werden.

84 Packungen Bio-Futter können ein Hamster 42 Tage lang versorgen.

Die Tierhandlung bringt pro m2 Fläche 3 Hamster unter.

Für den neuen Anbau werden pro m2 Fläche 3 Baumaterialien benötigt.

Figure 11: Generated word problem of hard difficulty

The last figure assumes a very good expertise of the student and represents one of the more difficult word problems the software can generate. Calculations now include decimals that the student has to consider. The magnitude of numbers is generally higher, as well as an increased template level, increasing the complexity of the single templates, as well as the whole word problem. The steps needed in this task compared to the one shown in figure 10 are higher, seeing that the amount of templates and also the difficulty of the single templates has increased.

It is important to note here that these three examples show generated word problems with difficulty parameters that are about the same for every parameter, which means that the overall difficulty and all of its sub-tasks are influenced in the same way. It is of course perfectly possible to only influence parts of the word problem, which could result in a task that is of much higher difficulty when it comes to the conceptual difficulty combined with easy arithmetic operations (see figure 12).

Eine Zoohandlung kauft Wasser für Meerschweinchen.

153 Packungen Wasser können ein Meerschweinchen 51 Tage lang versorgen.

10% der Einwohner in Bremerhaven besuchen die Zoohandlung jedes Jahr. 10% der Besucher geben im Durchschnitt 20 Euro aus. Die restlichen Besucher zahlen im Durchschnitt 10 Euro. Auf die Einnahmen müssen 10% Steuern gezahlt werden.

Die Zoohandlung bringt pro m Breite 5 Meerschweinchen unter.

Es muss Wasser für 10 Tage nachgekauft werden. Für eine Packung muss 1 Euro ausgegeben werden.

Aufgrund starkem Nachwuchs soll nun ein neues Gehege mit den Maßen 5m x 5m x 5m gebaut werden.

Figure 12: Detailed Difficulty Settings

In this example the difficulty parameter of the concept *General Number Handling* was set to an average value, while the concept *Area Measurement* was set on a very low difficulty. To put that into contrast the difficulty of the *Percent Calculation* concept was set to a maximum. This results in the latter influencing the third template in a

way that the template added the version of it with the highest complexity. The length of the task has not changed much because the low, medium and high difficulty settings of the concepts neutralize each other. If all of those concepts were set to a low difficulty, the length of the task would be much shorter and vice versa. The *division* difficulty setting stands in stark contrast with the set *multiplication* setting. The generated word problem poses difficulty division operations that the student has to solve, but easy ones that are about multiplication operations. Because of the low setting of the *Area Measurement* concept the fourth template that is about the amount of animals the Zoo can house, chose the *width* attribute as the determiner instead of for example the area of the Zoo. That means that the student does not have to first calculate the area to proceed. This example is just one of many, varying difficulty parameters can result in a vast amount of different combination of sub-tasks with different difficulties.

4.11.1 Modes of use

The software allows two different modes of how to use the difficulty adjustment process. A teacher can either set the process to be completely automatic without any need of manual input or fine tune the difficulty settings themselves. The automatic mode does however require a few settings to be set on first use.

- Automatic determination of difficulty parameters: Yes/No
- Allow calculations with decimals: Yes/No
- Sensibility Slider: : 1-5
- Concept: Area Measurement: Yes/No
- Concept: Percentage Calculations: Yes/No

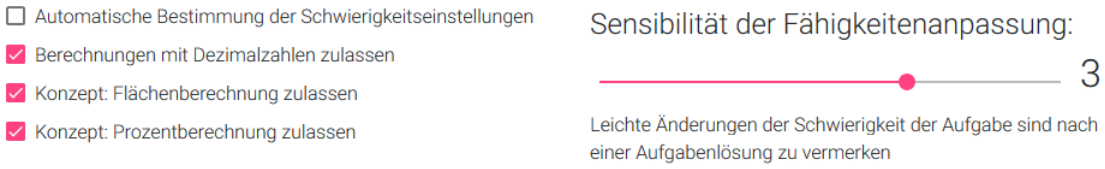


Figure 13: General Difficulty Options

The first option determines the mode of usage, while the second option enables the system to create tasks that include numbers with decimal places. The sensibility slider on the right determines how drastic the system should adjust the skill level of a student. The last two options let the teachers decide if templates of the respective concept are to be allowed. This allows the teacher to further customize what concepts shall be relevant to the learning session.

Figure 14: Overview Difficulty Settings

As seen in figure 14 the teacher has the possibility to adjust the sliders in the manual mode. Every top-level difficulty setting has a range of 1%-100%. This is done for several reasons. First of all this kind of range is intuitive to understand as long as it is clear that a higher setting results in a more difficult task and vice versa for a lower setting. It would be desirable to give the teacher full transparency instead of just allowing the setting of a single slider, but seeing that every slider has numerous effects in the system, it would be too technically challenging to implement, as well as too confusing for the teacher. For that reason more detailed optional sliders were introduced. Take the conceptual difficulty as an example. The teacher can set the top-level difficulty of it and the system will take care of all the detailed determination of the various difficulty settings that are part of the conceptual difficulty. The other option for the teacher is to set the sliders for each of the contained parameters, in this case *Replacement of attributes*, *Length of the task*, *Question Difficulty*, *Amount of distractors.* (see figure 15). These parameters are represented by levels. Therefore the sliders are only able to be influenced in steps of ten. An early developed prototype displayed the levels as 1-10, but seeing that all the top-level difficulty sliders have a range of 1%-100% the levels are simply multiplied by a factor of ten to avoid confusion for the user. At this point it is to be noted that these level-sliders can only be increased by increments of 10, while the percentage sliders can be increased by increments of 1.

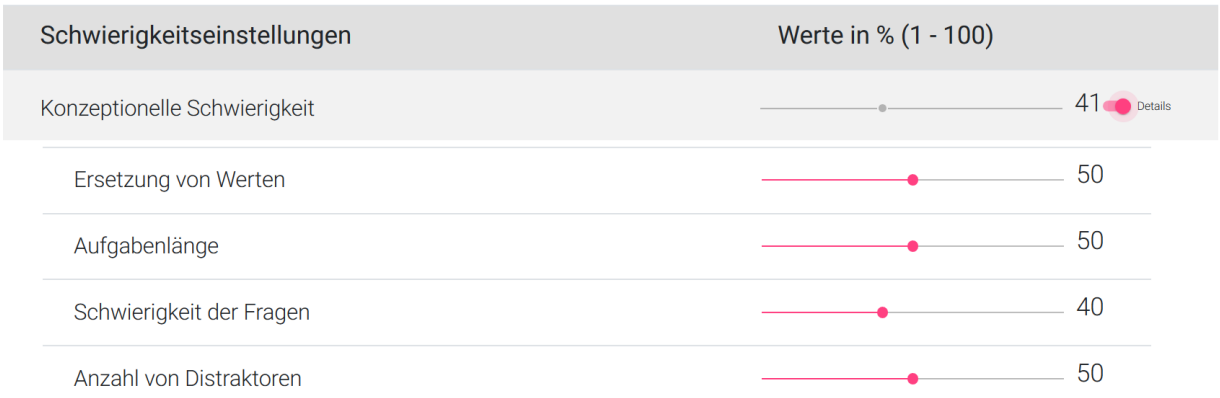


Figure 15: Detailed Difficulty Settings

Bottom-level difficulty determination: If the teacher decides to manually set all the available bottom-level difficulty sliders, no further difficulty parameter creation processes are necessary. However seeing that it is also possible for the teacher to just set the top-level parameters, it is necessary to somehow determine the contained parameter values.

The process starts with the teacher setting a percentage value for the top-level difficulty setting. In the case of the conceptual difficulty, we can assume a value of 75%. This value is now translated in a relative representation of the desired sum of the levels of all underlying difficulty parameters. Seeing that the four mentioned parameters all have ten levels, the sum of the levels is 40. In this case the desired level amount is $0.75 \cdot 40 = 30$. For easier handling, all values are multiplied by a factor of 10. At this point two possible approaches can be used to determine the underlying difficulty settings. Arguably the most trivial approach would be to split the desired level amount for all by the amount of parameters, which would result all parameters to be level 7. This approach would lead to a mundane difficulty presentation that does not ever change. In order to achieve more variety, instead an iterative process was used. Randomly, the level of one of the parameter is leveled up until the desired amount of levels is reached. Obviously seeing that a few parameters have level four as the maximum levels, it is necessary to normalize these parameters. Without the normalization the likelihood of these parameters achieving maximum level would be greatly increased. In the end these kind of parameters are normalized to have 10 levels resulting in a higher amount of levels to distribute.

In the current prototype version all levels have the same linear weighting associated with them except the difficulty parameter regarding the use of decimal numbers. It increases the difficulty of the task by a greater margin if the step from no decimal number calculations to calculations with decimal numbers is taken compared to increasing the number of decimal places from 1 to 2. An interesting avenue could be to explore the "cost" of a level up of certain levels for all parameters once they reach certain thresholds that increase the difficulty of the task more than other levels. An example could be the

Number Magnitude parameter that increases the digit-count at various levels. Kadosh et al. (2014) argues that digit count is a major determiner of the task difficulty. It is important to note that even though the difficulty parameters are not derived of the students mastery in the manual mode, the behaviour of the student still triggers the update process and reevaluates the skill levels of the student.

4.12 Question Generation:

The general approach for the question generation is to first create all possible questions to afterwards curate them and pick only the most suited ones. The curation process is explained in section 4.9 and is based on the base difficulty that is assigned to the question, as well as the amount of steps that are needed to be completed in order to solve it.

In the current prototype there are nine questions that can be asked if all the necessary requirements are set. In addition to that, questions are generated that ask for missing attribute information. The latter is a fairly simple process by scanning all available objects and attributes, asking the student to calculate the missing ones. These type of questions are only relevant if the missing attribute is not already included in the calculation process of the beforementioned type of generated questions.

Every question template stores information about possible questions that are relevant to said template. As an example a template that contains information about building a new building, it is then possible to create a question that asks about the expenses related to it. These are called question flags. Additionally, all information that is relevant to composing a question, like information about the building and the cost of materials, are also being stored for later use. Once the generation of questions start, the list of the nine supported questions have different required question flags. If the check is positive, the question is added to the pool of possible questions. Similar to how attributes store information about how to calculate them, the constructed questions store information about their concept affiliations, as well as valid calculations, including what formula is needed for the calculation. A similar process is employed for adding help in the way of optimal solutions to the question that is then later displayed by request of the student.

Seeing that later on the amount of steps that a question needs to be completed for it to be solved is crucial for the selection of what questions show what degree of difficulty level, the system looks through all the attributes that are needed to be calculated first in order to solve the question at hand.

4.13 Adaptive Task Generation

The purpose of this section is to wrap up the previous explanations of the various parts that make up the task generation process, as well as showing how the student's

mastery of skills are translated into difficulty parameters that influence the make up of the generated word problem. Assuming an automatic mode of difficulty parameter determination, the process starts with translating the student's skill into difficulty parameters as explained in section 4.9.

Now that all the necessary information about the task difficulty is available, the system starts to select question templates that make up the word problem in the end. Starting off, with the selection of a *Base Template*, a suitable *Connected Template* is randomly chosen to be added. Depending on the proficiency of the student in the concept that is affiliated with the connected template, the length of the task is influenced in a different way (see section 4.10). Each question template has variables that need to be filled by either objects or by created numbers. For the process of choosing relevant objects and generation of numbers, see section 4.11. Once the task is created, all possible questions are created. This process is explained in section 4.12 and results in a handful of questions that are eventually presented to the student. The word problem is now displayed to the student with a separation of the actual wording of the question and the object overview, including all the attributes (see section 4.4). The student now uses the flexible solving system to first calculate the missing information that is needed in order to give answers to the question (see section 4.6). During the student's attempt at solving the task, the system observes and logs the behaviour of the student (see section 4.7). The information that is gathered is then used to update the skills of the student with an implementation of Bayesian Knowledge Tracing (see section 4.8). As soon as the student decides to engage with a newly generated task, depending on if the teacher wants a completely automatic approach or a manual approach to set the difficulty parameters the way they like, the skills of the student are translated into difficulty parameters that are then used to create the task (see section 4.9).

4.14 Statistics

The prototype includes a mainly informational page displaying the skill mastery of the student. Similar to how it is done in the student model, each skill is represented with their respective level and current progress towards the next level. The level was previously named "Competency level" which was changed due to a suggestion by the expert in the user test, explaining that competency levels are used in a different context (see Add. Usability Test U39). Therefore to avoid confusion, the description was changed to "levels". Another benefit to the renaming is that levels usually have a gamification implication. Figure 16 shows the visual representation of the conceptual skills of a student. The statistics overview additionally includes a similar overview for the numerical skills of a student.

Konzeptuelle Fähigkeiten:

Kompetenz	Level (1-10)	Fortschritt in %
Rechenregeln/Arithmetik	6	75
Flächenberechnung	6	30
Zufall und Wahrscheinlichkeiten	5	50
Benötigte Schritte für Aufgabenlösung	5	50
Lösen mehrstufiger Aufgaben	4	70
Korrekte Verwendung von Formeln	7	30
Korrekte Informationswahl	7	30
Verwendung korrekter Werte	6	26

Figure 16: Overview of the conceptual skills of a student

4.15 User Registration

It could be argued that a newly registered student would start off with default difficulty settings. In some cases this would lead to a word problem that is either way too easy or way too hard. Eventually the equilibrium in difficulty should be found fairly quickly if the sensibility of the adaption was set to high. This process however can be improved in giving the student a way for an initial self assessment. The student can register to the E-Learning System by adding contextual information like grade, type of school and a self assessment. Additionally, the student can give information about the hobbies they are interested in. In the current prototype this registration process is only a mock-up and has no real implications. The mock-up was implemented because the expert revealed interest in the generated word problems containing content that is not only adjusted in difficulty but also in regards to the content (see Add. Guided Interview I43, I44, I45). Polozov et al. (2015) tried a similar approach.

5 Evaluation

This section presents the evaluation of the software proposed in this thesis. First of all, the findings of the initial interview with an expert that was of an explorative nature are presented. Additionally, the effect the interview had on the implementation details will be elaborated. Following that, the prototype that was presented, as well as the usability test results, will be shown.

5.1 Guided Interview

The guided interview took place early in the development phase at which time a small coded prototype was finished, sufficient enough to present an overview of the platform and its features. General questions regarding the use of E-Learning Softwares were asked, followed by more in depth questions regarding specific features of the prototype. The guided interview paraphrased transcript can be found in the Addendum. The following will refer to the paraphrases as either "I" as in initial guided interview paraphrase or as "U" which stands for an usability test paraphrase.

Current use of E-Learning Software: The school the expert teaches in is located in the city of Bremen, which subsidizes the use of E-Learning Softwares in schools. A widely used software is itslearning that enables the school to communicate with colleagues, create courses, invite students to the courses and even supply tasks (Add. I1). The tasks however have to be manually created by the teachers (Add. I3, I4). Additionally, Bettermarks is used one hour out of five that are allocated to math courses per week. At this point it is important to note that the school is only beginning to use E-Learning Softwares in their daily tuition plans (Add. I6, I7), they are however planning to increase their usage, while being open about any new software that could potentially help them in their goal of educating students (Add. I8, I21). The current usage of software is not limited to the school environment. Each student can access the learning software at home if they wish to do so (Add. I9).

The problem of clustering students: It is apparent that the students of a class of 20 students show different levels of skill, which can not easily be categorized. However teachers are forced to cluster the students into predefined categories of skill (Add. I12), which leads to a stigmatisation of students that are being put into a category that includes students with lower levels of skill (Add. I19). There is a wish that the E-Learning Software can identify knowledge gaps and then instantly create suitable tasks that revolve around the identified tasks automatically (Add. I19).

Allocation and creation of tasks: Even though E-Learning Softwares like Bettermarks provide teachers with packages of tasks that are labeled with difficulty settings,

manual effort by the teacher is still necessary. Knowledge gaps may be identified automatically, but the actual choosing of suitable tasks have to be done by teachers (Add. I10, I13, I18). Furthermore the amount of tasks that are already included in some E-Learning Softwares are limited (Add. I14). Redoing the same tasks can lead to an approach for solving the task that is not productive, because the students can fall back to remembering the last time they solved the task (Add. I51).

Individualization and difficulty adjustments: The goal of teachers is to facilitate the best possible learning experience for every student that is enrolled, which requires an unrealistic amount of individualization and time spent with each student (Add. I20). The expert values the idea of automatically adjusting the difficulty of tasks based on the student's individual skill level (Add. I35), as well as the idea of the teacher being able to influence the automated systems with manual adjustments (Add. I46, I47). It is however important to take the specific environment of the student into consideration like age and the type of school the student is enrolled in (Add. I49).

The structuring of help and support: Guidance and support in task environments have to be designed while keeping in mind that students of lower grades resort to asking for help as soon as they experiencing problems (Add. I27, I28, I30). Step by step help that focuses on informing the student how to solve a problem rather than just showing the correct results is heavily suggested (Add. I29, I36).

Automated creation of practice tasks: The idea of automated generation of practice tasks is valued (Add. I34). It is however important to structure the tasks based on the demography of the students. Simple, easy to understand language in short sentences are preferred (Add. I41, I42). Furthermore it is advised that the content in the generated task resembles topics that the students encounter in their daily lives (Add. I43, I44). They should also feature realistic outcomes. The expert emphasizes the need for self identification of the students in the task (Add. 45).

Gamification and reward systems: The expert suggests some kind of gamification and reward system (Add. I48). According to the expert's experience, the students are motivated to work with Bettermarks (Add. I31), because it features small amounts of gamification in the form of an animated figure (Add. I32).

To summarize, the guided interview revealed valuable information about the environment the proposed software would operate in. itslearning is widely used in schools in Bremen so it is important that the software proposed in this thesis is compatible with the system. Because of a variety of reasons this was not done in this thesis, which are elaborated in section 5.4. The ideas to tackle the problems that were initially identified

with the current landscape of E-Learning Systems (see section 1.1) that were proposed to the expert are seen as viable solutions and would be greatly accepted. The expert goes as far as to say that a good implementation of said ideas is exactly what teachers need to fulfil the demand of catering to each and every student as best as they can to facilitate an optimal learning experience. A lot of information regarding the use of help and guidance was revealed, which heavily influenced the implementation choices that were made in regards to presenting contextualized help (see section 4.6). The expert values the possibility of manually influencing the automated difficulty settings, which led to an implementation of a thorough user interface using sliders to adjust every difficulty setting that is relevant in creating tasks. It is important to note that the expert mentioned the benefits of the inclusion of a reward system or gamification multiple times. Because of the limited scope of this thesis, no gamification was implemented. Section 5.4.3 however elaborates on the compatibility of the proposed software and gamification.

5.2 Prototype

The prototype that was used in the usability test was a fully working coded prototype that was in development for about four months. It is fully functional and offers all features that are elaborated in the Implementation chapter (see section 4). All planned features were successfully integrated before the user test was conducted. The prototype resembles a fully operational product in its functionality regarding the solving system and difficulty adjustment. It features a task generation cycle of student behaviour analysis followed by an update of the skills and then finally the adjustment of difficulty parameters. It is however limited in the amount of content it offers. Currently only three concepts *General Number Handling*, *Area Measurement* and *Percent Calculation* are implemented. Each concept has multiple question templates to offer. Because of the completeness of the software, the results of the user test are predictive of how a fully operation ready software would be perceived in regards to the unique feature set.

5.3 Results

As outlined in section 3, a usability test was concluded and the transcript was paraphrased and then used for an heuristic evaluation to categorize the results from the test into twelve categories as seen below.

Category	Occurrences
Task Suitability	U4, U5, U6, U30, U57, U60, U61, U64, U65.1, U68, U70, U71
Process Suitability	U2, U8, U11, U13, U17.1, U27, U28, U29, U35, U37, U39, U51, U65, U71, U72, U73, U74
Self Explanatory Power	U3, U9, U12, U15, U36, U42, U43, U46, U49.2
Controllability	U19
Expectation Conformity	U10, U14, U55
Fault Tolerance	
System and Data Security	
Customizability	U1, U41, U45, U49.2, U50, U52, U53, U59, U68, U69
Promotion of Learning	U16, U17.2, U20, U21, U22, U23, U24, U25, U26, U31, U38, U45, U76
Cognitive Handling	U7, U18, U40, U44, U47, U48, U54, U62, U65.1, U75
Joy of Use	U32, U33, U34, U56, U63
Intercultural Aspects	

It is apparent that the amount of occurrences in each category differs heavily. In the following the results for each category, as well as the amount of relevant paraphrases, are commented on.

Task Suitability: Seeing that the software proposed new ideas on how to present and create mathematical word problems, it was important to inquire the expert's opinion on the task suitability of the software in regards to the school environment. The usability test started with an initial presentation of the software in which not surprisingly the suitability of task was being commented on. The majority of occurrences that fell into this category came about in the interview that followed up the usability test.

While the software was being presented to the expert, an inquiry was made about the possibility of a student being able to continue working on the task, even if the student was not being able to successfully complete a sub-task. This particular feature was implemented because of the comments of the expert in the initial exploratory interview. It was mentioned that help is constantly used by students (Add. I28) and that the students are more likely to just progress to the next task if they approach a wall and can not progress any further (Add. I29, I28). This point was further verified later on in the second use case (Add. U30).

The software does not offer a calculation system to digitally manually calculate arithmetic operations that need multiple steps. That is however not a problem, because

it is common practice for students to manually calculate these calculations not digitally, but rather in an analogue way (Add. U57).

The expert in general likes the way the solving system works, if certain improvements would be made. More elaboration on those points can be found in the relevant categories *Process Suitability*, *Self Explanatory Power*, *Promotion of Learning* and *Expectation Conformity*. To summarize, the most urgent problem is that the flow of the task solving process, essentially when, what and how to do something is not very clear for the student.

Being one of the core features aimed to solve one of the identified problems, the implementation of functionalities regarding the automatic and manual setting of difficulty is highly valued (Add. U65.1), which was a topic that already came up in the initial interview (Add. I47). Especially the possibility of only influencing parts of a word problem instead of the whole task was seen as a big success (Add. U67, U68).

The initial exploratory interview showed the importance of a large variety of practice tasks (Add. I51). According to the expert's opinion, the tasks that were presented during the user test were varied enough to deliver new experiences each time a new task is being generated (Add. 70).

Process Suitability: Process Suitability is about the degree of adaptiveness and optimisation of the software in regards to the school environment. This category is a critical factor for success because the user demography of this software are not adult customers but students in secondary schools. Therefore testing the special requirements was a focus in the user test. The software used competency levels to describe the current skill level a student has. It is however not transparent how these levels are calculated (Add. U2). Besides that, teachers use competency levels with a different meaning and context, which leads to confusion about what these levels are portraying (Add. U37, U39). As pointed out in the exploratory interview, students have problems with keeping prolonged focus on their tasks (Add. I27, I28).

While solving a practice task the expert identifies a problem with certain tasks requiring to many steps to be solved at once (Add. U8). It is also important to respect the language that the students are familiar with to avoid ambiguities and confusion (Add. U11, U13, U27, U28, U29, U35). It is apparent that the specific ambiguity of using ":" as an association symbol for attributes with their objects can easily be misinterpreted as a symbol for division, even for teachers. According to the expert it is generally the case that the degree of hand-holding and process suitability differs based on the type of school and grade the software would operate in (Add. 17.1).

With certain difficulty settings, the generation of tasks can possibly generate multiplication sub-tasks that use "1" as a multiplicator. Besides the obvious pointlessness of using such a multiplicator, it also might lead to confusion for the students (Add. U51). According to the expert, using a minimum of 2 would fix this problem.

Self Explanatory Power: During the user test the expert struggled multiple times not knowing what to do. Add. U3 showed a lack of clarity on how to calculate a missing area attribute. A similar problem was discovered in the first use case, when the expert failed to add a known attribute to the calculation (Add. U9). After a second explanation, the expert successfully completed the first two steps of building a calculation (Add. U12), but then failed to finish it because multiple steps have to be done within one calculation, instead of spreading it out in smaller steps (Add. U13). Later in the user test, the expert struggled again not knowing where to choose already calculated attributes from and how to approach solving a question (Add. U15).

The expert was tasked to find specific statistics of a student, which did not reveal any further confusion (Add. U36). Even though the user test showed that manually adjusting the difficulty sliders was a self explanatory task (Add. U44), there was confusion about what a low, respective high slider setting means (Add. U46). The use case further revealed that the optional manual settings reached by pressing the "Details" Button is being misunderstood to be more of an informational nature instead of showing a sub-menu with more options to set (Add. 43).

It is clear that the software shows problems in regards to self explanatory power of the actual solving process of the task. Multiple possible improvements to avoid confusion and to provide more guidance for the student were discussed. Besides that, the software was deemed self explanatory.

Expectation Conformity: The software follows a general procedure for solving tasks. First of the missing attributes have to be calculated in order to be able to then answer the questions that are being asked. There are two calculation tools that are each dedicated to either the attribute- or the question calculation. The expert had to solve a task in the first use case and noticed that the mentioned procedure was not applicable. A question was answerable without further calculation, which could confuse the students because they might think a further step of calculation is necessary (Add. U10, U14). A similar situation in the last use case duplicated this confusion (Add. U55).

Customizability: A major focus of this thesis was to create a software that is highly individualized to the skill level of the user. Large parts of the user test focused on this functionality of the software. Firstly the expert expressed that the possibility of adjusting the difficulty settings is valuable (Add. U1). The expert emphasized this point by arguing that it is the single most important feature of the software (Add. U59), because it fixes the problem of being forced to choose tasks for the students to solve that are of a fixed difficulty that affects all parts of the word problem (Add. U67, U68).

The expert set the sensibility slider in a way that was rationalised by the amount of tasks we already solved, which is the intention behind the sliders. Recently registered

students would profit from a more sensible skill adjustment compared to students that already solved multiple tasks and therefore have a skill level that more closely represents their actual skill (Add. U41, 49.1). The expert immediately noticed the change of difficulty after the automatic difficulty mode was activated (Add. U52).

A more specific difficulty parameter, the amount of distracting information in the question, was deemed as a valuable addition by the expert (Add. U52).

Promotion of Learning Clearly Promotion of Learning is a very important heuristic when it comes to E-Learning Software. The usability test revealed that there are problems not giving students enough guidance, so to not distract their focus on non relevant activities. Seeing that the user interface separates the information and the question itself, which is seen as valuable (Add. U18), and furthermore offers calculation tools for attribute calculation and question calculation, it is important that the offered guidance reflects this separation of concerns (Add. U16, U17.2, U20). The currently statically offered help to explain the general procedure is not inviting the user to use it when a problem arises (Add. U22, U23). The expert suggested a system that would guide the student along the task by dynamically analysing at which stage the student is currently at (Add. U24).

The fact that contextualized help regarding specific calculations is not visible until the student fails once is received positively (Add. U25), as well as the actual content of the help (Add. U26). The software offers the possibility of automatically solving sub-tasks. According to the expert it is however not needed to receive feedback that using this system will lead to negative impact on the estimation of the skill level of the student (Add. U31).

Cognitive Handling: This heuristic is about the minimalistic design and the direction of the user's focus on task relevant components. For that purpose the expert put herself into the place of a student and quickly identified where relevant informations regarding the solving of the question was to be found, when the teacher was tasked to solve an introductory word problem (Add. U7). The decision to seperate the concerns into information and the question itself was seen as a good implementation choice (Add. U18). The expert however remarked that the layout is a bit overwhelming at first (Add. U75).

At multiple times in the user test, the expert was asked to comment on the general presentation of information and on the way components are being structured. The findings are that the minimalistic design is simple and clean, which makes it easy to spot the relevant information (Add. U40, U44, U47, U54). Especially the simple overview of the user's skill was positively remarked, as other E-Learning Softwares that the teacher uses have more bloated overview panels (Add. U62).

Joy of Use: The exploratory interview already showed the enthusiasm the expert shows in regards to gamification of E-Learning Softwares (Add. I31, I32, I48). While working through the second use case, the expert inquired if the software features a reward system, in which the students get feedback in the form of an animation (Add. U32) or through the awarding of coins (Add. U33). According to the expert the positive impact of such gamification is still there even if the rewarded coins are of no real use (Add. U34). In the follow-up interview the expert again emphasizes that there should be more fun in doing the actual tasks (Add. U63).

It is noticeable that the categories *Fault Tolerance*, *System and Data Security* and *Intercultural Aspects* show no occurrences in the comments that were made by the expert. The software currently does not feature any kind of error messaging or fault tolerance, mostly because it is just not needed in the current prototypical state. For the same reason the software lacks any system and data security, because there is no data currently being stored that can be compromised. These two heuristics are of course relevant when it comes to using the software in a real environment but are outside of the scope of this thesis. Intercultural aspects were neglected because the adjustments to the demographic were mostly located in the *Process Suitability* category. Any additional adaption to any cultural characteristics of the users were not made or investigated. Further research in this regard is advisable.

At the end of the follow-up interview, some general questions summarizing the evaluation of the software were asked. It is of the expert's opinion, that the prototypical software differentiates itself from other E-Learning Softwares mostly in the possible individualization options in regards to difficulty settings (Add. U69). Especially the ability to only influence parts of the task was praised (Add. U68). According to the expert, a potential usage of the software could be found in the area of reinforcement learning or as private lessons. Students that are already done with the planned coursework could be tasked to use this software to further reinforce the concepts they have learned without needing manual effort by the teacher (Add. 71, 72). The students could also, in their own time, work on improving their deficiencies (Add. U73). In general the expert was surprised with the amount of individualization the software has to offer (Add. U74).

5.4 Further Development

This chapter briefly assesses possible further development of the software. To get a good picture about what kinds of improvements to the software is needed, it makes sense to differentiate between changes to the functionality regarding the new ideas proposed in this software and common features that are currently missing in the software that would be needed if it was to operate in a real environment.

5.4.1 Feature development:

The usability test revealed a handful of problems that would need to be fixed in order for the software to be usable by actual students. Most of the issues are problems regarding the *Process Suitability* like using the correct language that the students and teachers are familiar with. Plenty of suggestions were made by the teacher to improve the guidance and support for the student, like for example a contextualized support system that dynamically analyses at which solving stage the student currently is, to then offer help that can guide the student along. In the current version the flow of solving a task is interrupted by problems that do not have anything to do with the actual assignment of solving the task but rather navigational issues. In general the software needs to be more streamlined, as well as use more easy and intuitively understandable language. Another impactful feature addition would be the introduction of a reward system, even a simple one. The expert mentioned that the attention and motivation of students are reachable with even the simplest gamification systems.

E-Learning Softwares also usually offer expansive overviews about the behaviour of the student. The software does necessarily log the behaviour of the user in order to feed the cycle of updating the skill of said user, but it does not currently offer an overview that is reachable by a teacher.

Another obvious issue with the current software is the lack of content. As of right now three mathematical concepts with a handful of respective templates and sub-tasks are supported. In order to be used as an educational tool, it would be a necessity to include the whole curriculum of concepts and tasks to match the current plethora of E-Learning Softwares and analogue practice material. This problem is not a trivial one and would require a good amount of manpower and development time.

The software uses Bayesian Knowledge Tracing to update the skills of a student. Currently the same parameters are used for every skill. As described in R. S. J. d. Baker et al. (2008), the slip and guess parameters of a BKT are dependent on their context. For example having to choose the correct option out of 4 possible options results in a guess parameter that is at least 25% instead of having to choose out of a higher amount of options. In addition to these four parameters Yudelson et al. (2013) introduced a new parameter that describes the individual learn speed of a student. Individualizing each parameter for every represented skill of the student would lead to a more accurate representation of skill level. For certain skills like *choosing the correct information* and *disregarding distracting information*, the implementation is fairly trivial to do. Skills like measuring areas are much more complex in regards to adjusting the parameters.

5.4.2 Must have features:

In order for the software being able to be used in a real working environment, a large amount of common software features need to be implemented. In its current form

User Management only allows the choice of different predefined student accounts. Fully functional registration and login features would need to be implemented. As mentioned earlier in this thesis (see 2.1.4), and in the initial exploratory interview (Add. I1, I2), itslearning is deployed in a phased approach to schools in Bremen. The school that was visited for the interviews is already using itslearning for a few years. These systems already store the data of all students, colleagues, coursework and more. It would be essential for this software to be compatible with the dataset of the itslearning software. To be able to fulfil this task, data from the ITS Learning software would need to be extracted and then imported into this software, which is not a trivial task and was not pursued in this thesis, because the focus was on establishing new unique features and their usefulness. Data security features are also currently missing.

5.4.3 Gamification:

In both of the interviews the expert showed great enthusiasm towards the topic of gamification (Add. I31, U32). The ITS that is currently used in the expert's classroom only offers a very limited gamified experience in the form of animated figures (Add. I32). The students show great motivation, willingness and prefer to use E-Learning Software rather than being taught with traditional methods (Add. I31). This revelation conforms with the scientific findings on this topics such as in Brull and Finlayson (2016) and Papp (2017).

For those reasons it is important to examine the possibility of gamification in the software that was developed during the creation of this thesis. Morschheuser, Hassan, Werder, and Hamari (2018) explains the approach how gamified software should be engineered. First of all it is important to understand the motivation and needs of the user and the environment the software operates in (Morschheuser et al., 2018, 3.2 DP1). The participatory design approach of this thesis lead to an initial exploratory interview to reveal these kinds of information. It is however advisable to focus on the gamification methods that are currently used in the classroom, as well as to examine what kind of characteristics the gamified software needs to have so that it could function in an operational context. (Morschheuser et al., 2018, 3.2 DP2-DP4) shows that creating the E-Learning Software with gamification as one of the primary goals right from the start instead of having it as an afterthought is vastly superior, because it is an iterative process in which features have to be tested repeatedly. The following Design Principles continue laying out a framework of design ideas that benefit from an early consideration of how gamification can be included into the software. Based on these findings, two possible approaches of gamification for this particular software can be derived. Maximizing the effect that the gamification would have on the E-Learning Software's success in regards to motivation and willingness to partake in educational learning activities, redoing the software with a heavy focus on gamification would be

advisable. A completely new concept would have to be created in order to gamify the solving of math tasks. It is however the case that the current flexible structure of creating and solving a task could lead to creative and flexible ideas how to gamify the software. It would not be limited to just rewarding the student for solving a task, since the student has to solve multiple sub-tasks to eventually solve the whole questions. Actions like choosing the correct information out of the already calculated ones or calculating information that is eventually relevant for further calculations, can each be gamified.

Seeing that the expert voiced the opinion that even the small amounts of gamification have good effects on her student's motivation, the second approach of gamifying this E-Learning Software could be to add additional playful elements that do not require a great overhaul of the existing implementation. The statistics page of a student already evaluates the skill of the students in a level format. It is easy to see how the student could be presented with beating certain levels of tasks to further level up his or her levels. The expert especially mentioned a reward for solving tasks successfully with some kind of virtual currency. This would be a trivial task, since the positive effect can be seen even without presenting an actual use for the coins (Add. I33). In one of the interviews the implementation of small-step and contextualized help was discussed (Add. I36). Students resort back to using help as soon as they encounter problems with solving a task. The expert suggested to use the rewarded coins as some kind of payment in order for the help to appear to create an incentive of solving the task on their own to preserve the amounts of coins the students have (Add. I37).

The second approach could lead to a very beneficial cost and performance relation, since the task of implementing the mentioned gamified elements is trivial and would result in only minimal changes in the current implementation. Gamification was not a focus of this thesis, but further research regarding gamified E-Learning Software that has a focus on individualization and accommodation to the student's needs could fit together well, because the control of the generation and solving of the task is very expansive.

6 Conclusions

Three main problems regarding the use of E-Learning Systems in schools were identified. First, the availability of varied practice tasks are limited, secondly the mathematical word problems that the students have to solve are not individually adjusted to them and lastly students are being presented with a constrained way of how they can approach to solve a word problem. Participatory methods were used to gain valuable personal experience into how E-Learning Systems are used in a realistic environment. This process influenced the eventual implementation of the software.

To combat these problems a fully functional prototype of an E-Learning Software was created that automatically generates mathematical word problems with a dynamic difficulty scaling of every containing sub task. Participatory methods were used to inquire contextual information about the current usage of E-Learning Systems by conducting an expert interview. A skill representation model of the student was derived of the competency framework of the Ministry of Culture that at runtime translates into certain difficulty parameters that influence the conceptual and numerical difficulty of all sub-tasks contained in the task. The word problem gets generated by selecting numerous sub-tasks out of a collection of possible tasks that are added together. A question solving system that separates the question wording with information about the objects that are included in the word problem was created. The automatic generation of tasks allows the system to have full control of every part of the word problem, which in turn allows the student to freely decide on how to solve the task by constructing valid calculations. A skill model of a student needs to be updated based on the performance of the student. A comprehensive and detailed logging system analyses the behaviour of the student. The gained data is then used to update the skills via Bayesian Knowledge Tracing. A prototype test was conducted to evaluate the practical suitability of the E-Learning Software and to gain feedback to inspire further improvements.

The usability test evaluated the suitability of the software and found that the novel features of individual difficulty adjustments can be used to combat the problem of a need for a highly individualized learning experience for students. The generated word problems offer a high variety of possible tasks because the tasks are built iteratively by using a template-based building process. Additional variety is achieved by using a database of real-world objects that are inserted into the word problem and used as calculation values.

The first research question is about how the creation of word problems can be supported by using information that is freely available in the internet to create mathematical word problems. The prototype successfully uses object data that was manually extracted and curated into a database that is then queried by the system generating the word problem. Various amounts of different types of information was extracted in order to allow a multitude of different resulting word problems to be created. The semantic web was already used in other projects (see section2.4). This thesis took a different approach by combining the extracted data with a dynamic process that finds suitable objects while it is creating a word problem. Templates associated to mathematical concepts can be dynamically added together, while each template has variable sections that is filled by using real world object data. This results in coherent word problems that include realistic outcomes and real world objects, that can be presented to students as practice tasks. The expansion of the object collection is also trivial, because the creation of tasks and the object database are separated from each other.

A major focus of this thesis was the degree of individualization when it comes to the creation of word problems. The cycle starts with modelling the student, translating the skill mastery into difficulty parameters, which are then used in the task creation process and lastly logging of the behaviour of the student, which then results in an update of the student's skills. The cycle then begins again by presenting a word problem that is more and more adjusted to the student's skill level. The difficulty parameters were chosen in a way to allow the adjustment of sub-tasks within the word problem as opposed to the common way of classifying the whole word problem as a single difficulty. The thesis has shown that it is possible to use common student models and use that information to determine difficulty parameters that are then used to influence the resulting word problem. Existing E-Learning Systems do offer the choice of tasks of varying difficulty choices (see section 2.1), but run into the problem that word problems include multiple required expressions of skill by the student. The prototype shows that it is possible to adjust only the parts of the word problem that do not currently reflect the expertise of the student, instead of changing the difficulty of the whole problem.

One of the research questions that was explored in this thesis was regarding the variety of word problems that students encounter in their educational endeavours. It is clear that predefined tasks are limited in their amount and pose a problem for teachers looking for novel tasks that they can present to their students. In this thesis variety of generated word problems was achieved by the combination of multiple processes that each increase variety of the task, including a large collection of available objects, question templates and template levels. It was also shown that the addition of more templates increases the amount of combinations even more and is technically not a big task. The prototype can be expanded without infringing on the existing template material. Existing literature shows projects that aim to provide adjusted difficulties and customized content in their created task. The value this thesis adds in that regard is the dynamic difficulty adjustment of single elements of the task, that are then used to further analyse the student to iteratively adjust the difficulties of the word problem. The expert prototype test revealed that the generated word problems resemble already used ones. It was also validated that the effects different difficulty settings have on the task are noticeable. The amount of possibilities of manual difficulty adjustment was also praised.

In order to utilize the generation of word problems, the prototype implements a flexible question solving system. The thesis explored the research question how a solving process can be designed in a way that the student can solve the task however they want. For that reason a presentation of the word problem was chosen that separates the actual wording of the question with an overview about all the facts that are included in the question as an object overview. The student solves the word problem in two steps. First, missing attribute information are calculated, followed up by solving the actual questions. The student has to construct all the calculations that are necessary

for a successful completion of the word problem. A problem that was identified is that existing E-Learning Systems give the student unintentional hints in the way the task is presented. The implementation shown in this thesis manages to create a flexible solving system that further enables a detailed logging of the user's behaviour. Expert interviews revealed problems in the presentation and logical traversal of the solving system.

The automatic generation of word problems has numerous advantages because the system has a full model representation of the task at hand. This enables the addition of a flexible solving system that lets the student brick by brick finish sub-tasks that lead up to solving the whole word problem. Such a limitless approach to solve the system necessitates an implemented question solving system that allows the student to submit various distinct ways of constructing calculations. In the end the automated generation of questions was necessary to implement the other systems of adjusting the difficulty, as well as the solving system. Adjusting sub-tasks of a word problem would be impossible if the common approach of using predefined tasks was to be employed. The generation of tasks in an automatic fashion already has numerous approaches in the literature (see section 2.4). This thesis adds value to the scientific field in combining the automatic creation of tasks with the individualized adjustment of difficulty. It also showed that an automatic generation of tasks can lead to the enhancement of all other connected systems.

The findings in this thesis answer the research questions by showing how to implement an E-Learning System that individually adjusts the difficulty of word problems with an automatic generation process. It showed promising new unique approaches on how E-Learning Systems can be defined to tackle common problems. The expert that was first interviewed and then tasked to do a usability test, praised the prototype, as well as suggesting that this kind of approach is the way these systems have to be designed in order to satisfy the requirements that teachers nowadays are held responsible for. It is however important to note that because of time and budget constraints, only one expert was interviewed. More research, especially exploration in how to build such a system in a way that expanding the word problem generation process by the teachers themselves is possible by adding new template content to it, could lead to valuable results. The combination of automatic generation with an individual difficulty adjustment process shows promising results. It however needs more testing in what difficulty parameters represent the difficulty of word problems the best way, as well as how to optimally translate the mastery of a student in difficulty parameters.

6.1 Practical implications

The purpose of this thesis was to show that E-Learning Softwares that follow the approach that is laid out in this thesis can be of great value. The expert evaluation

showed that there are problems with the user interface of the current prototype version. However the software showed great task and process suitability and presented the need of heavily individualized word problems without needing manual input by a teacher. The current prototype implements all the core necessary components and features that would make up a productive ready E-Learning Software. It is however not ready to be used in a realistic school setting because major Quality of Life features are missing, as well as no current compatibility with existing systems like itslearning. A necessary requirement would also be the implementation of great parts of the student's curriculum. The current prototype only supports a fraction of the educational material that is needed in order to create a software that can be universally used in schools as a learning tool. The expert evaluated that an E-Learning System like the prototype presented, could be useful as a tool to support private lessons or to let students that already finished the coursework continue with practising what they have learned, with minimal teacher input (see Add. Usability Test U72). The usability test was restricted to only a single expert which needs further validation by larger studies. Furthermore standard usability issues were largely ignored in this thesis because the focus was on the task and process suitability and not on the actual readiness of the software to be used in an educational setting. It mainly showed that the novel features proposed in this software are technically and practically viable and should be further explored. Further research could analyse the suitability of such a software in the described way, especially in conjunction with participatory methods.

6.2 Contributions and Further Work

This thesis contributes to the existing scientific corpus regarding E-Learning Systems in that it shows new possible approaches on how to eliminate the need for teachers having to invest time to achieve an individual learning environment for their pupils. The automatic generation of tasks enables a logging system that can, in a detailed way, provide information about the student's behaviour that is then used to update the mastery levels of the student in various conceptual and numerical skills. These skills are then dynamically translated into difficulty parameters that are used to generate the tasks. The thesis emphasized on the importance of the combination of the three major components, an automatic process of generating tasks, a question solving system that takes advantage of that being the case, as well as a student model that can supply the word problem generation process with relevant information. Lastly this thesis showed that difficulty adjustments can be made on a sub-task level adjusting only the difficulty of parts of the word problem instead of the whole task. With this approach an individualized learning experience can be provided for the student.

Multiple new approaches were explored and implemented in the prototype. Numerous possibilities of further work to advance the methods used can be identified. The

accuracy of the generated word problems rely on the difficulty parameters that were used in this thesis. Further research into the choice of suitable difficulty parameters could lead to an improved determination of difficulty in the generated tasks. The influence of the difficulty parameters on the creation of tasks includes numerous subjective claims and decisions in regards to how many levels a difficulty parameter has or what each level entails in the final word problem. Additionally, the separation of concerns of the question itself and the object overview and the eventual solving of the question leads to confusion on first use. More usability testing that focuses on the user interface for the student using participatory methods could lead to the detection of additional improvement avenues, as well as currently existing usability problems. Further work in how to design an E-Learning System that includes the core components of the prototype in a way so that teachers can themselves easily add own templates to the system, which can then be used to generate word problems is also advised.

7 Glossary

E-Learning E-Learning encompasses learning with the support of electronic resources like videos, interactive learning material and online softwares that emulate classroom experiences (Arkorful & Abaidoo, 2014).

Intelligent Tutoring System An Intelligent Tutoring System provides the learner with a web-based learning environment in which the student receives learning materials, practice tasks, customized instructions and feedback (Psotka et al., 1988).

Adaptive ITS Adaptive Intelligent Tutoring Systems offer content that is adapted to the user's knowledge, goals and preferences by maintaining a model of the user (Phobun & Vicheanpanya, 2010).

Student model A model of a student representing skill, knowledge and social background of the student.

Natural Language Processing Natural Language Processing is a field of computer science that deals with analyzing natural languages and texts.

Natural Language Question Generation Natural Language Question Generation deals with the generation of questions by using NLP techniques.

Rule-based Cognitive modelling A cognitive model that gives strict requirements of how a solution to a task should look like (Aleven, 2010).

Constraint-based modelling Constraint based modelling sets constraints or requirements for a solution of a task. The student can then choose to solve the task in any way that satisfies the constraints (Mitrovic, 2010).

Bayesian Knowledge Tracing Bayesian Knowledge Tracing is an algorithm to traces the mastery of a student in a particular skill (Corbett & Anderson, 1994).

Hidden Markov Model A Hidden Markov Model adds unobserved states to a markov model, which describes probabilistic events that are dependent on each other (Rabiner, 1990).

Learning Analytics Learning analytics concerns the collection of data of learners and any relevant surrounding information so to understand and draw conclusions to better the learning experience for the student (LAK'11, 2010).

Item Difficulty Item Difficulty describes the determined difficulty of a task with a numerical value (University of Washington, n.d.).

Item Response Theory Item Response Theory uses latent traits of students and uses their ability scale to determine Item Difficulty (Hambleton, 1990).

Question Template A Question Template is part of a generated word problem. A word problem usually consists of multiple question templates. It is not to be confused with the actual questions that students are being tasked to solve.

API An Application Programming Interface is used for a client to communicate with a server via requests.

REST-API A REST-API is an Application Programming Interface architectural style (RESTfulAPI.net, n.d.).

Critical Success Factors Critical Success Factors are factors that are necessary to achieve a certain goal.

References

Aleven, V. (2010). Rule-based cognitive modeling for intelligent tutoring systems. *Studies in Computational Intelligence*, *308*, 33–62. doi:10.1007/978-3-642-14363-2_3

Andres, E., Heeren, B., & Jeuring, J. (2013). Towards automatic generation of domain-specific mathematical input support. *CEUR Workshop Proceedings*, *1010*.

Arkorful, V., & Abaidoo, N. (2014). The role of e-learning, the advantages and disadvantages of its adoption in higher education. *International Journal of Education and Research*, *2*, 397–410.

Baker, F. B. (1985). *The basics of item response theory*. Heinemann.

Baker, R. S. J. d., Corbett, A. T., & Aleven, V. (2008). More accurate student modeling through contextual estimation of slip and guess probabilities in bayesian knowledge tracing. In B. P. Woolf, E. Aïmeur, R. Nkambou, & S. Lajoie (Eds.), *Intelligent tutoring systems* (pp. 406–415). Berlin, Heidelberg: Springer Berlin Heidelberg.

Baker, R., Gowda, S., & Salamin, E. (2018). Modeling the learning that takes place between online assessments. In *Proceedings of the 26thinternational conference on computers in education*. Philippines: Asia-Pacific Society for Computers in Education.

Barr, A. (1979). *Handbook of ai* (E. A. Feigenbaum, Ed.). Preliminary Edition. Stanford, California.

bettermarks GmbH. (2018). *Bettermarks - erfolgreich mathe lernen*. https://de.bettermarks.com.

Bienkowski, M., Feng, M., & Means, B. (2012). *Enhancing teaching and learning through educational data mining and learning analytics: An issue brief*. U.S. Department of Education, Office of Educational Technology.

Brull, S., & Finlayson, S. (2016). Importance of gamification in increasing learning. *The Journal of Continuing Education in Nursing*, *47*, 372–375. doi:10.3928/00220124-20160715-09

Brusilovsky, P. (1994). The construction and application of student models in intelligent tutoring systems. *Journal of Computer and Systems Sciences International*, *32*.

Buckley, P., & Doyle, E. (2016). Gamification and student motivation. *Interactive Learning Environments*, *24*(6), 1162–1175. doi:10.1080/10494820.2014.964263

Butterworth, B., Marchesini, N., & Girelli, L. (1999). *Multiplication facts: Passive storage or dynamic reorganization?* Department of Psychology and the Institute of Cognitive Neuroscience, University College London, To appear in "The development of arithmetical concepts and skills". Pre-Publication Draft.

Butz, C. J., Hua, S., & Maguire, R. B. (2008). Web-based bayesian intelligent tutoring systems. In R. Nayak, N. Ichalkaranje, & L. C. Jain (Eds.), *Evolution of the web in artificial intelligence environments* (pp. 221–242). doi:10.1007/978-3-540-79140-9_10

Carnegie-Mellon-University. (2018). *Cognitive tutor authoring tools*. http://ctat.pact.cs.cmu.edu/.

Coniam, D. (1997). A preliminary inquiry into using corpus word frequency data in the automatic generation of english language cloze tests. *CALICO Journal*, *14*, 15–33.

Corbett, A. T., & Anderson, J. R. (1994). Knowledge tracing: Modeling the acquisition of procedural knowledge. *User Modeling and User-Adapted Interaction*, *4*(4), 253–278. doi:10.1007/BF01099821

Csikszentmihalyi, M. (2008). *Flow: The psychology of optimal experience*. Harper Perennial Modern Classics.

Cummins, D., Kintsch, W., Reusser, K., & Weimer, R. (1988). The role of understanding in word problems. *Cognitive Psychology*, *20*, 405–438. doi:10.1016/0010-0285(88)90011-4

Daróczy, G., Wolska, M., Meurers, W. D., & Nuerk, H.-C. (2015). Word problems: A review of linguistic and numerical factors contributing to their difficulty. In *Front. psychol.*

Dašić, P., Dašić, J., Crvenković, B., & Šerifi, V. (2016). A review of intelligent tutoring systems in e-learning. *Annals of the Oradea University – Fascicle of Management and Technological Engineering*, *15*, 85–90. doi:10.15660/AUOFMTE.2016-3.3276

Design Council. (2007). *Eleven lessons: Managing design in eleven global brands a study of the design process*. London: Design Council. Retrieved from https://www.designcouncil.org.uk/resources/report/11-lessons-managing-design-global-brands

Ehn, P. (1990). *Work-oriented design of computer artifacts*. Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Felder, R. M. (1988). Learning and teaching styles in engineering education. http://www4.ncsu.edu/unity/lockers/users/f/felder/public/ILSdir/styles.pdf.

Froschl, C. (2005). User modeling and user profiling in adaptive e-learning systems.

Gamalel-Din, S. (2018). The smart tutor: Student-centered case-based adaptive intelligent e-tutoring. Al-Azhar University, Systems & Computers Engineering Dept, Retrieved from: https://www.kau.edu.sa/Files/0053027/Researches/29114_Adaptive%20E-Training%207.pdf.

Grubišić, A., Stankov, S., & Žitko, B. (2013). Stereotype student model for an adaptive e-learning system. (Vol. 7).

Haftmann, R. (2014). *Aufgabensammlung zur höheren mathematikmit ausführlichen lösungen*. https://www-user.tu-chemnitz.de/~rhaf/Aufgabensammlung/Sammlung/Aufgabensammlung.pdf.

Hambleton, R. (1990). Item response theory: Introduction and bibliography. *Psicothema, ISSN 0214-9915, Vol. 2, Nº. 1, 1990, pags. 97-107*, *2*.

Heffernan, N. T., & Heffernan, C. L. (2014). The assistments ecosystem: Building a platform that brings scientists and teachers together for minimally invasive research on human learning and teaching. *International Journal of Artificial Intelligence in Education*, *24*(4), 470–497. doi:10.1007/s40593-014-0024-x

ISO 9241-210. (2010). *Iso 9241-210:2010: Ergonomics of human-system interaction – part 210: Human-centred design for interactive systems*. pub-ISO:adr: pub-ISO. Retrieved from https://www.iso.org/standard/52075.html

itslearning. (n.d.). Itslearning. https://itslearning.com/global/. Accessed: 2019-06-03.

Johnson, L., Smith, R., Willis, H., Levine, A., & Haywood, K. (2011). *The 2011 horizon report*. Austin, Texas: The New Media Consortium. Retrieved from http://horizon.unc.edu/HR2011.pdf

Kadosh, R. C., Dowker, A., Nuerk, H.-C., Moeller, K., & Willmes, K. (2014). Multidigit number processingoverview, conceptual clarifications, and language influences. Oxford University Press. Retrieved from https://www.oxfordhandbooks.com/view/10.1093/oxfordhb/9780199642342.001.0001/oxfordhb-9780199642342-e-021

Khajah, M., Huang, Y., González-Brenes, J. P., Mozer, M. C., & Brusilovsky, P. (2014). Integrating knowledge tracing and item response theory: A tale of two frameworks. In *Umap workshops*.

Kitchin, R. (2014). Big data, new epistemologies and paradigm shift. *Big Data & Society*, *1*, 1–12. doi:10.1177/2053951714528481

Knoedler, A. J., & Wenger, M. J. (1995). Handbook of usability testing: How to plan, design, and conduct effective tests by jeffrey rubin 330 pages, $34.95 new york: John wiley & sons, 1994 isbn 0-471-59403-2. *Ergonomics in Design*, *3*(4), 29–31. doi:10.1177/106480469500300408. eprint: https://doi.org/10.1177/106480469500300408

Kultusministerkonferenz. (2012). *Bildungsstandards im fach mathematik für die allgemeine hochschulreife*. https://www.kmk.org/fileadmin/Dateien/veroeffentlichungen_beschluesse/2012/2012_10_18-Bildungsstandards-Deutsch-Abi.pdf. Beschluss der Kultusministerkonferenz.

LAK'11. (2010). Learning Analytics 2011. https://tekri.athabascau.ca/analytics/. International Conference on Learning Analytics and Knowledge 2011, Accessed: 2019-06-03.

Lee, F.-L., & Heyworth, R. (2000). Problem complexity: A measure of problem difficulty in algebra by using computer. *Education Journal*, *28*, 85–107.

Li, N., W. Cohen, W., Koedinger, K., & Matsuda, N. (2011). A machine learning approach for automatic student model discovery. *EDM 2011 - Proceedings of the 4th International Conference on Educational Data Mining*, 31–40.

Mayring, P. (2000). Qualitative inhaltsanalyse. *Forum Qualitative Sozialforschung - Forum: Qualitative Social Research*, *1*.

Melis, E., Andrès, E., & et al., J. B. (2001). Activemath: A generic and adaptive web-based learning environment. *International Journal of Artificial Intelligence in Education*, *12*, 385–407.

Mitrovic, A. (2010). Modeling domains and students with constraint-based modeling. In R. Nkambou, J. Bourdeau, & R. Mizoguchi (Eds.), *Advances in intelligent tutoring systems* (pp. 63–80). doi:10.1007/978-3-642-14363-2_4

Mohammadi, N., Ghorbani, V., & Hamidi, F. (2011). Effects of e-learning on language learning. *Procedia Computer Science*, *3*, 464–468. World Conference on Information Technology. doi:https://doi.org/10.1016/j.procs.2010.12.078

Morschheuser, B., Hassan, L., Werder, K., & Hamari, J. (2018). How to design gamification? a method for engineering gamified software. *Information and Software Technology*, *95*. doi:10.1016/j.infsof.2017.10.015

Muth, K. (1992). Extraneous information and extra steps in arithmetic word problems. *Contemporary Educational Psychology - CONTEMP EDUC PSYCHOL*, *17*, 278–285. doi:10.1016/0361-476X(92)90066-8

National Research Council. (1993). *Measuring what counts: A conceptual guide for mathematics assessment*. Washington, DC: The National Academies Press.

National Research Council and Institute of Medicine. (2004). *Engaging schools: Fostering high school students; motivation to learn*. Washington, DC: The National Academies Press.

Naveh, G., Tubin, D., & Pliskin, N. (2012). Student satisfaction with learning management systems: A lens of critical success factors. *Technology, Pedagogy and Education*, *21*(3), 337–350. doi:10.1080/1475939X.2012.720413

Niebert, K., & Gropengießer, H. (2014). Leitfadengestützte interviews. In D. Krüger, I. Parchmann, & H. Schecker (Eds.), *Methoden in der naturwissenschaftsdidaktischen forschung* (pp. 121–132). doi:10.1007/978-3-642-37827-0_10

Nielsen, J. (1993). *Usability engineering*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Nielsen, J. (1994). Enhancing the explanatory power of usability heuristics. In *Proceedings of the sigchi conference on human factors in computing systems* (pp. 152–158). CHI '94. doi:10.1145/191666.191729

Nkambou, R., Bourdeau, J., & Mizoguchi, R. (2010). Introduction: What are intelligent tutoring systems, and why this book? In R. Nkambou, J. Bourdeau, & R. Mizoguchi (Eds.), *Advances in intelligent tutoring systems* (pp. 1–12). doi:10.1007/978-3-642-14363-2_1

Nkambou, R., Mizoguchi, R., & Bourdeau, J. (2010). *Advances in intelligent tutoring systems*. Berlin Heidelberg: Springer Verlag. Retrieved from https://www.springer.com/de/book/9783642143625

Palm, T. (2009). Words and worlds: Modelling verbal descriptions of situations - theory of authentic task situations. In *Words and worlds: Modelling verbal descriptions of situations* (pp. 3–19). Rotterdam/Boston/Taipei: Sense Publishers.

Papasalouros, A., Kanaris, K., & Kotis, K. (2008). Automatic generation of multiple choice questions from domain ontologies. In *Mccsis'08 - iadis multi conference on computer science and information systems; proceedings of e-learning 2008* (Vol. 1, pp. 427–434).

Papp, T. A. (2017). Gamification effects on motivation and learning: Application to primary and college students. *International Journal for Cross-Disciplinary Subjects in Education*, *8*, 3193–3201.

Pardos, Z. A., & Heffernan, N. T. (2011). Kt-idem: Introducing item difficulty to the knowledge tracing model. In J. A. Konstan, R. Conejo, J. L. Marzo, & N. Oliver (Eds.), *User modeling, adaption and personalization* (pp. 243–254). Berlin, Heidelberg: Springer Berlin Heidelberg.

Perrotta, C., & Williamson, B. (2018). The social life of learning analytics: Cluster analysis and the 'performance' of algorithmic education. *Learning, Media and Technology*, *43*(1), 3–16.

Phobun, P., & Vicheanpanya, J. (2010). Adaptive intelligent tutoring systems for e-learning systems. *Procedia - Social and Behavioral Sciences*, *2*(2), 4064–4069. Innovation and Creativity in Education. doi:https://doi.org/10.1016/j.sbspro.2010.03.641

Polozov, O., O'Rourke, E., Smith, A. M., Zettlemoyer, L. S., Gulwani, S., & Popovic, Z. (2015). Personalized mathematical word problem generation. In *Ijcai*.

Psotka, J., Massey, L. D., & Mutter, S. A. (1988). *Intelligent tutoring systems: Lessons learned.* Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc. Retrieved from http://psycnet.apa.org/record/1988-97858-000

Qiu, Y., Qi, Y., Lu, H., Pardos, Z., & Heffernan, N. (2011). Does time matter? modeling the effect of time with bayesian knowledge tracing. *EDM 2011 - Proceedings of the 4th International Conference on Educational Data Mining*, 139–148.

Rabiner, L. R. (1990). Readings in speech recognition. In A. Waibel & K.-F. Lee (Eds.), (Chap. A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, pp. 267–296). San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. Retrieved from http://dl.acm.org/citation.cfm?id=108235.108253

RESTfulAPI.net. (n.d.). What is REST. https://restfulapi.net/. Accessed: 2019-06-03.

Rosic, M., Glavinic, V., & Stankov, S. (2005). Intelligent tutoring systems for the new learning infrastructure. In *Intelligent learning infrastructure for knowledge intensive organizations: A semantic web perspective* (pp. 225–250). doi:10.4018/978-1-59140-503-0.ch008

Sarodnick, F., & Brau, H. (2006). *Methoden der usability evaluation: Wissenschaftliche grundlagen und praktische anwendung [usability evaluation techniques—scientific fundamentals and practical applicability]*. Hogrefe AG.

Schiaffino, S., Garcia, P., & Amandi, A. (2008). Eteacher: Providing personalized assistance to e-learning students. *Computers and Education*, *51*, 1744–1754.

Schley, D. R., & Fujita, K. (2014). Seeing the math in the story: On how abstraction promotes performance on mathematical word problems. *Social Psychological and Personality Science*, *5*(8), 953–961. doi:10.1177/1948550614539519

Self, J. A. (1994). Formal approaches to student modelling. In J. E. Greer & G. I. McCalla (Eds.), *Student modelling: The key to individualized knowledge-based instruction* (pp. 295–352). Berlin, Heidelberg: Springer Berlin Heidelberg.

Shernof, D. J., Ruzek, E. A., Sannella, A. J., Schorr, R. Y., Sanchez-Wall, L., & Bressler, D. M. (2017). Student engagement as a general factor of classroom experience: Associations with student practices and educational outcomes in a university gateway course. *Frontiers in Psychology*, *8*, 994. doi:10.3389/fpsyg.2017.00994

Siemens, G. (2013). Learning analytics the emergence of a discipline. *American Behavioral Scientist*, *57*, 1380–1400. doi:10.1177/0002764213498851

Sison, R., & Shimura, M. (1998). Student modeling and machine learning. *International Journal of Artificial Intelligence in Education*, *9*.

Slade, S., & Prinsloo, P. (2013). Learning analytics ethical issues and dilemmas. *American Behavioral Scientist*, *57*, 1510–1529. doi:10.1177/0002764213479366

University of Washington. (n.d.). Understanding Item Analyses. www.washington.edu/assessment/scanning-scoring/scoring/reports/item-analysis/. Accessed: 2019-06-03.

University of Wisconsin. (n.d.-a). Optimal Item Difficulty. https://www.uwosh.edu/testing/faculty-information/test-scoring/score-report-interpretation/item-analysis-1/item-difficulty. Accessed: 2019-06-03.

University of Wisconsin. (n.d.-b). Reliability & Validity. https://www.uwosh.edu/testing/faculty-information/test-scoring/score-report-interpretation/item-analysis-1/reliability-validity. Accessed: 2019-06-03.

Vines, J., Clarke, R., Wright, P., McCarthy, J., & Olivier, P. (2013). Configuring participation: On how we involve people in design. (pp. 429–438). doi:10.1145/2470654.2470716

Vrasidas, C. (2004). Issues of pedagogy and design in e-learning systems. In *Proceedings of the 2004 acm symposium on applied computing* (pp. 911–915). SAC '04. doi:10.1145/967900.968086

Wenger, E. (1987). *Artificial intelligence and tutoring systems: computational and cognitive approaches to the communication of knowledge*. A review of this book from Marie Bienkowski was published in ACM SIGART Bulletin, Issue 109, July 1989 (see link below). Los Altos (Calif.) : M. Kaufmann. Retrieved from https://hal.archives-ouvertes.fr/hal-00703016

Williams, S. (2011). Generating mathematical word problems. In *2011 association for the advancement of artificial intelligence (aaai) fall symposium on question generation* (pp. 61–64). Retrieved from https://www.researchgate.net/publication/266892784_Generating_Mathematical_Word_Problems

Wilson, A., Watson, C., Thompson, T. L., Drew, V., & Doyle, S. (2017). Learning analytics: Challenges and limitations. *Teaching in Higher Education*, 1–17. doi:10.1080/13562517.2017.1332026

wolframAlpha LLC. (2018). *Wolframalpha - computational intelligence*. https://www.wolframalpha.com.

Yamna, E., Mellouli, K., & Wuillemin, P.-H. (2010). A multicriteria bayesian intelligent tutoring system mbits. *2010 10th International Conference on Intelligent Systems Design and Applications*, 714–719.

Yerkes, R. M., & Dodson, J. D. (1908). The relation of strength of stimulus to rapidity of habit-formation. *Journal of Comparative Neurology and Psychology*, *18*(5), 459–482. doi:10.1002/cne.920180503. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cne.920180503

Yudelson, M. V., Koedinger, K. R., & Gordon, G. J. (2013). Individualized bayesian knowledge tracing models. In H. C. Lane, K. Yacef, J. Mostow, & P. Pavlik (Eds.), *Artificial intelligence in education* (pp. 171–180). Berlin, Heidelberg: Springer Berlin Heidelberg.

Zaharias, P. (2009). Usability in the context of e-learning: A framework augmenting 'traditional' usability constructs with instructional design and motivation to learn. *IJTHI*, *5*, 37–59.

Addendum

Paraphrase transcript of initial guided interview

Label	Timestamp	Paraphrase
I1	0:00:12	The school uses ITS Learning. It is subsidized by the City of Bremen and every school in Bremen is working with it. Teachers can use it to share learning material and to conduct communication within the school.
I2	0:00:55	It is possible to include the pupils into itslearning, to invite them to a course and to supply the platform with tasks. The school is using these features very rarely, which is something they want to work on.
I3	0:01:45	Software itself can not create tasks.
I4	0:01:55	Manual effort by teachers necessary to create tasks.
I5	0:02:05	Linking to sofatutor for relevant tasks, which is supplied by the city of Bremen.
I6	0:02:35	School is just starting to use E-Learning Software. Bettermarks is only used in 2 classes in the 5th grade.
I7	0:02:50	No concrete concept behind using Bettermarks, but it is used in one hour out of five math hours every week. It is only used for fundamental arithmetics. Expert would like to use it for pupils that are already done with the coursework so that they can use the software as reinforcement learning.
I8	0:03:45	School is very open for everything E-Learning Softwares have to offer.
I9	0:04:10	E-Learning Software is also used at home.
I10	0:04:20	Teachers have insight into the results of the students, including which tasks were solved, what knowledge gaps the students show. Expert criticizes that the softwares can not currently create tasks based on the knowledge gaps.
I11	0:04:45	Expert was talking with the Bettermarks founder at a conference to pitch the idea of the software being able to create tasks based on knowledge gaps on the spot. At the moment the teachers have to analyse the knowledge gaps and then supply the correct tasks.
I12	0:05:50	(Regarding the first use of Bettermarks) – The teachers were only mildly better informed about the software before they started using it. The teachers picked out tasks out of the three levels of skill that correlate to their estimation of students into elementary level students, base level students and advanced level students.
I13	0:06:40	Teachers had to manually pick out the tasks, the software did not do it itself.
I14	0:07:01	Expert criticizes the amount of tasks that are available in Bettermarks.
I15	0:07:12	All students have their own account for Bettermarks.

I16	0:07:30	Teachers explained the handling of Bettermarks on a beamer.
I17	0:07:42	For every package of tasks, a pre test was used to identify the knowledge gaps of the student. The results were inconclusive and not helpful.
I18	0:08:00	The teachers then manually picked the tasks that the student will be presented with based on the results of the test. The main reason for doing that was to familiarize the student with the procedure of doing a pre test, getting the actual tasks and then followed up with a final test.
I19	0:08:30	Experts favours the idea of an automated pre test, so that the software can identify knowledge gaps. With a system such as this, there would be less stigmatizing the student and less bias when selecting tasks.
I20	0:09:40	It is the goal of the teachers to facility the best learning experience for every individual student.
I21	0:10:10	The expert explains that they are willing to use these E-Learning Softwares because they do offer more individualized help for the students.
I22	0:11:01	Not much support from the university for using Bettermarks. Adds that they however do not use Bettermarks that often. The main support from the university comes from finding out better learning procedures or techniques.
I23	0:11:50	Current initiative of university students searching for suitable learning apps for the pupils in school, not only in math. Another imitative is for pupils to create tutorial videos.
I24	0:12:35	Pupils won the second place of a competition in which tutorial videos were created. The purpose of this initiative was to evaluate the reward of using this method of learning.
I25	0:13:25	(Regarding individual settings in Bettermarks) – Nothing was changed. It was not possible to remove tasks.
I26	0:14:00	(Regarding the procedure of the last time Bettermarks was used) - Bettermarks was used to identify knowledge gaps. Written addition and subtraction showed knowledge gaps. The concepts were then taught at the blackboard. Students who were already able to solve these tasks were told to solve tasks about multiplication in Bettermarks.

I27	0:15:25	(Regarding the interaction between teachers and students): A lot of interaction, because there are a lot of badly performing students, that struggle with the tasks a lot. As soon as a problem is identified, the students instantly seek out help from the teachers without trying to solve it on their own or to ask other students.
I28	0:16:12	Help given in the software is used constantly. Tips are read but not with the purpose of understanding them. As soon as an incorrect result is entered, the student seeks out the correct result and moves on to the next task. No learning success.
I29	0:17:05	(Regarding the suggestion to show only the approach to solving the task and not the end result): It is important to make the student interested into how the problem is solved but that usually does not interest the student.
I30	0:17:20	It is more akin to a game for the student, they just want to progress and advance to the next task. There is not much reflection done.
I31	0:17:55	(Regarding the interest for gamification): Extreme interest in gamification. The mention of going into the computer room leads to excitement by the students.
I32	0:18:10	Expert criticizes the lack of gamification in Bettermarks. Mentions that students of the teacher wrote letters to the software developers suggesting that the currency being earned by solving tasks should be usable to buy cosmetics for a certain figure that appears on the screen in the Bettermarks software.
I33	0:19:20	The currency earned in Bettermarks has no influence on the grading of the students. It is however planned that the currency can be used by the students in a way.
I34	0:26:10	Expert likes the idea that the software is automatically creates tasks.
I35	0:26:50	Expert likes the idea of the software automatically adjusting to the students needs.
I36	0:27:20	Expert advocates for step by step help that is dependent on the students progress in the task.
I37	0:27:40	Advocates a reward system that can be used to “buy” help.
I38	0:28:20	Expert expresses that the shown individualization possibilities in the system do not exist in current existing E-Learning Softwares.
I39	0:28:35	Expert comments on variety of tasks, that even though Bettermarks offers the choosing of different tasks, they are still limited in variety.
I40	0:28:55	Expert mentions that teachers are forced to cluster the students in certain levels of skill, which correlate to the three levels of difficulty of the tasks in Bettermarks. The individualization possibilities in the proposed software is according to the expert the way it should be like in the future in regards to E-Learning Software.
I41	0:30:14	(Regarding the automatic creation of tasks): Important to use easy language and no long sentences, because the students do not have the attention span otherwise.

I42	0:31:10	Complexity of sentences could be adjusted based on the student's age.
I43	0:31:35	The tasks need to resemble something the students are in contact with in their daily lives. Important that the tasks are of a realistic nature.
I44	0:32:05	Students' hobbies should influence the task generation.
I45	0:32:15	There is a need of self identification.
I46	0:33:10	Expert values the idea of being able to make manual adjustments of the difficulty settings.
I47	0:33:30	Expert likes the idea of being able to rely on the automatic difficulty settings of the software with the possibility of adding nuances to the difficulty manually.
I48	0:34:12	Expert advocates for including some kind of reward system.
I49	0:35:20	Difficulty settings should vary based on what level the school operates in.
I50	0:35:40	According to the expert, the proposed software idea is something the teachers are waiting for, it is exactly what they need.
I51	0:36:05	The student having to solve the same tasks over and over leads to less learning success because of the student remembering how the task was solved the last time instead of grappling with the task itself.

Paraphrase transcript of the usability test

Presentation of the software:

Label	Timestamp	Paraphrase
U1	0:06:50	Possibility of adjusting settings is deemed valuable.
U2	0:07:20	Ambiguity about how the competency levels are calculated.
U3	0:09:15	Lack of clarity on what steps are needed to calculate the Area attribute.
U4	0:13:00	Asks about the possibility for the student to continue working on solving the task even if a sub task was calculated incorrectly or the student was not able to calculate it at all.
U5	0:14:05	Assumes that after the task is completed, the teacher can review the behaviour of the student and find out that automatic solving was used.
U6	0:15:05	Selection of hobbies of the student deemed valuable.

Use Case 1: Solution of an easy task:

Label	Timestamp	Paraphrase
U7	0:17:51	Put herself into the place of a student, quickly finds out where the relevant information to solve the tasks are located in the information section. Quickly finds the list of already known information in the attribute calculation section.
U8	0:18:37	Pupils are not able to solve multiple steps of a calculation in one go. The attribute that is possible to be calculated is already one step too far ahead.
U9	0:20:24	Lack of clarity on how to add a known attribute to the calculation.
U10	0:21:10	Asks why this step is necessary, if she does not need to calculate something now to advance.
U11	0:21:39	Lack of clarity because of the used denominations for percentages (0.1 vs 10%). Pupils would not know what 0.1 means.
U12	0:22:20	Completes the first step of the calculation correctly. Adds attribute and multiplication operator to the calculation.
U13	0:23:15	Again confusion, because multiple steps have to be completed in one go.
U14	0:25:05	Questions the reason why the question asks for an answer that was already calculated in the attribute calculation section.
U15	0:28:15	Difficulties in understanding the solving system, what attributes to choose from and from where and where to solve the final question.

Label	Timestamp	Paraphrase
U16	0:29:19	Advocates for an annotation that in an easy to understanding wording explains at which solving stage the pupil is currently at. As an example, if there are still "?" for attributes that need be solved, the system should point out: "You are missing information of the turtle, water and animal shelter [...]". It is difficult to understand that you first have to solve the missing attributes, then solve the question. Advocates for a context sensitive system that guides the student along.
U17.1	0:30:45	This system could work without additional help at a higher school, but not in the school the expert works at.
U17.2	0:30:45	Small step tutorials are needed.
U18	0:31:36	Approval of the presentation, that information of the question and the actual question is separated. Gives structure and helps to solve the task.
U19	0:32:00	Likes that everything is able to be done with the mouse. Keyboards are not used very much by pupils.
U20	0:32:24	Program should make it clear that the solving of the task is to be done in two steps. Calculate all missing information in the attribute calculation section and then follow up with solving the question in the question solving section.
U21	0:33:25	Likes the idea of fading out the question calculation section if its not yet possible to calculate it.

Use Case 2: Usage of helps:

Label	Timestamp	Paraphrase
U22	0:34:15	The help that is being statically presented should be more context sensible. The expert would not use the button, because the expert would not know what kind of help is even needed.
U23	0:34:50	Expert would not know why to press the help button.
U24	0:35:02	Expert proposes a helper text that says "You do not know what you have to do right now, click here" so that the student knows when to click the help.
U25	0:35:50	Expert likes the fact that the calculation help is not directly visible.
U26	0:36:15	Step-by-step help is reasonable.
U27	0:36:44	Expert again shows confusion with the denomination (":" for division)
U28	0:37:35	Expert proposes the idea to leave out the „:“ used to describe association with an object like „Hude: Anzahl Einwohner“ (Hude: Amount of inhabitants) if there is only one City Object. Even if it does not make sense, students might get confused.

Label	Timestamp	Paraphrase
U29	0:38:12	Likes the idea of completely getting rid of the association symbol „:“ and instead use more fluid wording that expresses the association
U30	0:38:50	Expert values the possibility of automatically solving a task, because the students would just give up, especially if there are longer tasks or multiple ways of solving one.
U31	0:39:15	Feedback for using the help or automatically solving the task should not give feedback.
U32	0:39:25	Expert asks if the students if the students are being rewarded with some kind of feedback or animation after they solved the task.
U33	0:40:10	Use of levels to give the platform the resemblance of a gaming experience. Students could get rewarded with coins.
U34	0:40:58	Coins itself do not have to have a use. Students are only interested in collecting.
U35	0:41:25	Expert again shows confusion with the denomination “:” for division.

Use Case 3: Analyze the statistics of a student:

Label	Timestamp	Paraphrase
U36	0:42:55	Expert easily finds the statistics that the expert was tasked to find.
U37	0:43:20	Confused about what competency level means, if 1 means bad and 10 means good.
U38	0:44:15	Does not like the idea of setting a goal for individual students, because it is expected that every pupil reaches the highest competency level.
U39	0:45:10	It has to be explained what competency levels mean or it would lead to confusion since teachers use competency levels in a different context.
U40	0:45:45	Presentation of statistics is concise and arranged clearly.

Use Case 4: Manual settings of difficulties:

Label	Timestamp	Paraphrase
U41	0:47:00	Expert sets the sensitivity slider to 5 with the motivation that we were right at the start of using the platform.
U42	0:47:20	Expert thinks it is evident on how to control the difficulty adjustment.
U43	0:47:32	The function of "Details“ is not apparent. Expert things that “Details” means that, on keypress, an explanation pops up that explains the difficulty parameter the button is at. Expert thinks using “Topics” and “Sub-topics” would make it clear.
U44	0:48:40	The presentation and handling of the manual difficulty setting is clear.
U45	0:49:00	Not necessary to show example tasks based on the difficulty settings. For the expert the function of the manual difficulty settings is roughly change them based on the students needs.
U46	0:49:25	Confusion about if a high setting means the task will be more difficult or more easier.
U47	0:49:38	Presentation is good, because its very clean and simple.
U48	0:50:10	Easy to choose concepts that are permissible in tasks.

Use Case 5: Automatic setting of difficulties:

Label	Timestamp	Paraphrase
U49.1	0:50:50	Expert sets automatic difficulty scaling without problems. Expert then follows up by adjusting the sensibility because we already solved a few tasks.
U49.2	0:50:50	Positively notices the feedback that comes after saving the changes.
U50	0:51:25	Expert noticed the more complex generated task.
U51	0:51:45	Expert advices to get rid of tasks that include multiplicators that are 1, because it would irritate the students. A possible solution would be to use 2 as a minimum multiplier.
U52	0:53:45	Likes the idea of being able to adjust the amount of distractors that appear in a question.
U53	0:53:57	Expert says that the difference in difficulty is instantly noticeable.

Use Case 6: Solving of a difficult question.

Label	Timestamp	Paraphrase
U54	0:56:15	Expert says that it is easy to spot the information that you already have.
U55	0:57:00:	Expert goes through the steps that are needed to solve the task. It was noticed that with this particular question, just calculating the missing attribute answers the question, which is in opposition with the proposed two step system (Calculate missing attributes, then solve question).
U56	0:58:10	Advocates for some kind of animation when the question is solved.
U57	0:58:35	Students manually solve arithmetic tasks on paper.
U58	0:58:55	Expert solves one question and then explains the next few steps that have to be taken correctly.
U59	1:02:20:	The most important feature is the program lets you set individual difficulty parameters, which enables the student to focus on learning a specific concept.
U60	1:03:40	Writing out the whole identifiers for the different registration components would make it more clearer.
U61	1:04:40	Self assessment possibility is good and the students would do that. The expert especially mentions that the setting of hobbies is what pulls the attention in of the students.

Follow-up interview:

Label	Timestamp	Paraphrase
U62	1:05:00 (1)	Simplicity of the skill overview of the student is great. Dislikes bloated overviews that are present in other E-Learning Softwares.
U63	1:05:00 (2)	Likes the software overall. Would like to see an improvement in the presentation, so that it is more motivating. There should be more fun in doing the tasks.
U64	1:06:23:	Likes the solving system if the mentioned improvements are implemented. The steps for solving the task need to be more visible.
U65.1	1:07:05	Likes the manual and automatic difficulty settings. Very simple to use.
U65.2	1:07:05	Very simple to use. (manual and automatic difficulty settings)
U66	1:07:50	The generated tasks resemble tasks the expert uses in her classes.

Label	Timestamp	Paraphrase
U67	1:09:10	Expert likes the ability to influence all the sub tasks that are generated within a word problem.
U68	1:10:10	Expert says the general problem with using E-Learning Software is that you can not influence only parts of the task, but only choose task bundles that are associated with a difficulty level. It is a massive success to be able to individually adjust the difficulty of tasks for each student.
U69	1:10:45	No big differences between softwares like Bettermarks and the tested software. The big difference is the ability to better individually adjust the generated word problems student.
U70	1:12:10	Expert thinks the tasks are varied in their fidelity.
U71	1:12:25	Expert would use the tested software for reinforcement learning of fundamentals. One hour out of five would be optimal. The expert would also prefer to not enable only single concepts but rather use a combination of it so that it will not get boring.
U72	1:13:14	Expert sees the software as support for private lessons. Another use, if it runs on a tablet, would be that if a student is already done with the coursework, the teacher could just tell that student to further reinforce the concepts with the tested software. It is a current problem to give the student more advanced coursework material after he or she is done, because the the student will always be ahead of other pupils.
U73	1:14:05	Student can improve their deficiencies alone.
U74	1:14:50	Expected a similar software like this, but the way you can individualize to a student is much more complex. There is a lot of nuance to the difficulty settings. Also the possibility for teachers to interject in the settings is great.
U75	1:15:18	The layout is a bit terrifying at first.
U76	1:15:41	More practicability is needed. The flow should be more intuitive. It has to be as easy to handle as possible. The sole focus should be on the student's ability to solve the task.

List of Figures

Eidesstattliche Erklärung

Eidesstattliche Erklärung zur <-Arbeit>

Ich versichere, die von mir vorgelegte Arbeit selbstständig verfasst zu haben. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten oder nicht veröffentlichten Arbeiten anderer entnommen sind, habe ich als entnommen kenntlich gemacht. Sämtliche Quellen und Hilfsmittel, die ich für die Arbeit benutzt habe, sind angegeben. Die Arbeit hat mit gleichem Inhalt bzw. in wesentlichen Teilen noch keiner anderen Prüfungsbehörde vorgelegen.

$Unterschrift:$ $Ort, Datum:$