This is a preliminary version of an article published by
Schön, D.; Klinger, M.; Kopf, S.; & Effelsberg, W.:
A Model for Customized In-Class Learning Scenarios. An Approach to Enhance Audience Response Systems with customized Logic and Interactivity.
International Conference on Computer Supported Education (CSEDU), 2015.

A Model for Customized In-Class Learning Scenarios

An Approach to Enhance Audience Response Systems with customized Logic and Interactivity

Daniel Schön[1], Melanie Klinger[2], Stephan Kopf[1] and Wolfgang Effelsberg[1]

[1]*Department of Computer Science IV, University of Mannheim, A5 6, Mannheim, Germany*

[2]*Referat Hochschuldidaktik, University of Mannheim, Castle, Mannheim, Germany*

{*schoen, kopf, effelsberg*}*@informatik.uni-mannheim.de, klinger@ssl.uni-mannheim.de*

Keywords: Audience Response System, Peer Instruction, Lecture Feedback, Mobile Devices

Abstract: Audience Response Systems (ARS) are quite common nowadays. With a very high smart phone availability among students, the usage of ARS within classroom settings has become quite easy. Together with the trend for developing web applications, the number of ARS implementations grew rapidly in recent years. Many of these applications are quite similar to each other, and fit into many classroom learning scenarios like test questions, self-assessment and audience feedback. But they are mostly limited to their original purpose. Using another question types or little differences in the existing quiz logic cause considerable effort to develop as they have to be implemented separately. We searched for similarities between the different ARS implementations, matched them to a universal process and present a generalized model for all of them. We implemented a prototype that serves many known scenarios ranging from simple knowledge feedback questions up to complex marketplace simulations. A first evaluation in different course types with up to fifty students showed, that the model satisfies our expectations and offers a lot of new opportunities for classroom learning scenarios.

1 INTRODUCTION

Audience Response Systems (ARS) are quite common nowadays. The first implementations on PDAs and hand held computers were created to increase interactivity, activate the audience and get an realistic feedback of the students' knowledge. With today's availability of smart phones among students, the usage of ARSs within classroom settings has become quite popular. The technologies are so evolved, that programming a new ARS can be done by many informatics students within a few months.

Due to the didactic benefit, the number of ARS implementations grew significantly during the last years. Many of these applications are quite similar to each other and fit for the purpose of a specific classroom learning scenario, like test questions, self-assessment or simple audience feedback. We also implemented an ARS at our university three years ago and integrated it into the learning management system (Schön et al., 2012). *MobileQuiz* used the students' smart phones as clicker devices and has been adapted by many lecturers by now.

But with the increasing popularity, more and more requirements for extensions were brought to our attention. These ranged from simple layout adaptations, to enhancements of the number of different question types, up to new learning scenarios with their own customized logic. Beginning with simple feedback and self-assessment scenarios, our lecturers wanted to have customized learning scenarios with more complexity, adaptivity and increased student interactivity. They wanted to use guessing questions with a range of right answers, text input for literature discussion, twitter walls for in-class side discussions, game-theory and decision-making experiments for live demonstrations. Every lecturer had a precise picture of his needs that differentiated from the existing scenarios and the requirements of the other lecturers.

The problem is that meeting the lecturers' needs by adding new question types and extensions in the existing quiz logic requires an additional programming effort for every new scenario. Even little variations in appearance needed a change in the program code, and new features like an adaptive and collective quiz behaviour would have needed a full reconstruction of the existing application or an entirely new tool for this purpose. We wanted to meet all the requirements in one step by allowing the lecturers to configure the classroom learning application for their

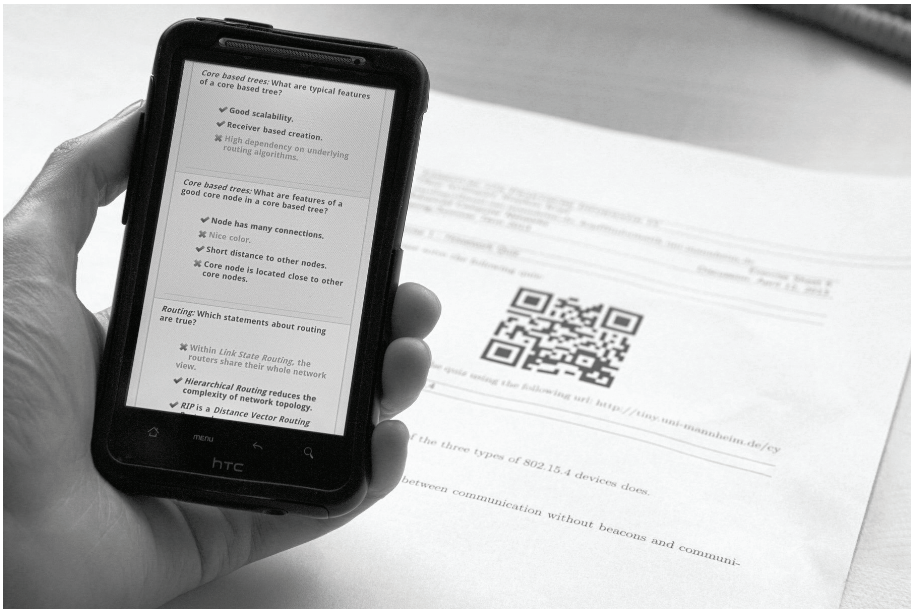

Figure 1: Example of an QR Code usage on an exercise sheet (Schön et al., 2013).

learning scenario without the limitation of existing tools and without implementing every change into the source code. We thus designed a model, able to depict many different kinds of in-class learning scenarios on mobile phones, and created an application, implementing that model. Our model is able to describe the appearance and behavior of many different scenarios by using a small set of predefined elements, and our prototype application comprises the features and scenarios of many other tools altogether within one system. Lecturers are now able to use a twitter wall, a knowledge quiz, a guessing game and further customized scenarios within their classroom using only one system. We tested the validity of our model in three different lectures with up to fifty students and altogether 27 classroom sessions last semester.

This paper is structured as follows: Section 2 discusses the previous ARSs with their benefits and drawbacks and the technologies we adapted from other fields of knowledge. In Section 3, we discuss the main requirements and present an abstract model to fit them all. We also match the procedure of using an ARS to a more embracing process and divide it into five phases. We describe the implemented prototype in section 4 and give an example, how a lecturer would use it. In section 5, we discuss the evaluation results of our approach using six different scenarios. Section 6 discusses the benefits, difficulties and limitations of our approach and gives a conclusion about the current state and an insight into the actual and future work.

2 RELATED WORK

2.1 Benefits of Audience Response Systems

In everyday teaching, time restraints and the number of students in a course can hinder the usage of beneficial methods like group works, feedback sessions or plenum discussions. In this case, e-learning technology can offer time-effective alternatives or supplements. A prominent example is the use of ARS. These systems can be used to anonymously test students understanding of a learning unit. Besides, students can be asked for their opinion concerning course contents or course design. These answers can be a starting point for further explanations or face-to-face discussions. Numerous research has shown positive effects of ARS: Students think their learning success is higher in courses with ARS (Ehlers et al., 2010; Uhari et al., 2003; Rascher et al., 2003), the courses are rated less boring (Tremblay, 2010), and students motivation is increased (Ehlers et al., 2010). The usage of ARS also leads to a significantly higher learning success (Chen et al., 2010). In summary, ARS can be highly recommended for higher education teaching. But besides high asset costs for commercial solutions, most of the ARS are functionally restricted to multiple choice and numeric answer formats. The didactic benefit is therefore limited to the realizable scenarios. When designing a course, the lecturer searches for the tools and methods he needs to realize his teaching goals (following the *'form follows function'* approach). At this point, lecturers often face the problem that there is no tool fitting exactly their needs and expectations. In the end, they have to arrange with semi-ideal solutions. In other cases, lecturers use the tools for the sake of the tools, not taking into account that the tools have to be subordinated to their teaching goals. Reasonable course planning therefore needs flexible instruments that can actually fit the stated goals. Hence, we want to offer a modular construction system that can be used according to the individual demands of the lecturers.

2.2 Current Audience Response Systems

The aim of early systems like Classtalk (Dufresne et al., 1996) was to improve the involvement of every single student. The teacher transferred three to four Classtalk tasks per lesson to the students devices, which were calculators, organizers, or PCs at that time. ConcertStudeo already used an electronic blackboard combined with handheld devices (Dawabi et al., 2003) and offered exercises and interactions such as multiple-choice quizzes, brainstorming sessions, or queries.

Basing on that, Scheele et al. developed the Wireless Interactive Learning (WIL/MA) system to support interactive lectures (Scheele et al., 2005; Kopf and Effelsberg, 2007). It consisted of a server and a client software part; the latter runs on handheld mobile devices. The components communicate using a Wi-Fi network specifically set up for this purpose. The software consisted of a multiple choice quiz, a chat, a feedback, a call-in module and was designed to be easily extendable. The main problem was that students needed to have a JAVA compatible hand-held device, and they needed to install the client software before they could use the system.

Murphy et al. and Kay et al. discussed advantages and disadvantages of commercial ARS (Murphy et al., 2010; Kay and LeSage, 2009). The authors pointed out that the purchase of hardware devices may cause a lot of additional overhead like securing devices against theft, updating, handing out and collecting the devices, providing large number of batteries, handling of broken devices, and instructing teachers as well as students about its usage. Other disadvantages are higher prices of dedicated devices as well as poor maintenance by students.

By now, Students' mobile devices evolved in technical functionality and propagation, that they not only allow an easy interaction with the device but also support the visualization or playback of multimedia content like audio, text, images, or videos (Schön et al., 2012). This new generation of lightweight ARS could also be used more easily in combination with different learning materials like lecture recordings and e-books (Vinaja, 2014). One major advantage of such systems is the possibility to get feedback about the learning progress and to allow teachers to continuously monitor the students' preparation for a course.

Recent research on audience response systems focused on improving the flexibility and expandability of such systems. Web-based systems have been proposed like BrainGame (Teel et al., 2012), BeA (Llamas-Nistal et al., 2012), AuResS (Jagar et al., 2012) or the TUL system (Jackowska-Strumillo et al., 2013). The research focuses on improved user interfaces that allow lecturers to add new questions on the fly, to create a collection of questions, or to check answers immediately. The lecturer may also export the data collected by the ARS for later analysis. Other functions include the possibility for students to update their votes or to support user authentication for authenticated polls. Considering the technical innovations, some systems use cloud technology for better scalability in the case of large groups of students.

When developing a tailor-made ARS in 2012, we defined three requirements for our application (Schön et al., 2012): (1) No additional software needs to be installed on the mobile devices. (2) Almost all modern mobile devices should be supported so that no extra hardware has to be purchased. (3) The system should be integrated into the learning management system of the university.

In our opinion, it is much more suitable to require only a web browser, and of course, every smart phone, notebook, or tablet PC is equipped with one. Figure 1 shows a situation where scanning a QR code printed on an exercise sheet is the only necessary step for entering the quiz. Other tools like *PINGO* (Kundisch et al., 2012) used similar approaches via a QR code for connecting to a quiz round. But all discussed ARS have a limited ability for adaptation. Predefined question types are typically supported well, but the effort for providing new functionality is typically very high.

2.3 Serious Games as specialized ARS

A *serious game* is a fully specialized application, which mostly supports only one single scenario. These games are distinctly more complex in development and usage than typical feedback systems. The collaborative *beer game* implementation from Tseng et al. (Tseng et al., 2008) is a good example for one of these very specific applications. The game illustrates supply-chain effects and let participants experience the impact of their actions in an interactive way. Another highly specialized application was developed by Kelleher and Pausch (Kelleher and Pausch, 2007), who implemented a storytelling game to inspire middle school girls' interest in programming. An important aspect of successful serious games is to combine good game design with good instructional design (Kapralos et al., 2013).

The game should be enjoyable, but basically it has to support the intended learning outcomes. Consequently, the learning progress needs to be assessed during the game. But useful assessment is difficult due to the complexity of a serious game (Bellotti et al., 2013).

Assessments should also provide dynamic, high-quality feedback and encourage students for further success (Al-Smadi et al., 2012). Dunwell et al (Dunwell et al., 2012) presented a prototype application that uses achievements (in-game) and external feedback from an assessment engine. This allows creating dynamic feedback without modifying the serious game. Web services and tools may be used for providing automatic assessment.

A main challenge that prevents the wide acceptance of serious games are high costs for developing a game and the limited reuse of existing serious games (Moreno Ger, 2014). Realizing the drawback of high effort in creating a game for a very small amount of users, Mildner et al. (Mildner et al., 2014) developed a game basis, which is independent of a specific topic. Every lecturer can feed the game with his own questions and answers, which then are integrated into a multiplayer first person shooter. Mehm et al. (Mehm et al., 2012) offered a higher flexibility for the basic game type and developed an authoring tool, which enables lecturers to create a complete story-based game without the need to face source code. However, developing a good game requires efforts in story writing and level design, which should not be underestimated.

Serious games can offer highly interactive learning scenarios with a motivating component where learning takes place along the way of playing the game, but they are also more difficult to use within a course (Moreno Ger, 2014). They often have time restraints and a certain necessity for hardware. Additionally, a serious game needs to fit the teaching style of the lecturer and, to some extend, the learning style of the students.

2.4 Generic Approaches

The main disadvantage of the discussed ARSs and serious games is the high effort when adding or modifying new functionality to existing systems. Usually, every new question type needs an additional implementation and every variation of a lecturer's scenario has to be developed for itself.

Few other systems were developed to offer more flexibility and extensibility like more task and answer formats or support of individual feedback. *MyMathLab* is one prominent example that was evaluated as highly useful to improve the math skills and learning success of the students (Chabi and Ibrahim, 2014). The system analyses the learning progress of students and generates equivalent tasks. Another example comes from Edgar Seemann who created an interactive feedback and assessment tool that 'understands' mathematical expressions (Seemann, 2014).

In our approach, we also integrated some conceptual ideas from MOOCdb (Veeramachaneni and Dernoncourt, 2013), which uses a generic database schema to store many different types of learning content within a small number of database tables. The benefits are, that analyses can compare different types of learning content more easily and new types can be added without changing the database structure.

Consequently, we also wanted to extend the flexibility, adaptivity and extensibility of our system and offer a highly generic software tool. Instead of focusing on math skills, our software tool is applicable to a wide range of subjects, easily extensible and can be fully customized to the lecturers learning scenario.

3 MODEL

Our concept is based on the following assumptions:

- the process of creating a learning unit is divided into five phases,
- every mobile learning scenario is constructed out of a few basic elements, and
- the visualization for the lecturer and the presentation for the students can be deduced automatically from the configuration of these basic elements.

The following subsections discuss the model that has been derived from these three assumptions.

3.1 Phases

We divided the process of a learning unit into the five phases listed in Table 1. The phases are *scenario* definition, create an *entity*, perform a *game* round, discuss the *results* and *analyze* the efficiency.

The first phase consists of stating the learning goals of the course unit and defining the concept of the learning scenario. According to the learning goals, this can be a personal knowledge feedback, an

Table 1: The five phases of our learning scenarios.

Phase	Description
Scenario	Definition of learning scenario.
Entity	Description of the entity of a scenario.
Game	Performance of a classroom activity.
Result	Presentation of the result.
Analysis	Analysis of students behavior and scenario success.

in-class twitter wall, a live experiment in game theory or an audience feedback on the current talking speed. The lecturer has to define the elements and appearance on the students' phones, the interactions between the participants and the compilation of the charts she wants to present and/or discuss in the classroom. Here, the decision is made whether the learning scenario consists of a single submit button or a more complex compilation of buttons, check-boxes, input fields and rules, for example, to simulate a full marketplace situation in an economics class. This phase needs a fair amount of structural and didactic input, because different scenarios can serve different functions. Every scenario has to be structured carefully in order to support the learning process of the students. So, if the scenario contains many buttons, charts and input fields, students are overstrained by the setting, unable to solve the task. On the other hand, an overview of the learners understanding of the current material is hardly possible with a twitter-wall when it comes to large groups, as the number of results does not allow a useful overview.

After defining the scenarios blueprint, lecturers can take a predefined scenario in phase two and build a concrete *entity*. If the scenario is, e.g., a classroom response scenario for knowledge assessment, the concrete questions and possible answers are entered now.

These entities are then used to perform *game* rounds with the actual students (phase 3). A *game* is one specific classroom activity. Every entity is persistent and can be used for many games in different lecture groups or at different times. Games can take a short time period, e.g for a small self assessment test at the beginning of a course, or a longer period, e. g. for self-regulated student exam preparation, for surveys or course evaluations.

In phase 4, the student input can be displayed as *result* charts. Depending on the type of input data, the results are displayed as text, coloured bar charts or summarized pie charts. The colors red, green and blue are automatically used for wrong, correct and neutral answers.

The last phase is mostly for analyses of the learning behavior and the didactic research. Everything entered by the students is logged, and this data can then be used to perform learning analytic research. The data also allows evaluating all learning scenarios under different conditions. We do not track any user specific data, we collect the data of the participating students or teachers anonymously.

3.2 Basic Elements

A classical board game has many similarities compared to an ARS when considering the most important elements. A classical game usually consists of different objects like tokens in different colors, cards with text, or resource coins in different values. *Objects* also exist in a learning scenario and could e.g. consist of questions with answers. Comparable to a board game, these objects also have attributes, for example, whether a question is correct in the case of multiple choice questions or which types of answers are allowed.

Beside the tokens and the board itself, a game has *rules*. They describe the logic behind the game and the way the players interact with the game elements and with each other. Rules are also required for the implementation of an ARS. *Objects* and *rules* already specify the main elements that are the basis for all learning scenarios (see Figure 2).

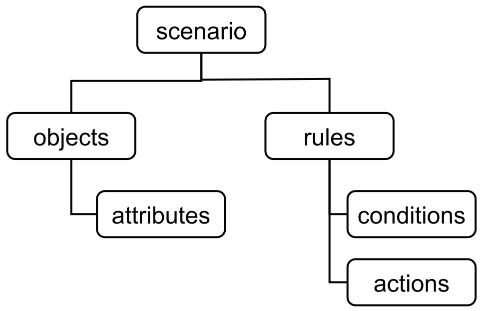

Figure 2: Relation of scenario elements.

Objects are elements holding information, and they have relations to each other. Every object can hold a set of *attributes*, which store the actual data. An attribute can be the text of a question, a check box for an answer or a button for a submit. Every rule consists of a set of *checks* and *actions*. Checks describe the conditions under which the rule gets activated, while actions describe the action that has to be performed. This covers very simple rules, like a button-click triggering a player's counter to increase by one, to very complex ones where a sum of an attribute of all players exceeds a given value, or every player with less than five correct answers gets a warning message on his display.

Every object and every rule belong to a defined context. Object contexts are *game* and *player*. Objects within a game context exist only once in every game (e.g., a time counter), while objects within a player context are created for every player (e.g., answers of a test). Rules can act in the context of *game*, *player* or *object*. While a game context rule checks and executes every attribute of the game (e.g., the game timer is less than 20 seconds), a player-context rule checks the attributes for every player. This could be e.g., an additional message if the player answered more than fifty percent of the questions correctly. Similar to the player context, object context rules are checked and executed for every game-object itself. Highlighting questions with wrong answers would be an example for an object context rule.

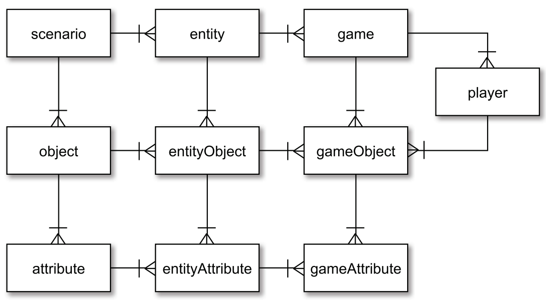

Figure 3: ER diagram of core database tables.

Whereas rules with checks and actions can be stored once in the database, the game elements (objects and attributes) have to be stored for every main phase (scenario, entity and game) separately. As seen in Figure 3 the objects and attributes of the scenario build the blueprint for the entities, which are then copied for every game and player. Scenario objects and attributes store information about the general structure of the learning scenario (e.g., a multiple choice test), entity objects and attributes store information about a concrete topic (e.g., questions about elephants), and game objects and attributes hold the data of one game and player (e.g., a player's answers to the test).

The information for the phases result and analysis are mainly stored in the game objects and attributes. Beside that, we log additional information about user behaviour in two separated tables, which are used for meta analysis and application improvement.

3.3 Generic Approach

We tried to follow a generic approach to keep complexity and configuration effort as low as possible. Therefore, we use the same information in every phase differently. This involves the visibility of objects and attributes, the appearances in form and color, as well as default texts. If an attribute has the type boolean, it is false by default and is displayed as a check box on the students' devices. If it is marked to be displayed as a bar chart, the true and false values of a game will be summarized and displayed as green and red bars. A bar chart about other values, such as different car models, will be displayed in neutral blue bars. A text attribute marked as writeable will be displayed as an input text element, whereas an attribute marked as readable will be shown as a simple HTML text label. An attribute of a numeric type only accepts numeric input.

4 PROTOTYPE

Our aspiration is to have an easy to use application providing the students with a fast and easy access via a QR code displayed on classroom screen and the lecturers with an integrated teaching tool. Besides using existing scenarios, such as classroom quizzes, the tool should be easy enough for lecturers to create their own, customized scenarios.

We have implemented a prototype of the described model as a proof of concept and to provide a powerful tool for learning scenario analyses, the *MobileQuiz2* It is written in PHP and uses the ZEND2[1] framework which already contains many useful functionalities. We used modern web technologies like AJAX, HTML5, CSS3 and the common web frameworks jQuery[2], jQuery Mobile[3] and jqPlot[4].

In an optimal process, a lecturer would start with an idea for a new in-class learning scenario. She then discusses this idea with a didactic expert and defines the scenario model. The expert defines the scenario as an XML file and uploads it to the prototype application. So far, the prototype supports typical attribute types like text, numbers, boolean values and more specialized ones like time, progress bars, images and buttons. For the implementation of the logic, the rules support logical operators, basic and medium complex math operators like arithmetic mean and sum, and vi-

[1]http://framework.zend.com/
[2]http://jquery.com/
[3]http://jquerymobile.com/
[4]http://www.jqplot.com/

sual operators for hiding and displaying specific attributes or objects.

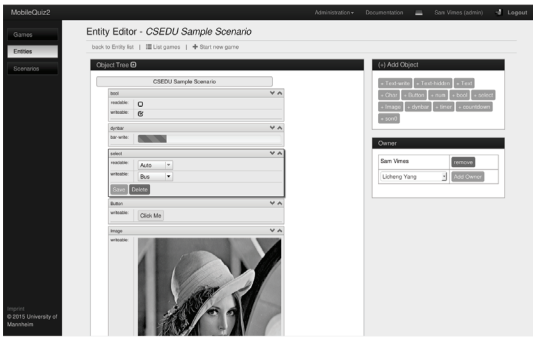

Figure 4: Screenshot of the entity edit page of the MobileQuiz2.

After defining the scenario, a lecturer can then choose it to fill out his own entity. Figure 4 shows the current editor for building a new entity. While new scenarios are available to every lecturer, entities and games have restricted rights for the respective owner.

Lecturers can start a new game round of any of their entities at any time. With the start of a game, a new QR code is created and can be used within the classroom or with any other medium like videos or exercise sheets (see Figure 1). We use a shortened URL to generate a QR code, which is the main connecting point for the audience. Students can then access the game with any JavaScript-supporting, internet enabled device via the URL. The page is only loaded once, and every change or user action is sent to the server via an AJAX request. After processing the rules and changes on the server side, the student's mobile browser gets an AJAX response with the relevant attribute information the page content and layout are changed through JavaScript without the need of a page refresh. After performing a game round, the results can be discussed with the audience or used for the preparation of an upcoming lecture. Our application stores meta data so that all kind of analyses of the students' behaviour can be done afterwards. Beside technical information about the participating devices, the users' activity, the line of action and the performance of the students are tracked.

In an actual lecture the system could be used as follows: A lecturer in game theory (business economics), wants to conduct a live experiment. The game is called *Guess* $\frac{2}{3}$ *of the average*. In this experiment, every participant has to choose a number between 0 and 100. The winner is the one closest to $\frac{2}{3}$ of the mean value. The lecturer first talks to our didactic expert and they decide to model a scenario with a numeric input field and a submit button. If students click the submit button without any number or with a number bigger than 100, a warning text is displayed, otherwise the button and input field disappear and a text message appears.

After a given amount of time, the mean value is calculated together with the corresponding $\frac{2}{3}$ value. The student with the smallest distance to this value sees a winning message. Our expert now models this scenario within an XML file and uploads it to the prototype application. The lecturer then chooses this scenario within the application to create an entity of it and gives it the name of his lecture and some describing text. She enters the entity editor and chooses an amount of time for this entity. During her lecture, she opens the course page within our university's LMS and starts a new classroom activity (game). The students participate in the experiment and while the winner is happy about his guess, the lecturer displays and discusses a chart of the distribution of all answers with the whole class.

5 EVALUATION

We evaluated the proposed model in real classroom settings to proof the practicability of our concept. The goal of this first evaluation was to observe the model's behaviour under real classroom conditions, to rate the overall model validity with its boundaries and drawbacks, to monitor potential performance parameters, and to gain experience with its validity in typical lecture scenarios. We modeled eight different learning scenarios and used the prototype application in five large exercises and twenty-one small exercise lessons in an informatics course and a media didactic seminar. About twenty students participated in the small exercises, about 30 in the seminar and up to fifty in the large exercise. The goal of the informatics scenarios was to investigate different scenario parameters on students' acceptance. The used scenarios were mostly multiple-choice feedback scenarios with differences in structure and appearance. Among other things, one of the scenarios investigated the differences between an open text answer format and predefined single choice answers. Later on, we hid following questions until the visible ones were answered and showed a progress bar to increase students' overview of how far they proceeded.

The purpose of this study was to give a proof-of-concept that our prototype enables the researcher to model the different scenarios and to investigate the impact of the different parameters without having an informatics background. We then used the experience of the lecturer and the generated data to get a qualitative feedback of our approach. The lecturer of the media didactic seminar used a five point likert scale scenario to perform a quick audience benchmark for a range of student talks. This section presents the technical results of these settings, whereas the qualitative feedback from our lecturers is discussed in section 6.

We recorded the delay times of every user action and collected over 37.000 delay times during the mentioned lectures.

Figure 5 shows the time delay for entering the active scenario. In about 70 percent, the scenario could be accessed in less than half a second. But slightly more than 10 percent of the joinings took more than one second.

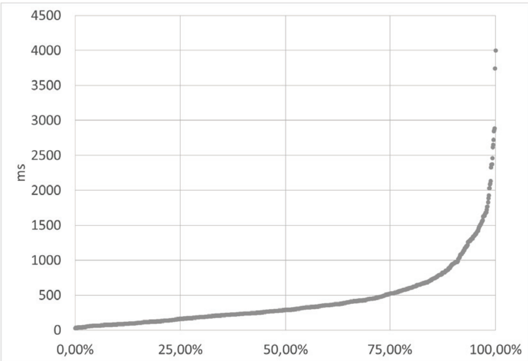

Figure 5: Delay time of student's access to page.

Figure 6 shows the delay for a data refresh. These refreshes perform in the background and are not triggered by an active user action. Data changes triggered by other users are transported via such refreshes. About 90 percent of the background updates are performed within one second.

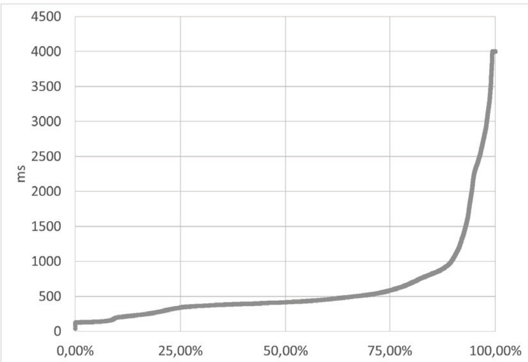

Figure 6: Delay time of regular background refresh.

We experienced, that users distinguish between delay times of different actions. They expect different reaction delays between clicking a button and selecting a check-box. Figures 7 and 8 show the delay times after actively clicking a button or a choice box. Whereas clicking a button typically triggers an event, that can be expected to take some time to be performed, selecting a choice box is expected to be accomplished immediately. Unfortunately, the delay times are high in both cases. Only 30 percent of button clicks, 40 percent of multiple choice selects and only 20 percent of single choice selects were executed in less than one second.

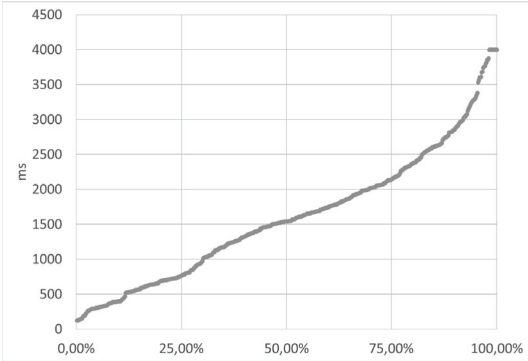

Figure 7: Delay time after clicking a button.

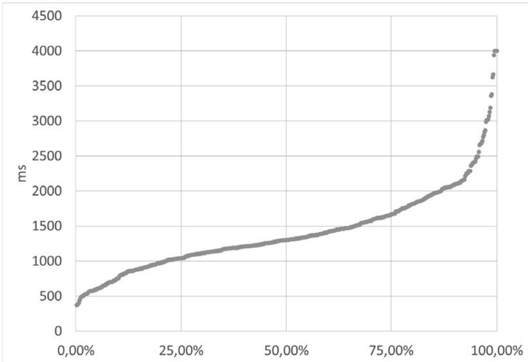

Figure 8: Delay time after using single choice menu.

To identify the bottlenecks of our implementation, we measured the delay time of every user interaction, divided in delay of the network, time to operate the scenario logic, gathering data from the database and updates of the browser cache on the students' smart phones.

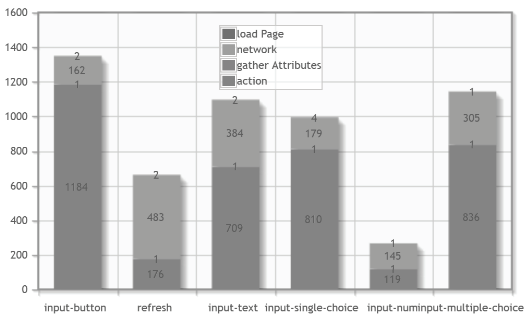

Figure 9: Split up delay times for different activities.

Figure 9 shows, that most of the time is needed to process the game rules and to send the data through the wifi network. Almost no delay arises to gather the user relevant information out of the database and to display it on the devices' browser.

6 DISCUSSION OF THE RESULTS

6.1 Model

As we know today, our model is able to represent every scenario mentioned so far. Objects with a variety of attributes can be created, and the rules are able to describe a complex, adaptive and interactive scenario behaviour. During the evaluation, we were not only able to model the scenarios we had planned from scratch, but we were able to react on unforeseen requirements, too. With the increasing understanding of the potential to customize their learning scenario the lecturers wanted further changes in existing scenarios which we could do by editing only the XML structure and without touching the applications source code. The lecturers mainly used scenarios with multiple-choice and text questions. They then extended the behaviour, so that further questions appeared depending the given answers during the quiz. For the big exercise, they modeled a complete question catalog of different topics and let the students decide for themselves which field they wanted to work on. The lecturers gave us a homogeneous positive feedback on the possibilities our model gave them to define their own lecture scenario. After getting customized to the objects, attributes and rules, they began to further enhance and diversify the scenarios on their own.

6.2 Performance

After our model-based tool passed the first scenario tests, we ran into vast performance issues during the first big exercise. The amount of game attributes (cf. Figure 3) increased rapidly with the number of participants, up to five thousand per game. Unoptimized algorithms of rule execution, together with a 5 seconds-refresh, finally crashed the web server. After optimizing the application's algorithms, we decided to track the times more specifically to identify potential bottlenecks and get better background information.

We fixed the first performance issues and achieved mean response times of less than 1.5 seconds. The usual response times were in a range that our students accepted as normal behaviour for a web application. The long response time for using a button is mostly accepted, because students expect something to happen after clicking at one. But even so, during scenarios with a complex body of rules, the experienced delay came close to the threshold of dissatisfaction. Refresh and data input actions mainly run in the background, so that they do not experience waiting times and don't recognize if they got an updated information with a delay. However, similar delay when selecting a choice box is not accepted. Students expect an immediate feedback when using a basic HTML element and do not tolerate unusual delays.

6.3 Usability

The handling of the system for the students is fast and easy. It affords as little technical understanding as writing a short message or using mobile applications, as we use the common application elements and libraries. Scanning a QR code has become a common task, and students are familiar with writing longer passages of plain text on their smart phones.

The usability for the lecturer depends on the phase. Designing a new scenario (phase one) still needs basic computer science knowledge and information of the system in greater depth, as the scenario has to be defined in an XML structure. Besides from not having a graphical scenario editor yet, creating objects, attributes and rules needs a basic understanding of programming concepts. As it is recommended to consider didactic principles in new scenarios, we are planning to support the creation of new scenarios by our university's didactic center. Depending on the complexity of the desired scenario, phase one can take from fifteen minutes up to several hours.

The handling of already defined scenarios is quite similar to the usage of tools mentioned in Section 2 as we can design scenarios to appear the way they these tools behave. But usability also depends on the scenario's complexity. Opening a new topic for a twitter wall would need far less effort than defining questions and answers for several rounds of adaptive multiple-choice questions. We have experienced no conspicuousnesses in the usability for the lecturers so far, but we have not examined feedback for a significant number of subjects yet.

6.4 Didactics

The advantages of our development are essential: at the current state of development, lecturers can create new scenarios that fit their needs, with only the initial help of an didactics expert. No programming expert is needed. They no longer have to search for other specialized tools to realize their planned learning scenarios. The *'form follows function'* approach is fully taken into account as the lecturer can independently plan his course and afterwards use (or create) equivalent e-learning scenarios. The usage of defined scenarios during preparation and execution of the lecture is manageable for the lecturer himself.

The current limitations that defining a scenario needs a basic understanding of programming will be eliminated within the process of implementing the system in the learning management system of the university. But, as we mentioned before, creating scenarios on the basis of the teaching and learning goals is not self-explanatory. Profound didactic competences or at least a good intuition are necessary to design a useful and effective learning environment. Therefore, we realized an educational consulting service that also includes help with the technical realization of individual courses and scenarios. With the new software tool, students benefit from even more varied and activating teaching methods in comparison to the original *MobileQuiz*. Against our assumption, experience showed that students even like to type longer free texts into their smart phones in order to answer questions. This offers additional didactic possibilities concerning the course design. Using our tool, we do not track any user specific data of students or lecturers. This allows us to perform many analyses without data privacy issues, but disables the opportunity for individual grading or long term tracking.

7 CONCLUSIONS

We presented a new model for describing different learning scenarios in a typical classroom setting. By focusing on objects, attributes and rules, we are able to describe a large amount of different scenarios within one model. We implemented a prototype application based on this model and did a first proof of concept in more than 25 university exercises with up to 50 students. Our results showed that the model is valid and satisfies the expectations. However, we were confronted with significant performance issues in scenarios with more than 30 participants and a non-trivial body of rules. Complex marketplace simulations in a course with 100 students would cause noticeable delays at current state.

We are now improving the prototype and are focusing on performance, stability and lecturers' experience. We are also implementing the tool in the university's LMS to grant an easy accessibility and usage for the lecturers, also for these without programming experience. Even in the current state of implementation, we experience a high interest of our lecturers who want to use the new application in their lectures.

The prototype will reach beta phase during the next Spring semester, and we are planning to steadily increase the number of lectures using the enhanced learning scenarios at our university. We will further extend the overall functionality and analysing capabilities of our tool. We established technical and educational expertise at the university's didactic center to support lecturers in defining new mobile learning scenarios and enhancing their classroom interactivity in novel ways. With the growing amount of users, we plan to evaluate the approach on further, more didactic centered focuses (like the creation of educational patterns) and to extend the approach to serious gaming scenarios.

ACKNOWLEDGMENTS

Special thanks to Licheng Yang for his frontier spirit in using the prototype and Dominik Campanella for his cutting-edge technology expertise in web programming. We also want to thank the lecturers who enthusiastically test and evaluate our developments and inspire our work in new directions, especially Philip Schaber and Henrik Orzen.

REFERENCES

Al-Smadi, M., Wesiak, G., and Guetl, C. (2012). Assessment in serious games: An enhanced approach for integrated assessment forms and feedback to support guided learning. In *Interactive Collaborative Learning (ICL), 2012 15th International Conference on*, pages 1–6.

Bellotti, F., Kapralos, B., Lee, K., Moreno-Ger, P., and Berta, R. (2013). Assessment in and of serious games:an overview. *In Advances in Human-Computer Interaction Volume 2013*.

Chabi, M. and Ibrahim, S. (2014). The impact of proper use of learning system on students performance case study of using mymathlab. In *6th International Conference on Computer Supported Learning*, pages 551–554.

Chen, J. C., Whittinghill, D. C., and Kadlowec, J. A. (2010). Classes that click: Fast, rich feedback to enhance student learning and satisfaction. *Journal of Engineering Education*, pages 159–168.

Dawabi, P., Dietz, L., Fernandez, A., and Wessner, M. (2003). ConcertStudeo: Using PDAs to support face-to-face learning. In Wasson, B., Baggetun, R., Hoppe, U., and Ludvigsen, S., editors, *International Conference on Computer Support for Collaborative Learning 2003 - Community Events*, pages 235–237, Bergen, Norway.

Dufresne, R. J., Gerace, W. J., Leonard, W. J., Mestre, J. P., and Wenk, L. (1996). Classtalk: A classroom communication system for active learning. *Journal of Computing in Higher Education*, 7:3–47.

Dunwell, I., Petridis, P., Hendrix, M., Arnab, S., Al-Smadi, M., and Guetl, C. (2012). Guiding intuitive learning in serious games: An achievement-based approach to externalized feedback and assessment. In *Complex, Intelligent and Software Intensive Systems (CISIS), 2012 Sixth International Conference on*, pages 911–916.

Ehlers, J. P., Mbs, D., vor dem Esche, J., Blume, K.; Bollwein, H., and Halle, M. (2010). Einsatz von formativen, elektronischen testsystemen in der prsenzlehre. *GMS Zeitschrift fur Medizinische Ausbildung*, 27.

Jackowska-Strumillo, L., Nowakowski, J., Strumillo, P., and Tomczak, P. (2013). Interactive question based learning methodology and clickers: Fundamentals of computer science course case study. In *Human System Interaction (HSI), 2013 The 6th International Conference on*, pages 439–442.

Jagar, M., Petrovic, J., and Pale, P. (2012). Auress: The audience response system. In *ELMAR, 2012 Proceedings*, pages 171–174.

Kapralos, B., Haji, F., and Dubrowski, A. (2013). A crash course on serious games design and assessment: A case study. In *Games Innovation Conference (IGIC), 2013 IEEE International*, pages 105–109.

Kay, R. H. and LeSage, A. (2009). Examining the benefits and challenges of using audience response systems: A review of the literature. *Comput. Educ.*, 53(3):819–827.

Kelleher, C. and Pausch, R. (2007). Using storytelling to motivate programming. *Commun. ACM*, 50(7):58–64.

Kopf, S. and Effelsberg, W. (2007). New teaching and learning technologies for interactive lectures. *Advanced Technology for Learning (ATL) Journal*, 4(2):60–67.

Kundisch, D., Herrmann, P., Whittaker, M., Beutner, M., Fels, G., Magenheim, J., Sievers, M., and Zoyke, A. (2012). Desining a web-based application to support peer instruction for very large groups. In *Proceedings of the International Conference on Information Systems*, pages 1–12, Orlando, USA. AIS Electronic Library.

Llamas-Nistal, M., Caeiro-Rodriguez, M., and Gonzalez-Tato, J. (2012). Web-based audience response system using the educational platform called bea. In *Computers in Education (SIIE), 2012 International Symposium on*, pages 1–6.

Mehm, F., Göbel, S., and Steinmetz, R. (2012). Authoring of serious adventure games in storytec. *E-Learning and Games for Training, . . .* , pages 144–154.

Mildner, P., Campbell, C., and Effelsberg, W. (2014). Word Domination. In Göbel, S. and Wiemeyer, J., editors, *Lecture Notes in Computer Science*, volume 8395 of *Lecture Notes in Computer Science*, chapter 7, pages 59–70. Springer International Publishing.

Moreno Ger, P. (2014). eadventure: Serious games, assessment and interoperability. In *Computers in Education (SIIE), 2014 International Symposium on*, pages 231–233.

Murphy, T., Fletcher, K., and Haston, A. (2010). Supporting clickers on campus and the faculty who use them. In *Proceedings of the 38th Annual ACM SIGUCCS Fall Conference: Navigation and Discovery*, SIGUCCS '10, pages 79–84, New York, NY, USA. ACM.

Rascher, W., Ackermann, A., and Knerr, I. (2003). Interaktive Kommunikationssysteme im kurrikurlaren Unterricht der Pädiatrie fur Medizinstudierende. *Monatsschrift Kinderheilkunde*, 152:432–437.

Scheele, N., Wessels, A., Effelsberg, W., Hofer, M., and Fries, S. (2005). Experiences with interactive lectures: Considerations from the perspective of educational psychology and computer science. In *Proceedings of th 2005 Conference on Computer Support for Collaborative Learning: Learning 2005: The Next 10 Years!*, CSCL '05, pages 547–556. International Society of the Learning Sciences.

Schön, D., Klinger, M., Kopf, S., and Effelsberg, W. (2012). MobileQuizA Lecture Survey Tool using Smartphones and QR Tags. *International Journal of Digital Information and Wireless Communications (IJDIWC)*, 2(3):231–244.

Schön, D., Klinger, M., Kopf, S., and Effelsberg, W. (2013). HomeQuiz: Blending Paper Sheets with Mobile Self-Assessment Tests. In *World Conference on Educational Multimedia, Hypermedia and Telecommunications (EDMEDIA) 2013*.

Schön, D., Kopf, S., and Effelsberg, W. (2012). A Lightweight Mobile Quiz Application with Support for Multimedia Content. In *2012 International Conference on E-Learning and E-Technologies in Education (ICEEE)*, pages 134–139, Lodz, Poland. IEEE.

Seemann, E. (2014). Teaching Mathematics in Online Courses - An Interactive Feedback and Assessment Tool. In *Proceedings of the 6th International Conference on Computer Supported Education*, pages 415–420. SCITEPRESS - Science and and Technology Publications.

Teel, S., Schweitzer, D., and Fulton, S. (2012). Braingame: A web-based student response system. *J. Comput. Sci. Coll.*, 28(2):40–47.

Tremblay, E. (2010). Educating the Mobile Generation using personal cell phones as audience response systems in post-secondary science teaching. *Journal of Computers in Mathematics and Science Teaching*, 29:217–227.

Tseng, C. C., Lan, C. H., and Lai, K. R. (2008). Modeling Beer Game as Role-Play Collaborative Learning via Fuzzy Constraint-Directed Agent Negotiation. *2008 Eighth IEEE International Conference on Advanced Learning Technologies*, pages 634–638.

Uhari, M., Renko, M., and Soini, H. (2003). Experiences of using an interactive audience response system in lectures. *BMC medical education*, 3:12.

Veeramachaneni, K. and Dernoncourt, F. (2013). MOOCdb: Developing data standards for mooc data science. *AIED 2013 Workshops*, pages 1–8.

Vinaja, R. (2014). The use of lecture videos, ebooks, and clickers in computer courses. *J. Comput. Sci. Coll.*, 30(2):23–32.