TECHNISCHE UNIVERSITÄT DRESDEN

DEPARTMENT OF COMPUTER SCIENCE
INSTITUTE OF SYSTEMS ARCHITECTURE
CHAIR OF COMPUTER NETWORKS
PROF. DR. RER. NAT. HABIL. DR. H. C. ALEXANDER SCHILL

Diplomarbeit

to obtain the degree
Diplom-Informatiker

Automated Control of a Web Service during a PowerPoint Presentation

Sascha Kath
(Born May 18, 1991 in Schlema)

Professor: Prof. Dr. rer. nat. habil. Dr. h. c. Alexander Schill
Tutor: Dr.-Ing. Tenshi Hara

Dresden, October 16, 2017

Hier Aufgabenstellung einfügen!

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die von mir am heutigen Tag dem Prüfungsausschuss der Fakultät Informatik eingereichte Diplomarbeit zum Thema:

Automated Control of a Web Service during a PowerPoint Presentation

vollkommen selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt sowie Zitate kenntlich gemacht habe.

Dresden, den 16. Oktober 2017

Sascha Kath

Abstract

In university only a few students pick up new knowledge in lectures. Most students who attend a lecture don't focus enough or loose touch. Many students even decide to skip lectures and just study by themselves. One major reason is that university lectures are neither engaging nor motivating enough. Audience response systems can help to increase teaching quality by engaging the audience and providing students with the possibility to test their understanding. At TU Dresden an audience response system named Auditorium Mobile Classroom Service (AMCS) is developed. In this work we investigate the automated control of the AMCS web service during a PowerPoint presentation by providing a PowerPoint add-in. First, audience response systems are explained and existing audience response systems PowerPoint add-ins are examined, including the existing lecturer add-in for AMCS. Secondly, different add-in types and implementation technologies and practices are investigated. Followed by the analysis of the requirements for the new lecturer add-in. Then, the concept, which is derived from the requirements, is given and the implementation of the concept is described. The created prototype is a web add-in that is runs in a taskpane in PowerPoint. Finally, the usability and acceptance criteria are evaluated. The usability evaluation is done through a usability study and field observations. The main goals are to investigate the user satisfaction, seamless integration and learnability. Consequently, changes derived from the evaluation results are applied to the system.

Contents

1 Introduction

In university only a few students pick up new knowledge in lectures. Most students who attend a lecture don't focus enough or loose touch. Many students even decide to skip lectures and just study by themselves, because they find lectures ineffective. One major reason for that is, that university lectures are neither engaging nor motivating enough. It is difficult for the lecturer to interact with a broad variety of students to check if they understood the explained concepts. Furthermore, for some students the inhibition threshold to ask a question in front of hundreds of other students is too high. Students may also loose focus, because the material is presented too fast or they are not interested in every bit or already know some concepts that are taught, but than miss the point to pay attention to the lecture again.
By using audience response systems (ARS) these problems can be solved. ARS can help to increase teaching quality by engaging the audience and providing students with the possibility to test their understanding. By using ARS a new channel between lecturer and students is opened. It can be used to ask students questions and evaluate them. Additionally, it is possible for students to ask the lecturer questions or notify the lecturer about the speed. Furthermore, messages with additional information can be sent to students. Messages can also be used to notify the student to pay attention, because a field of interest will be discussed. ARS can also be used to check the knowledge of the students before the lecture and assess their background knowledge. This can be used to adjust the lecture and use the time more effectively. In example, by only shortly reviewing well understood concepts while taking more time to explain difficult topics in more detail or to solve misconceptions.
The advantage of ARS is that it increases the teaching quality. On the other hand new challenges arise for the lecturer, like creating questions or using ARS while presenting. With this work we want to investigate a way to ease the use of ARS for a lecturer by providing a good integration of ARS into their existing work-flow. Auditorium Mobile Classroom Service (AMCS) is an ARS that is developed at TU Dresden. Currently, the PowerPoint integration is based on the old Visual Studio Tools for Office (VSTO), which only provides support for Windows and has to be installed via an executable.
In this work we investigate the new JavaScript API for Office. We want to achieve better integration into the existing AMCS system by using web technologies and a sophisticated work-flow that is similar to the website. Due to platform independence of the web add-ins it will be also possible to use the new add-in with Mac.
To achieve these goals, first, different ARS are investigated and compared. Furthermore, recent findings about seamless integration and state of the art web technologies are examined. Based on the analysis, the concept is developed and prototypically implemented. Finally, the prototype is evaluated.
The background chapter introduces ARS and investigates and compares related work. The second chapter analyses the existing AMCS, including the existing lecturer add-in. Especially focusing on the work-flow. Furthermore, the analysis chapter contains technological comparisons between the old add-in and the new JavaScript API for Office. Followed by recent findings about seamless integration and the examination of different technological aspects like state of the art web technologies and different architectural

design patterns. The analysis chapter concludes with the requirements analysis, which leads to the creation of acceptance criteria.
According to the analyzed requirements the concept is created, which is described in the fourth chapter. It provides details about the architecture, work-flow and tests. The following chapter explains the implementation of the concept by explaining general aspects, the actual implementation of the prototype and the realization of the tests.
The following chapter describes the evaluation of the prototype. In this chapter the usability tests are described and the results are presented. Finally, the satisfaction of the requirements through the acceptance tests is evaluated. The last chapter provides a summary and presents future work.

1.1 Definitions

In this section important terminology in the context of Office add-ins is defined.

An add-in is an application that extends the functionality of an Office product.

Web add-ins extend the functionality of Office products by using modern web technologies like HTML5, CSS3 and JavaScript. Web add-ins are provided as a service and integrated into an Office product through an eXtensible Markup Language (XML) file. The XML files are distributed through the add-in store.

Content add-ins are a way to insert additional functionality into a slide of a presentation.

Taskpane add-ins provide a way to interact with the slide set via a service and are not added to a particular slide, but docked at the right side in PowerPoint.

VSTO add-ins are extensions for Office that can interact with an Office application by using the COM-based interface. They are integrated into the ribbon but can also open a taskpane.

2 Background and Related Work

In this chapter the general background of this work is discussed and related works are presented and compared. First of all, audience response systems (ARS) are introduced, including Clicker systems. Followed by the presentation of related work. In this section five ARS that provide PowerPoint add-ins are examined. The chapter concludes with a comparison of the related work discussed before.

2.1 Audience Response Systems

Audience response systems open a new communication channel between the lecturer and the audience. This gives the possibility for two way communication. On one hand the lecturer can check if the students understood the taught material and on the other hand the students can ask questions and give feedback to the lecturer, e.g. about speed. The context of this work is the education in university. Hence, the audience would consist of students and each lecture is part of a course. In general, the interaction in a lecture with several hundred students is very limited. The lecturer can only interact with a few students or has to use some effort to reach more. However, it is almost impossible to get a significant measure of the students who are able to answer a question correctly to reassure that the taught material has been understood. Additionally, the anonymous participation provided through ARS reduces the fear of giving an answer, because students may be too anxious to give a wrong answer. Asking questions can also make students uncomfortable, because it could be an easy question that only shows that the student didn't understand the easiest things or didn't pay enough attention. Therefore, ARS can reduce the students stress and lower the barrier for interaction. Thus, increasing the students overall engagement.
In the past, several audience response systems have been introduced. Before smart phones were ubiquitous and a connection to the Internet could be established easily anywhere, Clicker Systems were introduced. Those systems use special hardware components, which are placed at the seats of the audience to collect the responds. For example, this was used in a famous quiz game show in Germany. A participant could use a joker to let the audience answer a question. The questions had four possible answers and so the Clickers at each audience seat had four buttons. After some time the result was accumulated and presented to the host. The advantage of those systems is, that the audience does not need to bring any device. Thus, nobody gets excluded. Furthermore, the learning effort to use such a simple system is negligible. However, the big disadvantage are the costs of such systems and the limited functionality and extensibility.
Through the wide spread use of smart phones and an almost consistently available data network, systems with extra hardware became obsolete. Hence, most modern ARS exploit these circumstances and use web-based solutions or apps for smart phones. There are some systems that use e-mail or sms as well. Some ARS also provide extensions for presentation software. In this way the ARS is better integrated into the existing work-flow and the lecturer doesn't need to use a separate program or website.
The general work-flow for a lecturer who wants to use an ARS begins with creating the presentation

and adding questions to the ARS. Usually, a question is created for a specific slide. Thus, when the slide is reached in the presentation, the ARS will send the questions to all the connected student devices. Then, the students answer the questions and the lecturer can present the results and use them to correct mistakes and focus on the parts that have not been understood that clearly. This also provides a way for the students to test their understanding.

Benefits	Classroom environment benefits	attendance
		Attention
		Anonymity and participation
		Engagement
	Learning benefits	Interaction
		Discussion
		Contingent teaching
		Learning performance
		Quality of learning
	Assessment benefits	Feedback
		Formative assessment
		Students can compare with class
Challenges	Teacher-centred challenges	Lecturer needs to give students time
		Coverage less than without ARS (but more depth)
		Developing questions
		Responding to student feedback
	Student-centered challenges	New method of learning
		Increased confusion in discussions
		Being monitored

Table 2.1: Benefits and challenges for students and lecturers when using ARS (according to Kay and LeSage [14]).

In Table 2.1 the benefits and challenges of ARS according to Kay and LeSage [14]) are shown. The benefits are ranging from classroom benefits over learning benefits to assessment benefits. The main benefits result from more engagement and higher student participation.

Classroom environment benefits result from engaging the students more and increasing students participation (i.e. the barrier to participate is decreased by providing anonymous participation). This also increases the students attention.

The learning benefits result from increased interaction and more discussions. Furthermore, the learning performance can be increased, meaning that students learn more in the lecture, because they pay more attention and have the possibility to test themselves with the given questions. The lecturer can also use the collected feedback and answers to emphasis certain concepts, resolve misconceptions or simply give a more detailed explanation, which ultimately leads to a much better teaching quality.

In conclusion, students mainly benefit from the use of ARS. However, they might have some trouble adjusting to the new method of learning, which may include the feeling of being monitored. Furthermore, the authors mention that discussions could actually increase the students confusion.

On the other hand for the lecturer mainly new challenges arise. The first and maybe most important challenge is to create good questions. Questions are the essence of ARS. They are meant to engage, motivate and assess students. Therefore, it's an important but challenging part for the lecturer that takes extra time when creating the slides. Furthermore, it's not easy to plan the time for a lecture beforehand, since students need an unpredictable amount of time to answer. Even if a timer is used to limit the available time and therefore make it better projectable, there is no point in using the system if the poll will end before most of the students had time to answer the question. Also the benefit of getting student

feedback and formative assessment (rate how well students understood taught concepts) needs extra time. Therefore, the overall covered material is less than without ARS. However, the understanding of the covered topics will be better, because students have to actively use the presented materials, which is in fact a benefit for the teaching quality.
Therefore, it's really important that an ARS provides the lecturer with good tools and a well-conceived work-flow. In the following section some existing ARS systems will be introduced.

2.2 Related Work

While the last section discussed the general aspects of ARS the focus of this section is to analyze systems that can be integrated into PowerPoint and are thus, more similar to this work than ARS in general. As discussed in the last section the lecturer finds himself confronted with many new challenges. To support the lecturer when using ARS a good work-flow is crucial. Hence, a special focus will be the work-flow of the described tools.

2.2.1 Poll Everywhere

The first examined system is Poll Everywhere, which is a proprietary system. However, non-profit use or use in higher education is free until a certain amount of participants. Poll Everywhere provides a web interface where the lecturer can create new polls and also see the results once the poll is active. The website offers three different main pages on the landing page: Polls, Participants and Reports. The landing page itself is Polls. So that the user immediately gets an overview of all the existing polls. It shows a small icon representing the poll type, the description, the active state and the number of responses. Furthermore, a few commands are provided, like Report, Delete and Edit.
Polls are the questions that will be sent to the participants. The major offered poll types are multiple choice, word cloud, Q&A, rank order, clickable image, survey, open ended, and other minor poll types under the category more, e.g. scale questions with numeric or emotion scales. Participants can respond to a question by either using the web interface or send their answer via sms. So even if the user has no

Figure 2.1: Slide with an inserted Poll Everywhere content add-in.

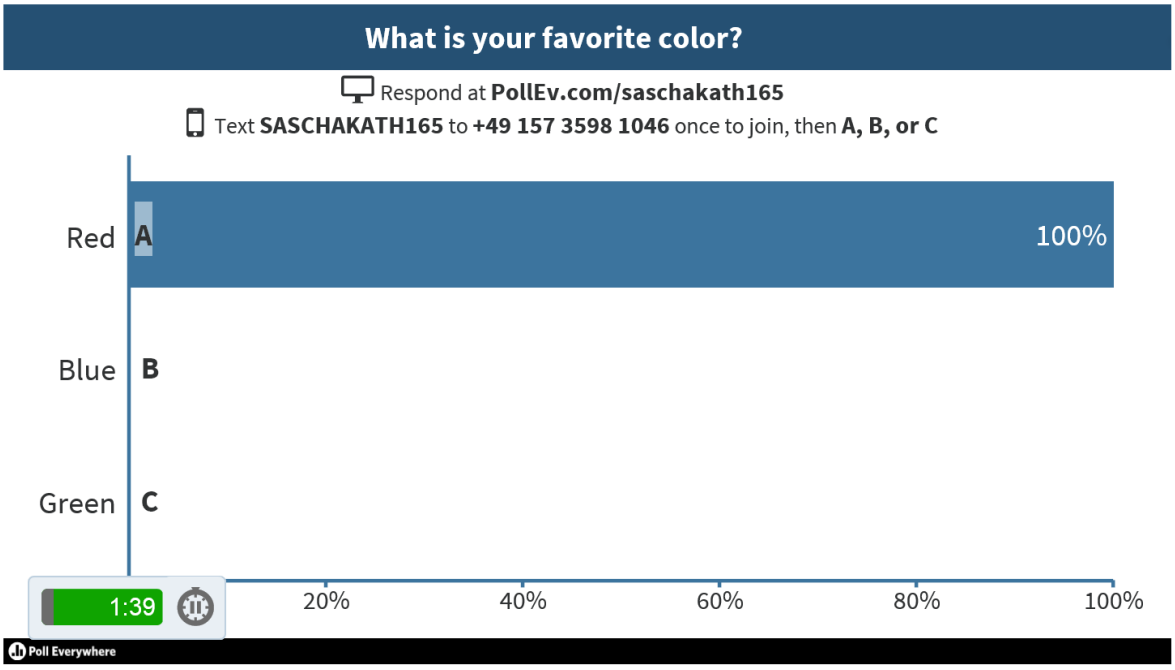


Figure 2.2: Slide in presentation mode with an active slide and a timer.

access to the Internet or doesn't own a smart phone they can still participate with a regular cell phone. Usually, there is no registration needed for giving an answer. The lecturer can force authentication of users, e.g. for grading. When using sms the user first has to subscribe by sending an initial sms. Both the phone number for sms and the web address will be shown on the slide (see Figure 2.2).

The second major page is Participants. This is different to AMCS, because it can be used to gather information of the participants and relate them to answers. The lecturer can set which details about the participant should be submitted with the question. This feature gives the possibility to grade students. In AMCS the participants are only identified by their self given nicknames and are thereby anonymous.

The third major page is Reports. On this page an overview of the results can be found. The answers are represented in charts and the results can also be exported.

To connect questions with slides, Poll Everywhere offers extensions for PowerPoint and Google Slides. Since this work focuses on PowerPoint, the PowerPoint add-in has been examined. The provided add-in type is a content add-in, which means that it will be inserted into a slide and not as a separate taskpane in PowerPoint. The work-flow to create a poll in the PowerPoint slides first requires the lecturer to insert the add-in on a slide. This is done through the 'My Add-Ins' button in the ribbon menu. Now an integrated web page will be shown on the slide. If not already logged in, it will point the user to the Login-Page. After a successful login an overview of all polls categorized by their assigned groups is shown (see Figure 2.3). In the list the poll type icon, the description and the number of responds can be found, very similar to the website. Furthermore, the polls can be edited or new polls created. The poll creation offers only the major poll types mentioned above. When editing a poll the description and the responds can be changed. In some cases the way responds are displayed can be changed (e.g., for open-ended can be chosen Text Wall, Word Cloud, Cluster or Ticker). It is not possible to change the slide or the poll type. When a poll is selected a page with the link to the respond page and the telephone number for sms response will be displayed (see Figure 2.1). Existing responds will be displayed immediately.

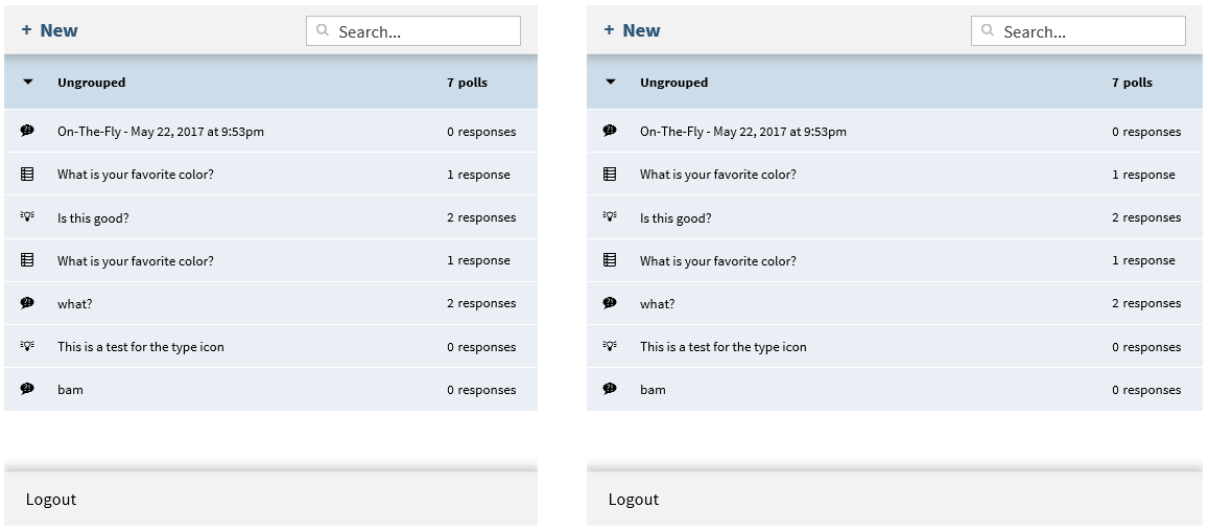


Figure 2.3: Slide with two polls. Both show the question overview - ready to select a question.

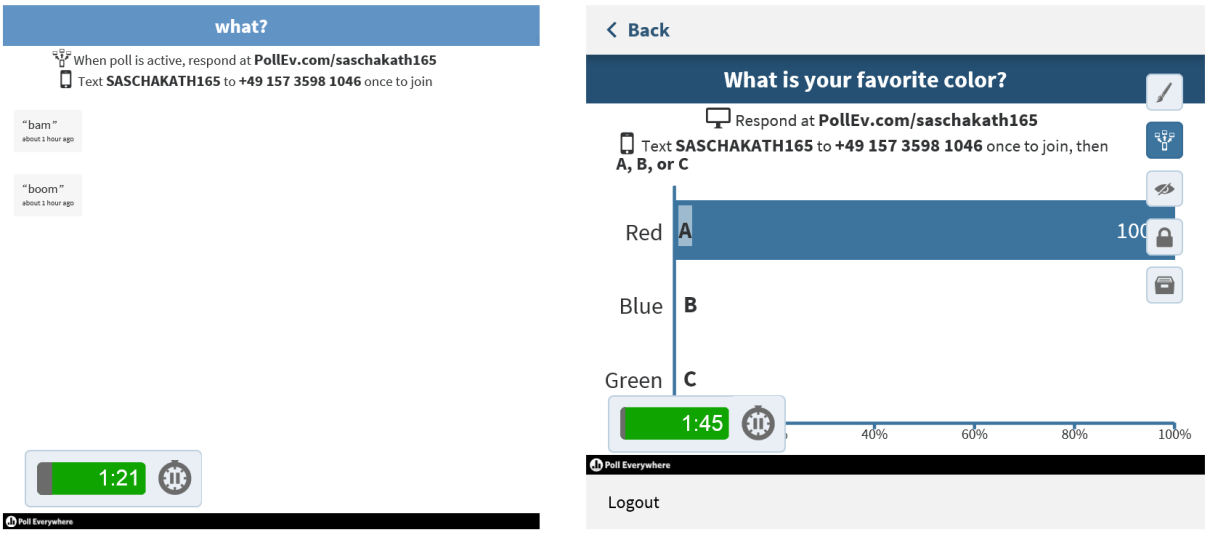


Figure 2.4: Slide with two polls.

The poll will be activated when the slide is active in presentation mode or it can be activated any time by pushing a button. Also the visual settings can be changed, the existing responds can be cleared and the poll can be locked so that no more responds can be given, this could be used for preparation polls before the lecture. It also provides a timer so the time doesn't get out of hand and it is shown so the audience knows how much time is left. When the time has elapsed the poll will be automatically locked. Nonetheless, this could be felt as pressure by the participants and it might be preferable to avoid using timers.

The Add-In provides a very clear, easy to learn and use user interface. Questions can be grouped to make question selection easier. Furthermore, the add-in provides a flexible way when to activate questions and collect data. It could be used before or after the lecture as well. On the other hand it also has drawbacks compared to a task pane Add-In. First of all, the Add-In needs to be inserted on every page where a poll should be placed. Furthermore, once a question is selected for the slide, the poll overview will be gone and only information about the specific poll is available. It is not possible to have a constant overview of

all existing polls in the slide set. Also two polls on one slide seem to cause trouble to the system. While it is possible to insert the content Add-In twice on the same slide (see Figure 2.3). The system doesn't even activate both polls in presentation mode. The second one even stays in editing mode, so that the menu icons stay on the right and the header and footer are displayed as well. However, by using the activate button in the right menu the second question can be activated manually, but stays in edit mode nonetheless as shown in Figure 2.4.

2.2.2 Mentimeter

Another considered ARS is Mentimeter [19]. It is also a proprietary system. The functionality is mainly provided through the website. Compared to the other systems it uses a different approach and offers it's own presentation software comparable to PowerPoint on the website. Even if a PowerPoint presentation is used, the presentations can only be created or edited through the website. The PowerPoint add-in only provides insertion of questions on a particular slide. IActually, it is not possible to even create questions in the Add-In. Hence, it is necessary to use the websites PowerPoint like presentation software, which means a user needs to learn another system. This also creates the feeling that the PowerPoint presentation and Mentimeter are not closely coupled. It is confusing to use 2 Systems for 1 presentation.

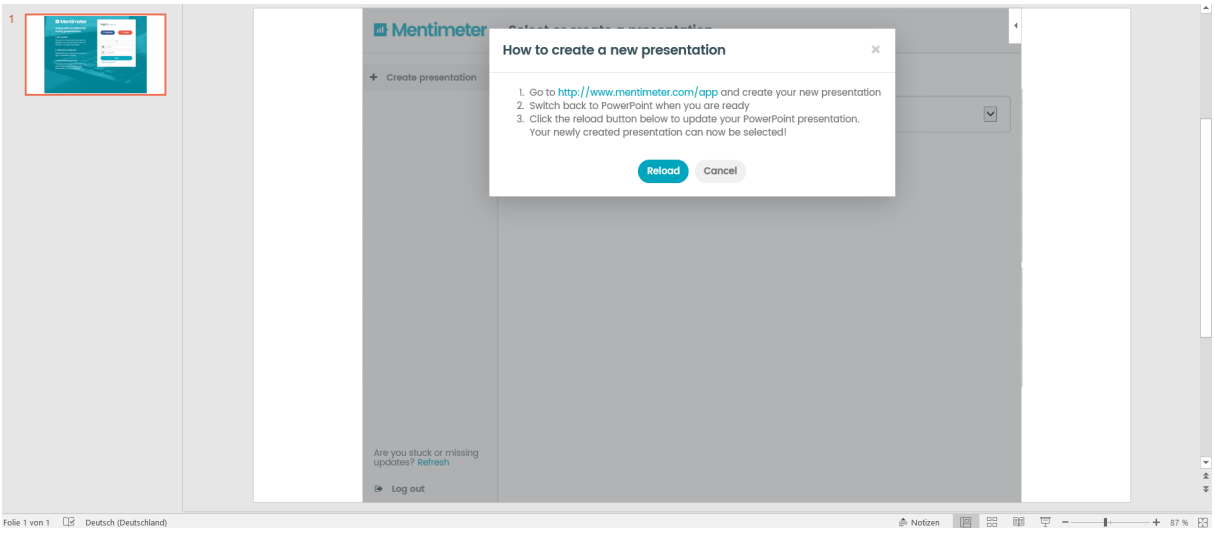


Figure 2.5: When clicking on 'Create Presentation' a popup window instructs the user to visit the website.

On the website similar question types as in Poll Everywhere are offered. The types are multiple choice, image choice, word cloud, quiz (single/multiple choice with correct answer), scales, open ended, 100 points, 2 by 2 matrix, who will win (vote with trophy and nice animation). In contrast to Poll Everywhere question types can be changed on the website. Mentimeter offers a system of giving suggestions for questions by tags for which kind of situation they could be used. In this way lots of question templates are provided. Filters can be applied to find suitable question types. This feature can help ease the barrier of the presenter to create different questions and make the presentation more interesting.

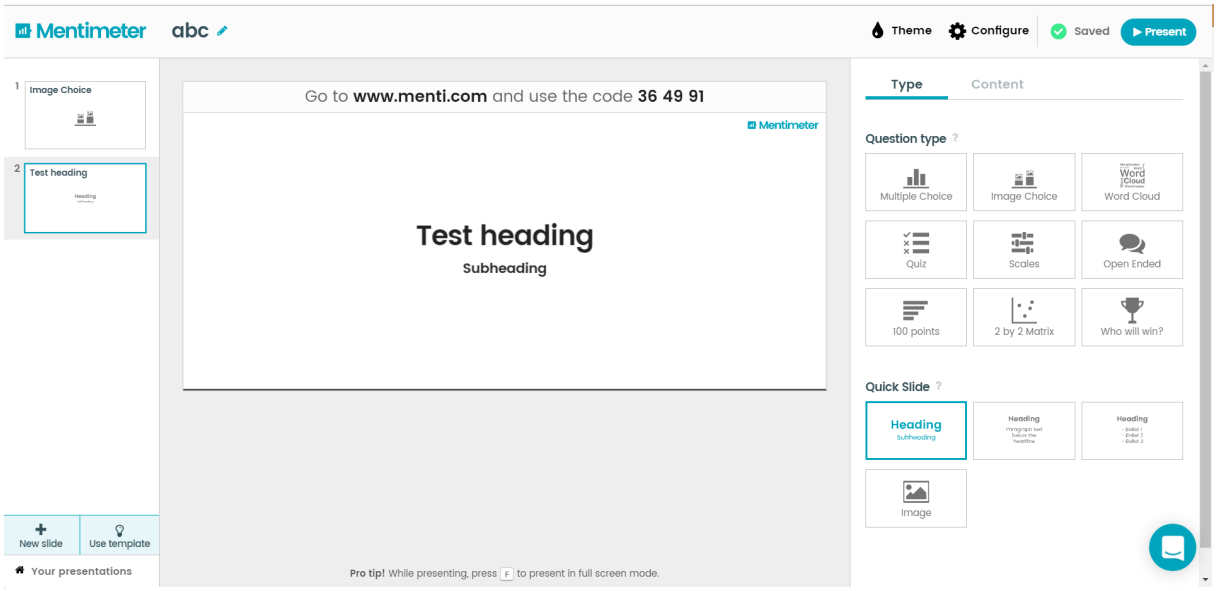


Figure 2.6: Mentimeter's website presentation tool with question and slide templates.

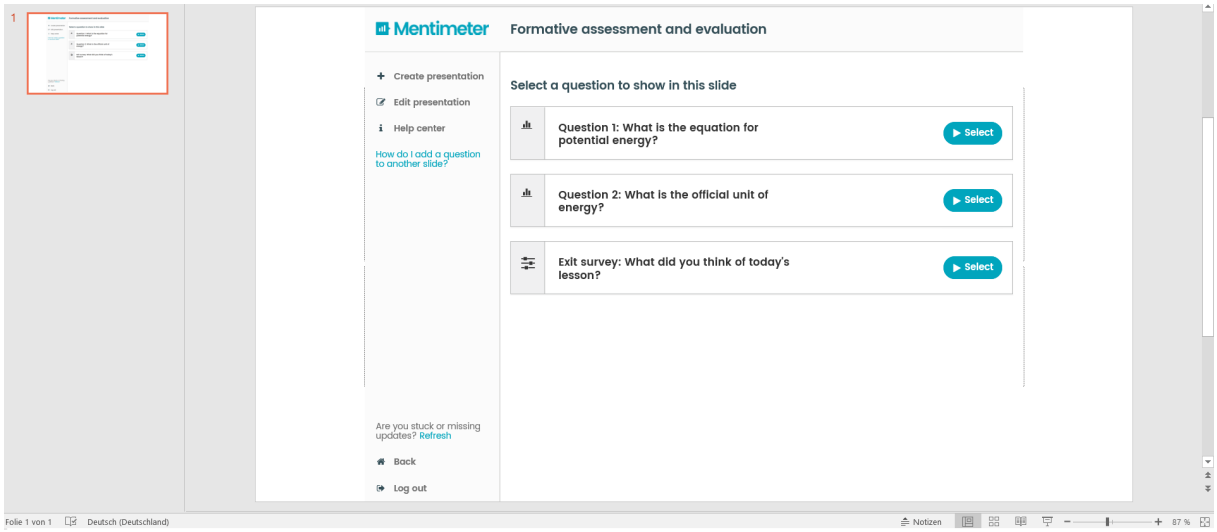


Figure 2.7: Question selection in the content add-in.

When using the presentation software on the website it is also possible to not only insert questions but also slide templates, to make the work-flow faster. However, it is a rather simple system that only offers 4 types of slides. Reactions can be added to a slide as well. The explanation however is rather short and doesn't explain the use well (e.g. no tooltips for icons).

The PowerPoint add-in is a content add-in. The provided functionality is rather limited compared to the website: no presentation creation, no question creation or editing; only lecture and question selection and display of results. As shown in Figure 2.5 when creating a new presentation the add-in points the user to the website. After a new presentation was created new questions can be created by using the presentation software provided on the website.

Figure 2.6 shows the PowerPoint like presentation tool provided on the website. On the left side is a small overview of all slides, the main area shows the active slide and the right side provides templates for new slides and or questions. In the lower left corner below the slide overview it's possible to add

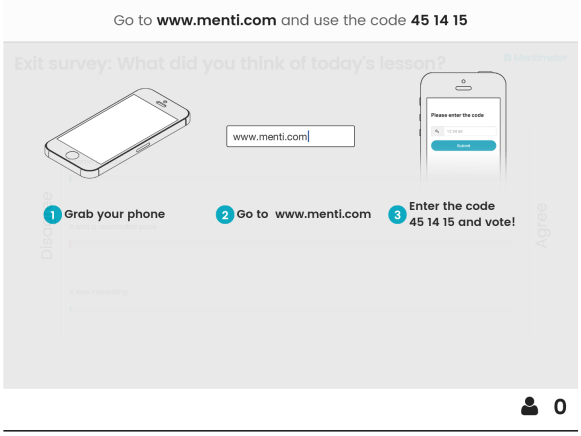


Figure 2.8: Explanation overlay for inserted question in PowerPoint.

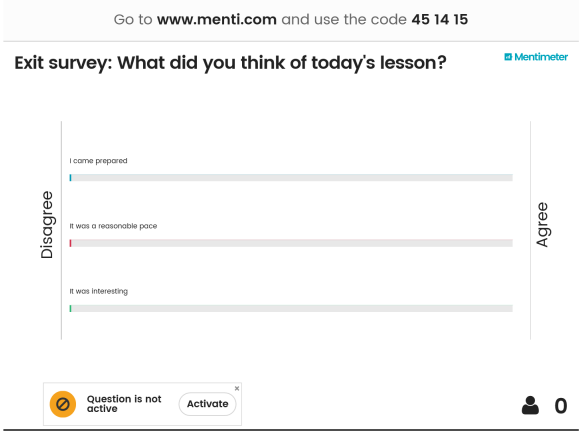


Figure 2.9: PowerPoint slide with a not activated question.

'New slides' or 'Use template'. The 'Use template' provides different question templates that are sorted by tags. Filter with the four main categories Occasion, Purpose, Question type and Audience size can be applied. After a presentation and questions have been created on the website they can be inserted into PowerPoint slides by selecting the desired question in the content add-in as shown in Figure 2.7.

To help the participants use the system each question provides an overlay that explains how to participate in 3 simple steps as shown in Figure 2.8. Each question must be activated manually by the lecturer even in presentation mode (see Figure 2.9). At anytime there can always only be one active question. It is possible to have two questions on one slide but they can only be activated sequentially, which might lead to waiting time for participants who answered quickly.

Users can participate by using the website. All they have to do is insert the code displayed on the top of each question. The results are displayed on the slide but can also be received by e-mail. The lecturer can also share the results by sharing the link to the results. Furthermore, the results can be exported to excel (for paying users) or as PDF (screenshots of the results). The results can be displayed using different layouts like bars, donut pie, sliders, spider charts, wall of text or word clouds.

The PowerPoint add-in is very intuitive and easy to use, but it is not possible to use the add-in as stand alone version. It is absolutely necessary for the lecturer to use the websites presentation software. The provided system is similar to PowerPoint and offers many templates, which both eases the effort to learn the system. However, the user has to learn a new system nonetheless. Instead of learning how to use the website and the content add-in it seems easier to just use the website for simple presentations. Activating every question separately is a bit annoying. It would be better to set the whole presentation to active or automatically activate the question when the slide is shown in presentation mode.

2.2.3 PPTclass Smart Classroom

PPTclass Smart Classroom is a system developed by a Chinese company. It is not quite clear to see, if the project is unfinished or has been shutdown. The servers are still running, but the website doesn't work. Nonetheless, the PowerPoint Add-in is working. No login is required to add a question to a slide. Only two question types are available, which are multiple choice and free text input. The question creation dialog is displayed in Figure 2.10. Participants just use a nickname and can answer the questions (see

Figure 2.11a). The answer choices are not displayed in the user view. The choices are only numbered alphabetically. The answers are displayed immediately on the slide as shown in Figure 2.11b.

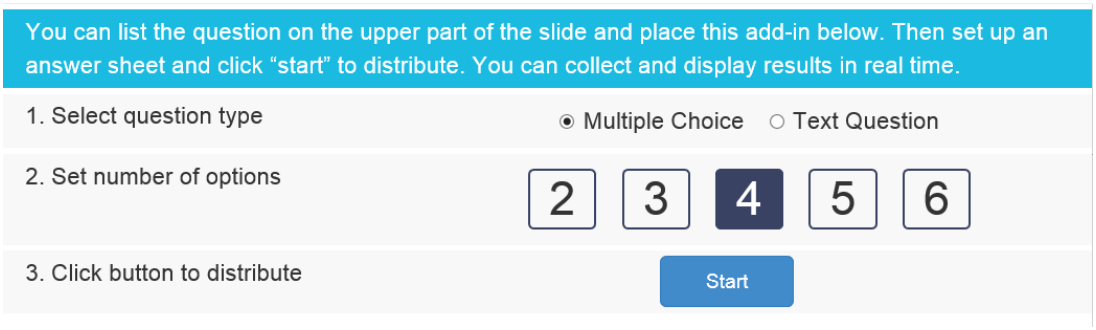
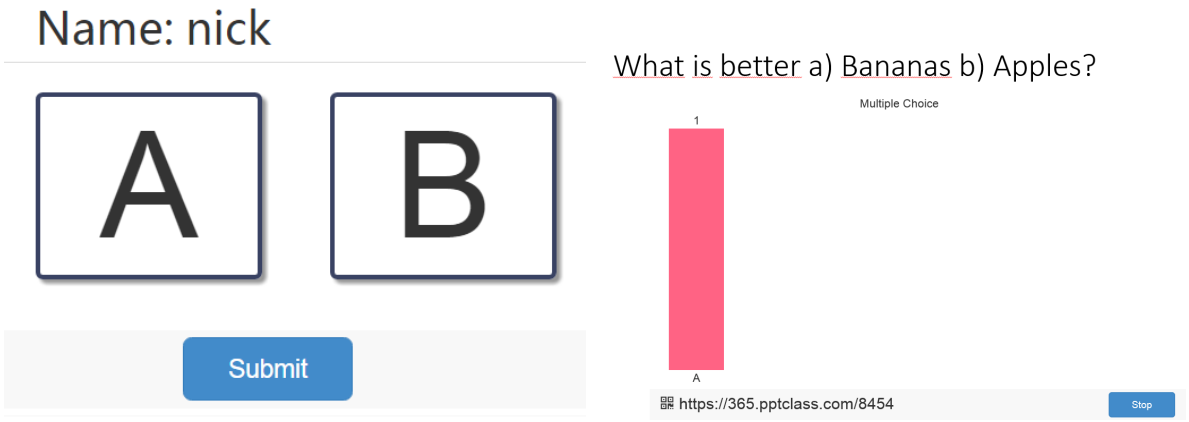


Figure 2.10: Creating a new question in PPTclass Smart Classroom.



(a) Participant view to answer a Multiple Choice question with 2 choices.

(b) Question and result display on the PowerPoint slide.

Figure 2.11: Choice and result screens of PPTclass Smart Classroom.

It is not possible to add a question description in the system, only on the slide. The system also doesn't offer question editing, but it is not really required, because the question just consists of answers since the question description is on the slide. Hence, the effort to just create a new question is very low. To start a question just hit start and the result page will be shown on the slide plus the link for participants to give answers. For every question a separate link is created, so participants have to 'login' with their nickname again for every question. The nickname however can be chosen freely, meaning that one user could use different names to answer different questions.

An advantage of this system is that no login or registration is required and it is possible to have multiple questions on one slide, which can also be active at the same time. This is possible because a different link for every question is used. Hence, the user has to type in a different link for every question and enter his nickname again. However, the overall functionality is very limited. For example the export of results only exists for free text questions. Furthermore, only two different question types exist in total. If the system would offer more question types and provide a single sign-on for users however, the system would not be much different to the two above discussed systems.

2.2.4 Sendsteps

Sendsteps is an Add-In from a dutch company. It is also proprietary, but offers free usage for most of the functionality. In contrary to the three above discussed systems it is a VSTO add-in. This means that the add-in uses the older version of the PowerPoint API for add-ins like the former Lecturer Add-In.
Like the other systems Sendsteps also provides a functionality rich website. While on the other hand the PowerPoint add-in itself provides rather little functionality. The PowerPoint add-ins work-flow is dialog based and the dialogs can be accessed through buttons in the ribbon menu as shown in Figure 2.12. No extra taskpane is given where an overview of questions or different lectures could be found. The two offered question types are Multiple Choice and Open Ended. When clicking on the ribbon button two new slides are inserted into the presentation. One showing the question (see Figure 2.13) and the other one showing the results (see Figure 2.14).

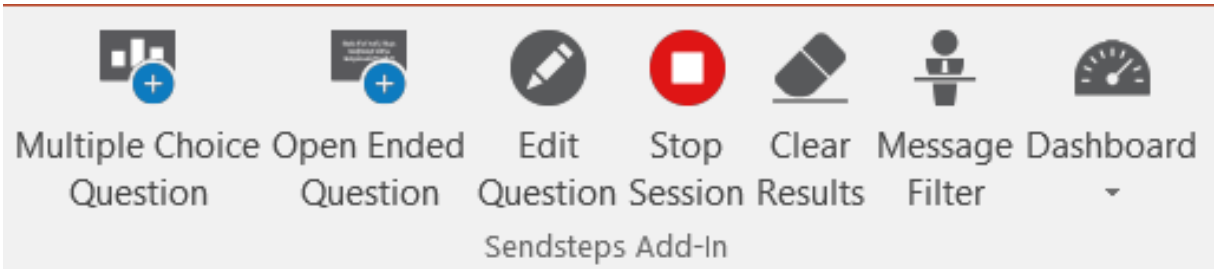


Figure 2.12: Slide in presentation mode with an active slide and a timer.

Since the question is inserted as a new slide, the question description and answer possibilities can be changed directly on the slide. Further options can be changed for a question like display (e.g. results in numbers or percent) or design of the question slide. The Multiple Choice questions can also be transformed into a quiz question, meaning it will have one or more correct answers.
More complex functionality is provided on the website. The website also offers surveys, which provide more question types including scale questions. A created survey appears as a special page for participants. When stopping a session the results can be found on the website. They can also be exported as excel or PDF files.
A disadvantage is that the website and presentation feels really decoupled. Neither the look nor the functionality seem to be cooperating. The website and the add-in feel like two different systems. It is also really confusing that the system does not offer the possibility to create different presentations

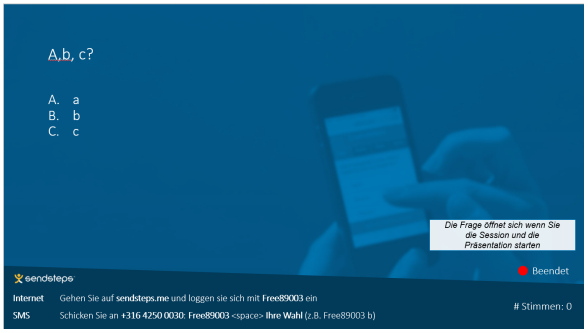


Figure 2.13: Participant view to answer a Multiple Choice question with 2 choices.

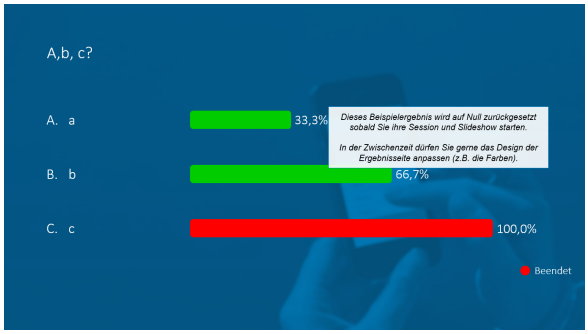


Figure 2.14: Question and result display on the PowerPoint slide.

and group the questions. Another disadvantage is that the presenter can never see an overview of all his lectures with all the questions. Only a list of all surveys can be found on the website. The survey however can only be turned on or off. It can not be connected to a presentation because they don't exist in the system, which is confusing. Furthermore, it is not possible to have multiple questions on one slide since every question creates 2 new slides.

An advantage is the nice integration of the system into PowerPoint through the ribbon buttons and dialogs. The user interface is intuitive, clear and it is much easier to add new questions then in the content add-ins. Furthermore, to edit a question just the slide text has to be changed, which is very intuitive.

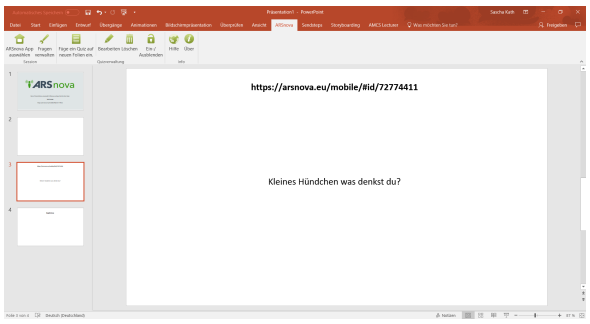
2.2.5 ARSNova

ARSnova is a project of the University TH Mittelhessen. In contrary to the above discussed systems it's an open source project that can be used under GNU General Public License. ARSnova is split into arsnova.click and arsnova.voting. Both are standalone systems that are distributed as Software as a Servic (SaaS). ARSnova.click is more playful and meant for use in schools while ARSnova.voting is designed for the use in university. In the following ARSnova.voting will be examined, because it is meant for use in an university environment, like AMCS.

The website provides a presentation software like Mentimeter. Presentations are organized in sessions. When a new session is created seven templates and the option to customize the session individually are available. The templates are a good feature to help the lecturer to get started. However, in the end it just narrows the available functionality to a few question types decreasing the complexity as well as the flexibility.

Figure 2.15: First slide inserted when using ARSnova.voting.

For each session the lecturer can see the questions the students asked, manage the slides, create preparation questions and flashcards, see the live feedback and get an overview about student answers. Every slide can have text, multimedia content or questions. The available question types on the website are multiple choice, single choice, yes/no, freetext, evaluation, score and hot spots. It also offers the

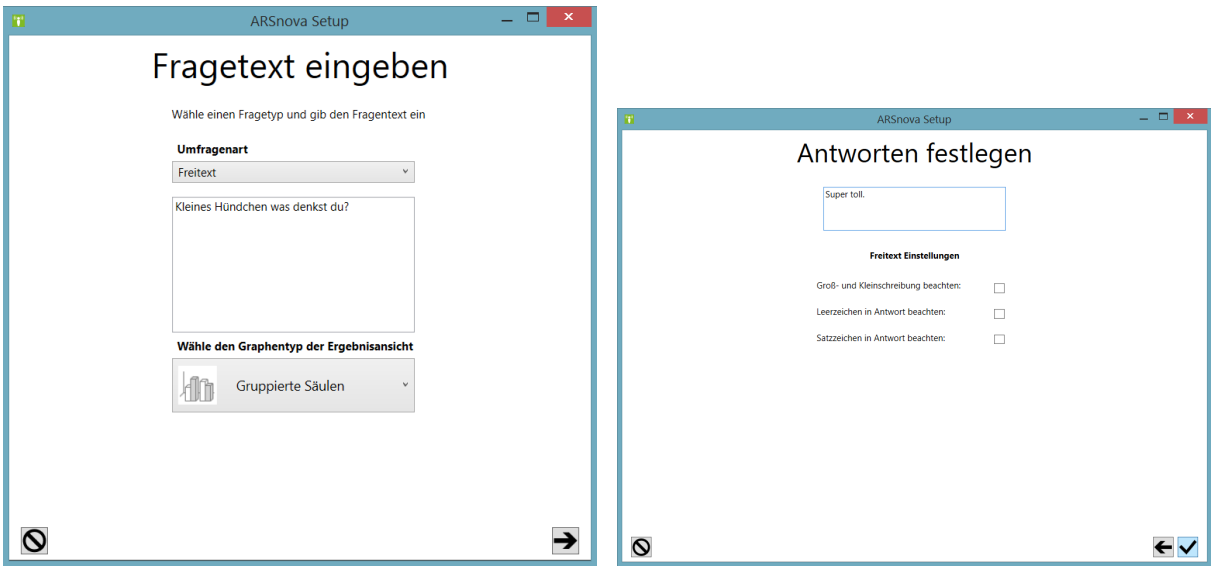


(a) Link and question display on the first slide.

(b) Question and result display on the second slide.

Figure 2.16: Slides inserted by ARSnova when adding a free text question.

possibility to import or duplicate questions. The lecturer can start the presentation on the website. It offers a full screen view so that it is suited for the use with a beamer.



(a) Selecting the question type, title and type of the diagram for the results.

(b) Setting the answers.

Figure 2.17: The two dialogs to insert a new question.

The integration into PowerPoint is done through a VSTO add-in and in contrary to Mentimeter it's not necessary to use the website as lecturer additionally to the PowerPoint add-in. It can be used with ARSnova.click and ARSnova.voting. The functionality of the add-in is accessible through buttons in the ribbon menu. The main interaction with the system is done through separate dialogs. Hence, every button except for the help button open a new dialog window. The ribbon menu is separated into three categories: session, quiz management and info. In the session category the desired ARSnova system can be chosen and an overview of all questions can be found. In the quiz management the lecturer can create a new question, edit, delete and hide or show an existing question. The last category, the info section, provides help and information about the project.

New question creation is done through the ribbon menu. A click on the button 'New Question' opens a dialog where the question type can be selected and a question text inserted (see Figure 2.17a). The add-in offers single choice, multiple choice, yes/no, true/false, estimation questions, freetext and survey as question types. According to the question type the answers have to be created as shown for a freetext question in Figure 2.17b. When the first question is inserted the add-in also creates a slide set cover as first slide showing the used system and the link to the quiz as shown in Figure 2.15. Furthermore, depending on the question type two or three slides are inserted per new question. For a free text question two slides are inserted: one slide shows the question (see Figure 2.16a) and the other slide shows the quiz results. The results will be displayed using excel diagrams (see Figure 2.16b).

The presentation software on the website uses a different work-flow and layout compared to PowerPoint. Hence, the ARSnova.voting website is confusing for first time users and it is definitely necessary to invest time to learn the system. There is no slide list on the left side and also the functions provided on each slide are only concerning this one particular slide. The user has to go back to the overview of all slides or click through the slides until the desired slide is found. The system provided by Mentimeter is clearer

and gives a better overview. Furthermore, the work-flow and layout of Mentimeter's system is similar to PowerPoint, which reduces the learning time, because it's more intuitive to use. However, in Mentimeter it is absolutely necessary to use the website for question creation. In ARSnova the add-in can be used as standalone version and that is a big advantage. On the other hand is the add-in not coupled to the website system at all, which is a drawback. Created sessions on the website and in the add-in can not be used in the other system, but the ARSnova add-in is still under development so if the system integration is achieved it will have an advantage over Mentimeter.
The add-in itself is very similar to Sendsteps. It is situated in the ribbon menu and all interaction is done through dialogs and new slides are inserted automatically on question creation. In contrary to Sendsteps, ARSnova uses excel diagrams to display the results. The diagrams have a good layout and fit neatly into the PowerPoint design. However, the first time creation of the result diagrams takes much longer than in the other add-ins. Also when first using the add-in without any prior knowledge it is a little bit confusing to distinguish between ARSnova.click and ARSnova.voting. Furthermore, it is confusing that the website and the add-in provide different question types. Another drawback is that for every interaction with the system another dialog is used. Every click on a ribbon button opens new dialog window and when closing a dialog, another dialog is opened to make sure the user really wants to close it. This issue is influencing the work-flow negatively. An advantage of the ARSnova add-in is that it is possible to have a list of all created questions. However, the list is also opened in a dialog, which means that while the question list dialog is opened it's not possible to interact with PowerPoint. Hence, the question list dialog has to be closed before further working on the presentation. It would be better to open this question list in a separate taskpane. This again would give ARSnova an advantage over the above discussed systems, because none of them offers the possibility to list all the questions in the presentation.

2.3 Comparison and Summary

In this section a short summary of the systems will be given, comparing advantages and disadvantages. The Table 2.2 provides an overall comparison of relevant features of the five discussed systems.
The first two examined systems Poll Everywhere and Mentimeter are very similar to each other in means of the used technology. Hence, the work-flow in PowerPoint is almost the same. First the content add-in is inserted into a slide, then a presentation (a group of questions in Poll Everywhere) is selected and finally a question/poll can be chosen to be displayed on the slide. The presentation of the results is also very similar.
Poll Everywhere provides a lot of it's functionality in PowerPoint. Hence, it can almost be used as standalone version improving the work-flow significantly, because there is almost no need to switch to the website while editing the presentation. Only poll types can not be changed in the add-in.
Mentimeter on the other hand only allows the user to create new presentations and questions on their website in their own presentation tool, which affects the work-flow negatively. However, Mentimeters own website implementation of a presentation software is closest to this works objectives. It provides a constant taskpane at the right side, where new questions or slides can be inserted or existing questions and slides can be edited. Questions are only on the slide, thus it is not possible to see a list of all questions. The interface however is very clear and a lot of slide templates are provided that can be filtered easily. This is a big advantage of Mentimeter.

PPTclass Smart Classroom uses the same technology as Poll Everywhere and Mentimeter. It's only advantages are that no login is needed and that two or more questions can be active on the same slide, which is not supported by any of the other systems. However, the offered functionality is almost none and the website is not working, so it can hardly be compared to the other systems.
The last two examined systems, Sendsteps and ARSnova, are VSTO add-ins. Therefore, they differ from the other systems in the used technology. The add-in has to be installed via an executable, which is a big disadvantage compared to the other add-ins that are deployed through the store using an XML-file. They both use buttons in the ribbon menu to insert questions. This is faster and more intuitive than inserting questions with a content add-in, because the add-ins have to be inserted every time through the 'Insert' -> 'My Add-Ins' menu. However, both add-ins insert at least two new slides on question creation using at least twice as much space as the other Add-Ins and don't provide the possibility of multiple questions per slide.
Sendsteps only offers two question types and the standard style of the inserted slides looks out-dated. Through the layout can be changed, the other add-ins just come with a clean and nice look without having to adjust the style.
ARSnova and Poll Everywhere offer the most features integrated into PowerPoint. ARSnova inserts two and mostly even three slides for a new question, while Poll Everywhere just needs one slide. Poll Everywhere's layout also looks clearer and more compact, all functionality is centralized in one point. The ARSnova add-in on the other hand has the advantage that it is always present through the buttons in the ribbon and is therefore easier to access. On the other hand the ARSnova add-in must be installed via an executable. Poll Everywhere provides the flexibility to create questions and slides separately and just insert the questions when needed. The question creation in ARSnova is closely coupled to the slide creation. The biggest drawback of ARSnova however is that the website and the add-in are totally decoupled.
In conclusion the two best add-ins among the five above examined systems are Mentimeter and Poll Everywhere. The website provided by Mentimeter provides very good usability and helps the lecturer by providing a lot of templates. However, it lacks integration into PowerPoint. Poll Everywhere achieves almost 100% integration of the system into PowerPoint and is therefore the best add-in.
In this chapter ARS was introduced. ARS enhances lectures by providing a new communication channel between lecturer and audience. By using ARS especially the learning quality can be improved. This is done by engaging the students more and reducing the barrier to participate, because users can interact anonymously. However, many new challenges arise and these mostly affect the lecturer. First of all, using ARS is time consuming and time management becomes a challenge. Secondly, the lecturer has to create meaningful questions and finally, the lecturer also has to respond to students feedback. These challenges consume extra time when preparing the lecture as well as in the lecture itself. This leads to less coverage in a lecture on one hand, but on the other hand it increases the depth of understanding. In example, the lecturer can use the feedback from the students to solve misconceptions. In conclusion using ARS mostly brings advantages to the audience but challenges the lecturer. Training and useful tools are important to help a lecturer exploit the benefits of ARS.
Earlier in this chapter different existing ARS were examined and compared. In the following chapter our existing system will be analyzed. Requirements for a new PowerPoint integrated ARS tool will be derived from the status quo and here drawn conclusions.

	Poll Everywhere	**Mentimeter**	**PPTclass Smart Classroom**	**Sendsteps**	**ARSnova**
Add-in type	JavaScript API for Office	JavaScript API for Office	JavaScript API for Office	VSTO	VSTO
Overall functionality integrated into the Add-In	++ (almost everything, just some question types and report/export is missing)	- (need to use the website)	++ (no need to use the website / can't use the website actually because it's broken)	+ (only necessary core functionality)	+ (but still under construction)
Where is the Add-In placed	slide / content Add-In	slide / content Add-In	slide / content Add-In	ribbon	ribbon
different question types provided in the PowerPoint Add-In	6	0	2	2	6
Create 'lectures' in the Add-In / organization of questions	polls organized in groups	organized in presentations	no lectures / no question list: questions are only on the slide where they have been inserted	no lectures / no question list: questions are only on the slide where they have been inserted	organized in session, but a session only exists in PowerPoint or on the website
Create questions	can create questions and insert them later (has a question list)	in the presentation tool on the website	question is created on insert in the content Add-in (on the slide)	question is created on insert through dialog. inserts 2 new slides	question is created on insert through dialog; inserts 2 or 3 new slides
insert a question	insert the content Add-In on the slide and select the question	insert the content Add-In on the slide and select the question	insert the content Add-In on the slide and create a new question	click on ribbon button and create a new question	click on ribbon button and create a new question
different questions than slide questions	can activate the slide questions anytime (surveys are possible)	questions only for slide - question is only active when slide is active	questions only for slide - question is only active when slide is active	survey open whenever lecturer opens it, until it is manually closed (can be used as preparation, lecture or post-processing question)	questions only for slide - question is only active when slide is active

Table 2.2: Comparison of the investigated related works.

3 Analysis

In the last chapter the background of this work was explained. In this context, ARS was introduced and five different ARS systems were examined and compared.
In this chapter first of all the ARS developed at TU Dresden Auditorium Mobile Classroom Service (AMCS) will be described. A special focus will be the existing PowerPoint add-in. In the following the new API for Office is analyzed and the old and new API's will be compared. After the technological considerations seamless integration is discussed. Seamless integration is important to increase the User Experience (UX) and to achieve the same look and feel for the whole system making it easy to switch between different platforms like the PowerPoint add-in, the browser or the app. Following, different technical approaches are discussed. Starting with state of the art web technologies and then examining different architectural design patterns. The next section presents quality assurance aspects, which include a description of test centered development approaches and explain unit, integration, regression and acceptance tests. The chapter ends with the requirements analysis. In this section the functional requirements, nonfunctional requirements, user stories, scenarios and acceptance criteria can be found.

3.1 Status Quo

In this section an overview of the existing system will be given. The main focus will be the description of the existing PowerPoint add-in. However, the other parts of the system will be introduced briefly as well, including the website, apps and back-end.

3.1.1 AMCS

AMCS is an ARS that is developed at the TU Dresden in cooperation between the faculty of computer science and the faculty of psychology. It is implemented as web-service and all of the functionality can be accessed through a website. AMCS is mainly focusing on use in higher education and has therefore been optimized for the structures of courses that are split into lectures as typical in university. Hence, the main elements of AMCS are courses, lectures and questions. Each course can have one or more lectures. In the system design one lecture equals one presentation and questions exist in a certain context. The available contexts are slide, preparation, lecture, post rocessing and course. Preparation, lecture and post processing questions are all in the context of a certain lecture, but are meant for different purposes and will be activated at different times. Questions can be created for a whole course, a lecture or a particular slide of a presentation. Offered question types are single choice, multiple choice, scale, study single choice, study multiple choice, freetext and grouped scale questions. Study questions are questions with correct and wrong answers. In this way the student can verify the given answers. Participants can respond by using the website or by using either the Android or iOS app. To use the system, participants only need to enter a pseudonym and a password. Also every course has a pin, which students can use when logging in, to enter the course directly. All the different parts of the system communicate with the web-service.

It offers a RESTful API that can be accessed via Hypertext Transfer Protocol (HTTP). There is also a stream that provides additional information for the lecturer about the lecture state and slide changes. The channel is used to synchronize the different parts of the system with each other.

Navigational Structure

The navigational structure plays an important role in creating a similar user journey on different channels of the system and is a part of seamless integration (see section 3.4). In figure 3.1 the relevant parts of the navigational structure of the website is shown. Forms are pictured as separate pages, because forms are opened as overlay and are hence blocking interaction with the underlying page. Starting from the landing page with general information the user can navigate to the login page to enter the system. Directly after a successful login the website offers an overview of all courses as well as a list of active lectures. A lecture can be in one of four states: offline, before, during or after. Active lectures are lectures that are not in offline state. If a course is chosen from the course list, the list of lectures of the course is displayed and an existing lecture can be selected or a new one can be created. The list also shows the current state of each lecture. When a lecture is chosen, the user will be directed to the lecture page. The lecture page is divided into three sub-pages. The initial page is the dashboard. The dashboard can also be reached by a click on an active lecture in the first page after login. In the dashboard page the active slide and the lecture state can be found in the control center. The dashboard page also provides the evaluations of all questions. The second sub-page is the question overview. It contains a list of all questions sorted by slide questions, preparation-/lecture-/post processing questions and course questions. New questions can be created through a modal dialog. The third sub-page is messages. Here, the lecturer can create messages that will be sent as notifications to the participants. While on any of the lectures sub-pages the lecture can be edited at anytime. In the Figure 3.1 modal windows appear to be sinks, because no action is leaving them. This is only done for means of simplicity. After submitting or closing a form displayed in a modal window the website will still show the page that was displayed before the modal window was opened.

Presenting Work-Flow

When a presentation is set to 'before', the preparation questions will be activated and sent to all subscribed users. When the lecture starts it should be set to 'during', which activates the lecture questions and delivery of the slide questions and notifications for the active slide. When only using the website without the PowerPoint add-in the active slide must be updated explicitly by the lecturer. Hence, the lecturer has to open the browser as well and when changing a slide in PowerPoint he also has to change the active slide on the website. This affects the work-flow negatively. To enhance and smoothen the work-flow a PowerPoint add-in is provided to automate the slide changes. So the lecturer can just follow the usual work-flow while the add-in is communicating with the service in the background. Nonetheless, the add-in also offers the back way from the website to the presentation. Therefore, the presentation can be controlled from the website.

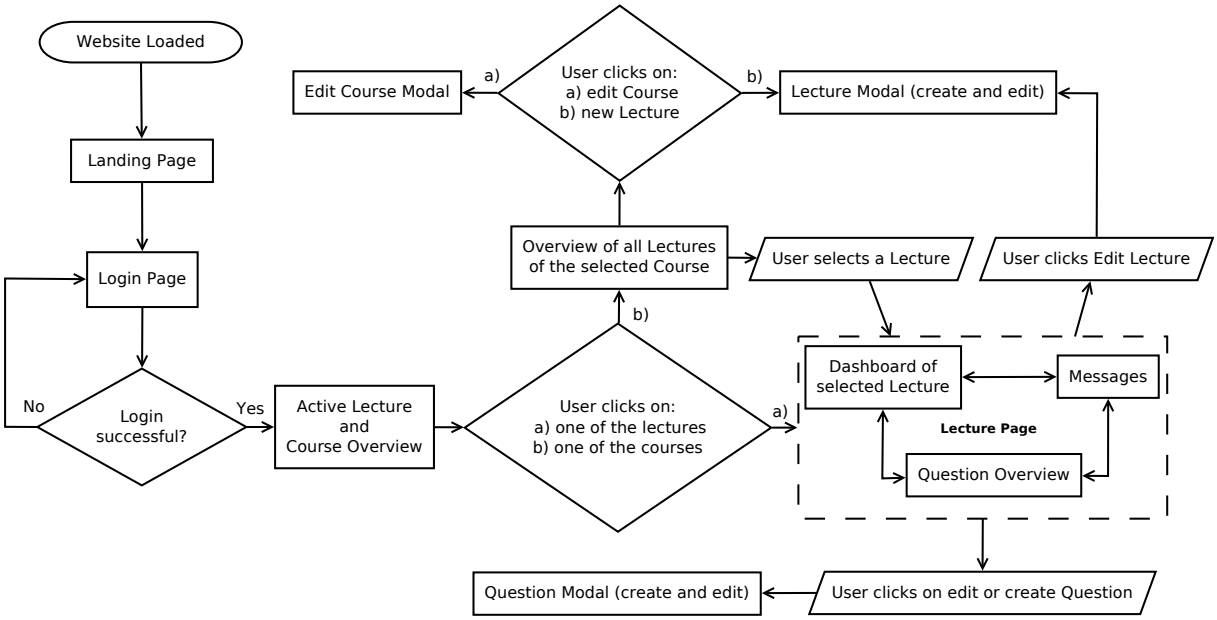


Figure 3.1: Navigational work-flow diagram of the AMCS website.

3.1.2 The Existing Lecturer Add-In

The existing PowerPoint add-in prototype was created as part of a bachelor thesis by Antje Schubotz [27]. In her work she discussed the different technical approaches for the realization of the integration of AMCS into PowerPoint. The four possibilities that were considered are the JavaScript API for Office, Visual Studio Tools for Office (VSTO), Macros and a standalone application using the COM-interface. The last two will only be mentioned here and will not be taken into further consideration, because as the author stated these two techniques don't offer the right functionality for a well integrated work-flow.
In the further discussion of the different add-in types the author states that JavaScript runs in a sandbox and hence, its interaction with the computer environment is limited, making it much more secure but also limiting its functionality. The usage of the COM interface is also very reduced and adaption of the PowerPoint user interface is only possible to some extend. Another point mentioned is the platform and version support. The prototype was only to be developed for windows, which is the only platform VSTO add-ins support. On the other hand VSTO add-ins support older versions of PowerPoint. Concluding mainly from the stated points the author chose the VSTO add-in type for the prototype.
Using the VSTO add-in type gives the big advantage to dynamically adapt the ribbon menu at runtime. Therefor, the author decided to elaborate this fact and integrated most of the functionality into the ribbon menu. The login, course, lecture, slide set selection and further commands for the presentation can be found there. In order to display relevant information in more detail an additional taskpane is used. The list of all questions as well as the question creation are both integrated into a separate taskpane.

Work-Flow

First a login is required. This can be done through a ribbon button that opens a login dialog window. After a successful login the ribbon menu is extended by the course selection, lecture selection and slide-set selection, all as drop down lists in the ribbon menu. A new lecture can be created through a dialog from the lecture drop down list. When a slide set has been selected, the ribbon is extended by further

buttons, which provide the functionality to add a new question, show the list of questions, start the presentation or add an explanation. The two buttons 'Add question' and 'List of questions' both open a taskpane. In my opinion the creation of lectures through a dialog window, but the creation of questions through a taskpane is an inconsistency. Also having two task panes next to each other reduces the space for the actual editing area of PowerPoint. On the other hand a taskpane does not cover any slides like a dialog would do as stated by the author. However, it is arguable if an overlay is really disturbing, because the slide must not necessarily be seen while the question is created and the dialogs position could be changed by moving it away, in case it blocks vital information.
Concluding from the just described work-flow, similarities to the websites work-flow can be found. Basically, the navigational structure is very similar. After a successful login a course can be selected, followed by a lecture selection or creation of a new lecture, which leads to the possibility to select a slide set and then create, view or insert questions. Except for the step of 'linking' a lecture to a presentation it equals the navigational hierarchy of the website. However, this 'linking' is done automatically by the application. On the other hand the UI approach is obviously different. In example, considering that most of the functionality is just simply a menu in the ribbon and the dialogs are new windows and not just modal overlays. However, the user can see the whole systems functionality in the menu at once, which is definitely an advantage for clarity. Given the very similar order of steps to add questions and start a presentation, the system helps users who are familiar with the website to ease learnability and increase seamless integration. A disadvantage for seamless integration is the automated 'link' between slide-set and selected lecture, because from the website a user would expect to be able to check on other lectures as well. Unfortunately, this is not possible. When the user tries to inspect another lecture, for example to see its questions, the opened slid set would be automatically linked to this newly selected lecture.

3.2 JavaScript API for Office

The newest technology for developing add-ins is the JavaScript API for Office. Add-ins using this API can be content or taskpane add-ins. A content add-in is directly inserted into a slide, while a taskpane add-in is run in a separate taskpane that is visible until it's closed. Microsoft says that the content add-ins are for enhancing a slide with dynamic HTML5 content and the taskpane add-ins are meant to be used for bringing in "reference information or insert data into the slide via a service" [21]. The distribution of the add-ins is done through the Office store.
A comparison to the older VSTO add-in type was already done in Antje Schubotz work and discussed briefly in the section above. Compared to the time Antje Schubotz wrote her thesis the functionality of the JavaSript API for Office was extended. While the newest updates underline that most of the effort is spent for extending the JavaScript API for Word and Excel, the PowerPoint API also received some updates. However, the development is not finished yet and the JavaScript API is still missing some crucial parts to give the developer real flexibility.
The used technology to integrate the add-ins into PowerPoint is an Internet Explorer (IE) 11 iFrame. This is done for means of backwards compatibility and is a major drawback. While Edge, the newly developed browser by Microsoft, supports most of the newest web standards for JavaScript released by Ecma International [10] the IE11 is stuck on Ecma 5th Edition and only provides very limited support for Ecma 6th Edition [13]. However, a lot of work-arounds can be used to accomplish similar functionality.

The major drawback about the only partly available user interface adaption mentioned by Antje Schubotz [27] on the other hand is still present and can not be fixed easily. In her work she integrated a lot of functionality into the ribbon, which is changed at runtime. The web add-ins are distributed through an XML-File that describes the add-in and lets the developer integrate some static functionality into the ribbon, like command groups and buttons. This can not be changed, once the add-in has been added.
One of the other technical disadvantages is that the interaction with the slide set is very limited and is selection based. The possible interaction is currently only available through the commands listed below. They can be grouped into two sets: 1) only available for PowerPoint add-ins 2) also available in add-ins for other Office products.

3.2.1 1) Office API only available for PowerPoint

- The presentation's active view can be read. It is either read or edit and it's possible to register a handler to the ActiveViewChanged event to recognized when the presentation mode is triggered.

- Slide navigation is another important API functionality. It offers the possibility to navigate to a particular slide in the presentation or go to the next or prior slide, which can be exploited for iteration over all slides.

- The presentations URL can also be read via an API function. It returns the local file path, including the name and type of the file.

2) Office API also available for other Office products

- It is possible to persist add-in states and settings. The data will be stored in the property bag of the PowerPoint file [23].

- Data can be read and written to the active selection in a document or spreadsheet. In PowerPoint this would be a text field on a slide. Just like for the active view an event handler can be registered to detect changes that alter the selection [24].

- The last available function offers to get the whole document from an Add-In. This is available for PowerPoint and Word. This functionality offers the possibility to "provide one-click sending or publishing of a Word 2013 or PowerPoint 2013 document to a remote location" [22].

All functions of the API are asynchronous for the means of better performance. As described above, the interaction is selection based. That means that the Office API only provides access to selected data. This leads to a not straight forward data access at some points. Also a few important parts are missing, for example concerning the active view. While the active view can be read and its change can be detected, there is no way to actively set the presentations active view from an add-in. Another problem is that the supported coercion types are very limited, only text and images can be added to slides. When using web technologies for the Add-In development, it would seem logical to insert HTML5 contents into the slide as well.
To use the API the Office JavaScript file must be linked and the Office-Object has to be initialized. To achieve good integration into Office products, Microsoft provides guidelines for the UI design and offers a fabric stylesheet file and a JavaScript file for the dynamic interaction with the styled objects.

Following the guidelines and using the fabrics, the same look and feel as in other Microsoft products can be provided, which increases the user experience.

3.3 Comparison Between the Different Approaches

In the last chapter existing ARS were discussed. Mostly the systems lack integration of the PowerPoint add-in into the rest of the system. Some provide different or less question types. Most of them didn't offer the possibility to have a constant overview of all questions. Furthermore, only one of them allowed more than one question per slide. Most systems needed to be used together with the website. Two systems didn't provide the possibility to group questions into lectures or a similar structure. Additionally, none of the systems gave the possibility to control the PowerPoint presentation from the website. With the AMCS add-in it is possible to change the active slide of the PowerPoint presentation by changing the active slide on the website.

The goal of this work is to integrate the main functionality into PowerPoint, so that the creation and presentation of a lecture can be done only in PowerPoint. However, on the other hand it should also be possible to apply changes using the website, i.e. when the lecturer switches to a device that doesn't have PowerPoint. Furthermore, the system should provide the possibility to add multiple questions to one slide and provide a list of all questions for a better overview.

The main advantage of the JavaScript API is that it is platform independent, making it possible to also support MAC OS and Office 365. The drawback is that it can not be used in PowerPoint versions older than PowerPoint 2013. However, PowerPoint 2013 is not the latest version; hence, the JavaScript API is supported by the two most recent Office versions. The other technical advantages are the high security, because it is running in a sandbox, and that no installation is required. Simply an XML file is used to distribute the add-in through the Office store. On the other hand the functionality of VSTO add-ins is superior in means of interaction with PowerPoint, in example changing the ribbon at runtime or inserting new slides, which increases the flexibility of the developer.

As Antje Schubotz mentioned in her work, she uses windows forms to get the same look and feel with other windows applications. However, this is not really an advantage above the JavaScript API, because Microsoft provides a fabric that helps to integrate the add-in seamlessly. What might be even more important, with a web-based approach the add-in can have the same or at least a very similar look and feel as the website, helping users who worked with the website to get around easier in the add-in. Nonetheless, in one point the VSTO add-in is more powerful: it can add new slides to the presentation and easily alter their content. With the JavaScript API no new slides can be inserted and only the user selection on a slide can be altered. And through this method only text and images can be inserted. Also the amount of slides can not be extracted. The only information the API provides are for the currently selected slides. To access every slide an iteration over all slides is necessary, which might cause the user to feel like their presentation is controlled by someone else.

One of the main goals of this work is to improve the lecturers work-flow and to provide the add-in to a broader audience. However, another objective is to ease the work of the developers integrating it better into the existing system. With this two goals in mind the newly developed prototype will use the JavaScript API for Office. Not only platform independence, but also the similarity to the website can be provided in this way. It is also likely that the JavaScript API will get more powerful in the future.

Another objective of this work is to research if a different approach with a more PowerPoint centered setting can enhance the work-flow by offering most of the systems functionality directly in PowerPoint.

3.4 Seamless Integration

Seamless integration is a term that became relevant when smart phones and tablets were introduced. The same system must run on many devices that differ a lot, especially in size and interaction method. While displays for PCs get bigger, Laptops exist in many different sizes and tablets and smart phones have small displays and use touch input. This brings a new challenge to system design. One system running on many devices and on different platforms. However, it should still look and feel like the same system. First the difference of User Interface (UI) and User Experience (UX) must be discussed. While UI is describing visual components and input methods, UX describes the personal experience of a user with the system and its services. Seamless integration is one important part of a good UX. One step forward to seamless system integration can be seen by achieving the same look and feel, easing the effort of a user to use the system on different devices or platforms. In this case, look would be the UI so the user can easily find visual elements and feel would be the experience. A seamless experience means that the user can expect behavior that is already known. So the system does not simply look the same, but it feels the same, because the interaction is the way the user expects it to be.

Kim [15] defines seamlessness as follows: "Seamlessness is a quality of any crosschannel customer journey where the transitions (or handoffs) from one channel to the next involve zero or minimal overhead for the users. Basically, if you can pick up where you left off, the user experience will be seamless. But if users have to reestablish their contexts and/or redo work when switching to a new channel, then the experience will feel bumpy."

Following this definition, seamlessness is about reducing the costs to switch from one channel to another. The definition is mainly focusing on the functional and data integration. However, the author also mentions that the users should be able to reestablish their contexts. To achieve this, the systems must feel similar. If the user interface, structure or behavior of the different channels is very different the context will be lost.
In his article Kim states further that the rise of mobility forces the users to do work in more than one session and use different platforms. As an example Kim names a system for selling movie tickets of a cinema. There is a website, a mobile version of the website and an app to access the system from mobile devices. The closest movie theater will be suggested by using the users location and when the user arrives at the cinema he can get the tickets by using a machine or the counter. Following Kim's statement about the limited time for one session, a user might check the times and free seats for a movie with his Laptop at home. Then the user leaves the house and buys a ticket using the app and later goes to the cinema to pick them up using a code he got with the purchase in the app. Kim claims that a user will not use a system if it is difficult to use. Hence, if the app was too difficult to use, the user might not choose to buy tickets in the app and later find out at the cinema counter that the good seats are gone and decides to do something else. Also when picking up the tickets at the machine is too difficult, the users will all use the counter, resulting in long waiting lines.

To achieve a good system in terms of seamlessness the author suggests to examine the customer journey. The examination should research when the user uses different channels or when the system even leads the user to another channel. Roadblocks are also a very important matter that should be considered. Meaning a channel breaks the experience because it is not working consistently with the rest of the system. The author states that, when adding new technologies to a system they are more strapped to the system than really integrated. Hence, Kim claims that roadblocks not only arise from front-end design, but also from the poor integration into the back-end. As an example Kim names the Walmart system, which "[...] was not able to process returns on online orders" [15], because of the poor back-end integration.

Bianca Ignacio describes seamless UX as follows: "A seamless UX design doesn't make the user think twice about what they should do next. Instead, it should adapt to users' needs and behaviors, and feel natural to them." [8] Like Kim she underlines that the user journey is a crucial part. It should be hassle-free and not frustrating for the user. Furthermore, she emphasize the importance of consistent behavior.

Consequently, Ignacio gives 10 tips how to achieve this. In general the tips can be grouped as follows:

The UI should be simple, consistent, only use simple animations and significant colors. A clear design that uses colors to emphasize important information and is consistent over all the platforms helps the user to find the right information and functionality easier.

Achieve ease of use through short journeys, prioritization of the landing page, a search feature and clear feedback. Short journeys are about the steps a user needs to finish certain progresses from start to end in the system. The author especially suggests to keep forms and surveys short. Also the system should provide very clear and significant feedback so that the system seems transparent without flooding the user with information.

Be adaptive through personalization and an age responsive design. The best system is a system that fits the needs of all users. However, different users have different expectations and may use the system differently. Hence, a good system should adjust to each user.

Seamless integration is a part of a good UX. It can help to make the user feel like the same system is used, even when accessing the system through different channels. The user should feel familiar with the system even through it is used on a different platform, because the user can expect the behavior. The user journey also plays a crucial role. It describes the way from start to end of certain processes the user does when using the system. The journey should also research when a channel is changed and how this change works. Additionally, the journeys should be short and as least frustrating as possible by giving clear feedback for user actions. Using good design can also help to achieve this. The UI should be consistent and clear. Another important point for good UX is adapting to each user. However, both the same look and user adaptation are not an essential part of seamless integration.

The system should integrate seamlessly into PowerPoint by using the Microsoft fabric and UX suggestions. On one hand the integration into PowerPoint is important but on the other hand the integration into the existing system is important as well. A user who knows the website should be able to expect the behavior of the add-in. To achieve a good UX a trade-off between integration into PowerPoint and into AMCS is necessary.

3.5 State of the Art Web Technologies

In recent years the world of Javascript has been flooded with numerous libraries and frameworks. However, until now none of those libraries could establish itself as a de facto standard. They differ quite much in purpose and implementation design. While some are simple libraries to solve one certain problem others are frameworks that almost make Javascript look like a new script language. In this section a few outstanding and interesting solutions will be discussed.

One of the first and maybe most well-known and in the early days also most used libraries is jQuery [12]. jQuery is a library to ease interaction with the Document Object Model (DOM) but is not limited to this. It also provides other features like ajax calls and also provides an abstraction for cross-browser support.

Angular [9] on the other hand is a full grown modularized framework that consists of several libraries, which are not all mandatory to use. Essentially, it consists of HTML templates, component classes, services and modules. The templates are enriched with angularized markup and the component classes handle the templates. Application logic is added through the services, which are grouped into modules.

React [2] is a library to build user interfaces. It uses components that encapsulate their own states. JSX is a syntax extension for JavaScript that tries to makes writing markup easier. It is not necessary to use JSX, but it has become a de facto standard for writing templates in React.

Bootstrap [18] is a UI component library. It doesn't provide an architecture or template system like Angular and React. It is a mere toolkit for developing with HTML, CSS, and JS. The technique of Bootstrap is comparable to jQuery. It also provides a responsive grid system.

Microsoft's Office UI Fabric [20] is similar to Bootstrap. It provides UI components with their functionality in the Microsoft corporate design. It's possible to use the react or pure JS fabric.

The project should be light weight and easy to understand. Hence, big frameworks as Angular are not used. React also introduces unnecessary overhead by using JSX and forcing a specific architecture on the user. However, it would be better suitable as Angular because it's only for the UI part and the rest of the application stack can be chosen freely. The libraries that will be used in this project are jQuery and Microsoft's fabric. jQuery makes DOM manipulation much easier and also provides simplified ajax functionality, which reduces implementation effort. For giving the user the feeling to be still using a part of PowerPoint Microsoft's fabric will be used to have the same colors, icons, fonts and behavior of UI components. For UI components that are very similar, or even equal to the website (e.g. certain forms) should be displayed using Bootstrap, because Bootstrap is used on the website and it is desirable to achieve seamlessness across the different channels. As discussed above, the main parts of seamless integration are a similar user journey providing a consistent navigational structure and same behavior across different channels (section 3.4). The UI however is not necessarily an essential part of seamless integration. It will help the users to utilize their existing knowledge about the system nonetheless, i.e. by using similar icons and same looking buttons in the same button group structures.

3.6 Discussion of Different Design Patterns

The use of separate presentation patterns gives huge advantages concerning maintainability, testability and reusability. When the code is interacting directly with the UI than it is coupled very closely to it. This violates the single responsibility principle, because changes can be caused by different reasons. For

example, when changing the UI it should not affect the logic or data representation and vise versa. The testability is greatly improved, because the logic and data can be unit tested in isolation. It is rather hard to test user interaction. This is mostly "useable as integration tests, because they need the entire application to be set up." [30] Maintainability is improved because the code is decoupled and as long as the API doesn't change, the changes won't affect the other modules. Reusability also benefits from that fact, because the modules are self-contained and hence can be easily used independently in another context.

Model View Controller (MVC) is one of the most famous design patterns and became like a standard for developing web applications. The pattern has been invented by Trygve Reenskaug in 1979. Since then the used technology improved greatly and today a vast set of functionality to build web applications exists. The MVC pattern divides the system into 3 parts, the Model, View and Controller. The Model contains and persists the data. The View holds the UI and the Controller handles the user interaction with the system and updates the model. Depending if an active or a passive View is chosen either the Model or the Controller updates the View. In case of an active View a Model update also triggers a View update and in case of a passive View a Controller update leads to a View update.

Model View Presenter (MVP) is a generalization of the MVC pattern and works on a more abstract level. The main difference is that the Presenter talks to the View through an interface. The View does not handle any user input it simply passes the user action onto the Presenter. Thus increasing the testability.

Presentation Model is another way to achieve the separation of Model and View. It can be seen as an abstraction of the View. The behavior and state are pulled out of the View into the Presentation Model. A Presentation Model is a self-contained module that holds all the data of the View that might change. The View effectively only renders the data and passes the user interaction to the Presentation Model. A problem with this pattern is that View and Presentation Model need to be synchronized. This can be done in the View, but making it less testable, or in the Presentation Model, which couples the Presentation Model closer to the View decreasing reusability and maintainability. However, Fowler [6] states that synchronization errors are rather easy to fix and he also suggests to add a mapper. This will increase the code base but simplifies error tracking and therefore increases maintainability and testability.

Model View ViewModel (MVVM) is another derivation of the MVC pattern and is a variation of Fowlers [6] Presentation Model that was developed by Microsoft. The only difference is the usage of data bindings between the View and the ViewModel. This way the ViewModel only exposes very easily bindable properties and methods, that's why it is also called Model-View-Binder. The objective of using data bindings was to remove the 'code-behind' the View. When not using .NET this data binding has to be implemented by the developer. Anyhow, many frameworks exist using the Model-View-Binder pattern that offer a data binding implementation (e.g. KnockoutJS for JavaScript [16]).

3.7 Quality Assurance

In this section different approaches for quality assurance and acceptance criteria satisfaction are discussed. Furthermore, different types of tests are explained. Finally, different test approaches will be compared.
Quality assurance is a crucial part of software development, because mistakes are inevitable. Some may be caused by developers others arise from misconceptions. In the past many possible solutions have been researched to solve the communication problem between customer and developer. Over time the customer has been involved into the developing process more and more. Tighter feedback loops and measures for acceptance criteria have been introduced to assure that the end-product works as requested by the customer.
Therefore, quality assurance is not only important for the quality of the code but also a way to prove for both sides that the final software product is according to the requirements created in the analysis phase.

3.7.1 Test Centered Development Approaches

Agile methods were introduced to simplify the software development process. Some of the goals of agile methods are reduced upfront design and increased customer collaboration. Extreme Programming (XP) is one widely used agile method that uses very short iteration cycles. Pair programming, refactoring and Test Driven Development (TDD) are essential parts of the XP methodology. TDD is also called test-first development and may seem like just a testing technique but is in fact much more than that. David Janzen and Hossein Saiedian say: "The driven in test-driven development focuses on how TDD leads analysis, design, and programming decisions." [11] The authors further state that TDD subsumes many analysis decisions when using XP. However, it is not a methodology but rather a practice, which should be used together with other practices (like in XP).
Williams et al. [31] describe the TDD practice as an iterative process that starts from the functional requirements and rapidly cycles through the following five steps to produce code:

- Creating automated unit test cases, which describe the behavior of the code that should be implemented. The behavior is derived from the functional requirements.
- Running the unit tests - they must fail, since there is no code yet.
- Implementing the code.
- Re-run the tests. Now all tests should pass.
- All tests should be re-run periodically. This ensures that newly added features don't break existing code. This testing technique is called regression testing.

Bhat and Nagappan [1] describe the iteration cycle very similarly except they add a very important step before the last step (regression testing step). This introduced step is the refactoring step. Refactoring is necessary if the code structure or requirements change. This step doesn't only affect the actual code base of the implementation, but it's also necessary to refactor the tests, because the tests are closely coupled to the codes behavior and have to be changed accordingly.

A special type of TDD is Acceptance Test Driven Development (ATDD). It uses acceptance tests to represent the stakeholders requirements. Carlos Solis and Xiaofeng Wang describe the benefits of ATD as follows: "ATDD helps developers to transform requirements into test cases and allows verifying the functionality of a system." [29]
Behavior Driven Development (BDD) can be seen as an evolution of TDD and ATDD [29]. It encourages collaboration between all different parties of a project [17]. Therefore, a ubiquitous language is needed. BDD provides a small domain independent language that is used to structure stories and scenarios, which is mainly used in the analysis phase. The user stories are user centric and give a context for the features the system should provide. The template, which is used for user stories typically looks like shown below (according to [25] and [29]):

StoryTitle (One line describing the story)

As a [Role]

I want a [Feature]

So that I can get [Benefit]

In the following a template for scenarios is shown (according to [25] and [29]):

Scenario *Number*: Scenario Title

Given [Context]

AND [Some more contexts]

WHEN [Event]

THEN [Outcome]

AND [Some more outcomes]

A scenario describes how the system that implements a feature should behave when it is in a specific state and an event happens. The outcome of the scenario is an action that changes the state of the system or produces a system output.

3.7.2 Tests

Tests are essential to the just described development practices. They are used to evaluate the functionality of a software system. Tests differ mainly in purpose and how frequently they are run. Furthermore, it is important to get a measure that tells how well tested a system is. In the following different kinds of tests will be described concluding with a comparison and coverage aspects.

Unit Test

Unit tests are tests that verify the correct functionality of a unit. A unit is the smallest self-contained element in a program. Although sometimes it is discussed if a unit should be on the level of classes, most commonly and also in this work units are functions. Unit tests simply show that the unit doesn't

fail under certain conditions. Unit tests are white box test, meaning that the code to be tested is known to the tester. Code coverage is used to determine how well a unit is tested. Hence, tests need to be picked carefully, in example testing boundary values and taking all possible branches.

Integration Test

Integration tests are evaluating the whole system and not only parts in a separate manner like unit tests. Therefore, integration tests are done in later phases of the development cycle. In contrary to unit tests, which show the correct functionality of one part of the program, integration tests focus on the interaction of the modules (units) of the system.

Regression Test

Regression tests are tests that are run repeatedly. The goal is to show that the system still works the same way after changes were applied. After a system has been changed all regression tests should be run. This can be very expensive in means of time consumption. Therefore, often only parts that are affected by the applied changes are tested. However, it is not always clear which parts are affected and integration tests should be done frequently as well.

Acceptance Test

The acceptance tests are the tests that validate if a system works according to the specified requirements. Following the BDD practice all acceptance criteria should be manually or automatically executable tests. Furthermore, in BDD the acceptance test cases are directly derived from the scenarios, which are elaborated from the user stories.

3.7.3 Conclusion

As E. W. Dijkstra said: "Program testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence." Testing is not a verification of the correctness of the code. However, if done thoroughly bugs can be found and therefore the code quality can be increased. Code Coverage is used to measure how good the tests actually test the code. One common way is to use statement coverage. However, only because every statement was executed once, doesn't mean that every branch was taken and also it says nothing about special boundary cases. So for unit testing knowledge about the implementation and requirements should be used to increase code quality. To help developers write better tests BDD introduces the practice to test behavior of code rather than a certain functionality. In BDD tests are written as sentences making it clear what and how much at once is tested. Furthermore, this makes the tests better understandable for the stakeholders.

3.8 Requirements Analysis

In this section the functional and non-functional requirements of the later proposed system will be specified. Additionally the acceptance criteria will be defined.

3.8.1 Functional requirements

In the following the high-level functionality of the system will be described. The overall functions of the new lecturer add-in should at least equal the functionality provided by the old lecturer add-in. In the following list of high priority items the first 5 objectives are almost equal to the high priorities of Antje Schubotz work. They have only been adjusted to the newly used technologies and A2 can be omitted because the new add-in will be located on a web-server and, thus, it's not possible to load the add-in without an Internet connection. Furthermore, some middle priority items (B2, B3, B4) are high priority items now.

A) High Priority

1. Must at least work with the newest PowerPoint version on Windows and Mac.
2. Sign-in and Sign-out to AMCS must be possible.
3. Must be able to link a presentation with a lecture.
4. Control commands (next slide, previous slide, particular slide) must be sent to server in presentation mode.
5. The Add-In must listen to active slide changes on the server, after a presentation has been linked to a lecture.
6. Must display existing courses, lectures and questions.
7. It must be possible to create and edit questions for a linked lecture.
8. After linking a presentation to a lecture, changes in the presentation must be persisted on the server.

B) Medium Priority

1. Create, edit and unlink lectures must be possible.
2. Provide a light weight version, which is limited to linking lectures and sending control commands to the server in presentation mode.
3. It must be possible to set the lecture state.

C) Low Priority

1. Should provide a language option for German and English.
2. System should give feedback (e.g. popups, notifications) to confirm user actions.

3.8.2 Nonfunctional requirements

In contrary to the functional requirements, which describe actual system behavior, the nonfunctional requirements describe system architectural requirements.

Usability and Learnability

A menu should be provided that gives the user the possibility to take shortcuts. Additionally, a back button should provide an easy way to navigate to the previous page. Pages should show the necessary information only. The information should be self-describing and use self-explaining icons, which should be chosen from Microsoft's Office fabric and with regard of the icons used on the website. Thus, supporting seamless integration, because the users can use their knowledge and predict the meaning of data without needing extra tooltips. The colors suggested by Microsoft should be used and every page should use a different color to support the user navigation. The user journey should also be similar to the website, concerning the page order, question and lecture creation and edit. This helps to increase learnability for users who are familiar with the website and also improves seamlessness. System feedback (e.g. popups, notifications) should be short and clear.

Maintainability

Achieve Maintainability by the usage of design patterns and modularization that provide a clear and well testable architecture. Create single points of truth through encapsulation and provide clear interfaces.

Extensibility

The system architecture should provide an easy way to add new question types and incorporate new views.

Testability

An architectural design should be chosen that decouples view from logic, which improves testability of the domain model. For the not easily testable parts (e.g. system components that need user input) test scenarios should be provided.

3.8.3 User Stories

In the section explaining BDD the importance of a ubiquitous language was underlined. According to Dan North [25] every user story should contain a story title, the role of the user, the feature the user wants and the benefit for the user. In our case there is only one user group - the lecturer. Hence, the role will always be the lecturer and therefore this field will be omitted to shorten the user stories. The used template will be:

> *Number*. Story Title
>
> **I want** Feature
> **So that I can** Benefit

The user stories derived from the functional requirements are listed in the following. The user stories are important to create scenarios, which are used as acceptance tests.

1. System running on different platforms

   I want to work with the newest PowerPoint version on Windows or Mac.

   So that I can choose the working environment.

2. Sign-in and Sign-out

   I want to sign-in and sign-out.

   So that I can get access to the system after sign-in

   AND protect my data when signed-out.

3. Linking a presentation with a lecture

   I want to link a presentation to a lecture.

   So that I can persist my changes applied to the presentation on the server

   AND automatically control the lecture through PowerPoint.

4. Send control commands to the server

   I want that my active slide and slide changes are sent to the server.

   So that I can fully concentrate on the lecture, because my commands will be sent automatically.

5. Remote Control of the presentation

   I want to control my presentation from another platform (website, app or smart watch).

   So that I can choose my preferred way to present.

6. Displaying courses, lectures and questions.

   I want to see lists of my courses, lectures and questions.

   So that I can choose a lecture to link to a presentation

   AND have an overview of my questions.

7. Creating and editing Questions

   I want to create and edit questions for my presentation.

   So that I can improve my work-flow, because I can create and edit questions directly in PowerPoint.

8. Persisting changes on the server

   I want that my changes are permanent.

   So that I can sign-out and use my changed data later.

9. Creating and editing lectures

   I want to create

   AND edit

   AND unlink my lectures.

   So that I can create a new presentation and insert a new lecture to link it to the system

   AND apply changes later.

10. Light weight version

    I want to use only slide command automation.

    So that I can use/learn the system easier.

11. Set the lecture state

    I want to set the lecture state.

    So that I can start sending questions to students.

12. Language option

    I want to select the system language.

    So that I can use the system even if I don't speak German.

13. Feedback

    I want to get feedback.

    So that I can know that my actions were successful.

3.8.4 Scenarios

From the user stories scenarios are elaborated. Every user story can have one or more scenarios. The scenarios are used as acceptance tests, which are used to prove that the requirements were satisfied. In this section the scenarios will be described followed by the acceptance criteria section that explains how the scenarios will be evaluated.

1. System running on Windows (story 1)

   GIVEN PowerPoint on Windows
   WHEN inserting the Add-In
   THEN it opens and works.

2. System running on Mac (story 1)

   GIVEN PowerPoint on Mac
   WHEN inserting the Add-In
   THEN it opens and works.

3. Sign-In (story 2)

   GIVEN PowerPoint Add-In loaded and open
   WHEN inserting correct credentials and clicking on login
   THEN Add-In checks credentials
   AND shows course overview.

4. Sign-In with pin (story 2)

   GIVEN PowerPoint Add-In loaded and open
   WHEN inserting correct credentials and clicking on login
   THEN Add-In checks credentials
   AND shows lecture list of course with the provided pin.

5. Linking a presentation with a lecture (story 3)

   GIVEN an unlinked presentation
   WHEN user clicks on link lecture in lecture overview
   THEN lecture and presentation are linked by saving the lectures Id in the presentations property bag
   AND subscribe to the websocket to listen to server changes
   AND highlight the linked lecture
   AND synchronize slides in presentation with server slides.

6. Control lecture from PowerPoint (story 4)

   GIVEN a linked presentation
   WHEN user enters presentation mode
   THEN send active slide to server
   AND check for slide changes
   AND send any slide changes to the server.

7. Control PowerPoint presentation from server (story 5)

   GIVEN a linked presentation
   WHEN user changes lecture state or active slide
   THEN change state in Add-In accordingly
   AND go to active slide.

8. Course, lecture and question lists (story 6)

   GIVEN the user has successfully logged in
   WHEN user clicks on course, lecture or question overview
   THEN display the requested data items in a list.

9. Question creation (story 7)

   GIVEN a linked presentation
   WHEN user clicks on new question
   THEN open question creation dialog
   AND create question after submit.

10. Question editing (story 7)

    GIVEN a linked presentation and a question
    WHEN user clicks on edit question
    THEN open question editing dialog
    AND edit question after submit.

11. Persisting changes on the server (story 8)

    GIVEN a linked presentation

WHEN changes are applied to presentation
AND synchronized
THEN changes are persisted on the server.

12. Creating and editing Lectures (story 9)

GIVEN user is logged in
WHEN user clicks on create or edit lecture in lecture overview
THEN open lecture dialog
AND create/edit lecture after submit.

13. Unlink lecture and presentation (story 9)

GIVEN presentation is linked to a lecture
WHEN user clicks on unlink lecture in lecture overview
THEN remove lecture Id from presentation property bag
AND close websocket
AND remove highlighting.

14. Light weight version (story 10)

GIVEN user is successfully logged in
WHEN user uses light weight version
THEN only show course and lecture list
AND only show linked lecture in lecture list.

15. Set lecture state (story 11)

GIVEN presentation is linked to a lecture
WHEN user sets lecture state
THEN lecture state is set
AND sent to the server.

16. Language option (story 12)

GIVEN language is German
WHEN user changes to English
THEN texts will be displayed in English.

17. Synchronization feedback (story 13)

GIVEN linked presentation
WHEN user clicks on synchronize
THEN display 'synchronizing' notification.

18. Lecture creation/edit and question creation/edit feedback (story 13)

GIVEN lecture or question dialog is open
WHEN user submits form
THEN positive or negative feedback about persisting data on server is displayed.

3.8.5 Acceptance Criteria

The above shown scenarios were derived from the user stories, which are according to the functional requirements. Following BDD, every scenario should be an executable acceptance criteria. However, not all parts of the system can be easily tested automatically. User interaction and interaction with the environment make automated testing complicated. In example, the system needs to interact with the PowerPoint API. Additionally, UI components have to be tested manually and also some scenarios require user input and must therefore be run manually by using the above given scenarios. First the system state described under [GIVEN] must be set up, followed by the execution of the action explained under [WHEN] and finally the new state given under [THEN] has to be evaluated.
The System must at least provide the functionality defined through the high priority items of the functional requirements. Furthermore, the correct functionality of unit testable components (no need for outside interaction) should be shown by unit tests that provide overall code coverage above 90%. By using a design pattern that separates view from logic, the unit testable components are increased. Furthermore, an integration test should be provided that covers at least the high level requirements.
In this chapter the system requirements were analyzed, which lead to the formulation of acceptance criteria. The proposed system will be described in the next chapter and is according to the defined requirements and will be evaluated against the here specified acceptance criteria.

4 Concept

In the last two chapters the background of this work and the analysis of the existing systems and requirements for this work were given. Concluding by giving acceptance criteria and test scenarios to evaluate the outcome of this work. In this chapter the concept of the proposed system will be discussed. Starting with an overview, which is followed by a description of the system and the software architecture. After the discussion of the work-flow, aspects for quality assurance can be found, including automated unit tests and test scenarios for UI and system components that require user interaction.

4.1 Overview

The chosen technology for the add-in is the new Javascript API for Office. This offers two ways to implement the add-in: first, as content add-in or second, as taskpane add-in. A content add-in is inserted on one particular slide and can only be seen and accessed on that slide. The system however should provide a constant overview of all questions and lecture details. Hence, the system will be implemented as taskpane add-in. The application will run on the client side and only communicate with the AMCS back-end via HTTP requests and responds (see Figure 4.1. The add-in can be accessed in PowerPoint through a menu in the ribbon. In the ribbon menu, some essential functions will be provided. The buttons that are included in the ribbon are: a button to open the taskpane, a button to open the website, buttons to set the lecture state and a synchronization button.

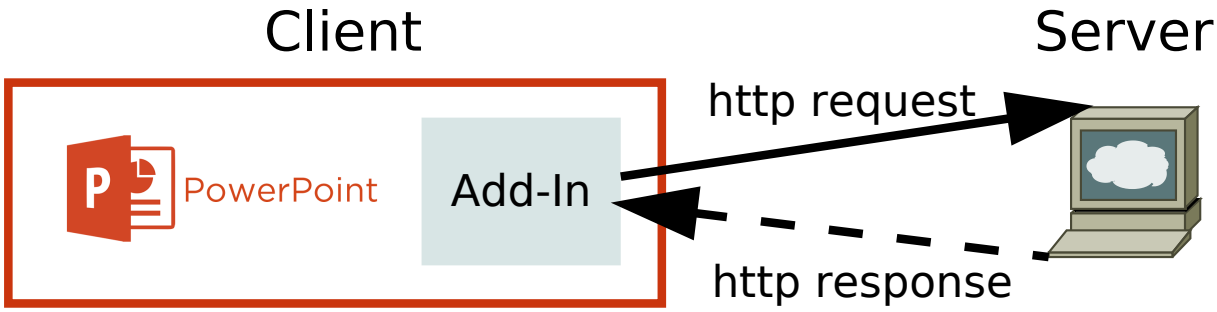


Figure 4.1: Coarse grain system overview.

The taskpane is initially docked at the right side of PowerPoint but can be undocked by the user to appear in a separate window that can be moved freely. The add-ins UI will mostly use the UI components suggested by Microsoft for building add-ins. As discussed in section 3.4 the same look supports seamless integration and the add-in should fit into PowerPoint to make it seem as natively integrated as possible. Additionally, users who switch from the website to the PowerPoint add-in should also feel as they still use the same system. To achieve this, the basic navigational structure is designed similarly to the AMCS website. In the website the dialogs for creating and editing lectures and questions are displayed in a

modal overlay. In the add-in the dialogs are opened in a separate dialog window, because the space in the taskpane is very limited. However, the dialogs in PowerPoint are similar to the modal overlays, because they also block interaction with the rest of the system while they are open. To support seamlessness the dialogs will be designed to look similar to the dialogs on the website. While the UI structure and interaction possibilities (inputs, buttons, etc.) will be displayed exactly as on the website, the colors, fonts and borders will be changed to fit into PowerPoint. This way the user finds the same structure and can expect the same behavior with which seamless integration with the website is achieved. On the other hand the user still feels like he is using a part of PowerPoint, which is achieved through the changes of the look.

4.2 Architecture

As software architecture the Presentation Model pattern has been chosen. As discussed in section 3.6 it provides a clean separation of UI and domain model. Hence, increasing the testability greatly, which also leads to a better maintainability and extensibility. The Model View ViewModel has not been chosen, because the objective is to keep the implementation simple and slim. Implementing data bindings is not trivial and is best done by existing frameworks or libraries. However, the prototype should be implemented by using as little frameworks as possible as discussed in section 3.5.

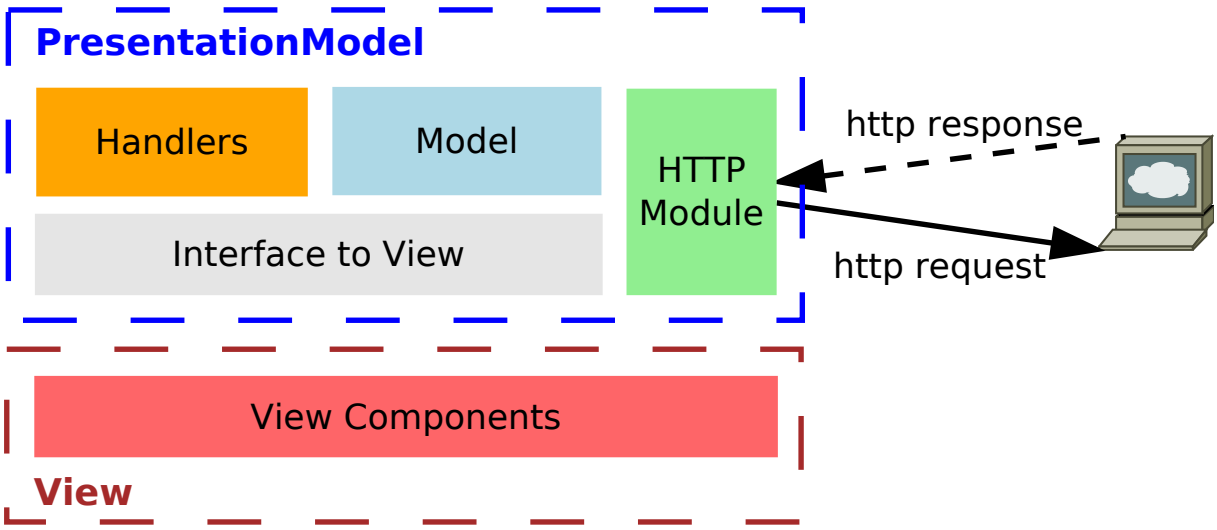


Figure 4.2: Overview of the system architecture.

The Presentation Model pattern divides the whole system into the View, which contains the graphical representation, and the Presentation Model that holds the domain logic. To further structure those components the Presentation Model is split into handler and model classes (see Figure 4.2). The handlers are similar to supervising controllers from MVC and handle user interaction. They also update the model if necessary. Following the pattern, the handlers are independent of the view and only provide an interface, which provides the View with requested data. To simplify the interaction with the View and the interface of the Presentation Model a mapper is used as suggested by Fowler [6]. In this way all communication goes only through one point. This simplifies testing and bug tracking. The same technique is used to communicate with the server. The communication to the server is wrapped by a separate module.

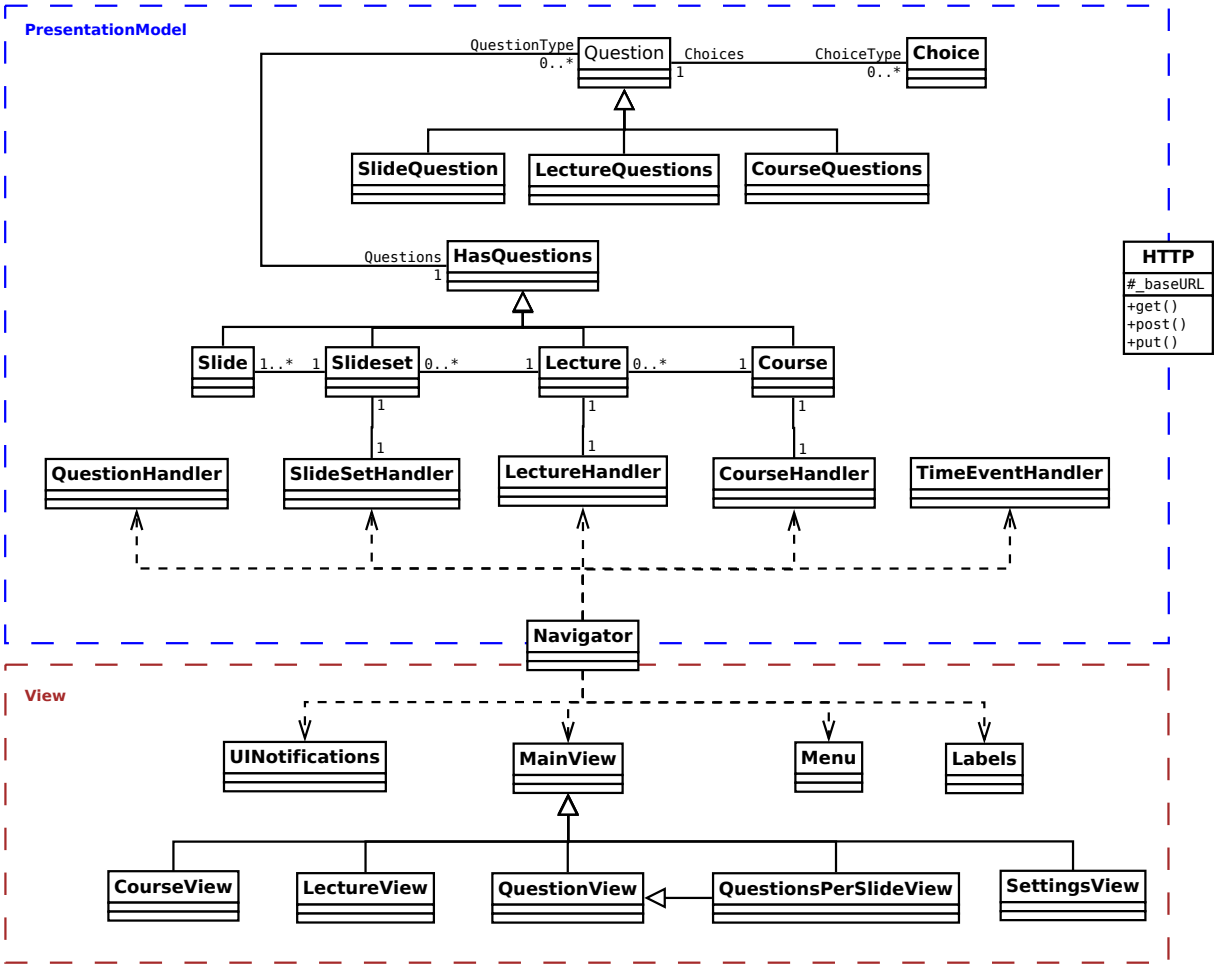


Figure 4.3: UML class diagram.

As shown in Figure 4.3 the Presentation Model consists of the handlers and the domain model. The domain model is a local representation of the server data and is kept consistent through the persistence of changes on the server. The classes of the domain model are the Question, SlideQuestion, LectureQuestion, CourseQuestion, Choice, Slide, Slideset, Lecture, Course and HasQuestions classes. The Slide, SlideSet, Lecture and Course classes all inherit from HasQuestions. HasQuestion is an abstract class that has a question list. Each question can be a SlideQuestion, LectureQuestion or CourseQuestion, depending on their context. Every Question child class inherits from the Question class, which implements common functionality. Every question object can have 0 or more choices. The Choice and Question class are mere wrappers for the back-end data objects, which are retrieved via the HTTP class from the server. Because of that, and means of simplicity splitting the questions into a separate class per actual question type (SingleChoice, MultipleChoice, etc.) has been disregarded. The Navigator class manages all communication between the view classes and the handlers. Per page a separate class is used. All common functionality is inherited from a parent class called the MainView.

4.3 Work-Flow

In this section the work-flow will be described. The work-flow equals the user journey discussed in section 3.4 as part of seamless integration. They both describe the way for a user to finish a process in a

system from start to end. However, subsequent only work-flow will be used as terminology.

The work-flow of the old lecturer add-in is mainly done through the ribbon menu as described in section 3.1.2. It only uses a taskpane to display the list of questions and another taskpane to create new questions. With the new JavaScript API for Office the ribbon can not be changed dynamically. Hence, only static features will be included in the ribbon and most of the functionality will be accessed in one taskpane.

The general work-flow for the taskpane application of the new lecturer add-in is shown in Figure 4.4. After the taskpane load and a successful user login the course or lecture overview will be displayed. If the PIN of a course was provided then the user will be forwarded to the lecture list of that course. From the lecture overview a user can reach the question overview. Thus, making the essential work-flow very similar to the website. The main differences are that the course overview doesn't show active lectures, the lecture page is reduced to the question overview and a questions per slide view was added. First, the active lectures overview is not necessary, because this add-in is only for the lecturer and hence, it's not physically possible to hold multiple lectures at the same time. Second, the lecture page on the website consists of the three sub-pages dashboard, messages and questions. The dashboard shows the current state, active slide and question results. The state is integrated into the question view, the active slide is implicitly given through the selected PowerPoint slide and the result integration is not part of this works objective. The question view in the add-in is derived from the question sub-page. Finally, the questions per slide view is a new page, which allows the user to only see the questions created for a particular slide. The work-flow will be described in more detail in the following sections by examining significant use cases and navigational paths.

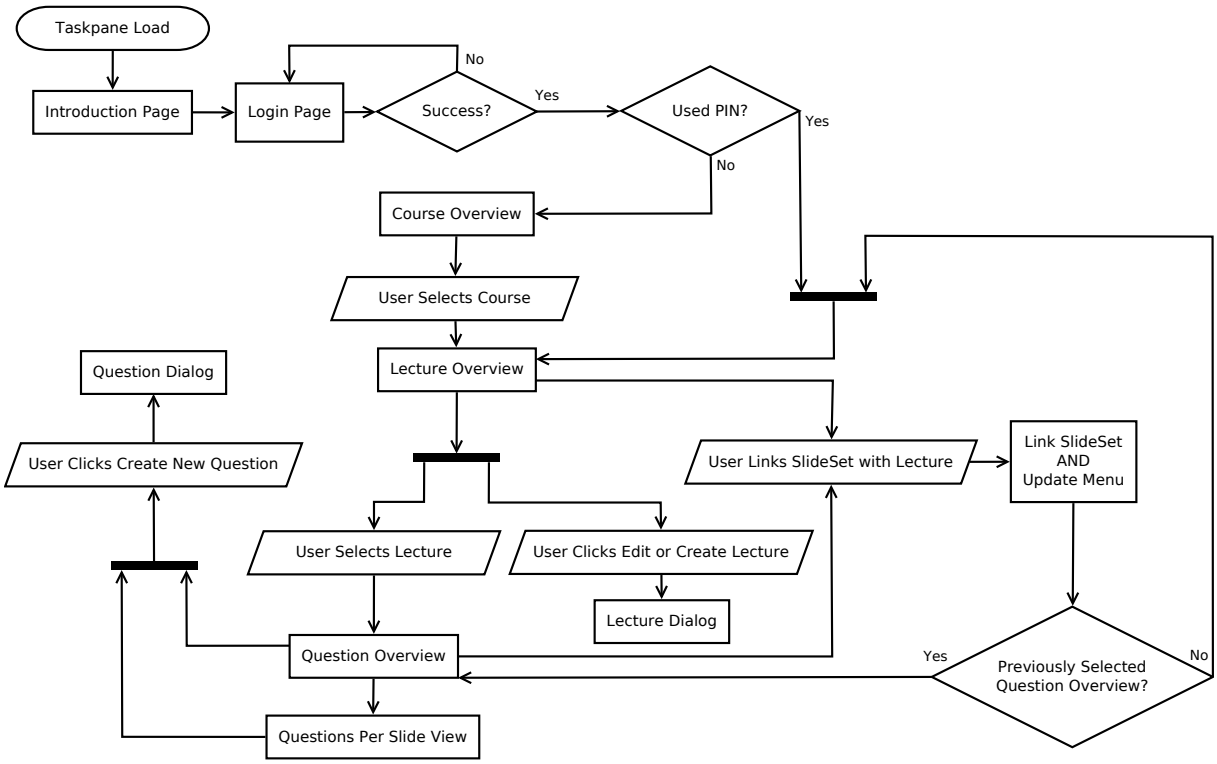


Figure 4.4: Navigational work-flow diagram of the add-in.

4.3.1 Use Cases

In this section the three most significant use cases of the system will be shown in detail. Starting with the process of linking the slide set to a lecture. Followed by the use case showing the creation of questions and the last use case that is considered describes the presentation of a lecture. Synchronization is also a very crucial scenario and will therefor be discussed separately.

Linking a Lecture

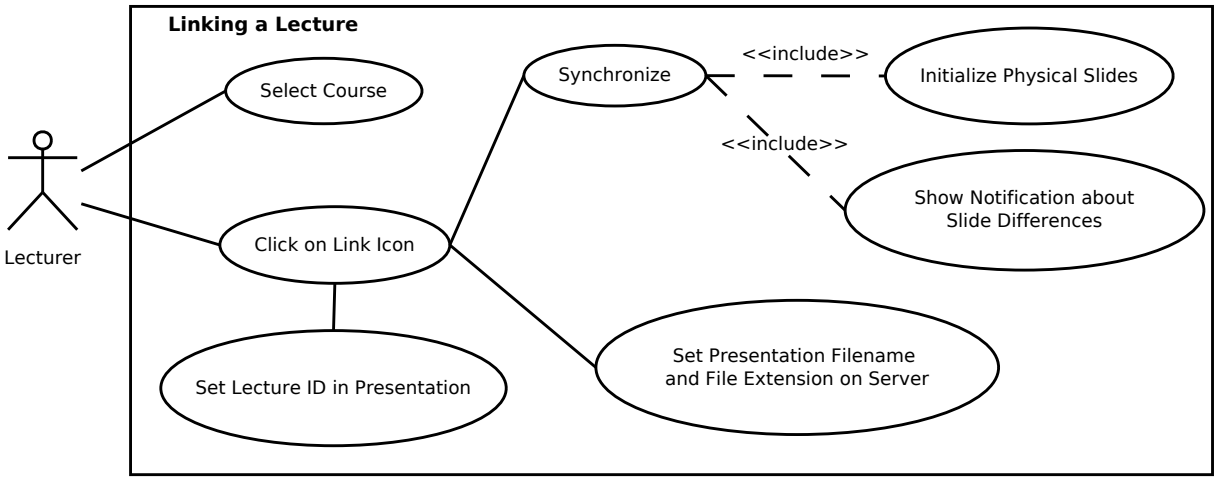


Figure 4.5: Use case diagram for the process of linking a lecture to a presentation.

Linking a lecture is essential to unlock a lot of functionality, like question creation, synchronizing the slide set with the server and automated slide changes in presentation mode. The linking process is shown in Figure 4.5. It is done by navigating to the lecture overview and clicking the 'link' button. After linking a lecture with a slide set an initial synchronization will be done. This first synchronization initializes the slides by linking logical slides with physical slides. Roles are used for that. The slide model object stays the same, it just changes its role from logical to physical slide. In the case, that more logical slides than physical slides exist, only as much logical slides take the role of a physical slide as slides in the PowerPoint slide set. The logical slides left over remain unlinked. Should the user add a new physical slide then it will be linked to an unlinked logical slide, if one exists. If there is no unlinked logical slide then a new slide object will be created and immediately linked to the physical slide. Hence, the new logical slide takes the role of a physical slide immediately after being created. Finally, the user will be informed about the successful linking process and the difference of the amount of physical slides in the slide set and logical slides on the server.

Creating Questions

Pagination is used on the website to split the lecture page into three sub-pages: dashboard, questions and messages. Hence, in the website the creation of questions is integrated into the lecture page. The question list is also part of the lecture page. The add-in however, is not as spacious as the website, and thus, the lecture page will be just a list of questions containing the slide, lecture and course questions. The dashboards control panel information and functionality is implemented implicitly and the display

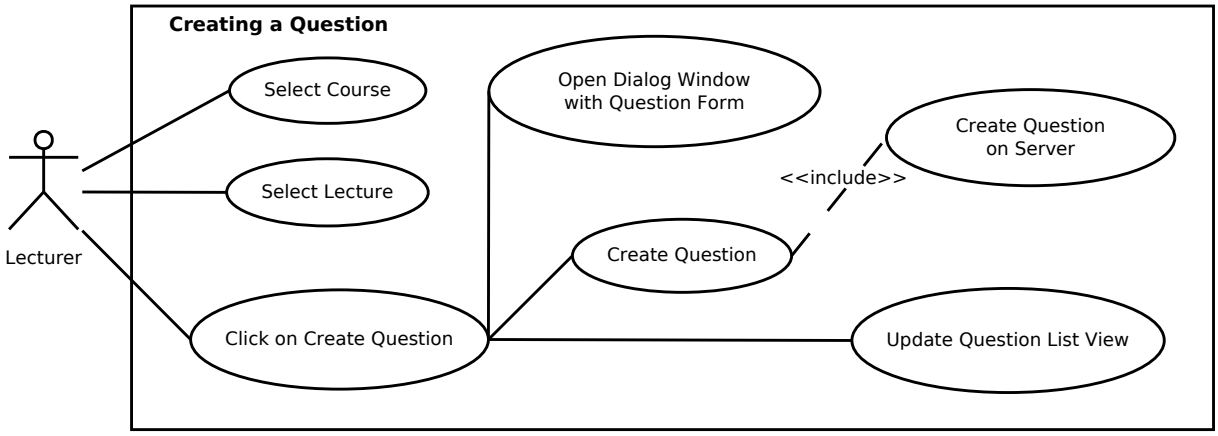


Figure 4.6: Use case diagram for the process of creating new questions.

of results is not the focus of this work. In the lecture page new questions can be created by simply clicking on 'add new question'. The question form will open in a new dialog window while the taskpane still displays the question list. After finishing, the dialog window will be closed and the question list is updated showing the newly created question. This process is shown in Figure 4.6.

Presenting the Lecture

Presenting the lecture is the process when the lecturer presents the slide set to an audience, as shown in the use case diagram in Figure 4.7. To reduce the efforts of the lecturer, the lecture state is set automatically. When starting the presentation the state will be set to 'during' and when the lecturer exits the presentation mode in PowerPoint the lecture state will be set to 'after'. Furthermore, when the lecture is started, the add-in has to observe slide changes and send them to the server. The channel for slide change is be bidirectional, so that the presentation can also be controlled from the website. To achieve the backward direction the add-in listens to slide changes on the server as well.

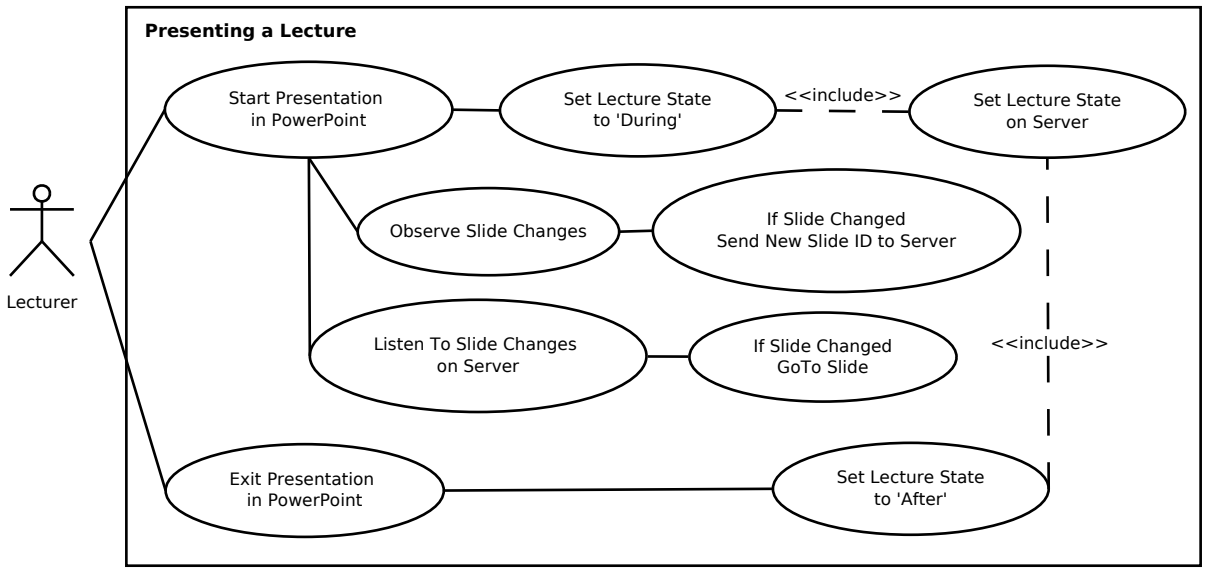


Figure 4.7: Use case diagram showing the process of presenting a lecture.

4.3.2 User Interface-Navigational Paths and Screen Mock-ups

In this section the UI navigational paths and screen mock-ups will be shown.

Navigational Structure

The navigational structure describes the path a user can take to navigate through the system. The page hierarchy is shown as part of the navigational work-flow in Figure 4.4. However, the figure also shows the necessary user actions. Dialogs are opened in a new window and are therefor not really like a new page, because they don't appear in the navigation of the system (when using the back-button). Furthermore, after closing a dialog window the page from which the dialog was opened will be still displayed. From the landing page it is possible to go to the Login page, which leads to the course overview right after a successful login. From here a user can go to the lecture overview by selecting a course. By choosing a certain lecture, the question overview can be reached. The question per slide view can only be accessed via the menu and if the presentation has been linked to a lecture. If the Light Weight version is used, the question overview and the question per slide view can not be reached. Hence, the question creation is not possible in this version.

Screen Mock-Ups

The screen mock-ups will cover the most important UI components of the add-in. Starting with the ribbon, which enables the integration into PowerPoint. Followed by the menu, which provides some short-cuts and special functionality. Also the lists for courses, lectures and questions will be shown and finally, the dialog window mock-ups are presented.

The ribbon is the part of the PowerPoint menu that can be used by the add-in. It is used to open the taskpane, access the website, set the lecture state and synchronize manually. It is integrated into PowerPoint automatically when the add-in is inserted through the XML file. The ribbon UI is static and can not be changed after it was loaded initially. The ribbon menu is split into three groups: general commands with buttons to open the taskpane and the website, the set linked lecture state and the synchronization.

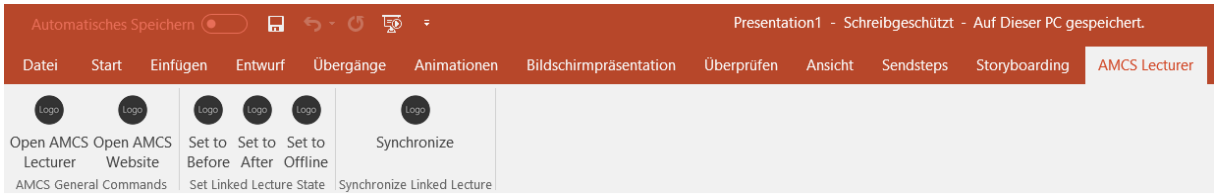
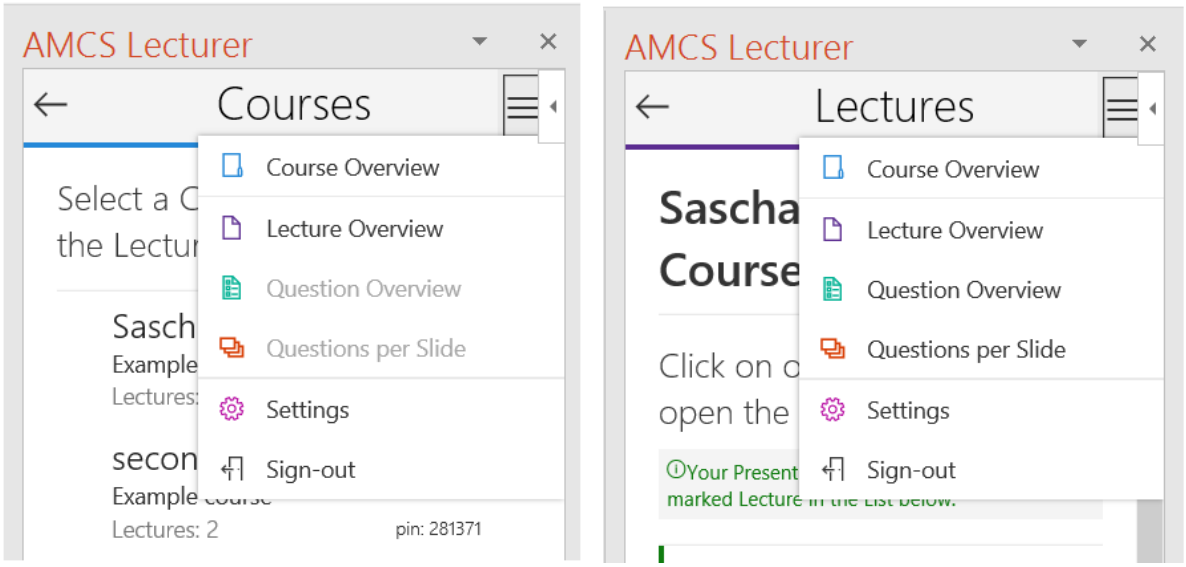


Figure 4.8: PowerPoint menu with the AMCS lecturer add-in ribbon extension.

The menu is part of the header. It gives an overview of all pages. It can be used as short-cut to go to the course, lecture or question list. However, some functionality can only be accessed through the menu. This includes the question per slide page and the sign-out. Initially, some parts of the menu are locked. As shown in Figure 4.9 the 'Question Overview' and 'Questions per Slide' will be locked until the presentation is linked to a lecture.



(a) Menu before linking a lecture.

(b) Menu after linking a lecture.

Figure 4.9: The add-in menu.

The lists that are used are the lists suggested by Microsoft's fabric. Each list item can have primary, secondary and tertiary text as well as meta text (see Figure 4.10). Furthermore, the items can have actions that appear at the top right of the list item. Below a mock-up of every list can be found.
Starting with the course list shown in Figure 4.10. Every course item has the course title as primary text. Secondary text is the course formulation, if it is too long it will be clipped. The number of lectures in the course is displayed as tertiary text. The last element of each course item is the pin of the course, shown as the meta text element.

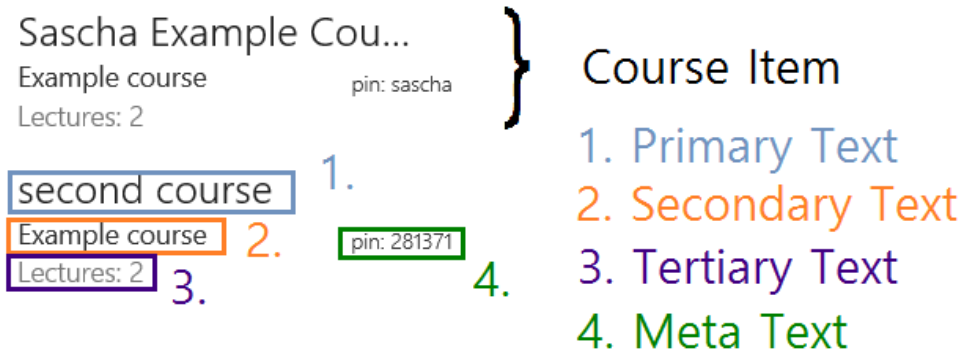
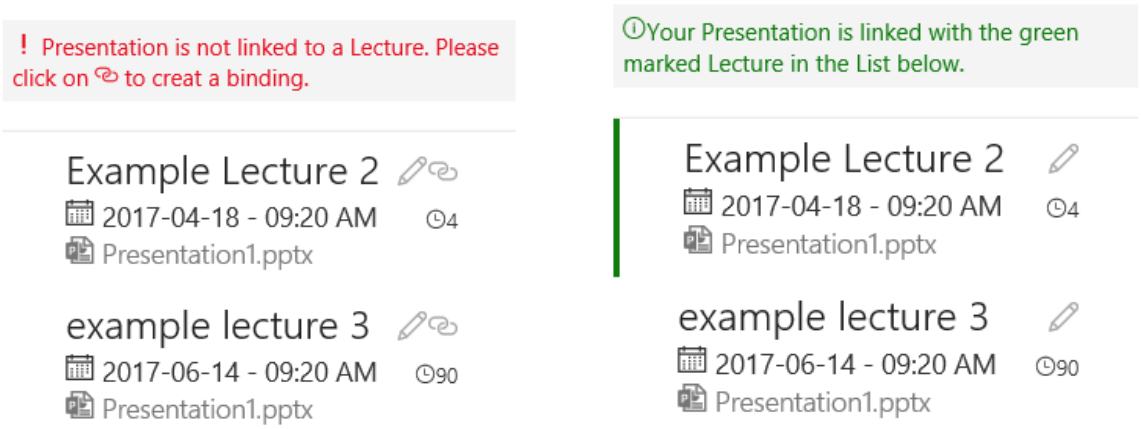


Figure 4.10: Course list with explanation of the list item elements.

The second list is the lecture overview, which lists all the lectures of the selected course (see Figure 4.11a). Every lecture list item contains the lecture title, the date, the duration and the PowerPoint filename and extension with which it is linked. If the presentation is not linked to any lecture a warning will be displayed, see Figure 4.11a. If the presentation is linked to a lecture in the list, then a positive notification will be displayed and the lecture will be highlighted by a green border on the left. Again following the style guides of Microsoft's fabric. The link action icon will be removed as shown in Figure 4.11b.
The last list displays all the questions of the slide set, lecture and course. A question can belong to a

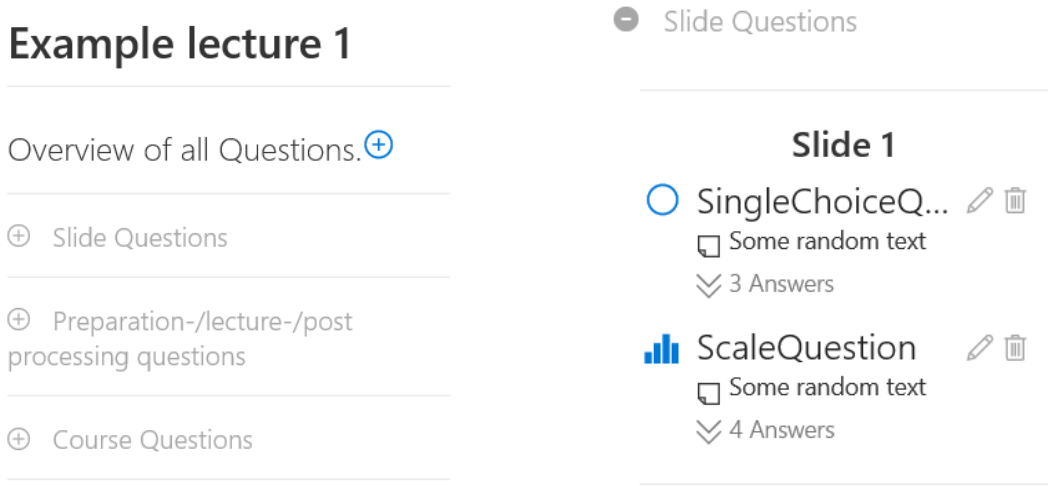


(a) The lecture list if the presentation hasn't been linked to a lecture yet.

(b) The lecture list after a link was established.

Figure 4.11: The lecture list.

particular slide, the lecture or the whole course. Hence, the list of questions is split into three sublists, the same way as on the website. When the page is first loaded all the lists are collapsed and must be expanded by a click on the list name (see Figure 4.12a). The Figure 4.12a shows the expanded Slide Question list. It holds all questions for all slides. Every question list item has a question type symbol, the question title, the question formulation and if the presentation is linked, the question items action component provides the possibility to edit or delete the question. When clicking on the question item, a list with all answers is displayed.



(a) The collapsed question list only displaying the three categories.

(b) The questions in the expanded slide question category.

Figure 4.12: The question lists.

The dialogs will be displayed using the dialog API from Microsoft. In total, five different dialogs exist. The first four dialogs are the lecture, the question, the synchronization and the set lecture state dialog. The fifth dialog is used to open the AMCS website, because it is not possible to open a new window directly from the ribbon menu. Therefor, a new dialog window is opened and there a new

browser window is opened to display the website. The dialog terminates itself after a delay. The lecture and the question dialogs are built like their counter part on the website and therefor use bootstrap for the UI components. Just a few parts like header and footer are adjusted, because they are a new window and not just an overlaying modal. In Figure 4.13 an opened dialog window can be seen. It will always be on top and centered in PowerPoint.

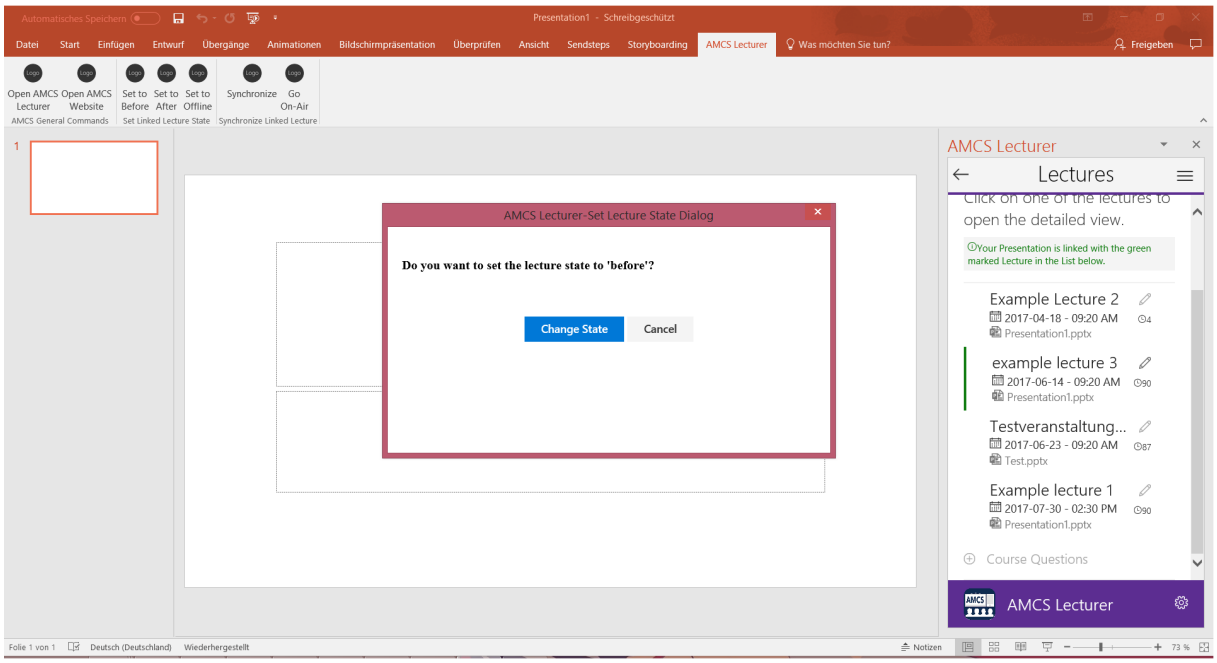
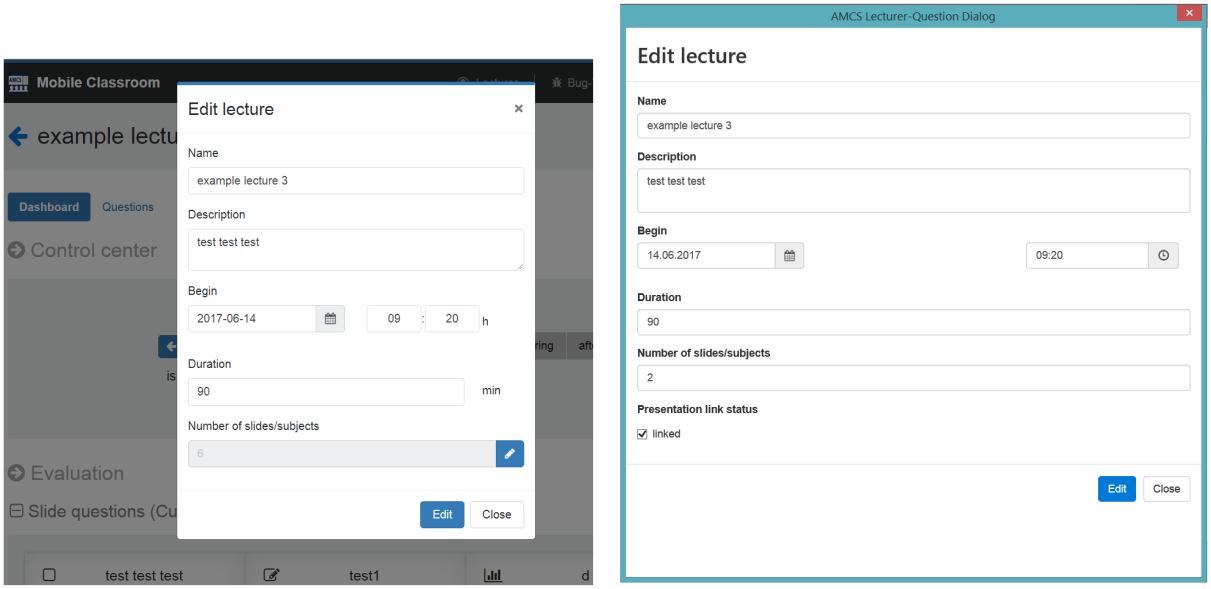


Figure 4.13: PowerPoint with the loaded taskpane add-in and an opened dialog window.

The lecture dialog can be accessed from the lecture list after selecting a course. A click on edit lecture in the action panel will open the dialog window filled with the information of the lecture. In Figure 4.15a the dialog from the website and in Figure 4.15b the dialog from the add-in are displayed. The round corners of the header and footer have been removed and also the blue line in the header is removed in the add-ins dialog. Furthermore, the header title uses the styles suggested by Microsoft. The main functional difference is the extra option to unlink the presentation from the lecture. This functionality could be integrated into the server as well, but makes much more sense in the add-in and could be even confusing on the website. Small style changes are made for the time selection and the number slides. The time selection now uses a time selector as used by many smart phones. The number of slides only affects the logical slides and not the physical slides in the slide set anyway. So it is arguable to keep this option and maybe it would be better to provide a mechanism to adjust the server slides to the physical slides.

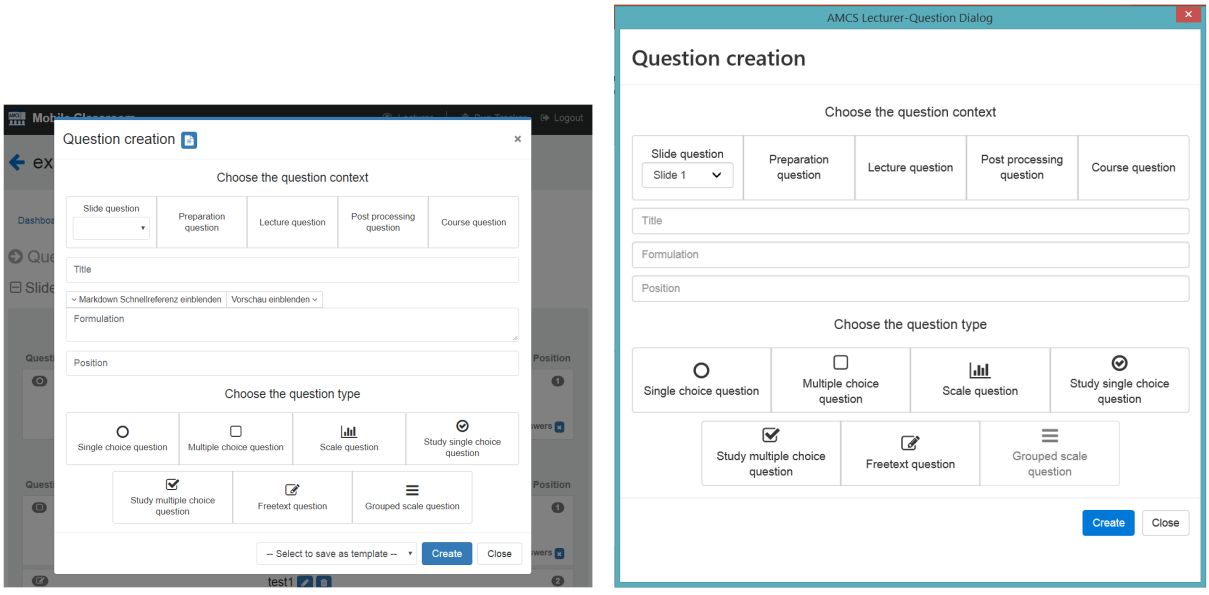
The question dialog is very similar to the one on the website (see Figure 4.14a). The header and footer have been changed like in the lecture dialog (round corners in header and footer and blue line at the top of the header have been removed). The main difference is, that the add-in doesn't offer templates yet. Furthermore, the slide question field will be preset to the currently active slide. In the Figure 4.14b the active slide is Slide 1.



(a) The dialog overlay in the website.

(b) The dialog window in the add-in.

Figure 4.14: The lecture forms.



(a) The dialog overlay in the website.

(b) The dialog window in the add-in.

Figure 4.15: The question forms.

The dialogs to set the lecturer state are used to set the state of the lecture. The lecture state can be set to 'before', 'after' and 'offline'. The dialog is opened by clicking on the respective button in the ribbon menu (see Figure 4.8). The dialogs only differ in the formulation and hence, only the dialog to set the state to before is shown. It is displayed in Figure 4.13. If the presentation is not linked to a lecture setting the state is not possible. The respective dialog window is shown in Figure 4.16. It displays a warning message and disables the 'Change State' button.

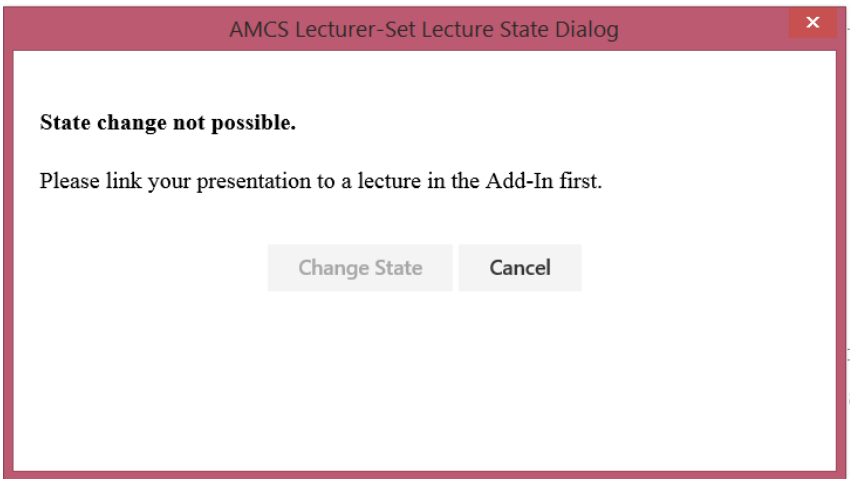


Figure 4.16: The set state dialog - denying the state change.

The Synchronization Dialog is an information popup that tells the user that synchronization will be done soon. Hence, it just displays a warning in case that the presentation is not linked to a lecture, because then synchronization can not be executed. Or in the case of a linked lecture, it will display the following message: 'Preparing Synchronization.' Essentially, it looks the same as the set lecture state dialog (compare Figure 4.16).

4.3.3 Synchronization

In the context of the lecturer add-in synchronization means the update of the model state according to the changes the user made in the PowerPoint slide set. Each slide has an index giving the position of the slide in the slide set and a unique id. The model keeps a logical representation of the slide set and all the slides. So it is not important if content on the slide is edited, but if a slide is added, moved or deleted. Because multiple slides can be added, moved or deleted it is necessary to iterate over all slides for a full synchronization.

Synchronization is a big issue for the system. As described in section 3.2 only the currently selected slide can be accessed from the add-in. However, for synchronization the system must iterate over all slides. In order to get the information about the next slide in the PowerPoint presentation the go-to-slide-command can be used. So starting from the first slide the slide index and slide id are read and then the next slide is selected until the last slide in the slide set is reached. This way every slide will be shown shortly and the user will feel like the presentation is externally piloted and the slides might be flickering on the screen. This two facts may result in a very unpleasant and confusing experience. Furthermore, if the presentation contains many slides (more than 20) or a lot of pictures then the synchronization will

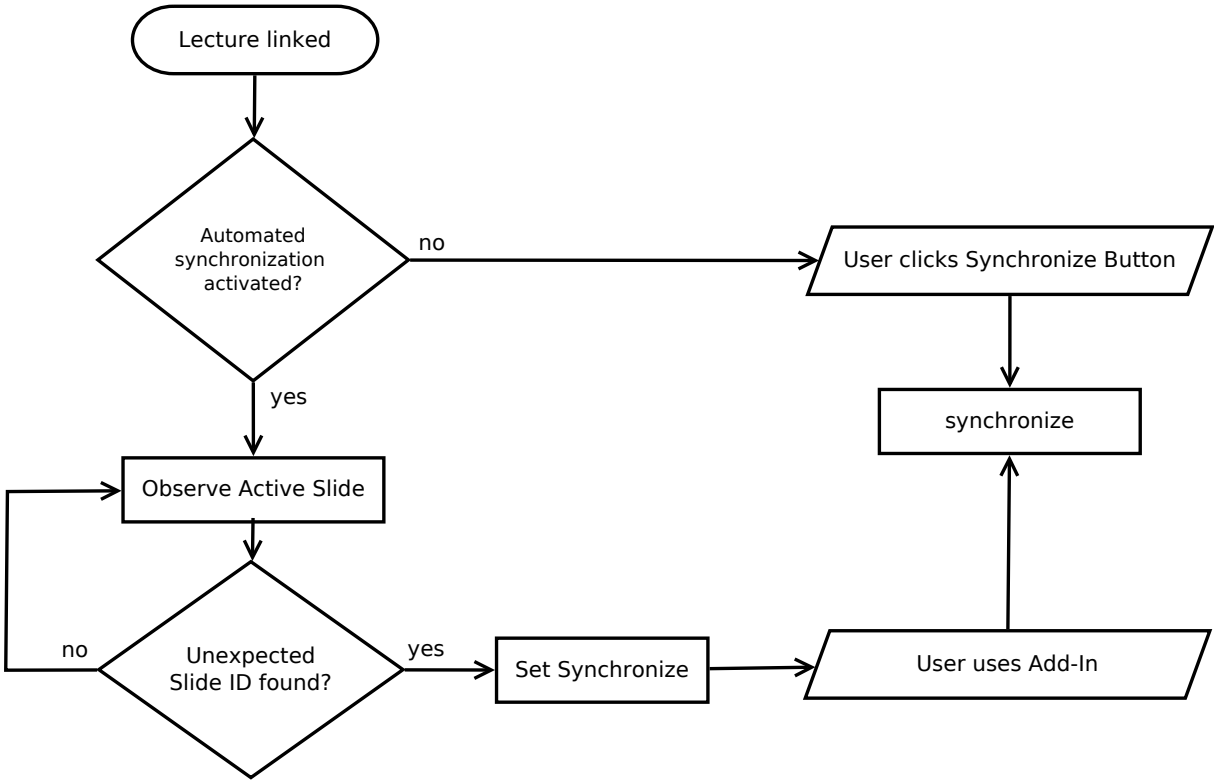


Figure 4.17: Synchronization flowchart.

take noticeable time (greater than 1 second). Because it will not be possible to use the system while synchronizing, this may result in user frustration. Concluding from these facts two major requirements arise for synchronization: first a clear notification should be provided to reduce confusion and frustration and secondly synchronization should only be done when necessary. In the following three different approaches will be discussed.

The three options are: 1. Synchronize frequently using a fixed time interval. 2. find out when the user isn't using the add-in; and 3. only synchronize if a button is clicked. Synchronizing frequently makes it easy for the application to continuously retrieve information about the slides in the slide set. It also automates the synchronization process for the user. However, because of the above described issues the synchronization can only be done in large time intervals, because if the time interval is too small the application will always synchronize, which makes it impossible to edit the PowerPoint presentation. Considering a large time interval, there might still be times when synchronization is done even through the user didn't add, move or remove any slides, which makes the synchronization unnecessary. Furthermore, when the user edits the slide set shortly after a synchronization it takes until the next interval to synchronize. These issues make this option impractical.

The second option is still offering automated synchronization, but reducing the constraints given through fixed time intervals. However, the application must recognize when the user might need a synchronization. The users mouse movement can be used to detect this. Two possibilities are to check when the mouse enters the add-in or when the add-in is focused. Then a one time synchronization is useful. Because while the user is working with the add-in, no changes at the slide set can be made and vise versa.

When the user edits the presentation the add-in is not needed. However, only because the user enters or focuses the add-in doesn't mean that a synchronization is necessary.
The third option is explicit synchronization through a click by the user on a button. Hence, it doesn't offer automated synchronization. On the other hand, synchronization is only done if necessary, because the user knows when he applied changes to the slide set. However, if the user forgets to synchronize it could lead to confusion why the data is not correct. So this approach requires special care by the user. But it also only triggers the synchronization and thereby also the synchronization message exactly when the user expects it. A comparison of advantages and disadvantages of the three options is shown in Table 4.1.

Method	Synchronize frequently using a fixed time interval.	Only synchronize if the user is using the add-in.	Only synchronize if a special button is clicked
pros	+ add-in has a continuous information flow and can keep the data synchronous all the time	+ synchronizing automatically + synchronizes only when user uses the add-in and not while the user doesn't need it	+ very simple to implement + user expects the synchronization + only triggered when user made some changes (so only when necessary)
cons	- destroys the work-flow - if done too often it makes editing slides impossible	- need to find out when user uses the add-in - could still synchronize when it is unnecessary	- forces user to explicitly synchronize

Table 4.1: Comparison of different approaches for synchronization.

As discussed above the first option is impractical. The second option still lacks the problem of unnecessary synchronization. To overcome this problem. A smarter option has been derived from this automated approach. The add-in will observe the active slide frequently every fixed interval. If a slide with an id is detected that is not expected, then the application will synchronize the next time the user uses the add-in. Using the application will be detected as described above for the second option.
However, the third option will be implemented too. In this way the user can decide, if the second or third option is preferable for him. As discussed above, adapting to the user is an important part of good UX. The synchronization process with both options is shown in Figure 4.17.

4.4 Tests

Tests are important for quality assurance by proving the right functionality for given tests. With different test types the software components can be tested in isolation and when interacting with each other. However, not every part of the system is suited for automated unit testing. Therefore, scenarios were described to ensure the functionality is according to the requirements. Hereafter, the test strategy will be described and the automatically testable software parts are identified. Furthermore, the concept and objective of the integration test are explained.

4.4.1 Using Behavior Driven instead of Test Driven

Dan North, who first described BDD, says that he got asked frequently about the following issues when teaching agile practices (like TDD): "Programmers wanted to know where to start, what to test and what

not to test, how much to test in one go, what to call their tests, and how to understand why a test fails." [25] He further claims that if tests don't describe the behavior comprehensively it can create a false sense of security. So he strongly advices to have the desired behavior in mind and test if the code behaves correctly, instead of testing against surrogate tests. He states that many misunderstandings about testing could be solved by replacing the word test with behavior. His advice mainly focuses on three things he found to be misunderstood easily by programmers. First the name of each test should be a sentence that describes the desired behavior. Secondly, what to test. North states that it is already given in the test name - one test should check the correctness of the behavior described in the name. And finally, how to handle a failing test: in the case of a failing test either a bug has been introduced, the behavior was moved, or the test is not relevant anymore.

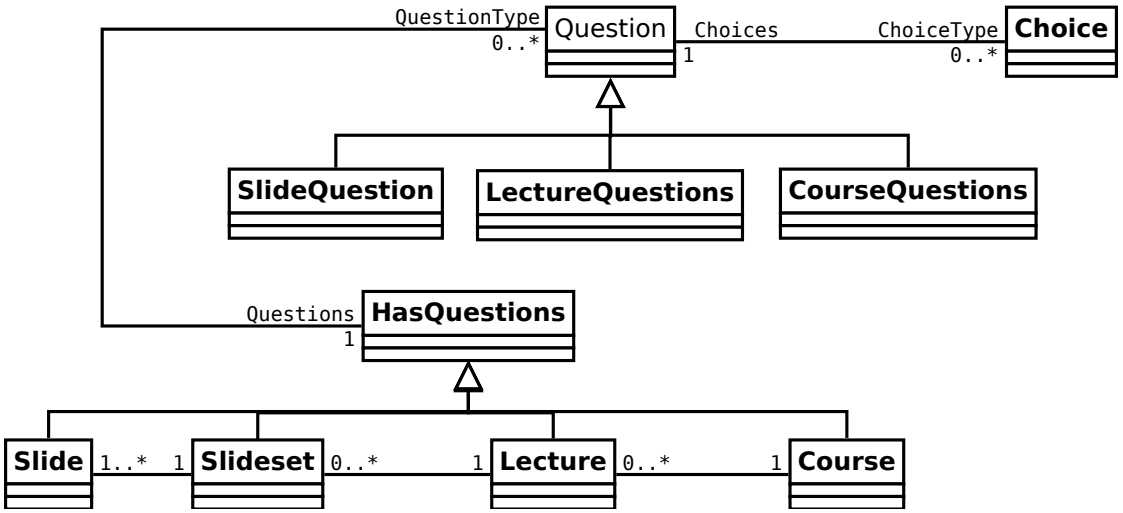


Figure 4.18: Unit testable part of the Presentation Model.

4.4.2 Unit Tests

The easily unit testable parts are the model classes shown in Figure 4.18. They don't need user interaction and can be instantiated with mock-up data.

4.4.3 Integration Test

In contrary to the unit tests, which can be run in isolation, the integration tests need interaction with the slide set and the server. Hence, they must be run in PowerPoint. They also cover the handlers and the mapper functionality. The UI is not automatically tested. Therefore, view components have to be checked manually. However, the View doesn't have much logic to begin with, it is a mere display of passed data. The user interaction is just forwarded to the Navigator class, which serves as interface between View and Presentation Model. In the integration test some user input will be simulated and thus, the Navigator and the handlers functionality is tested as well.

5 Implementation

In the last chapter the concept was described giving an overview of the system. The system architecture and work-flow were exploited as well. The chapter concluded with a discussion about the testability. In this chapter the implementation of the concept will be explained. Starting with general aspects, followed by the description of the prototype and concluding with the test implementation.

5.1 General Aspects

As mentioned in section 3.2 the new PowerPoint JavaScript API for Office supports much less features for interaction with PowerPoint than the old VSTO add-ins. For example it only provides limited access to the PowerPoint ribbon. The ribbon can only be set statically and hence, can not be changed at runtime. Furthermore, new slides can not be created and all data interaction is only selection based. Selection based means that only the currently by the user selected data can be accessed (i.e. slide index, id and slide content). Another limitation is that only text and image coercion types can be inserted in the selection. However, as discussed in section 3.3 the API also has advantages (e.g. platform independence) and it is still actively being developed. Nonetheless, especially the interaction with the slides needs many work-arounds, like for the communication between ribbon and taskpane. The biggest challenges arose when creating the synchronization, because slide access is not straight forward.

However, this is not the only technological barrier. The PowerPoint add-in is run in an IE 11 iFrame. This limits the supported JavaScript functionality drastically, because IE11 is one of the weakest browsers in means of JavaScript functionality support (i.e. no ES6 support). To make JavaScript computably efficient the JavaScript API for Office uses asynchronous functions and since promises are not supported by ES5 the prototype relies heavily on callbacks. Callbacks are functions that are passed into another function that executes the callback after finishing its computation.

JavaScript is an object oriented language, which means that everything is an object with a prototype. It does not really provide classes, at least not in ES5. However, 'classes' can be created by using anonymous functions. An anonymous function has no name and creates a separate namespace, which prevents global namespace pollution and provides a private scope. To make it globally accessible, the constructor function within the anonymous function is saved in a global variable. Inheritance can be achieved by copying the prototype of the parent and adding more functions or overriding existing ones.

When using anonymous functions it is possible to create private variables that are only existent in the namespace, but they are not added to the prototype making it impossible to inherit them by children. In order to make an attribute inheritable it has to be added to the prototype, which unfortunately also makes it globally accessible. For this case the '_private' convention is used, meaning any private attribute beginning with '_' should never be called from outside the context where it is declared.

5.2 Prototype

The actual implementation of the prototype differs from the class model shown in Figure 4.3. First, the Choice class has been omitted. Choices are mere data objects that are part of a question. Therefore, it was decided to implement them as part of the question. The choices are nested in the question object retrieved from the web service and are managed by the Question class. Furthermore, the three child classes of the Question class were omitted. They are realized through roles. Each instance of a Question takes the role of a slide, lecture or course question. Furthermore, inheritance is done by prototype chaining, which means that when an attribute (i.e. a function) of an object is called a search for this attribute is conducted. If an attribute in the root of the prototype chain is searched than this can be time consuming. Therefore, prototype chaining is bad for performance and it is desirable to keep the hierarchies low. [1].

The HTTP, Navigator, UINotifications, Menu and Labels classes are all implemented as singletons. In contrary to the other classes they are just global objects and not anonymous functions. The HTTP object wraps the communication with the web service by using Asynchronous JavaScript and XML (AJAX) calls. For the AJAX calls the jQuery library is used. The actual request handling is done in the Handler classes. The Navigator is the interface between handlers and view. It handles all user interaction and forwards it to the handlers. The UINotifications object provides functionality to display toast messages and dialogs. The Menu object handles all menu functionality and the Labels object contains all labels that need to be present in German and English.

Lazy load is a design pattern that only loads data when it is needed. Fowler [5] categorizes the possible implementations into four main variants. The first one is Lazy Initialization. In this case the not loaded variables are marked by a certain value (i.e. null) until loaded. So when accessing a data field it needs to be checked if the marker is set. The second variant is the Virtual Proxy. As the name says it is just a proxy that provides the same interface as the real object. When a method is called the proxy loads the real object and delegates all calls. The Value Holder is the third variant. When using this method the object provides a getValues method that, if called, loads the data. And the last variant is a Ghost. In this variant the real object is used, but no data is loaded until the first method is called. The implemented prototype uses the third variant - a Value Holder. Meaning the data is loaded when a specific load function is called. However, the getValues methods were given more meaningful names.

5.2.1 Handlers

The handlers decouple the model completely from the view and the web service. They are implement using the lazy load pattern. If the requested data is not present in the model yet, the handlers will send a request to the server to load the data. The data is only loaded once. After that all changes are applied locally and then persisted on the server. Only the lecture state and the active slide are updated by listening to the web-socket after a slide set has been linked to a lecture. This enables the possibility to remotely control the PowerPoint presentation from the website or other platforms.

The Navigator class has a set of connected handlers. Handlers are initialized on start up and registered at the navigator.

The SlideSetHandler manages a linked slide set, including all data connections with the server, concern-

[1] `https://developer.mozilla.org/en-US/docs/Web/JavaScript/Inheritance_and_the_prototype_chain`

ing any slide data. It also manages the web-socket connection. Furthermore, it handles all user input for the SlideSet object. The SlideSetHandler also has a slides array, but only for synchronization reasons. The SlideSet object holds the actual slides.
The course, lecture and question handler all wrap the specific functionality for courses, lectures and questions.

5.2.2 Model

In this section the model part of the Presentation Model will be described. The actually implemented classes are as shown in Figure 5.1. The model contains the Course, Lecture, SlideSet, Slide, HasQuestions and Question classes. They are all implemented by using anonymous functions. However, none of this classes has private attributes. All attributes are added to the prototype and are therefore global, but also allow further inheritance. To mark private attributes the common practice was used to indicate those attributes by starting their names with a '_'. Any prototype attribute starting with '_' should never be called from outside the object scope.

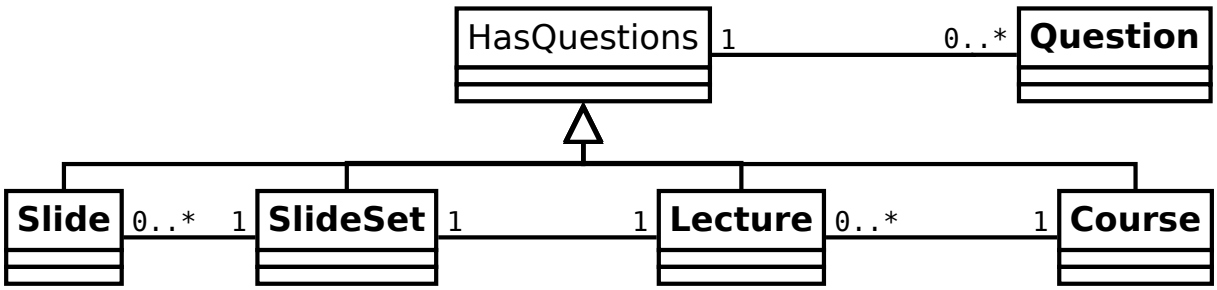


Figure 5.1: SlideSet and Slide class diagram.

Every Course can have lectures and every lecture has a SlideSet. Each SlideSet object has slides. The SlideSet class (see Figure 5.3) is an abstraction of the physical slide set in PowerPoint. Therefore, the _slides array starts at index 1. At index 0 it has a null value, because the index of the first slide is 1 in PowerPoint. It is important that Slide objects take the role of a logical and physical slide. When a slide is loaded from the web service it is only a logical slide. It becomes a physical slide when it is linked to a physical slide in the actual presentation by saving the id given by PowerPoint. If a SlideSet object has more logical slides then physical slides (i.e. too many slides have been set when creating the lecture) they are just left unlinked. If a new slide is created in PowerPoint the SlideSet can get an unlinked slide by using the getNotLinkedSlide function. The Slide class on the other hand has a _indexAtSlideSet and the position on the server in the _slideObj.position variable. Both are important for consistence between the local data and the server. HasQuestions is an abstract class (shown in detail in Figure 5.2) and the Course, Lecture, SlideSet and Slide classes inherit from HasQuestions. However, the SlideSet itself doesn't have ques-

HasQuestions
+_questions: array +_questionsLoaded: boolean
+getLastQuestionPosition(): integer +numberOfQuestions(): integer +hasQuestions(): boolean +questionsLoaded(): boolean +setQuestions(questions): void +getQuestionByID(id): Question +getQuestionUIObjs(): array +addNewQuestion(questionObj): void +addNewQuestionAt(questionObj): void +removeQuestionByID(questionID): Question or null

Figure 5.2: HasQuestions class diagram.

tions, hence, it overrides the two functions setQuestions and getQuestionUIObjs. When setQuestions is called the SlideSet calls the setQuestions functions of each Slide instance. When calling getQuestionUIObjs the SlideSet collects all question objects from all slides and returns them. The Question class also wraps the choices and handles all data manipulation.

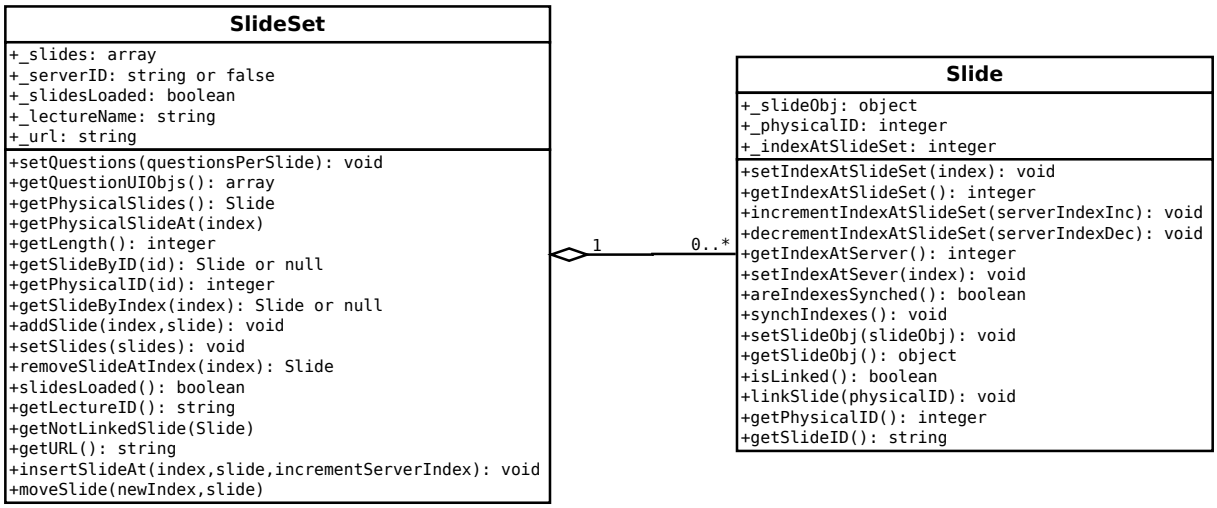


Figure 5.3: SlideSet and Slide class diagram.

5.2.3 The Dialogs

Through the Microsoft dialog API a dialog window can be opened. Only one window can be opened by one context at a time. Therefore, it is possible to open a dialog window from the ribbon and at the same time open a window from the taskpane add-in. The host can receive messages and close the dialog window. Unfortunately, it is possible to close the dialog with a click on the close button in the upper right corner, which is not detectable by the host. Both the beforeunload and unload event can't be used to send a message to the taskpane informing it about the close. Hence, it is not possible to notify the host manually by sending a message when the window is closed. First, it was planed to have a dialog variable to make sure to not open a dialog if another one is open already, since it would throw an exception. However, if the close button is used, the close can not be detected, which destroys this mechanism by making it impossible to open another window dialog until the taskpane is reloaded. In the final implementation, when a dialog is opened it is not checked if another dialog is open and just simply the exception is caught.

The communication between dialog windows opened from the ribbon and the taskpane is also not trivial. The ribbon function-file and the taskpane code run in different contexts and hence, can't send messages from ribbon to taskpane. It is necessary to communicate through a central point. This can be the server or the local storage. The session storage can't be used, because it introduces stricter domain / context rules. Sending data over the server is cumbersome, because the server API needs to be adjusted and it introduces extra delay. Therefore, any data or commands that have to be shared between ribbon and taskpane are saved in the local storage and the local storage is checked frequently if changes occurred. In JavaScript a storage event exists to check for changes applied to the local storage, but it is not supported by IE11. Therefore, a timer is used to check for data changes frequently.

As described in the concept (paragraph 4.3.2) the dialogs were designed to be similar to the website dialogs. This is not only supporting seamless integration, but also supports reusability, because the same dialog code could be used across the different platforms. Therefore, a special dialogs project has been created, which is hosted as a service. The dialog service provides a small API to change some styles, the language and set and get all form data. To use the dialogs from the service a wrapper is used that loads the actual dialog from the service and applies some small adjustments, like changing the round header and footer and remove the blue line in the header. In total the dialog service provides two dialogs: the question and the lecturer dialog. The wrapper gets a variable passed that determines if the dialog is for creation or editing the question or lecture. In case an existing lecture or question should be edited the wrapper has to load the data and pass it to the dialog. The dialog can send data back to it's parent. This feature is used to send the data back to the taskpane add-in. If a lecture or question is edited the dialog keeps track of the changes and sends all the data and a list of changes back to the parent. This way only changed data will be updated on the server to reduce the communication effort.

5.2.4 Synchronization

In section 4.3.3 the synchronization was described and different approaches were compared. Only two of the discussed methods were rated practically usable: the first option is to use a smart auto-synchronization algorithm and the second option is to only synchronize when the user clicks on a designated button. Both options were implemented for usability comparison and to adapt to the users needs.
The smart auto-synchronization works by checking the currently selected slide frequently and comparing it to the logical slide set. If any slide id is found that differs from the expected id, the algorithm knows that it needs to synchronize. However, the synchronization is not done immediately, but when the user uses the add-in the next time. The usage is detected by checking when the courser enters the taskpane area. The good point is, that this approach can insert new slides automatically without synchronizing at all. Also moving one slide is no problem and reordering can be done automatically by moving all slides up or down that are affected. This algorithm may have a lot of advantages, but it has one huge flaw that can not be neglected. It is impossible by only checking the currently selected slide if the last slide or last slides are deleted. The reason for this is, that when the last slide is deleted the current slide will be the second last. So the algorithm checks the existing id and everything is in place. There are two ways to overcome this problem, when the second last slide was active and the user enters the taskpane the next time, the last slide could be automatically checked. It is much less time consuming if only a few slides are checked instead of all. However, deleting multiple slides at the end could still be a problem. The second possibility is to introduce a frequently but not very often executed auto synchronization, just like an auto-save function. Both options were implemented in the prototype, because this way the user has the possibility to choose if he only wants to use auto-save or the smart synchronization or maybe only the manual synchronization button.
The synchronization button is integrated into the ribbon menu. When pressed, it will set the variable for manual synchronization in the local storage to true. So the next time the taskpane checks this variable it knows that a synchronization should be executed. The recognition of the change may take some time. That's why a dialog window will open after clicking on the 'synchronize' button so the user is informed that the synchronization will be done shortly.
The implemented synchronization algorithm is shown in Figure 5.4. First of all, the slides array is reset

and the active slide is saved. This is necessary, because after the synchronization process the active slide is always the last slide. Therefore, the active slide is needed to navigate back to it. The algorithm checks recursively if another slide was found at the next index. If a slide is found than the id of the slide is saved in the slides array at the index where it was found. After this, the id is checked. If a corresponding logical slide is found, then the algorithm executes the next recursive call. If no slide exists for the id then a new slide is created and the logical slide is moved to the index in the SlideSet object.

The recursion ends when no next slide exists. Now the algorithm iterates over all logical slides that are linked to a physical slide. If the slide id is found in the slides array then this means that the slide is present in the PowerPoint slide set. Hereafter, the slide indexes are compared and the slide is moved respectively. If the slide id is not present anymore, then the slide was removed.

For all create, delete and move commands the splice function is used. It automatically reorders all the array indexes after an insert or delete. So a slide move consists of a deletion and an insertion at the new index, hence, no reordering is needed locally. However, this works because the splice function returns the deleted slide. The server API on the other hand does not return the slide, but just deletes it. Therefore, it is necessary to keep track of the slide positions on the server. This is done by decreasing the logical server indexes of all slides after a deleted slide and increasing all logical server indexes of all slides after an inserted slide.

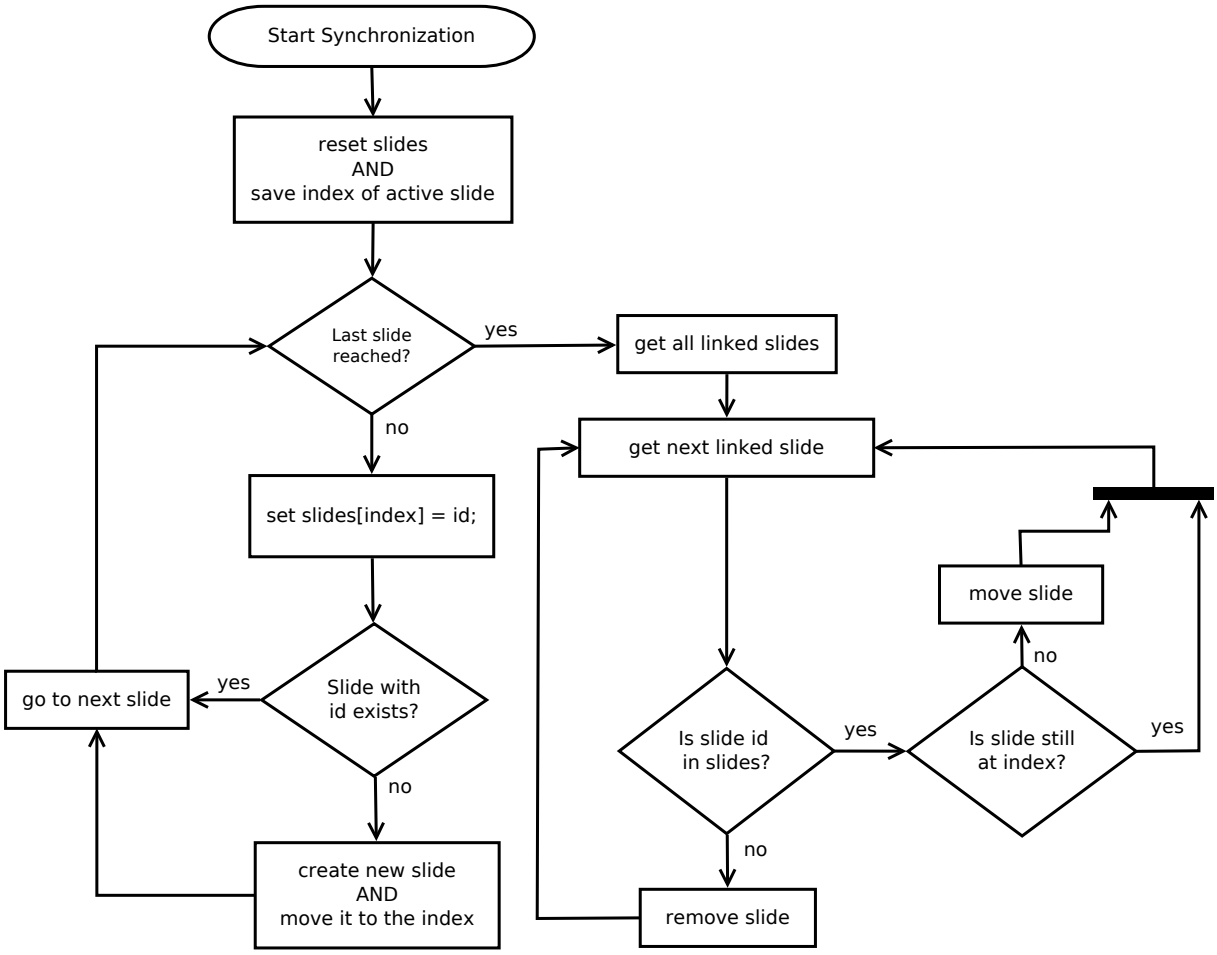


Figure 5.4: Synchronization algorithm flowchart.

5.2.5 Views

For each page a separate view is created. Views are also instantiated by using the lazy load pattern. The navigator checks if the requested view is in the set of _connectedViews and if not present, the view is instantiated. Each view inherits from the MainView class. The MainView provides functionality to set the language and the view color. The QuestionsPerSlideView additionally inherits two functions from the QuestionView, this is possible because inheritance is just copying the prototype of the parent and adding further functions. Thus, the desired function attributes from another parent are just copied to the prototype as well, which allows multiple inheritance.

5.3 Tests

The tests were implemented using Mocha [4]. Mocha is a JavaScript testing framework supporting BDD style tests. It is possible to use mocha with Node.js or in the browser. The ability to run tests in the browser is very helpful for this project, because it is entirely run on the client side. With the usage of an in-browser test framework there is no need for extra adjustments to get the tests running on Node.js. For the assertions the Chai Assertion Library [26] is used. It provides two different assertion APIs: the expect / should API, which covers BDD assertion styles and the Assertion API, which covers the TDD assertion styles. Since the BDD style is used for the tests also the expect / should API from Chai is used. To measure the code coverage Blanket.js [28] is used. Blanket.js can also be used in the browser or with Node.js. The tests are setup in the Blanket.js environment, which initializes mocha in BDD style. Also in the main test file all JavaScript files that should be tested and all tests are loaded. Each test file initializes Chai and provides the tests. After the tests were run, Blanket.js displays the results as an html page. In the end all testing and reporting is orchestrated into one html file. The actual tests are implemented in the style of BDD. The tests in BDD should be readable as sentences and are directly derived from requirements. Therefore, successful tests show that the required behavior was achieved rather than testing the function for bugs. The model part of the Presentation Model can be completely covered with unit tests. The other parts (the handlers and views) need user interaction, data from the web service, data from PowerPoint or have timing constraints, which makes them unsuitable for unit testing. However, the integration test covers the main behavior of these components.

5.3.1 Integration Tests

The integration test is integrated into the system and can be run from the settings view. It is only accessible when the Navigator is in debug mode though. The problem with running the integration test is that it takes a long time. This leads PowerPoint to the assumption that the add-in is broken and stops running the script or just reloads the add-in. In both cases the test has to be re-run and all interesting information for bug tracking is usually lost. The second problem is that many parts of the program run asynchronous. Especially, the data fetches. Therefore, the test cases must be serialized. The testing framework offers the possibility to pass a callback to any function that is tested and waits with any further execution until the done callback is called. However, this needs extra test callbacks. The third problem is that opening dialogs that load content from a web server take too much time and hence, crush the test script. To fix this issue dialogs are only opened very shortly by using a timeout.

6 Evaluation

In the last chapter, the implementation of the before described context was shown. In this chapter, the methods and results from the questionnaire will be presented. Furthermore, the changes that the evaluation results suggest will be discussed, and the concluding changes that were applied to the system will be shown. Finally, the test results will be presented, including the automated unit tests, integration test and acceptance tests.

6.1 Usability Tests

DIN EN ISO 9241, 210 describes usability as: "A person's perceptions and responses that result from the use and/or anticipated use of a product, system or service." Holzinger describes usability as "the ease of use and acceptability of a system for a particular class of users carrying out specific tasks in a specific environment." [7] Following these two definitions usability can be seen as a measure of how well a user can use a system and how satisfying the system is for that user.
Ferré et al. [3] state that the term usability is too abstract and that it is usually divided into learnability, efficiency, user retention over time, error rate and satisfaction. They describe the five terms as follows:

Learnability describes how easily a system is accessible for new users and how fast novices can gain proficiency to complete certain tasks. Learnability can be measured by comparing the time needed to complete a job by a new user and an expert. The authors state that this is a very important attribute for novice users.

Efficiency measures the maximum tasks the user can perform per unit of time. Therefore, this is a measure of maximum speed of task completion by the user. The authors conclude that a higher speed means higher system usability. Efficiency is mainly important for expert users. The authors later suggest using key access for features as a possibility to achieve higher efficiency.

User retention over time describes the ease of use after a period of non-usage. The authors state that this attribute measures how well a user can remember the use of the system. Holzinger calls this also memorability [7].

Error rate measures the amount of mistakes a user does while completing a task. It contributes negatively to usability. The authors conclude that errors reduce efficiency and user satisfaction. Furthermore, they say that a high error rate "can be seen as a failure to communicate to the user the right way of doing things" [3]. which ultimately means that the usability analysis should be revised.

Satisfaction is based on the user's subjective perception and preference of use. It is like the sum of all parts of usability and hence, gives an overall perspective of how well the systems usability is.

Usability testing is used to uncover design problems. It is mainly done by observing users using a prototype or a mock-up of the system. Ferré et al. [3] say that the first step of usability testing is to decide how many users from which user group should perform the tests. The second step is to create the test task that will be used for observation. For this the authors suggest to consider the following aspects:

1. whether the participant can ask the evaluator for help;
2. should two participants jointly perform each test task to observe the remarks they exchange in the process;
3. what information participants will receive about the system prior to the test; and
4. whether to include a period of free system access after completing the predefined tasks to get the user's overall impression of the system.

Holzinger [7] names three usability tests: thinking aloud, field observation and questionnaire. Thinking aloud is a test method where users tell their thought process while using the system. It is helpful to find out why users do something. Holzinger also states that thinking aloud will be even more effective when done in groups of two, because it feels more natural to verbally express their thoughts for the participants when using the system together with someone.
The second method is field observation and is the easiest to conduct. In this method the participant uses the system in the real environment. Therefore, it is best if the observer is invisible and the tester can use the system freely and naturally.
The third method is a survey using questionnaires. The simplest form of a questionnaire is an interview. Questionnaires are especially useful to estimate the subjective satisfaction of the users.

6.1.1 First Test: Thinking Aloud and Questionnaire

The first test used the thinking aloud method and a survey approach to rate the satisfiability. However, when using survey approaches it is not always clear why a user decided to give a specific score. Therefore, the questionnaire was followed by a short interview clarifying some user choices. The tests were conducted using the system described in Table 6.1.

Platform	Windows 10
Office PowerPoint	2016

Table 6.1: Test system setup.

A general overview of the test setup can be found in Table 6.2. The two expert users had seen the system before, while the 4 novice users didn't have any previous knowledge. However, every novice user had used the AMCS website before. Otherwise they wouldn't be able to rate the seamless integration. Furthermore, the light weight version is not meant as stand alone version, but more as an extension to the AMCS website, hence, some knowledge of the website is necessary to rate the add-in.

Goal

The main goal of the first test was to evaluate the learnability, efficiency, error rate, satisfaction and how well seamless integration was achieved. Efficiency and error rate as well as parts of learnability can

Date	between 04.07. and 13.07. and the last on 14.08.2017
Novice Users	4
Expert Users	2
Time	approximately 30 minutes

Table 6.2: First test setup.

be quantified for objective comparison. Satisfaction and seamless integration however, are subjective values. They are rated by the results from the questionnaire and the interview.
The interview was used to check on very poorly rated questions and also clarifying confusing things said in the think aloud test. As well as checking general satisfaction and giving free space to suggest improvements and state what was perceived positively.

Test Execution

As suggested by Ferré et al. [3] the following four general aspects were chosen for the test:

1. the participant can ask the evaluator for help if really necessary;
2. participants should perform the tests alone;
3. participants will receive a short introduction showing all features necessary for the given evaluation; and
4. a period of free system access is included after completing the predefined tasks.

The test persons first received a short introduction, which explained the features of the system necessary for the task. After this, the users were handed out the task. The participants were instructed to think aloud while solving the task. They were allowed to ask questions if stuck. When the task was finished each tester was asked to answer the questionnaire. The questionnaire used a five step Likert scale between '- -' and '++'. '- -' meaning total disagreement, '++' meaning total agreement and 'o' meaning neutral. During the task the test persons used the full version of the add-in, which provides complete functionality and always shows maximum information available. Before the final interview, the light weight version was presented and each user had time to freely use the add-in for up to 5 minutes. During this period, the user was again instructed to think aloud.
The given task simulated the real usage. In the first part, the participants had to link the lecture and create questions. The second part on the other hand focused on the presentation. To underline this, the participants had to close the application between the two parts. In the real situation the lecturer would usually do the two parts separately and even on different days, maybe even with a big time gap.

Results

In this section the results from the first test are presented. The results don't appear in chronological order for anonymity reasons and it also allows a better categorization of the results. First, the results are ordered by level of experience starting with the novice users followed by the results from the expert users.

Before the test started every participant was asked about existing experience with AMCS and PowerPoint add-ins. Additionally, questions about the primary use and important features of AMCS were asked. The results are shown in Table 6.3. Most of the users only used the system for 1 semester. Two users had used the systems for 4 or more years. Except for two users all had experience with other PowerPoint add-ins. However, only two had experience with the new add-in type from the store that uses the JavaScript API. The most important features for the participants are the question creation and presentation of questions and results and the automated slide changes. Except for the presentation of results the system provides the integration of the other features into PowerPoint. Thus, the general acceptability of the system should be high amongst the test persons.
In the following the participants number 1 to 4 are rated as novice users and the participants number 5 and 6 as expert users. Only testers number 5 and 6 had seen the new lecturer PowerPoint add-in before.

Nr.	Experience with AMCS - how long	Experience with PPT add-ins	Primary system usage and important features
1	website - 1 semester	none	question creation
2	website - 1 semester	yes	only used during lecture questions before, because they are activated automatically; participant didn't know that slide questions can be used
3	website and old lecturer add-in - 5 years	yes, but none from the store	question creation and presentation
4	website - 1 semester	none	presenting questions and results
5	website, old lecturer add-in - 4 years	yes, but none from the store	presentation with server synchronized - automated slide changes; results and next slide commands on the website
6	website and new lecturer add-in - 1 semester	yes	question creation and presentation

Table 6.3: Interview questions that were asked before the test.

The error rate is not rating the users action, but how intuitive the use of the system is. Because errors come from bad understanding and hence indicate a bad usability. In Table 6.4 the error rates of each participant are shown. If a test person asked for help it was counted as an error, but only rated by 0.5, because it wasn't actually an error. However, it indicates bad usability nonetheless. Only one expert user solved the task without any mistakes. The other expert user however just asked one question resulting in 0.5 error points. The reason for the error is that the lecture state wasn't highlighted enough and the expert user hadn't used that feature before, showing clearly that it is not obvious enough.
The novices made more mistakes. Some confusion was caused by the different views for questions. It seemed that first time users couldn't distinguish easily between the 'Question Overview' and the 'Question per Slide' view. Participant number 4 clicked on the wrong link icon and was confused why the link didn't work. In fact, the same icon had been used to symbolize if a lecture was linked to a PowerPoint presentation and it was also used for the link button. The test person number 2 didn't associate a lecture with a presentation and therefore asked for clarification. Participant number 4 didn't know how to use the pin, because the meaning wasn't understood clearly and suggested to rename it to course id or something similar. The error rate is mainly important for new users and is significant to rate how intuitive the system is. The efficiency in the contrary is mainly important for expert users, because experts want to finish tasks as fast as possible and take short-cuts. However, the measured time is also used for comparison

between novice and expert users showing the learnability.

Nr.	**Level of Experience**	**Error Count**	**What was wrong?**
1	novice	1	Nr. 7: went to question view instead of question per slide view
2	novice	0.5	Asked: Was confused that one lecture is equal to one presentation.
3	novice	1	Nr. 4: clicked on wrong link symbol.
4	novice	1	Nr. 12: didn't know how to use the pin.
5	expert	0.5	Asked at nr. 16: couldn't find the lecture state
6	expert	0	-

Table 6.4: Error rate per participant.

The effficiency rating is based on the measured times for each participant, which are shown in Table 6.5. The second column shows the time needed to complete the given task in minutes and the third column provides information about what was especially time consuming. The novice users needed between 18 to 35 minutes and in average 24 minutes. The expert users needed between 6 to 8 minutes and in average 7 minutes. In average the expert users completed the task 3.4 times faster. This shows that experts are much more efficient. The big gap also indicates that learning the system is not easy. However, when the novice users completed the second part of the task they found navigational paths easier.

Nr.	**Level of Experience**	**Time in minutes**	**comment**
1	novice	35	Couldn't find where to add a new question easily.
2	novice	23	-
3	novice	18	Messages disappeared too fast, so user sometimes repeated the action to trigger the message again (i.e. for F5 warning).
4	novice	20	-
5	expert	8	Only lecture state took some time to find.
6	expert	6	-

Table 6.5: Efficiency measured through speed of task completion by participant.

The questionnaire is used to gather data about the testers subjective view of the system. It is mainly used to rate the satisfiability. The questionnaire consists of 17 statements and the participants were asked to rate them on a Likert scale from '- -' total disagreement to '++' total agreement.
Each of the symbols used in the questionnaire is translated to a score to determine the level of agreement. The level of agreement for each statement in the questionnaire is computed by the sum of each score multiplied by their total count, as shown in equation 6.1 below. Total disagreement '- -' scores -2 and the other scores are incremented respectively, up to total agreement '++' scoring +2. Therefore, a higher score means that the participants agreed more with the statement. If the score is negative, it means that most of the participants disagreed. The maximum score for 6 participants is 12 and the minimum score is -12. Scores between -2 and +2 indicate neutrality, 3 to 5 indicates weak agreement, 6 to 9 strong agreement and 10 to 12 absolute agreement, respectively for negative scores and disagreement.

$$overallAgreementScore = \sum_{i=0}^{4} (i-2) * countOfAnswers_i \tag{6.1}$$

The first 3 questions are designed to get a general impression of the users opinion about the add-in. Focusing on easy use and if it improves the work with AMCS.

Question Nr.	- -	-	o	+	++	score
1	0	0	1	4	1	6
2	0	2	1	2	1	2
3	0	0	1	2	3	8

Table 6.6: Results from the first three questions of the questionnaire.

The following questions 4 to 8 are used to investigate if the systems navigation and the integration into PowerPoint is intuitive, which is especially interesting for new users. Statements number 6 and 7 have a rather poor score. Later questions in the interview showed that participants didn't understand statement 6 that clearly and after a clarification they mostly agreed with the statement. Number 7 however was criticized. The test persons said they would like to have the questions and evaluation results on the slides and not just in the add-in.

Question Nr.	- -	-	o	+	++	score
4	0	0	1	5	0	5
5	0	1	0	4	1	5
6	0	1	3	1	1	2
7	0	3	1	2	0	-1
8	1	1	0	2	2	3

Table 6.7: Results from the 4th to 8th question of the questionnaire.

The questions 9 to 14 are designed to estimate the user satisfaction with the usability of specific system parts, asking for preferences and impressions about the use, displayed information and system feedback. The results for statements 9 to 14 are shown in Table 6.8 The statements 9, 12 and 13 were stating the opposite state of the implementation and were meant to justify the design choice, hence, a negative value was expected. While the scores for 12 and 13 are clearly negative more test persons agreed with statement 9 or were at least neutral. Number 14 got the lowest score among the other statements. This shows that the feedback messages are not satisfying enough yet.

Question Nr.	- -	-	o	+	++	score
9	3	0	1	1	1	-3
10	0	0	2	4	0	4
11	0	1	1	5	1	6
12	2	3	0	0	1	-5
13	4	1	0	1	0	-8
14	0	1	1	4	0	3

Table 6.8: Results from the 9th to 14th question of the questionnaire.

The last 3 questions conclude the questionnaire with general questions about the overall user satisfaction. The testers are asked about learnability, seamless integration and if the light weight version would be sufficient for their use. Question 15, which was about learnability, was rated poorly especially by novice users. However, the test persons agreed strongly that knowledge from the website helped to use the add-in. Showing that seamless integration was achieved in means of similarity in use. The last question was about finding out if users actually do like to have a stand-alone add-in or if they prefer to use the website and only use the add-in to control the web service during the presentation. The results show that half

of the users would like to use the add-in only for the automated web service control. One user totally disagreed and the other users said they would like to keep both versions.

Question Nr.	- -	-	o	+	++	score
15	0	2	1	2	1	2
16	0	0	0	4	2	8
17	1	1	1	0	3	3

Table 6.9: Results from the from the 15th to 17th question of the questionnaire.

The think aloud and interview section both help to get subjective data and user opinions like the questionnaire. The think aloud test also give insight on the users behavior. In Table 6.10 the most substantial improvement suggestions from the think aloud and interview section are shown.
When asked about the positive aspects all of the participants said that the add-ins design was appealing and that the PowerPoint integration is helpful. However, everyone of them also asked for the possibility to show questions and results on the slides. Most of the participants also underlined this as a very important part of the system for them. One even said that the system is not usable without this feature. This clearly shows that the system is not satisfying enough yet. This feature however, is out of the scope of this work.
The second most negatively mentioned issue was the loading circle. This appears when retrieving the current slide data. The add-in runs in a sandbox and has to send AJAX requests to get the slide data from PowerPoint. This however causes the IE11 iFrame to display a loading circle, because it appears as if data was loaded from a web service or server.
The third most negatively mentioned issue was about synchronization. Mostly the participants just said they would like to have no synchronization, but synchronization is necessary for obvious reasons. Two participants mentioned that it was too often and requested to use it only with the button.
The other comments were mainly about style and intuitive use and only require minor changes. However, the comment 2 a and 2 b are more conceptual. The participant argued that one presentation might be used for more than 1 lecture in case that the topic was too large for one single lecture. When suggested to split the presentation into several lectures to fit one lecture each, the participant replied that the 'lecture' should just be called presentation to reduce confusion.

6.1.2 Second Test: Field Observation

The second test was a field observation. It was done by giving the participants a beta version of the prototype and let them use it in their actual lecture. In total 2 participants took part in the second test. Both of them also participated in the first test and were therefore no novice users anymore. The first field observation was done with an earlier stage of the system than the second one. Therefore, the first observation could be seen as an alpha and the second observation as a beta test. Furthermore, the first tester had much less knowledge about the system than the second one.

The first field observation was carried out in a psychology lecture. In a test session the lecturer received another introduction to the system, which explained the changes since the first test. Additionally, the add-in was prepared on the computer of the lecturer. The used platform was PowerPoint 2016 for Mac.

Nr.	Comments	
1	a	Color from header also in menu.
	b	Rename 'Questions per Slide' to 'Questions for current Slide'
	c	Synchronization is too often - synchronize through button only.
	d	Information in which date format the date is given for a lecture.
2	a	One lecture per presentation is not intuitive.
	b	What if a presentation is used for more than one lecture.
	c	Why is the loading circle still there?
3	a	Link symbol used to for link (action) and to show if a slide was linked - confusing.
	b	Too much scrolling.
	c	Questions with horizontal scrolling.
	d	After linking a lecture, only show that lecture in the lecture list - introduce a third system version.
	e	Smaller font or at least adjustable fonts.
	f	Question creation dialog is too difficult (too many things to select and decide by the user).
4	a	The loading circle is annoying.
	b	Pin is not intuitive.
5	a	highlight linked lecture more
	b	lecture state information doesn't stand out enough
	c	footer smaller and in the color of the header
6	a	Synchronization is done too often.
	b	The loading circle is a bit disturbing.

Table 6.10: Most substantial suggestions for improvements by each participant from the think aloud test and the interview.

Since there was no question evaluation on the slides at this moment, the lecturer had to use the website in parallel. When the lecturer reached the slide with the questions, the lecturer exited the presentation mode in PowerPoint to open the website. However, in this moment the questions were deactivated. This happened because the add-in sets the lecture state to 'offline' when the presentation mode is exited and in 'offline' state no questions are displayed for the subscribed students. Even through the other functionality worked without any problems, this issue caused the lecturer a lot of stress and confusion.

In the following discussion, it was decided with the lecturers consent to introduce extra buttons to set the lecture state manually. The lecturer emphasized that the rest of the systems functionality was an improvement to the work flow without the add-in. However, the lecturer also underlined that the new main priority should be to integrate the questions and the question evaluation on the slides.

The second field observation was conducted on a workshop. The used platform was PowerPoint 2016 for Windows. The tester didn't receive any extra instructions before the test and also set the add-in up without further help. However, the lecturer forgot about the fact that the lecture state had to be set manually now, but an assistant recognized it and set the lecture state to 'during', by using the website.

From then on the add-in worked without any problems until the first slid with animations and questions was reached. The questions on the users devices reloaded frequently and so fast that answering was not possible. Furthermore, messages were resent every few seconds until the next slide was shown.

This issue was caused by a mistake in the PowerPoint API. When an animation is active, the API returns 0 as current slide index. However, the slide index should be in the range of 1 to number of slides. When the add-in read the index 0 it sent this new active slide index to the server. After the animation was

complete the slide index was reset to the current slide index and the add-in sent the new active slide to the server. After that a 0 was read as index again caused by more animations and so on. This lead to the frequent reload of the user devices, because the active slide changed very fast between 0 and the actual slide index.
The lecturer himself had no problems with the add-in and was satisfied with the overall functionality. Especially, the adaptability through different settings was appreciated. In this test the footer color and medium version settings described below (section 6.1.3) were already introduced.

6.1.3 Conclusion and Changes

In this section the evaluation of the test results and the derived changes that were applied will be described.

Usability Ratings

The goal of the usability tests was to investigate and rate the five main components of usability: learnability, efficiency, user retention over time, error rate and satisfaction. The main focus however was learnability and satisfaction. The subjective opinion of the users was to be investigated so that the usability as well as the overall acceptance of the system could be increased. Most of the tests were done in an early stage of the prototype so that it is hard to draw well founded conclusion about the efficiency, user retention over time and error rate.
The main factors that influence learnability are an intuitive design and usage, and the seamless integration into the existing system. The better the seamless integration the more helps prior knowledge of the system. The results from the usability tests show that users agree that the system is well integrated by stating that prior knowledge with the AMCS website helps to use the add-in. Additionally, test users agreed to statement number 4 and 5 with a total score of 5, which is at the upper boundary of weak agreement. These two statements were about page navigation and how well the navigation was supported by different colors per page. However, novice users still found the add-in hard to learn in general. The learnability was only rated with an overall score of 2. Through no user totally disagreed with the statement 15, only 1 expert user agreed completely. Showing that the system has a fairly high learning curve in the full version. However, the neutral score also shows that the users don't think the system is too complex to learn. The users complaining about the high learning curve especially mentioned the different views for questions to be confusing. Furthermore, those users asked for less information and reduced functionality to reduce the learning curve. In example, participant number 2 suggested a medium version that only shows the linked lecture and a reduced question creation dialog.
The overall satisfaction was investigated through statement number 1 (add-in is easy to use), 2 (system integration), 3 (add-in simplifies working with AMCS) and 17 (light weight version is sufficient). The sum of the first 3 questions got an overall score of 15 out of -36 to 36. While number 3 received strong agreement and number 1 weak agreement, question 2 was rated neutral This is due to the fact that questions can't be displayed and evaluated on the slides. However, when taking into consideration that users didn't disagree and adding the 36 points of disagreement, then the overall score is 51 points out of 72, which is slightly above 70 percent. This shows that users are basically satisfied with the existing functionality, but also think that it is still missing an important component. Furthermore, the fairly high

learning curve influenced the satisfiability negatively. The results of the last statement number 17 and user comments show that the users appreciate the different versions with different function complexity.
Efficiency can be rated by comparing the time needed to complete the given task. It was found, that expert users needed 3.4 times less time in average than novice users. This indicates that users who surpass the steep start of the learning curve can be much more efficient. However, one expert user didn't have that much hands-on experience and still solved the task three times faster. Furthermore, users had a much better understanding of the system in the test section where they could use the system freely. Indicating that learning the system doesn't take too much time.
The error rate in the first test is not that significant, because the first test was conducted in a fixed test environment. Participants had a specific task and an observer was watching them. In the second test, the field observation, the two participants used the system without a specific task or help during the presentation. Thus, the error rate measure is more significant. The two users of the field observation both made one mistake. The first users mistake was due to a conceptual design flaw and the second one forgot about a change in the system. Therefore, both were only minor mistakes that can be avoided when using the stable version of the prototype.
The tests results are also not that significant for user retention over time or memorability, because only two users tested the system twice with a time gap in between. Furthermore, the first user from the field observation received an extra instruction again, because too many changes were applied to the prototype since the first test. The user from the second field observation however, used the system only with his prior knowledge. Showing that a time gap of 2 months is no problem for an expert user.

Changes

In this section the applied changes suggested by the user evaluation are described. According to the suggestions from the first field observation, extra buttons to set the lecture state to 'before', 'after' and 'offline' were introduced. Additionally, the automated set lecture state to 'during' was removed, because this also activates questions when only testing. This means that if a student was subscribed to the lecture, this student could answer the questions even through the lecturer just wanted to check his slides before the lecture. Therefore, also a button for on-air was introduced. Only the light weight version still sets the lecture state to 'during' when the presentation mode is started. The functionality is kept in the light weight version, because it is assumed that the add-in is only used in the actual lecture in this case. The adjusted ribbon menu is shown in Figure 6.1.

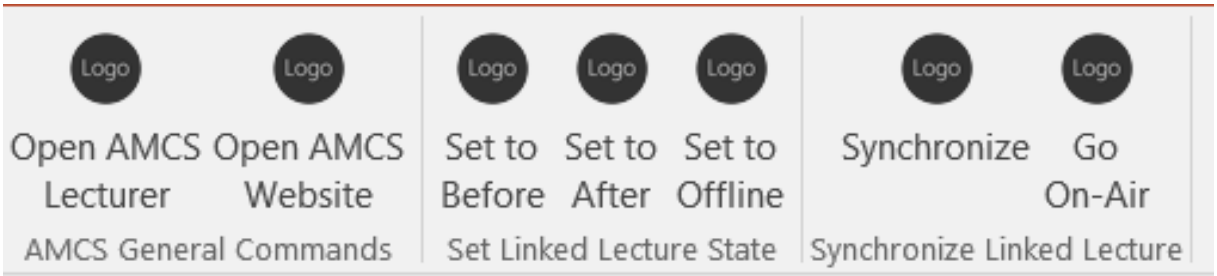
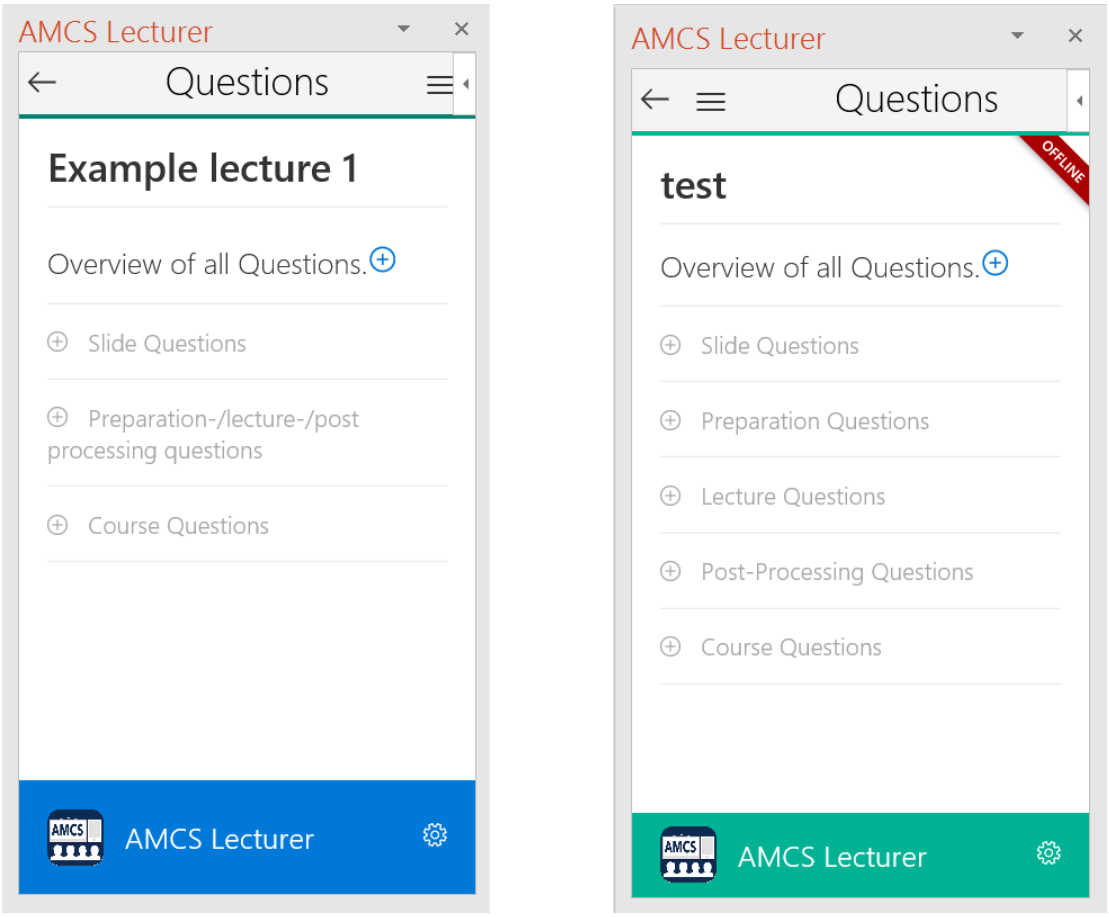


Figure 6.1: The new ribbon menu with set state, synchronize and go on-air buttons.

The second field observation encountered an error when using the add-in with animations on the slides. To fix this issue the add-in now only sends the active slide index if it is greater than 0.



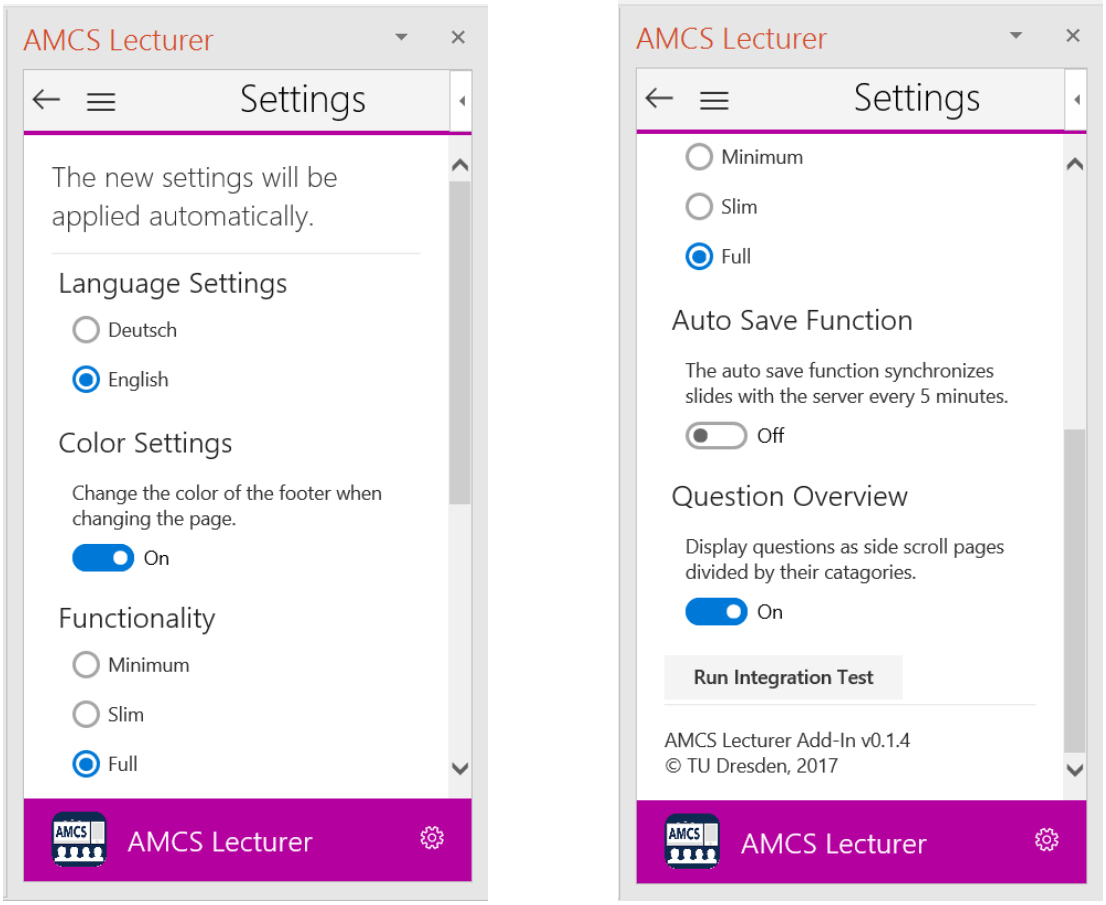
(a) The initial design for the question categories with three categories.

(b) The revised question list with five categories and the current lecture state.

Figure 6.2: The question view.

Test user number 1 suggested to use five question categories in the question list, like in the question creation dialog. The new question list with five categories is shown in Figure 6.2b. The image also shows the initial question list under 6.2a. The question list has also been extended by the current lecture state shown in the upper right corner below the menu.
Furthermore, the menu button was moved to the left, because under Mac on the top right is a big info button, which is inserted by PowerPoint. The info button position is exactly the same as the menu buttons initial position.

Optional Settings The adaptability was widely appreciated by the test users and further adaptability was suggested. Therefore, more adaptation options were introduced. The settings view with all possible settings is shown in Figure 6.3. The Figure 6.3a shows the language, color and functionality settings. The color option activates the footer color changes. The functionality setting now also provides the Slim version. In Figure 6.3b the auto save, question overview and integration test button are shown. If the question view is on then the questions will be presented in the side scroll view. The 'Run Integration Test' button is only displayed in debug mode.



(a) Language, Color and Functionality setting. With the additional slim functionality option.

(b) The newly added Auto Save Function, Question Overview located horizontally and Run Integration Button only displayed in debug.

Figure 6.3: The settings view.

The footer was changed as shown in Figure 6.2a. The users suggested to use the color for the page in the whole header and not just a small line below. However, this needs too much changes, because the menu and back button would need to be changed as well. To make the navigation more intuitive the footer color can be changed now. This can be seen in Figure 6.2b. In the same Figure 6.2a the old footer is displayed. It had the same blue color on every page. The footer height was reduced according to user comments, which stated that too much scrolling was required and that the footer looked too heavy.

The synchronization has been mainly reduced to the synchronization button, because for auto synchronization the current slide needs to be accessed to retrieve the index and id. However, this causes a loading circle, because the accesses is done through an AJAX request and many test users complained about the loading circle. Hence, only the auto-save option stayed. When this option is selected the add-in will synchronize every 5 minutes, but only when the user uses the add-in. To determine the add-in usage by the user the same method as before is used: when the courser enters the add-in and the time is over, then a synchronization will be done.

Sidewards scrollable question categories were introduced. Test user number 3 suggested to use horizontal located question categories, because the user found the scrolling inconvenient. The participant suggested to use the same or similar way as for the 'Questions per Slide' view. Therefore, a second menu bar was introduced, which has a previous and next category button, a list to go directly to a specific category and also the create question button. In Figure 6.4 the new horizontal question category layout is shown. In the image the Course Questions are selected. Since the questions are now just a list and not a sublist of the categories list, the width has been extended, which gives a better overview; especially, when using long question titles and descriptions.

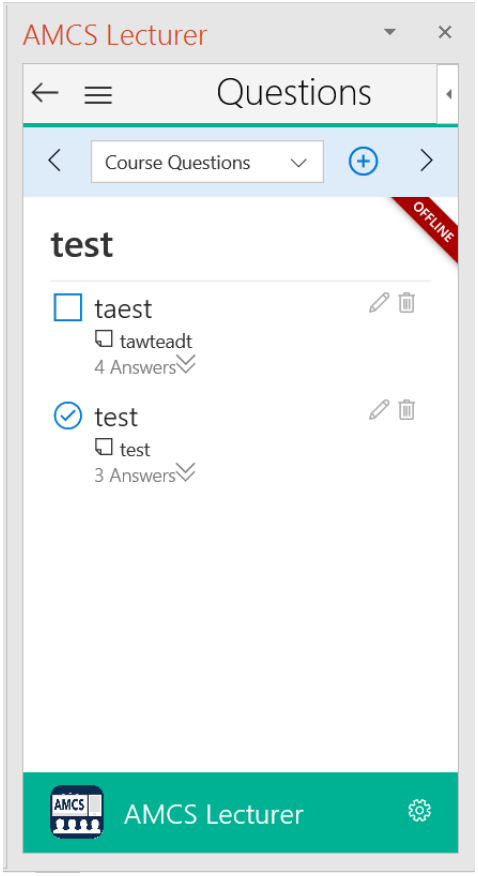


Figure 6.4: Horizontal question category layout.

The medium version was also suggested by participant number 3, because this user felt there were too many unnecessary information. However, the light weight version didn't have enough functionality in the users opinion. Therefore, a third version was introduced: the Slim version. It can be activated in the settings under functionality. The slim version is equivalent to the Full version until the presentation is linked to a lecture. After a successful link the course overview won't be accessible anymore and the lecture list will only show the lecture to which the presentation is linked.

6.2 Requirements and Acceptance Criteria Evaluation

In this section the satisfaction of the requirements and acceptance criteria will be evaluated. First the software test results will be given, including the unit and integration tests. Secondly, the satisfaction of the acceptance tests will be shown. Thus, proving the satisfaction of the acceptance criteria, which were to satisfy the unit tests and user scenarios. The functional requirements are implied in the user scenarios.

6.2.1 Software Tests

In this section the software tests are presented. The model part has been completely unit tested and an integration test was provided to test the most critical system functions.

Unit Tests

The unit tests are testing the functions of a specific class in isolation. A list of all unit tests can be found in the appendix B.2.1. As specified in the requirements the test coverage is above 90% as shown in

Figure 6.5.

Blanket.js results	Covered/Total Smts.	Coverage (%)
1. https://localhost:44368/PresentationModel/Model/Question.js	37/37	100 %
2. https://localhost:44368/PresentationModel/Model/HasQuestions.js	69/69	100 %
3. https://localhost:44368/PresentationModel/Model/Course.js	37/37	100 %
4. https://localhost:44368/PresentationModel/Model/Lecture.js	23/23	100 %
5. https://localhost:44368/PresentationModel/Model/Slide.js	49/49	100 %
6. https://localhost:44368/PresentationModel/Model/SlideSet.js	106/106	100 %
Global total	321/321	100 %

Figure 6.5: Test coverage results for the unit tests.

Integration Test

The objective of the integration test is to show that all separate parts of the system work together. In the case of this work, this includes the different parts of the software (handlers, views and model) and the interaction with the web service and PowerPoint. However, because of the technical issues described in section 5.3.1 the main focus was to automatically test as many features of the acceptance criteria as possible rather than achieving 100% code coverage. In the end the overall coverage achieved by the integration test is almost 70% as shown in Figure 6.6. A list of all test cases and results for every tested file can be found in the appendix B.2.2.

Global total	650/930	69.89 %

Figure 6.6: Overall test coverage result for the integration test.

6.2.2 Acceptance Test

In BDD the acceptance criteria should be executable through the acceptance tests. In the ideal case the acceptance tests would be all automatically executable. However, there are parts of the system that have to be tested manually. Table 6.11 shows all 18 acceptance tests derived from the functional requirements. Furthermore, it specifies the method and gives additional information about the test execution. Finally, the last column shows the test result. If the result is '✓', then the test was passed. Otherwise the result will be 'x'.

Nr	Scenario Title	Method	Comment	Result
1.	System running on Windows	semi automated test	Need to insert the add-in manually, but then the integration test can be run, which tests the requirements.	✓
2.	System running on Mac	semi automated test	Need to insert the add-in manually, but then the integration test can be run, which tests the requirements.	✓
3.	Sign-In	manual test	Integration tests can only be run if logged in.	✓
4.	Sign-in with pin	manual test	Usual sign-in with pin, but then directly forwarded to lecture view.	✓
5	Linking a presentation with a lecture	automated test	Included in integration test case: Navigator - should link lecture to presentation.	✓
6.	Control lecture from PowerPoint	manual test	When changing the active slide in presentation mode in PowerPoint the active slide on the server is changed. This can be checked by using the website.	✓
7.	Control PowerPoint presentation from server	manual test	When changing the active slide on the website the active slide in PowerPoint is changed as well.	✓
8.	Course, lecture and question lists	automated test	Included in integration test cases: 1) Navigator - should open course overview when requested. 2) Navigator - should load lectures for course and show lecture overview when requested. 3) Question Views - should load questions and show question overview.	✓
9.	Question creation	automated test	Included in integration test cases: Question Dialogs - 1) should create a new slide question. 2) should create a new before lecture question. 3) should create a new course question.	✓
10.	Question editing	automated test	Included in integration test cases: Question Dialogs - should edit a slide question.	✓
11.	Persisting changes on the server	automated test	Included in integration test cases: implicit in all tests that modify server data. The tests fail if the server response is not positive.	✓
12.	Creating and editing lectures	automated test	Included in integration test cases: both Lecture Dialogs tests.	✓
13.	Unlink lecture and presentation	automated test	Included in integration test cases: The after hook of the integration test unlinkes the presentation and the lecture.	✓
14.	Light weight version	automated test	Included in integration test cases: Navigator - should be able to set the version to light weight (minimal) or full.	✓
15.	Set lecture state	automated test	Included in integration test cases: Navigator - should set lecture state.	✓
16.	Language option	automated test	Included in integration test cases: Navigator - should be able to set language to German or English.	✓
17.	Synchronization feedback	automated test	Included in integration test case: Navigator - should link lecture to presentation.	✓
18.	Lecture creation/edit and question creation/edit feedback	automated test	Included in the two Lecture Dialogs integration tests	✓

Table 6.11: Satisfaction evaluation of all 18 acceptance test cases.

7 Summary and Conclusion

The Goal of this work was to improve the lecturers work-flow when using AMCS. First of all, the background was given, explaining ARS and analyzing different existing systems. In the analysis, five systems were examined in total. The systems can be divided into two groups: 1) VSTO add-ins and 2) add-ins using the JavaScript API for Office. The comparison showed that VSTO add-ins integrate into PowerPoint nicely and inserting questions by inserting new slides is very intuitive and comfortable. A drawback is that the systems mainly used dialogs for interaction. This means that every button in the ribbon opens a new dialog. Furthermore, questions are only on the slides, hence, there is no central place to see all created questions in the add-in. One system offered a list of all questions, but in a new dialog window, which overlaps the slides in PowerPoint and blocks all further interaction until the dialog window is closed. Furthermore, the VSTO add-ins look out-dated and don't fit in the new PowerPoint style that well. And finally, VSTO add-ins can only be used on Windows and have to be installed via an executable.

The second group of add-ins are using the JavaScript API for Office. The API offers two types of add-ins: 1) taskpane add-ins and 2) content add-ins. Taskpane add-ins are run in a separate taskpane and are docked in PowerPoint on the right side. The content add-ins on the other hand are inserted on a particular slide. The examined systems all use the content add-in type. Since they are inserted on a specific slide, it is also not possible to have a constant overview of all questions. However, the add-ins fit much better into PowerPoint, when speaking of style. Adding a new add-in on every slide where a question should be is a drawback, but it is inevitable, because the JavaScript API doesn't provide the possibility to insert new slides in the slide set yet.

Only one of the examined systems provided the possibility to have more than one active question on a slide. This is definitely a disadvantage compared to AMCS, which offers this feature. It is desirable to have a question block and not just ask one question at a time, because it is less time consuming. Furthermore, none of the systems offered the possibility to control the PowerPoint slide set from the website.

The existing PowerPoint add-in of AMCS was also a VSTO add-in, but in contrary to the other examined VSTO add-ins it used a separate taskpane, each to show the question overview and create new questions. The existing lecturer add-in uses Windows Forms, and hence, doesn't fit into the new PowerPoint style that well. Furthermore, the system is only available on Windows and needs to be installed via an executable, as all VSTO add-ins.

The JavaScript API for Office on the other hand provides an easy way to distribute the add-in and offers platform independence. An add-in using the JavaScript API is distributed using an XML-file that describes the add-in and must be hosted as a services. The add-ins are run using an IE11 iFrame. Unfortunately, the IE11 technology is rather out-dated and doesn't support new JavaSript features. However, especially the platform independence and easy distribution lead to the selection of the new technology for the new lecturer add-in.

After discussing state of the art web technologies including widely used frameworks and libraries, architectural design patterns that improve testability were discussed. Concluding from the analysis the requirements for a new lecturer add-in were specified. From the functional requirements user stories and scenarios were derived leading to acceptance criteria formulation, following the BDD practice.

According to the requirements, a concept for the lecturer add-in was created. The suggested prototype uses the JavaScript API for Office and is implemented as a taskpane add-in. As architectural design pattern the Presentation Model was chosen, because it increases unit testability by decoupling domain logic from view components. The View doesn't hold any states. All requests or user inputs are passed to a mapper, which connects the Presentation Model with the View. An important part of the conceptual phase was to find a good work-flow. In order to achieve that, use cases for the most important features were examined and UI-navigational paths were specified. While creating the concept, seamless integration was considered. Especially, when creating the navigational paths, because the user journey is an important step to achieve seamless integration. However, one of the most important parts of the concept is the synchronization, because a badly implemented synchronization could affect the usability very negatively and destroy the work-flow. Three options for synchronization were discussed: 1) frequent automated synchronization; 2) smart automated synchronization; and 3) manual synchronization. For the prototype option 2) and 3) were chosen to support adaptability and check the usability of the automated synchronization. Finally, different test practices and types of tests were described. The project follows the BDD practice, which means that the tests are testing the behavior of system components according to the specifications. Furthermore, the concepts for unit and integration testing were described. The unit tests are used to provide coverage of the model part of the Representation Model, which is an acceptance criteria. However, most of the acceptance tests need interaction with PowerPoint or the server and are hence part of the integration test.

In the implementation chapter the actual realization of the before described concept was shown. The Presentation Model was split into handlers and model classes. The handlers cover all user interaction and server communication. In this way the model components can be tested in isolation. For each page a view was implemented and views communicate with the Presentation Model through a mapper. The dialogs were built according to the websites dialogs using the dialog API provided by Microsoft. For the synchronization two methods were implemented: 1) the smart auto-synchronization and 2) the manual synchronization.

The tests were implemented using Mocha as test framework and Chai for assertions in the expect BDD style. The integration tests were integrated into the add-in, because they have to be run in PowerPoint. To evaluate the code coverage Blanketjs was used.

Via a survey, the usability of the system was evaluated. Participants had to solve a given task, and then, answer a questionnaire. After that, they could use the system shortly and where asked some questions in an interview to clarify some answers they gave. While using the system, participants were instructed to think aloud, to get additional feedback and an insight on user behavior. The goal was to investigate the main components of usability: learnability, efficiency, memorability, error rate and satisfaction. However, the main focus was learnability and user satisfaction and acceptance. In general, the usability was rated mostly positive and users were satisfied with the functionality. Especially, the UI design was appreciated. However, participants felt that the learning curve was rather steep, because the system seemed complex to novice users. Furthermore, through the users were satisfied with the core functionality, which

is creating questions and automating the slide changes, most participants asked for the possibility to integrate questions and question results on the slides. Thus showing, that the users are missing some functionality. Nonetheless, participants stated that the add-in improves the work-flow.
The second test was a field observation. It was done with two participants who both used the system to hold an actual lecture. Both tests identified smaller problems, which were fixed afterwards.
Concluding from the survey results and the field observations the prototype was changed. The most significant changes were the set lecture state, the introduction of a medium version and the changes for synchronization. The automated lecture state change was removed, because it lead to unexpected behavior. The medium version was introduced to give the users an easier overview and ease the learning effort. Finally, the auto-synchronization was removed, because it produced a loading circle, which participants found annoying.
Finally, the satisfaction of the requirements was shown by providing the unit, integration and acceptance tests. Most of the acceptance tests were evaluated as part of the integration test, but some had to be run manually.
The main goal of the work was to provide more ease of use for the lecturer by creating the integration of the system into PowerPoint. To achieve that, a good work-flow and a concept with regards of seamless integration into the existing system were elaborated. The new lecturer add-in achieves better integration into the system than the old one by using very similar looks and navigational structures. Furthermore, the newly used JavaScript API for Office now gives the possibility to use the system on Mac. Additionally, the distribution of the add-in is much easier. As shown by the results, the users were generally satisfied with the existing features of the prototype.
However, most of the participants of the evaluation asked for a feature extension, which allows to integrate questions on slides. As discussed, currently there is no good way to insert data on slides. Especially, to insert the question results, because no HTML coercion type data can be inserted on the slides. Therefore, in future works the question integration on the slide should be done. Unless the JavaScript API will provide a way to insert new slides or HTML content, the best option would be to use a content add-in. The content add-in should be designed to work seamlessly with the system implemented in this work. This can be achieved by using the property bag and the localStorage to share common data, like the login or lectureId of the linked lecture. Furthermore, the dialogs can be reused. Especially, the question dialog, so new questions can be created directly in the content add-in.

Bibliography

[1] Thirumalesh Bhat and Nachiappan Nagappan. Evaluating the efficacy of test-driven development: industrial case studies. In *Proceedings of the 2006 ACM/IEEE international symposium on Empirical software engineering*, pages 356–363. ACM, 2006.

[2] Facebook. React.
`https://facebook.github.io/react/`, [accessed: 15.08.2017].

[3] Xavier Ferré, Natalia Juristo, Helmut Windl, and Larry Constantine. Usability basics for software developers. *IEEE software*, 18(1):22–29, 2001.

[4] JS Foundation and contributors. Mochajs.
`https://mochajs.org/`, [accessed: 23.08.2017].

[5] Martin Fowler. *Patterns of Enterprise Application Architecture*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2002.

[6] Martin Fowler. Presentation model.
`https://martinfowler.com/eaaDev/PresentationModel.html`, [accessed: 24.07.2017].

[7] Andreas Holzinger. Usability engineering methods for software developers. *Communications of the ACM*, 48(1):71–74, 2005.

[8] Bianca Ignacio. 10 tips for a seamless user journey through ux web design.
`http://www.aumcore.com/blog/2017/02/04/10-tips-for-a-seamless-user-journey-t`
`#`, [accessed: 14.08.2017].

[9] Google Inc. Angular.
`https://angular.io/`, [accessed: 15.08.2017].

[10] Ecma International. Ecma javascript 5th edition.
`http://www.ecma-international.org/news/PressReleases/PR_Ecma_finalises_major_revision_of_ECMAScript.htm`, [accessed: 18.07.2017].

[11] David Janzen and Hossein Saiedian. Test-driven development concepts, taxonomy, and future direction. *Computer*, 38(9):43–50, 2005.

[12] jQuery Foundation. jquery.
`https://jquery.com/`, [accessed: 15.08.2017].

[13] kangax on github. Ecmascript support 6th edition.
`http://www.ecma-international.org/news/PressReleases/PR_Ecma_finalises_major_revision_of_ECMAScript.htm`, [accessed: 18.07.2017].

[14] Robin H Kay and Ann LeSage. Examining the benefits and challenges of using audience response systems: A review of the literature. *Computers & Education*, 53(3):819–827, 2009.

[15] Flaherty Kim. Seamlessness in the omnichannel user experience. https://www.nngroup.com/articles/seamless-cross-channel/, [accessed: 14.08.2017].

[16] knockoutjs.com. Knockoutjs. http://knockoutjs.com/, [accessed: 13.10.2017].

[17] Ioan Lazăr, Simona Motogna, and Bazil Pârv. Behaviour-driven development of foundational uml components. *Electronic Notes in Theoretical Computer Science*, 264(1):91–105, 2010.

[18] et al. Mark Otto, Jacob Thornton. Angular. http://getbootstrap.com/, [accessed: 15.08.2017].

[19] Mentimeter. Interactive presentations, workshops and meetings. https://www.mentimeter.com/, [accessed: 18.07.2017].

[20] Microsoft. Office ui fabric. http://dev.office.com/fabric#/, [accessed: 15.08.2017].

[21] Microsoft. Docs for powerpoint add-ins. https://dev.office.com/docs/add-ins/powerpoint/powerpoint-add-ins, [accessed: 18.07.2017].

[22] Microsoft. Get the whole document from an add-in for powerpoint or word. https://dev.office.com/docs/add-ins/develop/get-the-whole-document-from-an-add-in-for-powerpoint-or-word, [accessed: 19.07.2017].

[23] Microsoft. Persisting add-in state and settings. https://dev.office.com/docs/add-ins/develop/persisting-add-in-state-and-settings#how-to-save-add-in-state-and-settings-p [accessed: 19.07.2017].

[24] Microsoft. Read and write data to the active selection in a document or spreadsheet. https://dev.office.com/docs/add-ins/develop/read-and-write-data-to-the-active-selection-in-a-document-or-spreadsheet, [accessed: 19.07.2017].

[25] Dan North. Introducing bdd. http://dannorth.net/introducing-bdd, [accessed: 01.09.2017].

[26] Open Source Project on GitHub. Chai assertion library. http://chaijs.com/, [accessed: 23.08.2017].

[27] Antje Schubotz. *Entwicklung einer Lecturer App zur Kopplung einer Präsentationssoftware mit der ARS-Plattform AMCS*. Master's thesis, TU Dresden, 2016.

[28] Alex Seville. Blanketjs.
http://blanketjs.org/, [accessed: 23.08.2017].

[29] Carlos Solis and Xiaofeng Wang. A study of the characteristics of behaviour driven development. In *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*, pages 383–387. IEEE, 2011.

[30] Erwin van der Valk. the difference between model-view-viewmodel and other separated presentation patterns.
https://blogs.msdn.microsoft.com/erwinvandervalk/2009/08/14/the-difference-between-model-view-viewmodel-and-other-separated-presentation
[accessed: 18.07.2017].

[31] Laurie Williams, E Michael Maximilien, and Mladen Vouk. Test-driven development as a defect-reduction practice. In *Software Reliability Engineering, 2003. ISSRE 2003. 14th International Symposium on*, pages 34–45. IEEE, 2003.

List of Figures

List of Tables

A Acronyms

AJAX Asynchronous JavaScript and XML 58

AMCS Auditorium Mobile Classroom Service 5

ARS audience response systems 5

ATDD Acceptance Test Driven Development 34

BDD Behavior Driven Development 34

DOM Document Object Model 31

HTTP Hypertext Transfer Protocol 24

IE Internet Explorer 26

TDD Test Driven Development 33

UI User Interface 29

UX User Experience 23

VSTO Visual Studio Tools for Office 5

XML eXtensible Markup Language 6

XP Extreme Programming 33

B Evaluation

B.1 Evaluation Survey

Aufgabe

1. Öffnen Sie das Add-In.
2. Loggen Sie sich mit den folgenden Login-Daten ein: (Name, Passwort) (lecturer, lecturer)
3. Verschaffen Sie sich einen Überblick über die bestehenden Veranstaltungen des Kurses „Evaluationskurs“ und betrachten Sie die Fragenübersicht der Veranstaltung „Evaluation“.

4. Verlinken Sie die Präsentation mit der Veranstaltung „Evaluation“ aus der Liste.
5. Fügen Sie ihrer PPT-Präsentation eine weitere Folie am Ende hinzu.
6. Fügen Sie der 3. Folie eine Frage vom Typ „Mehrfachauswahl“ hinzu.
7. Navigieren Sie zu der „Fragen je Folie“-Übersicht.
8. Verschieben Sie in der PPT-Präsentation die 3. Folie an die 2. Position.
9. Schalten Sie in der „Fragen je Folie“-Übersicht wieder auf die dritte Folie.
10. Erstellen Sie eine Vorbereitungsfrage vom Typ „Skalenfrage“.
11. Loggen Sie sich aus.

12. Loggen Sie sich erneut mit den oben gegebenen Daten ein und benutzen Sie diesmal auch die PIN: eval123, um direkt zu der Fragenübersicht zu gelangen.
13. Starten Sie die Präsentation indem Sie F5 drücken.
14. Klicken Sie bis zum Ende der Präsentation und schließen diese mit ESC.
15. Navigieren Sie zur Fragenübersicht der verlinkten Veranstaltung.
16. Prüfen Sie, dass die Lehrveranstaltung auf „Danach“ steht.
17. Loggen Sie sich aus.

Bitte beantworten Sie den beiliegenden Fragebogen „Plattform-Feedback“.

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch“, ++ „vollkommene Zustimmung“ und o „Neutral“.

Datum der Versuchsdurchführung: 2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.					
2.	Die verschiedenen Funktionen des Systems sind gut integriert.					
3.	Das Add-In erleichtert die Arbeit mit AMCS.					
4.	Die Seitenführung ist intuitiv.					
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.					
6.	Das Menü bietet sinnvolle Shortcuts.					
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.					
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.					
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.					
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.					
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.					
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.					
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.					
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.					
15.	Das System ist leicht erlernbar.					
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.					
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.					

Interview-Fragen

Vor dem Test und den Plattform-Fragen

- Erfahrung mit AMCS?
- Wie lange?
- Erfahrung mit PPT-Add-Ins?
- Wie wird das System primär genutzt? Welche Funktionalitäten sind der befragten Person wichtig?

Nach dem Test und den Plattform-Fragen

- Was wurde als umständlich zu nutzen empfunden?
- Was wurde als positiv empfunden?
- Wo sind noch Verbesserungen möglich?
- Erweiterungswünsche?

Wenn nicht sowieso schon angegeben:

- Falls „nicht gut erlernbar angekreuzt“: Warum?
- Falls „nicht übersichtlich“: Warum?
- Falls „abgespeckt ausreichend“: Warum?
- Falls „14. Schlecht bewertet wurde“: Die Popup-Nachrichten sind hilfreich? Mehr als eine Popup-Nachricht ist zu viel? Die Nachrichtenanzeigedauer ist zu kurz?

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 14.08 .2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.				X	
2.	Die verschiedenen Funktionen des Systems sind gut integriert.		X			
3.	Das Add-In erleichtert die Arbeit mit AMCS.			X		
4.	Die Seitenführung ist intuitiv.				X	
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.		X			
6.	Das Menü bietet sinnvolle Shortcuts.			X		
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.		X			
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.	X				
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.					X
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.			X		
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.				X	
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.					X
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.				X	
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.				X	
15.	Das System ist leicht erlernbar.		X			
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.				X	
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.					X

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 06. 07. 2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.				X	
2.	Die verschiedenen Funktionen des Systems sind gut integriert.		X			
3.	Das Add-In erleichtert die Arbeit mit AMCS.					X
4.	Die Seitenführung ist intuitiv.			X		
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.				X	
6.	Das Menü bietet sinnvolle Shortcuts.			X		
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.		X			
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.				X	
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.	X				
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.				X	
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.					X
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.	X				
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.	X				
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.				X	
15.	Das System ist leicht erlernbar.		X			
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.				X	
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.		X			

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 12.07 .2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.			X		
2.	Die verschiedenen Funktionen des Systems sind gut integriert.			X		
3.	Das Add-In erleichtert die Arbeit mit AMCS.				X	
4.	Die Seitenführung ist intuitiv.				X	
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.				X	
6.	Das Menü bietet sinnvolle Shortcuts.		X			
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.			X		
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.		X			
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.	X				
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.			X		
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.				X	
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.		X			
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.	X				
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.				X	
15.	Das System ist leicht erlernbar.				X	
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.				X	
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.					X

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 7.7. 2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.				X	
2.	Die verschiedenen Funktionen des Systems sind gut integriert.				X	
3.	Das Add-In erleichtert die Arbeit mit AMCS.				X	
4.	Die Seitenführung ist intuitiv.				X	
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.				X	
6.	Das Menü bietet sinnvolle Shortcuts.				X	
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.		X			
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.				X	
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.				X	
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.				X	
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.				X	
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.	X				
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.	X				
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.			X		
15.	Das System ist leicht erlernbar.				X	
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.				X	
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.			X		

beide Versionen gewünscht

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 04.07. 2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.					11
2.	Die verschiedenen Funktionen des Systems sind gut integriert.					11
3.	Das Add-In erleichtert die Arbeit mit AMCS.					11
4.	Die Seitenführung ist intuitiv.				11	
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.				11	
6.	Das Menü bietet sinnvolle Shortcuts.					11
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.				11	
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.					11
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.			11		
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.				11	
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.				11	
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.		1			
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.		11			
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.		11			
15.	Das System ist leicht erlernbar.					11
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.					11
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.					11

Plattform-Feedback

Vielen Dank, dass Sie sich die Zeit nehmen, Ihr wertvolles Feedback zur Plattform zu geben. Bitte bearbeiten Sie die gegebene Testaufgabe und beantworten anschließend die untenstehenden Fragen.

Die Bewertungskriterien bedeuten: -- „vollkommener Widerspruch", ++ „vollkommene Zustimmung" und o „Neutral".

Datum der Versuchsdurchführung: 05.07.2017

		--	-	o	+	++
1.	Das System ist einfach zu nutzen.				X	
2.	Die verschiedenen Funktionen des Systems sind gut integriert.				X	
3.	Das Add-In erleichtert die Arbeit mit AMCS.					X
4.	Die Seitenführung ist intuitiv.				X	
5.	Die Farbgebung ist intuitiv und hilft bei der Orientierung.					X
6.	Das Menü bietet sinnvolle Shortcuts.			X		
7.	Die Verknüpfung von PPT-Foliensätzen mit dem System ist intuitiv.				X	
8.	Die Erstellung von Fragen integriert sich gut in den Prozess der Folienerstellung.					X
9.	Das Erstellen von Fragen sollte auch für nicht verlinkte Veranstaltungen verfügbar sein.	X				
10.	Das System ist übersichtlich und bietet dennoch die nötigen Informationen.				X	
11.	Die gegebenen Informationen in den Kurs-, Veranstaltungs- und Fragenübersichtsseiten sind ausreichend.				X	
12.	Bei der Fragenübersicht sollten standardmäßig alle Fragen beim Aufruf sichtbar sein.		X			
13.	Die Kursfragen sollten besser am Anfang der Veranstaltungsliste platziert sein.	X				
14.	Die in den Popup-Nachrichten bereitgestellten Informationen sind klar und bereichern die Nutzung des Systems.				X	
15.	Das System ist leicht erlernbar.			X		
16.	Erfahrung mit der AMCS-Website erleichtert die Bedienung des Add-Ins.					X
17.	Der Funktionsumfang der abgespeckten Version ist ausreichend.	X				

B.2 Test Cases

B.2.1 Unit Test Cases

passes: *53* failures: *0* duration: *1.18*s 100%

Question

when is slide question
- ✓ should have slide index.

when updating attribute
- ✓ should update attribute if attribute exist or do nothing.

choice
- ✓ should return choice at index.
- ✓ should sort choices after insert of new choice.
- ✓ should return the choice when deleted or undefined.

HasQuestions

when adding questions
- ✓ should have questions.
- ✓ and a new Question with the same position is inserted at the end, it will be pushed to the next position.
- ✓ and a new Question is inserted then the existing Questions will be reordered
- ✓ and a new Question with the same position is inserted at the end using addNewQuestionAt, it will be pushed to the next position.

when retrieving the last question position
- ✓ should return 1 when there are no questions.
- ✓ should return the last position when there are questions.

when question with ID is removed
- ✓ should remove the question and return it or return null if the question ID doesn't exist.

when question with ID is requested
- ✓ should return the question or null if the question ID doesn't exist.

Course

Constructor
- ✓ should set the course object
- ✓ should set lectures to not loaded yet
- ✓ should initialize list of lectures as empty array

Course Object Access
- ✓ should return the course name

- ✓ should return the course id.

Lectures
- ✓ should add lectures to _lecture array and set lectures loaded to true.
- ✓ should return the lecture with a specific id.
- ✓ should return null if a lecture with a specific id doesn't exist.
- ✓ should return the list of all lectures.
- ✓ should return the UI objects of all lectures.

passes: *53* failures: *0* duration: *1.18*s

Lecture

Constructor
- ✓ should set the lecture object
- ✓ should set lecture object
- ✓ should initialize the slideset

Lecture Object Access
- ✓ should update the lecture object.
- ✓ should set and get the current lecture states (0,1,2,3).

SlideSet

Constructor
- ✓ should set the slideSet id, name and url.
- ✓ should initialize slides.
- ✓ should initialize slidesLoaded to false.

slide
- ✓ when adding slides they should be added to the slides array in order and the slides should be loaded.
- ✓ when a slide is requested by index it should return null if the index is not in range.
- ✓ when slide is requested by id it should return null if id doesn't exist or the slide with the id.
- ✓ when slide is added at index the old slide at index should be exchanged with the new one.
- ✓ when removing a slide at index and the index is in range it should be returned and the slide positions of the following slides should be decreased.
- ✓ when removing a slide at index and the index is not range it should return null.
- ✓ when requesting a not linked slide it should return the first not linked slide.
- ✓ when slide is inserted at a specific index the slide should appear at the index and all the other slide indexes should be adjusted.
- ✓ when slide is inserted at a specific index and incrementServerIndex is true than the slide objects position attribute should be incremented as well.
- ✓ when moving a slide it should apear at the new index and the other slide indexes should be adjusted.
- ✓ when adding questions to slides the questions should be added to the slides and questionsLoaded should be true.

when slides are linked
- ✓ and requesting physical slides all linked slides should be returned.

✓ and requesting a physical slide at index the linked slide at the index should be returned or null if the index doesn't exist or the slide is not linked.
✓ and requesting a physical slide by ID the linked slide with the ID should be returned or null if the ID can not be found.
✓ and requesting a not linked slide it should return null.
✓ and moving a slide it should apear at the new index and the other slide indexes should be adjusted and the indexes at slide set should be adjusted as well.

passes: *53* failures: *0* duration: *1.18s*

Slide

when unlinked the Constructor

✓ should set the slide object, physical id, state and index at slide set.

server and physical indexes

✓ when index at server is changed it should set the new position in the question object.
✓ when index at slide set is incremented it should increment the index at slide set if the slide is linked and if serverIndexInc is true it should also increment the slide object position.
✓ when index are not synched and synch indexes is called then both server and physical index are the same.

when slide object is changed

✓ id should be different.

when slide state is changed

✓ should have new state.

Blanket.js results	Covered/Total Smts.	Coverage (%)
1. https://localhost:44368/PresentationModel/Model/Question.js	37/37	100 %
2. https://localhost:44368/PresentationModel/Model/HasQuestions.js	69/69	100 %
3. https://localhost:44368/PresentationModel/Model/Course.js	37/37	100 %
4. https://localhost:44368/PresentationModel/Model/Lecture.js	23/23	100 %
5. https://localhost:44368/PresentationModel/Model/Slide.js	49/49	100 %
6. https://localhost:44368/PresentationModel/Model/SlideSet.js	106/106	100 %
Global total	321/321	100 %

B.2.2 Integration Tests

passes: *29* failures: *0* duration: *20.82*s 100%

Integration Test

Navigator

- ✓ should have 4 registered handlers.
- ✓ dialog shouldn't be open and toast should be initialized.
- ✓ should be able to set language to German or English.
- ✓ should be able to set the version to light weight (minimal) or full.
- ✓ should initialize courses. 65ms
- ✓ should open course overview when requested.
- ✓ should load lectures for course and show lecture overview when requested. 93ms
- ✓ should link lecture to presentation. 192ms
- ✓ should set lecture state.

Synchronization

- ✓ should link a new slide and move it to the position if a new slide was discovered and unlinked slides exist.
- ✓ should create a new slide locally and on the server. 130ms
- ✓ should move a slide locally and on the server. 114ms
- ✓ should delete a removed slide locally and on the server. 113ms

Lecture Dialogs

- ✓ should create a new lecture. 225ms
- ✓ should update an existing lecture. 160ms

QuestionHandler should

- ✓ load all course questions. 280ms
- ✓ load all lecture questions. 411ms
- ✓ load all slide questions for every slide in the slideSet. 152ms

Question Dialogs

- ✓ should create a new slide question (of type single choice). 621ms
- ✓ should create a new before lecture question (of type multiple choice). 249ms
- ✓ should create a new course question (of type single choice with correct answer). 287ms
- ✓ should edit a slide question. 166ms
- ✓ should move a question from one slide to another. 151ms
- ✓ should change the question context from course to lecture. 178ms
- ✓ should delete a slide question.
- ✓ should delete a lecture question.
- ✓ should delete a course question.

Question Views

- ✓ should load questions and show question overview. 71ms
- ✓ should show question per slide overview 62ms

Blanket.js results	Covered/Total Smts.	Coverage (%)
1. https://localhost:44368/PresentationModel/HTTP.js		75.51 % 37/49
2. https://localhost:44368/PresentationModel/Handler/TimeEventHandler.js	2/15	13.33 %
3. https://localhost:44368/PresentationModel/Handler/SlideSetHandler.js		63.79 % 185/290
4. https://localhost:44368/PresentationModel/Handler/CourseHandler.js		86.27 % 44/51
5. https://localhost:44368/PresentationModel/Handler/LectureHandler.js		69.35 % 43/62
6. https://localhost:44368/PresentationModel/Handler/QuestionHandler.js	97/124	78.23 %
7. https://localhost:44368/PresentationModel/Navigator.js		71.39 % 242/339
Global total	650/930	69.89 %

Acknowledgments

Thanks to Tenshi Hara, Iris Braun and the rest of the AMCS team, who always supported me with feedback and helped me when I was stuck. My parents for letting me choose my own path and my friends who accompanied me on the way. And most importantly to my biggest supporter in Korea and probably also the world - Kkotsong. Arigatou minna-san and kamsahamnida.