Design document: Classic

Group 8

Maxime Fernández Alonso, Caroline De Brouwer,
Jorsi Grammens, Driek Struyve, Jan Vermeulen,
Juta Staes, Kevin Debeuf

February 13, 2016

Contents

Chapter 1

Product Vision

1.1 Target group

Our target group consists of students and their teachers. Since we realize that this is a very large group, we have opted to focus on computer science students pursuing a master's degree and our professors. However, this does not mean we will focus less on usability and user-friendliness. Should we succeed in our objectives for this target group, we can always broaden our horizons.

1.2 Needs

1. We believe this target group has certain unsatisfied needs. When students come across an unclear syllabus or lecture, they will (in most cases) try to understand and extend this syllabus by themselves. In our opinion, this work could be done more efficiently by crowd sourcing the syllabus and extensions of it.

2. Students often deal with long or hard commutes, which would be unnecessary when lectures could be accessed on demand. When lectures overlap, for students with a personalized schedule or elective courses, the lectures would still be accessible.

3. In class, it occurs very frequently to professors that students actually know the answer, but are too shy to answer. The amount of feedback a professor gets from a lecture is also very small.

4. The concentration level of a student during a lecture is typically not very high. When a teacher moves too fast and a student was not yet able to process the information, the concentration level can quickly become near zero due to the fact that the new information depends on previously explained information. Being able to pause a lecture or rewind it can make drastic improvements.

1.3 Product

The idea of the product is an overarching (web-based) application that contains a section for every course that contains all the information of a course. This information includes lectures which can be accessed on demand, the course syllabus, slides, etc.

To this course material, questions can be added in case something is not clear. These questions can be answered by colleagues, assistants or the professor. Annotations can also be added to course material to provide extra information on a certain subject to clarify a slide, piece of a lecture, ...

The essence and strength of the application actually lies in the uniform cross-referencing system. Often, course material is provided redundant. When a course video lecture, the course syllabus and the slides all provide information on the same subject, we want to be able to create a cross-reference which references all 3 of them. For example, when 10 minutes of a lecture is about virtual memory, the slides that were used during the lecture are uploaded and a course syllabus was also provided. If students would want to add information about this subject they should be able to cross-reference the relevant slides, syllabus and the part of the lecture to an annotation with this additional information. The same principle can be applied to questions. We would also provide a rating system for these questions, so the application would be able to provide the user watching a lecture with the most relevant cross-referenced questions.

1.4 Value

This application would provide the students with a dynamic way of navigating, extending and processing course information. By considering additional information provided by peers, students will develop more insight into the subject matter. Students would also be provided with a uniform communication channel. To encourage users to actually use this communication channel, comments and annotations should optionally be anonymous. We think that this is the major reason why students do not use the Minerva forum more frequently (more on this in section 1.5). Anonymous comments would also take away the social barrier for providing feedback on the course.

1.5 Competitors & alternatives

- **Khan Academy** is an excellent platform, however when processing the information of a certain course, it is very cumbersome to find every piece of information included in a course. It is also too limited for a lot of courses. The comment section does not include a timestamped cross-reference to the material.
- **Swivl** focuses more on the hardware necessary to capture a course, however in their marketing campaigns they announced to pursue the development of software to facilitate crowd-sourced learning.
- **Minerva** provides a uniform and mandatory platform for every course, however collaborated learning is not extensively supported and would only be possible via the Minerva forum, which is not anonymous.

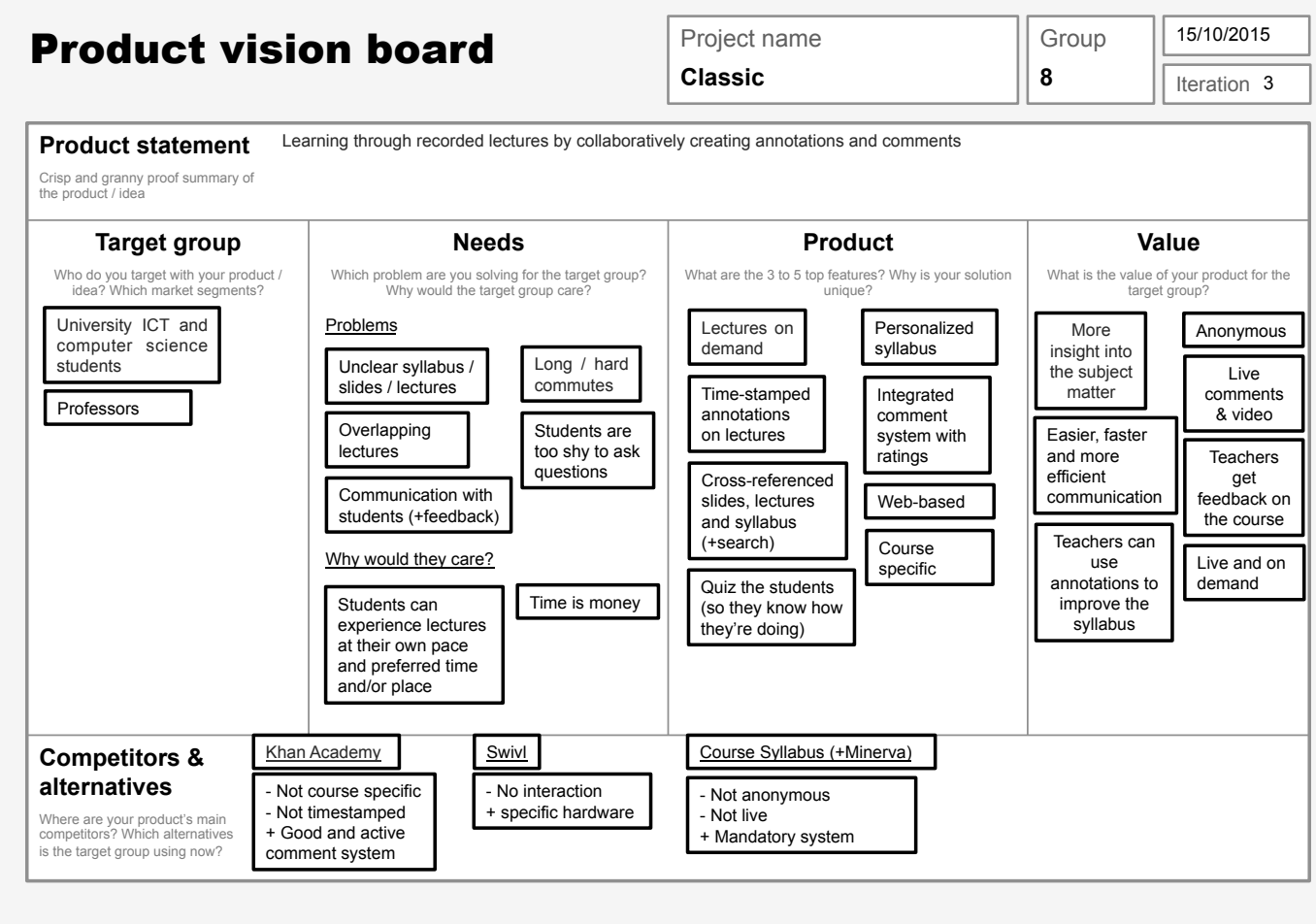


Figure 1.1: Product vision board: Classic

Chapter 2

State of the Art Analysis

2.1 Scientific research

A lot of research has been done in the field of annotation of video lectures. Most of this research is focused on the collaborative aspect. One example is The Collaborative Lecture Annotation System, or CLAS [Evan F. Risko, 2013]. This tool has been developed to extract important information out of lectures. It does this by relying on semantically constrained annotation. This restriction will not be implemented in our product. The user will be able to freely add annotations.

"One of the most popular and well-researched platforms for collaborative video annotation in educational contexts is the Microsoft Research Annotation System (MRAS). Briefly, this platform presented the lecturer, the slides, and annotations anchored to time points in the lecture. Annotations consisted of comments and threaded responses to those comments. An example of an annotation might be a question about the presented material at a given point in the lecture (e.g., "what does she mean when she says X?"). These annotations could be viewed and responded to by users. In addition to public annotations, there were also private notes users could store. Thus, through reading, responding, and posting annotations the platform provides a video annotation tool that users can use to collaboratively and asynchronously interact with one another." [David Bargeron, 1999] MRAS was only implemented as a prototype and never released to the public. This means that we can't build our application upon this platform. Our platform would be different from MRAS by using cross-references. These references are links between different course materials and a time point is not required.

"A similar and more recently developed tool, virtPresenter (now part of the OpenCast Matterhorn Project), represents a similar tool. This tool captures, processes, and distributes educational audio and video. It also provides tools

for students to engage with the material including in-video searching of text, bookmarking, and navigation. Included in the tool is a form of social navigation that allows users to traverse material by following traces caused by the actions of other users. Specifically, users can review the accumulated viewing history of all users (i.e., what parts of the lecture other users have viewed) and also compare their own viewing history with that trace." [Evan F. Risko, 2013] This tool is different from our platform as there is no annotation support. In our application, the students will be able to annotate themselves. They will also be able to ask questions and it will be possible to up-vote answers.

Other research concluded that most students do not use electronic devices to take notes. Most students prefer pen and paper [Steimle J., 2009]. This will have a very big impact on the project. A platform will have to be created that students want to use. The researchers of the paper build a concept and system called CoScribe. It makes use of an Anoto pen that automates the conversion of handwriting into digital data. Their design has been user validated which indicates that it efficiently supports student annotation and can be easily integrated into a system.

2.2 Industry trends

There are a lot of MOOC platforms and learning management systems out there, but none of them focus on the cross-linking of different course materials. Most of them offer a place where course material can be collected with a forum for students. Every university has a student platform like that, UGent has Minerva, KU Leuven has Toledo, VUB has PointCarr..

There are also MOOC platforms that offer courses through videos: a few examples are Coursera and CodeAcademy. Thinkful is a platform that offers more personal mentorship through online videos. Those, however, do not cooperate with certain schools or universities and usually don't offer much personal correspondence with the teachers or other students. Our application would be a supporting platform for an existing course at universities and colleges, not to create entirely new online courses.

Swivl is something that comes the closest to what we're trying to do, with the feature of time-stamped comments. However, their focus is more on the hardware to capture the lectures and present multimedia presentations, rather than the software to support it.

2.3 Standard bodies & professional organisations

2.3.1 Video encoding standards

There are several commonly used media encoding standards like "H.265/MPEG-H HEVC", "H.264/MPEG-4 AVC" and "H.263/MPEG-4" which we could choose for the lectures in our application. However we will be using the format that will be given to us by the challenger, since we will not provide the lectures ourselves. Thus, we will obviously not be choosing this standard ourselves.

2.3.2 Addressing Media

""Media Fragments URI 1.0" is a media-format independent standard that describes the syntax to refer to a media fragment with a unique URL and to describe the relevant timeframe, the relevant track and an optional restriction for the space dimension. The standard makes use of URI's (Uniform Resource Locator) to specify the location of a media object. The Media fragments standard consists of "URI fragments" and "URI queries". A URI fragment provides extra information about a media resource, namely spatial, temporal and track info. A URI query restricts the media resource with track, temporal and spatial information, in order to create a new resource. Depending on the application, one should be chosen or both should be combined.
For our intentions, Media Fragments could be used for defining the cross-references to lectures. Multiple video-sharing websites like YouTube make use of the media fragments standard.

2.3.3 Media annotations

For the annotation of audio and video, only several standards are available. To the best of our knowledge, we believe these only present itself in the context of web applications. This is a very recent topic gaining popularity with the semantic web. The W3C describes "SMIL", "Media annotations" and "Timed Text". We will not elaborate on "Timed Text" and "SMIL", since these standards have the objective of describing information about certain frames inside the video. For example, if you would like to add subtitles to a video, this could be done with "SMIL" or "Timed text". This does not comply with our application.

The W3C media annotations working group proposed "Ontology for Media Resources 1.0". The purpose of this standard is to create a standard language to describe media annotations for the web. The mapping to the media resource is independent to the media format. By defining such a standard, the working group tries to enable applications to retrieve standardized information. This way, search engines and other applications can filter and navigate through foreign media data.
This standard does not completely align with the idea we have of our applica-

tion, since we wish to describe annotations about multiple data formats at once, making use of multiple cross-references with only one actual annotation.

2.3.4 Portable document format

For the representation of our syllabi, PDF 2.0 will be used, since this is the most common format for course syllabi. PDF 2.0 describes two types of metadata, namely "Document Information Dictionary" and "Extensible Metadata Platform". The "Document Information Dictionary" can be used to add key/value fields to the whole document, like the title, author, date, etc. This is the least interesting of the two for our application. "Extensible Metadata Platform" is an ISO standard conforming to the XML standard and is embedded within the PDF 2.0 Document. The metadata can be attached to any document stream. We will probably not use these metadata standards, for the same reason as described in 2.3.3, namely that the annotations must be referenced to multiple media formats at once.

2.4 Intellectual property rights

When building a new application it is very important to check if you are not using any patented things, such as software and algorithms. Luckily these days there are a lot of open-source alternatives all over the internet.

2.4.1 Video Annotation System

Annotating a video in different ways is something we will surely need to do and it obviously has been done before. Be it through highlighting a time-span or drawing a box around a certain part of the video. Of course in our application we would simply like to highlight a time-span and let the user add some text, including cross-references, to that certain time-span of the video.
There is currently only one patent concerning this type of annotating a video, which is patent No. EP 2248288 (A2). It's a patent called 'ANNOTATING VIDEO INTERVALS' by Google Inc. (`http://www.google.com/patents/EP2248288A2?cl=zh`).

Even though there is a patented way of doing this, there are tons of alternatives. There is a group at the University of Harvard that study the usage of annotations (`http://www.annotations.harvard.edu/`). In their chapter about video, they show the different ways of annotating video as well as some of the best open-source alternatives. One of these open-source alternatives is `http://www.openvideoannotation.org/` which is perfect for our application.

2.4.2 Displaying Different Files in the Application

A very good piece of software is the HTML 5 document viewer: GroupDocs.Viewer (`http://groupdocs.com/html5-document-viewer`), but to use it to develop an

application, you have to buy the software.
An open-source alternative is ViewerJS (`http://viewerjs.org/`). Its GitHub repository is shared on the website and you can freely fork it.

2.5 Useful technologies and frameworks

2.5.1 SCORM

Sharable Content Object Reference Model (SCORM) is a set of technical standards for e-learning software products. SCORM gives programmers a standard to follow when writing their code so that it can interoperable with other e-learning software. SCORM governs how online learning content and Learning Management Systems (LMSs) communicate with each other. The standard does not speak to instructional design or any other pedagogical concern, it is purely a technical standard. In our application we will make use of SCORM to be interoperable with other learning platforms. SCORM can be fitted into the application using the SCORM driver. This driver can convert content to be SCORM compliant. Note that the SCORM driver is only free to use for non-commercial use.

2.5.2 Tin Can

The Tin Can API (sometimes known as the Experience API or xAPI) is called the next generation of SCORM. Tin Can differs from SCORM in that it adds the possibility to collect data about the wide range of experiences a person has. This API captures data in a consistent format about a person or groups activities from many technologies. Tin Can works by sending statements whenever an interaction with the content needs to be recorded. Simple Tin Can statements take the general form of "[actor] [verb] [object]", for example "Anna watched math lecture 3". All of the statements made are stored in the Learning Record Stores (LRS). This API is free to implement and can be used in our application to provide teachers with statistics on how students use their course. The previously described SCORM driver also has Tin CAN support.

Chapter 3

Product requirements

3.1 Use case list

id	Use cases	Priority	Complexity	Frequency
1	Add cross-reference	High	High	High
2	Add (cross-referenced) questions + answers	High	Medium	High
3	Add (cross-referenced) annotation	High	High	High
4	Open and synchronize course	Medium	Medium	High
5	Create a course	Medium	Low	Low
6	Upload a lecture	Medium	Low	Medium
7	Upload course material	Medium	Low	Low
8	Search engine	Low	Medium	High
9	Privacy settings for lectures	Low	Low	Low

The use cases "Add (cross-referenced) annotation", "Add (cross-referenced) questions + answers" and "Add cross-reference" form the basis of our application. This is what we believe will make it actually useful. With this in mind, it is obvious that all three receive the frequency and Priority "High". Since it is quite cumbersome to be able to cross-reference the content of multiple documents of different media formats, "Add cross-reference" receives the complexity "High". Annotating these different media formats is also quite complex. We estimate that adding questions and answers will be a little easier, so that received complexity "Medium".
"Create a course", "Upload a lecture" and "Upload course material" are use cases of the second most important group. They received Medium Priority. The complexity of these actions will be limited, since this will probably only consist of database manipulation. If this application would be used in a university similar to Ghent University, the frequency of uploading a lecture would be once or twice a week. We assigned it a medium frequency. Creating a course and uploading a syllabus received low frequency.
Our "Search engine" and "Privacy settings for lectures" have a low Priority

assigned, since these are only accessory use cases. They respectively received a medium and low complexity, a High and low frequency.
The last use case "Open and synchronize course" was added to the list of use cases because we had no use cases in which an external source (ex. SCORM), and thus the adapter package, was used.

3.2 Use case diagram

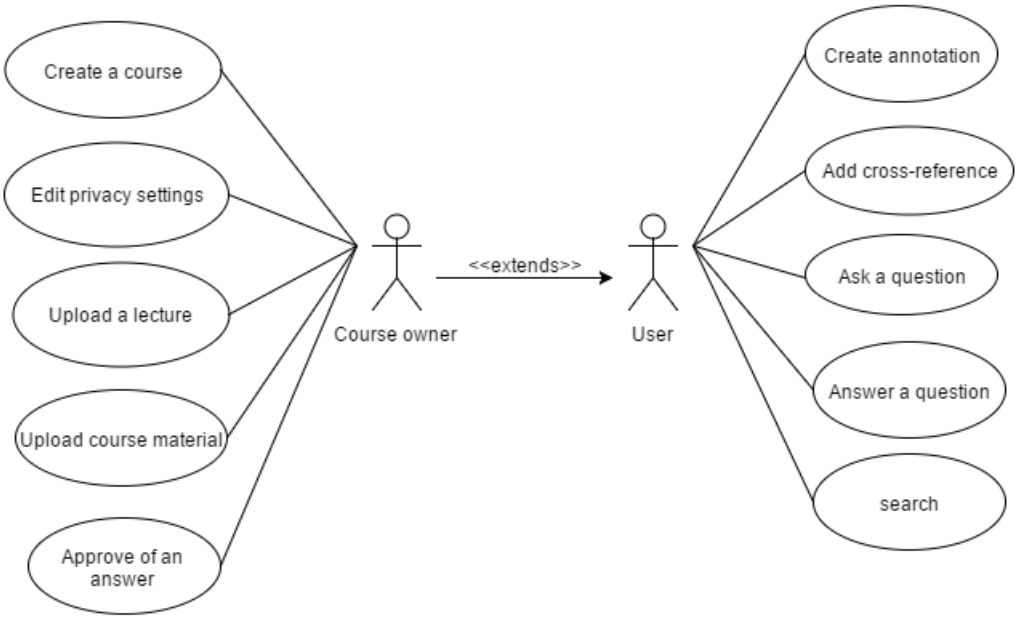


Figure 3.1: Use case bubble diagram

3.3 Quality attributes

The two most important quality attributes for the project are usability and interoperability. First of all, the platform needs to be easy to use. Using the platform cannot feel like it is a hassle. Such a program would not keep its user base, nor would people change to this platform. This quality attribute would be tested by asking real users about their experience of using the program. Secondly, the platform needs to be able to handle many different file formats. This was chosen because teachers and students all prefer different file formats. Before implementation is started, potential users would be queried about what file extensions they like to use.

3.4 Elaboration on elementary use cases

3.4.1 Add cross-reference

Adding cross-references is a big part of the application. Therefore, this process needs to be user-friendly. This would be tested by asking users if cross-references are easy to add.

3.4.2 Ask a Question

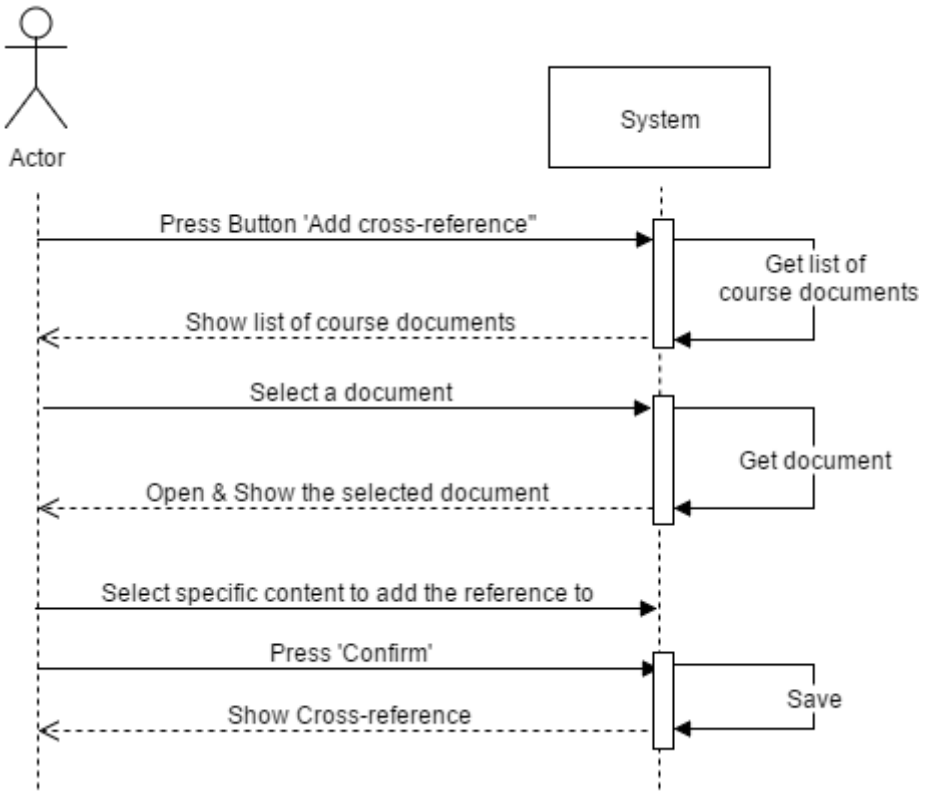
Adding a question must be very simple and should take at most 2 mouse clicks ("Ask question", "Post question").

3.4.3 Add Annotation

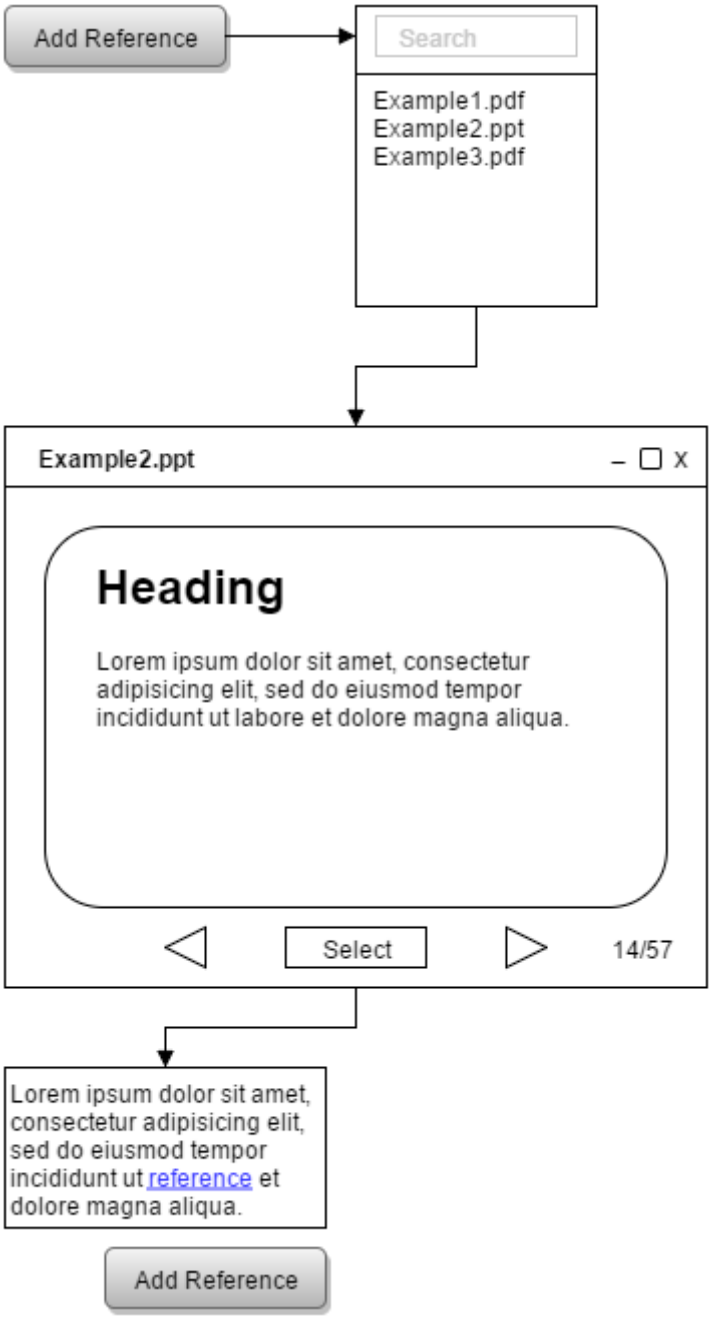
Adding an annotation must be very simple and should take at most 2 mouse clicks ("Add annotation", "Save annotation").

Use Case ID:	1		
Use Case Name:	Add cross-reference		
Created By:	Caroline	Last Updated By:	Caroline
Date Created:	28/10	Date Last Updated:	13/12

Description:	Adding a cross reference to an annotation, a question, an answer or a document.
Actors:	User, System
Trigger:	Press the “add cross reference” button.
Preconditions:	The user is looking at an annotation, question, answer or document. The user is logged in.
Postconditions:	The cross-reference has been added.
Normal Flow:	1.0.1 The user presses the “add cross reference” button. 1.0.2 The system shows a list of the different course materials. 1.0.3 The user selects one of them. 1.0.4 The system shows a more detailed view of the selected material. 1.0.5 The user selects the content he wants to reference. 1.0.6 The user presses the “Confirm” button. 1.0.7 The system confirms the cross-reference has been added by showing the document with the cross-reference.
Exceptions:	**1.0.E.1 The cross-reference cannot be saved.** 1.0.E.1.1 The system displays an error message. 1.0.E.1.2 The system redirects the user back to step 1.0.1.
Priority:	High
Frequency of Use:	High
Complexity:	High
Special Requirements:	Must be very simple to do, as this is the main focus of our application.
Assumptions:	We assume the app can work with a lot of different file types.



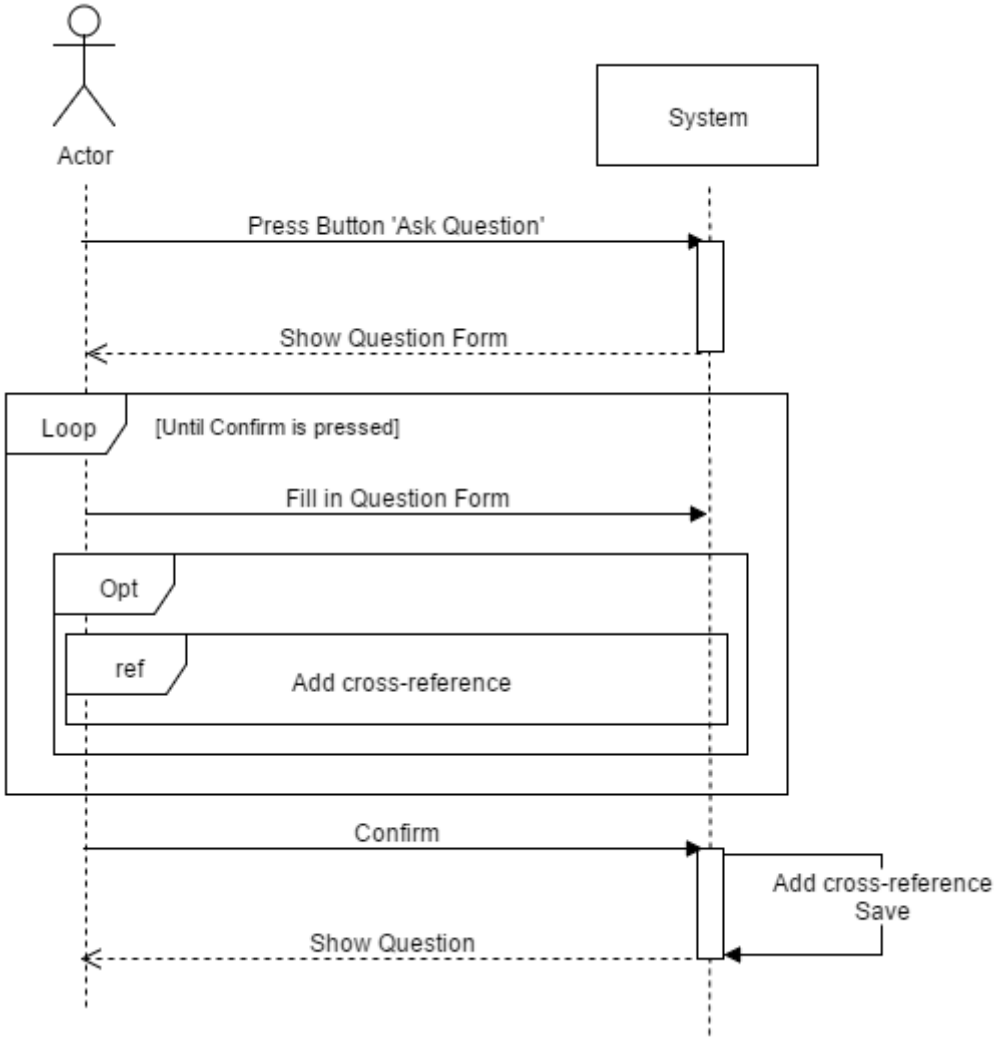
Black box diagram of use case 1: Add a cross-reference



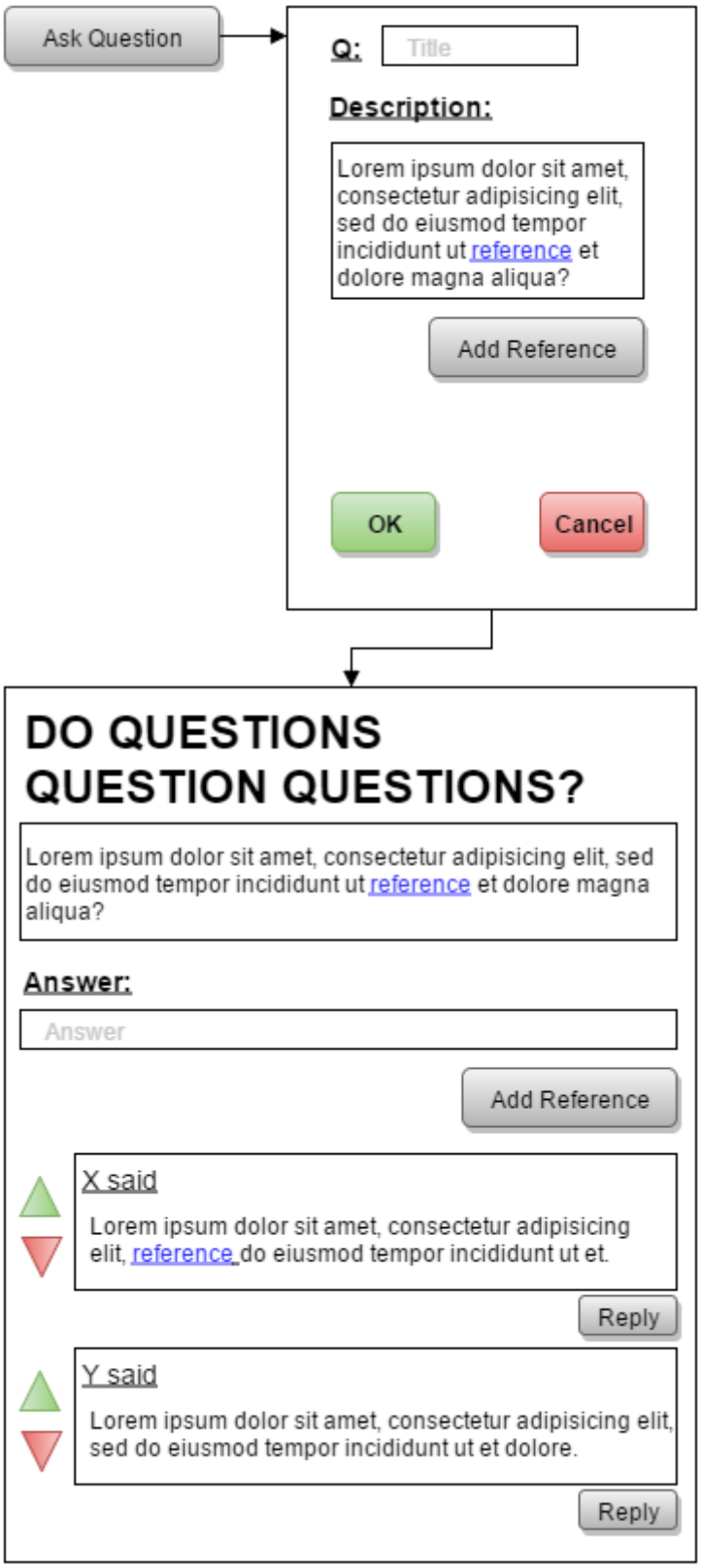
Mock-up of use case 1

Use Case ID:	2		
Use Case Name:	Ask a Question		
Created By:	Maxime	Last Updated By:	Caroline
Date Created:	2/11	Date Last Updated:	13/12

Description:	Adding a question containing a self-reference and none or more cross-references to the course or course material.
Actors:	User, System
Trigger:	Press the “Ask Question” button.
Preconditions:	The user is logged in. The user is subscribed to a course.
Postconditions:	A question is added to the course or course material.
Normal Flow:	2.0.1. The user presses the “Ask Question” button. 2.0.2. The system shows the question form. 2.0.3. The user fills in the form, which can contain cross-references. 2.0.4. The user presses the “Confirm” button. 2.0.5. The system adds a cross-reference to the course or document the user was viewing. 2.0.6. The system saves the question. 2.0.7. The system shows the question to the user. 2.0.8. The system notifies the course owner and subscribed users of this question.
Exceptions:	**2.0.E.1 The question cannot be saved.** 2.0.E.1.1 The system displays an error message. 2.0.E.1.1 The system redirects the user to step 2.0.1.
Includes:	Use case 1: ‘Add cross-reference’
Priority:	High
Frequency of Use:	High: We expect this feature to be used frequently as it is one of the main focuses of our application to have a lot of interaction between users.
Complexity:	Medium
Assumptions:	Cross-references should be clickable inside the questions so it is easy to navigate to a certain part of a document.



Black box diagram of use case 2: Ask a Question

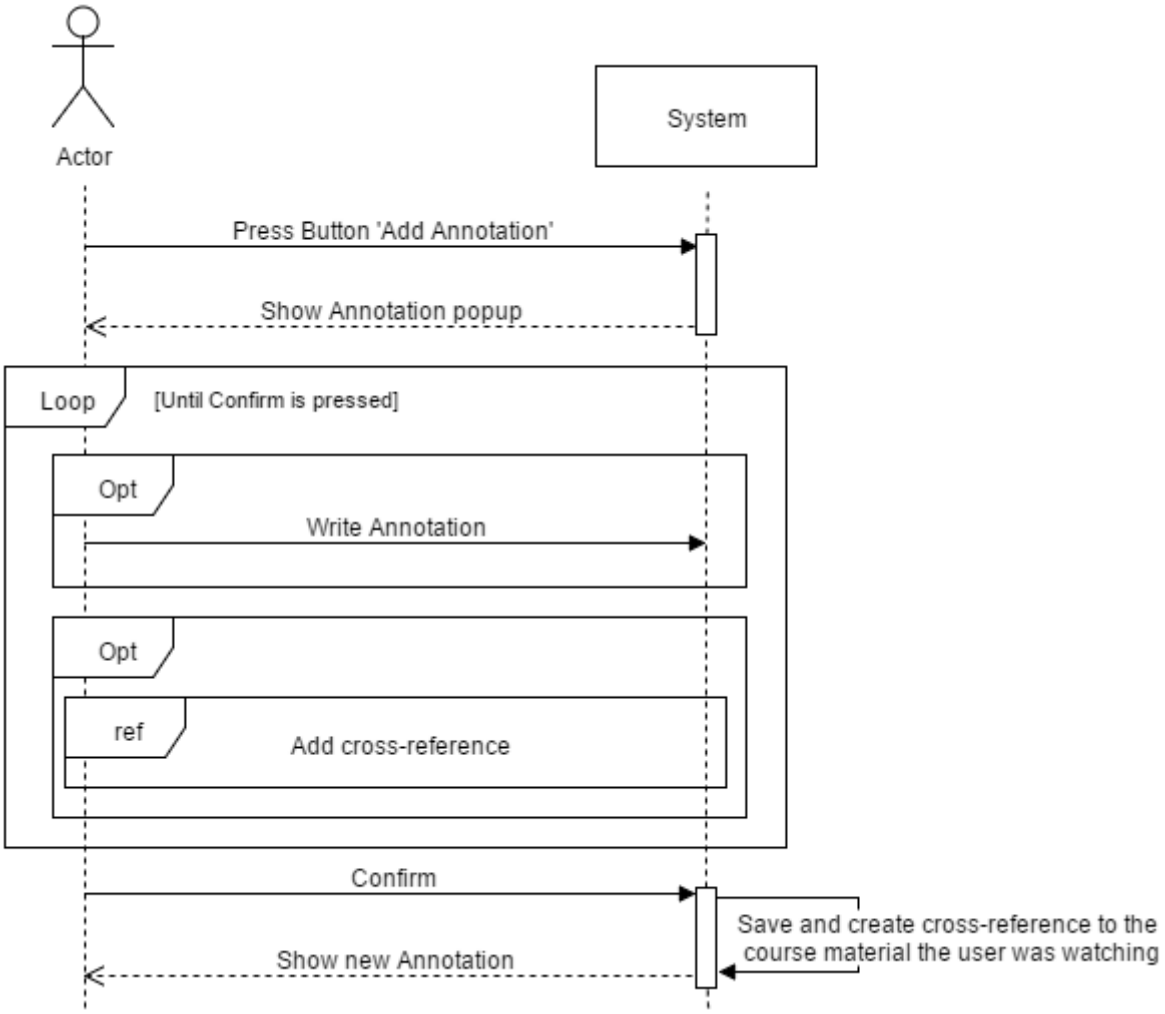


Mock-up of use case 2

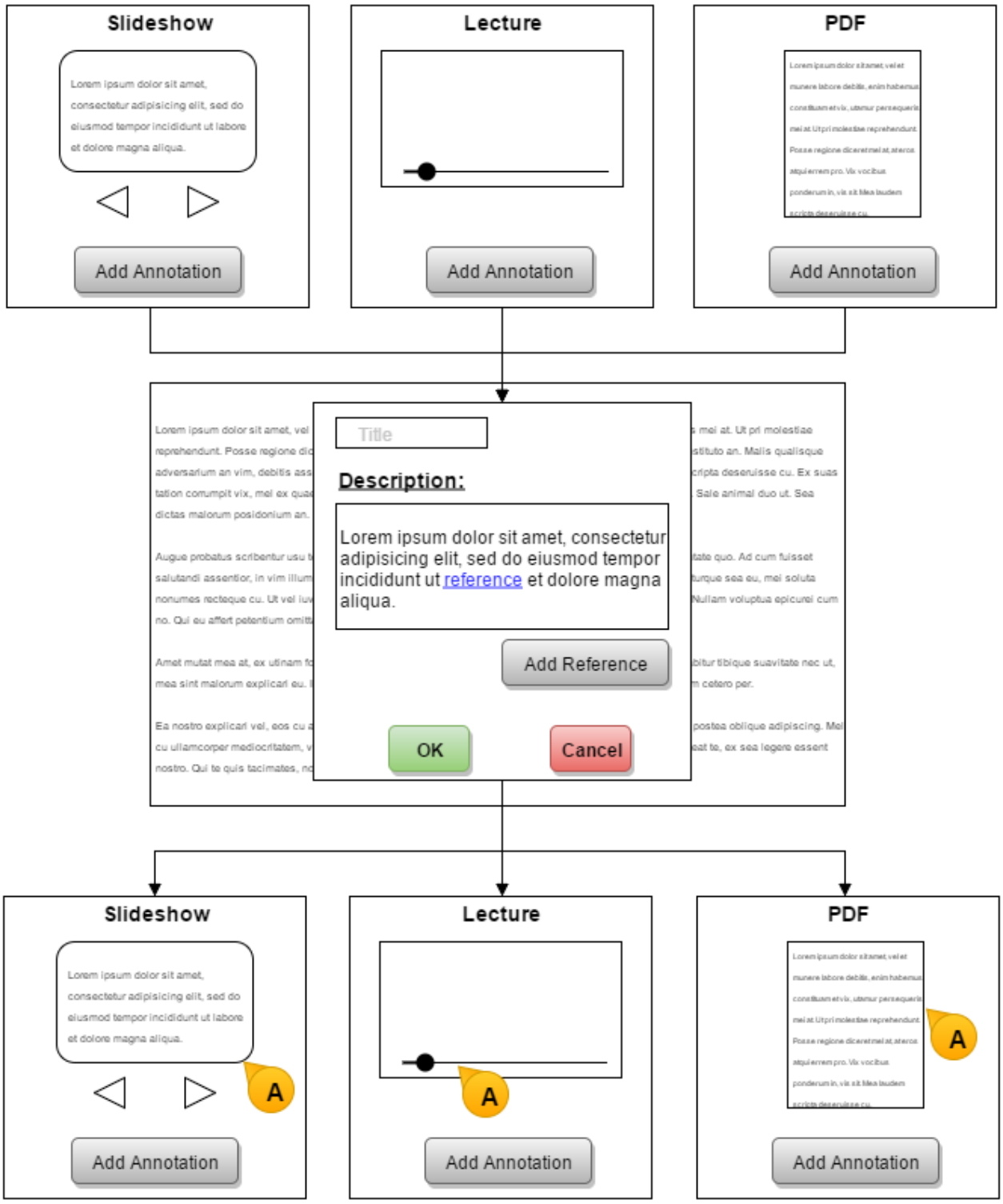
Use Case ID:	3		
Use Case Name:	Add Annotation		
Created By:	Driek	Last Updated By:	Caroline
Date Created:	3/11	Date Last Updated:	13/12

Description:	Adding an annotation to a lecture containing a cross-reference to course material.
Actors:	User, System
Trigger:	Press the “add annotation” button.
Preconditions:	The user is subscribed to a course. The user is logged in.
Postconditions:	The annotation is added to the lecture.
Normal Flow:	3.0.1. The user presses the “add annotation” button. 3.0.2. The system shows the annotation create popup with timestamp set on what time the user clicked the “add annotation” button. 3.0.3. The user writes the text for the annotation in the text field. 3.0.4. The user adds a cross-reference to course material. 3.0.5. The user presses the “Confirm” button. 3.0.6. The system saves the newly added annotation to its database 3.0.7. The system confirms the annotation has been added by showing the document with the annotation.
Alternative Flows:	**3.1.1 The user presses “Cancel”.** 3.1.2 The annotation create popup is closed with no changes saved.
Exceptions:	**3.0.E.1 The annotation cannot be saved.** 3.0.E.1.1 The system displays an error message. 3.0.E.1.2 The system redirects the user back to step 3.0.3.
Includes:	Use case 1: ‘Add cross-reference’
Priority:	High

Frequency of Use:	High: this is the main feature of our application
Complexity:	High
Special Requirements:	The annotation must automatically add a self-reference, meaning it holds a cross-reference to the document you want to add the annotation for.



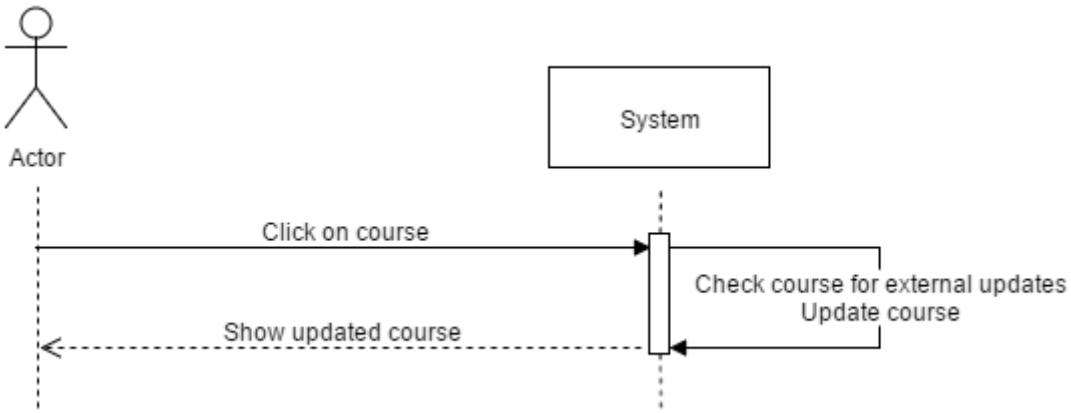
Black box diagram of use case 3: Add Annotation



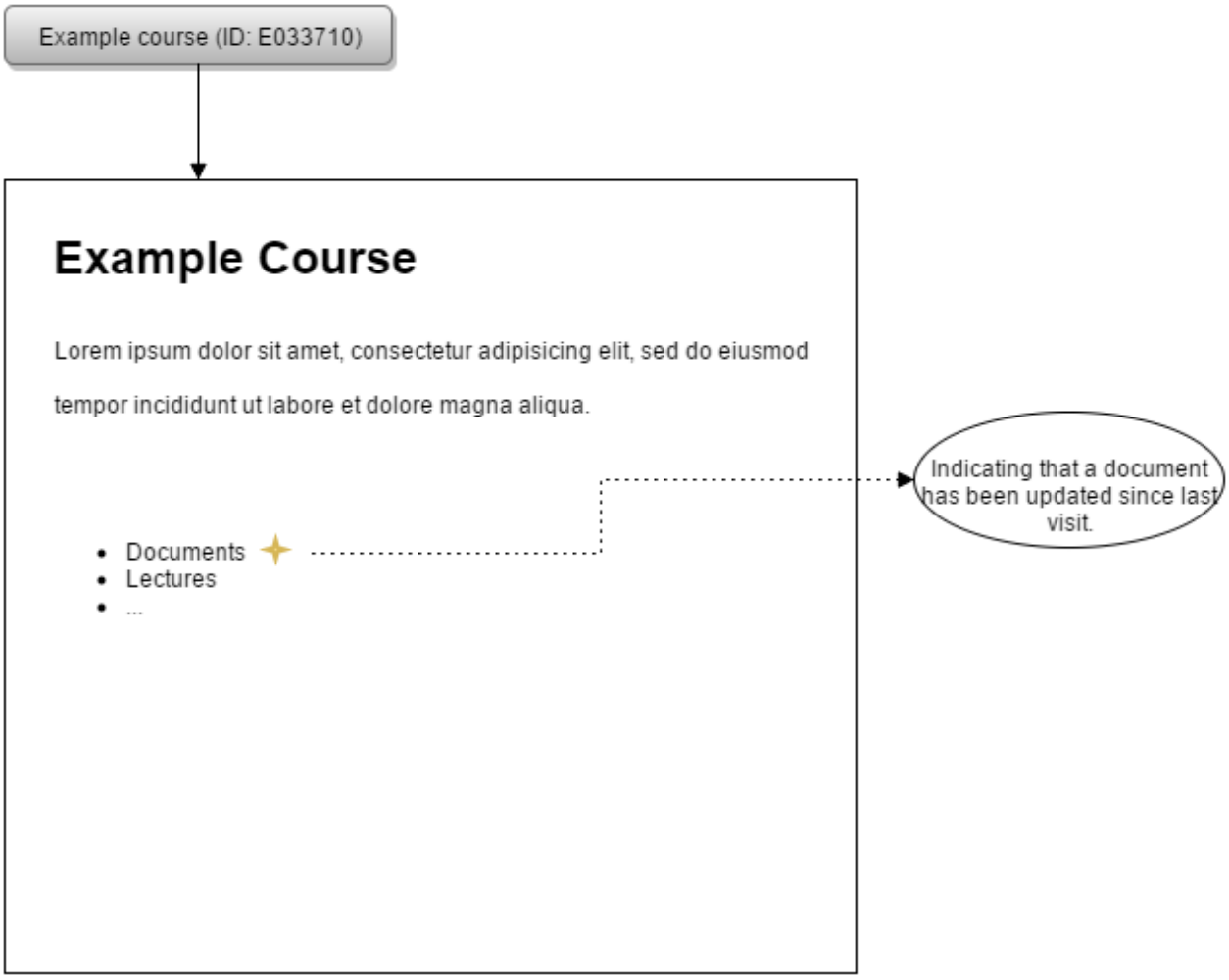
Mockup of use case 3

Use Case ID:	4		
Use Case Name:	Open and synchronize course		
Created By:	Maxime	Last Updated By:	Caroline
Date Created:	13/12	Date Last Updated:	13/12

Description:	Clicking on a course causes it to automatically synchronize with the external source it was originally imported from.
Actors:	User
Trigger:	Click on a course.
Preconditions:	The user is subscribed to a course. The user is logged in. The course has been originally imported from an external source.
Postconditions:	The course is opened and up-to-date.
Normal Flow:	4.0.1. The user clicks on a course he is subscribed to. 4.0.2. The system shows the updated content of the course.
Alternative Flows:	**3.1.1 The course has no new content.** 3.1.2 The system shows the course.
Priority:	Medium
Frequency of Use:	High
Complexity:	Medium
Assumptions:	The external source must be saved and still available.



Black box diagram of use case 4: Open and synchronize course



Mockup of use case 4

Chapter 4

Design

4.1 Component and connector UML diagram

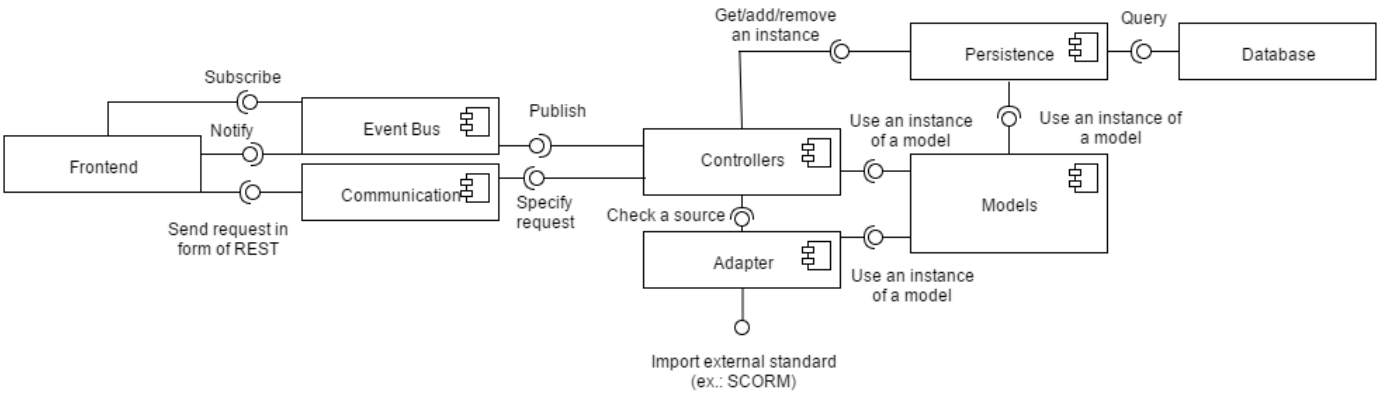


Figure 4.1: The component diagram

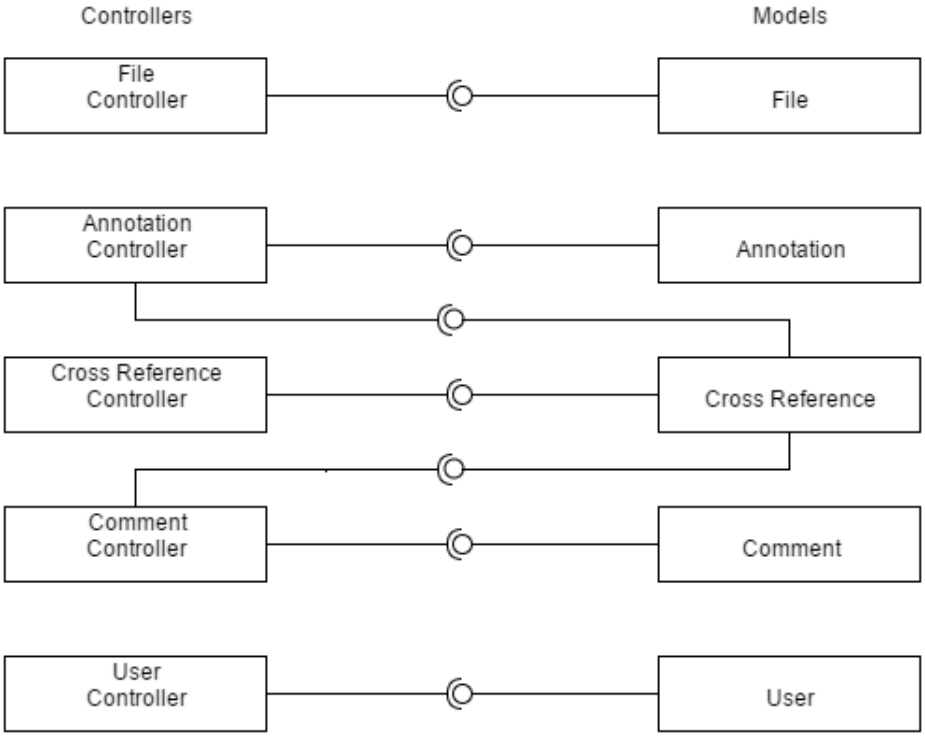


Figure 4.2: An in-depth view of the Model-Controller relationship in the component diagram

The **Communication** component is responsible for catching all requests that come from the front end and translate or parse them before passing them on to the controller.
The **Controllers** component will be split up in different controllers (Fig. 4.2). These controllers will be responsible for the processing of requests, and passing the right requests to the right components. This will mostly consist of updating models, communicating with the persistence layer and publishing events to the EventBus. They act like controllers of a standard MVC model.
The **Models** component will be split in different models in a similar manner (Fig. 4.2). They will represent the data the application works with. These act like models of a standard MVC model.
The **Adapter** component is responsible for parsing external sources. It will consist of different modules to parse different standards (e.g. JSON, SCORM, ..)
The **EventBus** component handles communication in a Publish/Subscribe pattern. Every event will be published to this EventBus and subscribers will register through the EventBus.

The **Persistence** component is responsible for updating and querying the database.

4.2 Quality Attributes

4.2.1 Interoperability

We included an **Adapter** component to ensure that our system is interoperable with a wide variety of other systems. Collaboration with another system can be achieved, only by implementing a new adapter. This way, our system is independent of any data format.

4.2.2 Usability

The **EventBus** component contributes to the usability of our system. When course data is added or updated, we will notify every front-end currently subscribed to the notifications of that particular course. A front-end will be subscribed, for example, when it is browsing the contents of that course. Notifying will be implemented using push-notifications.

4.3 White-box sequence diagrams

4.3.1 Use case 1: Add cross-reference

This sequence diagram contains the necessary steps to perform when adding a cross-reference. The diagram can be found in Figure 4.3. Note that which controllers and models are invoked, are not specified in detail, because this depends on the document type we are working with at that moment.

4.3.2 Use case 2: Ask question

This diagram contains the necessary steps to perform when wanting to ask a question. The diagram can be found in Figure 4.4. As can be seen in the mockup of use case 2, the question form contains a title and a body. In the sequence diagram, we fill in the title first, then we add text to the body of the question and add inline references. This is why we chose for a loop with the two (optional) actions “Append question body” and ”Add cross-reference”.

4.3.3 Use case 3: Add annotation

This diagram contains the necessary steps to perform when wanting to add an annotation. The diagram can be found in Figure 4.5. This use case is quite similar to the last one. We also work with inline cross-references in the annotation body.

4.3.4 Use case 4: Open and synchronize course

This is the use case we added to illustrate the use of the adapter module. The diagram can be found in Figure 4.6. When we try to load a course, the system will automatically synchronize its contents with the appropriate sources. This involves the adapter module and external sources.

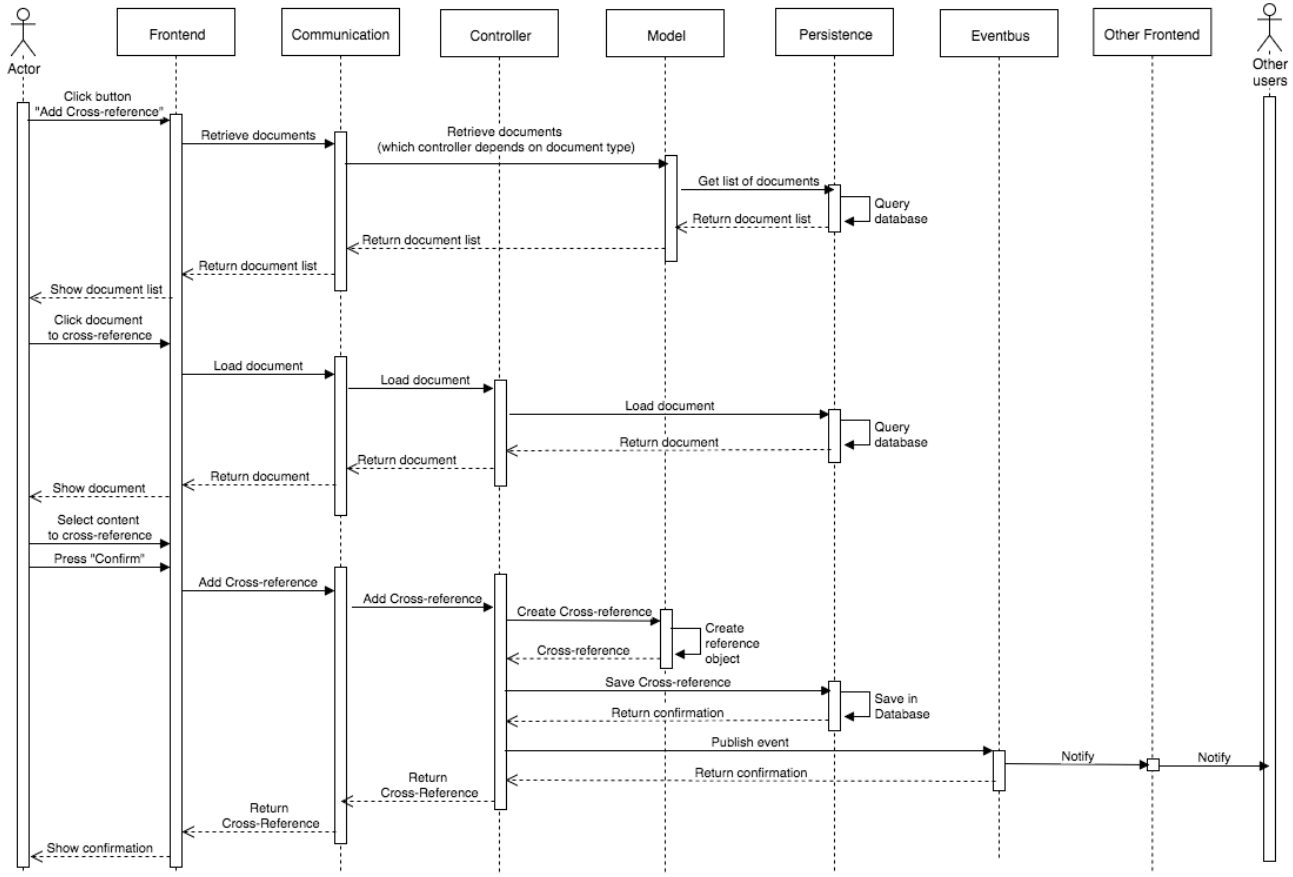


Figure 4.3: Use case 1: White-box sequence diagram (Add cross-reference)

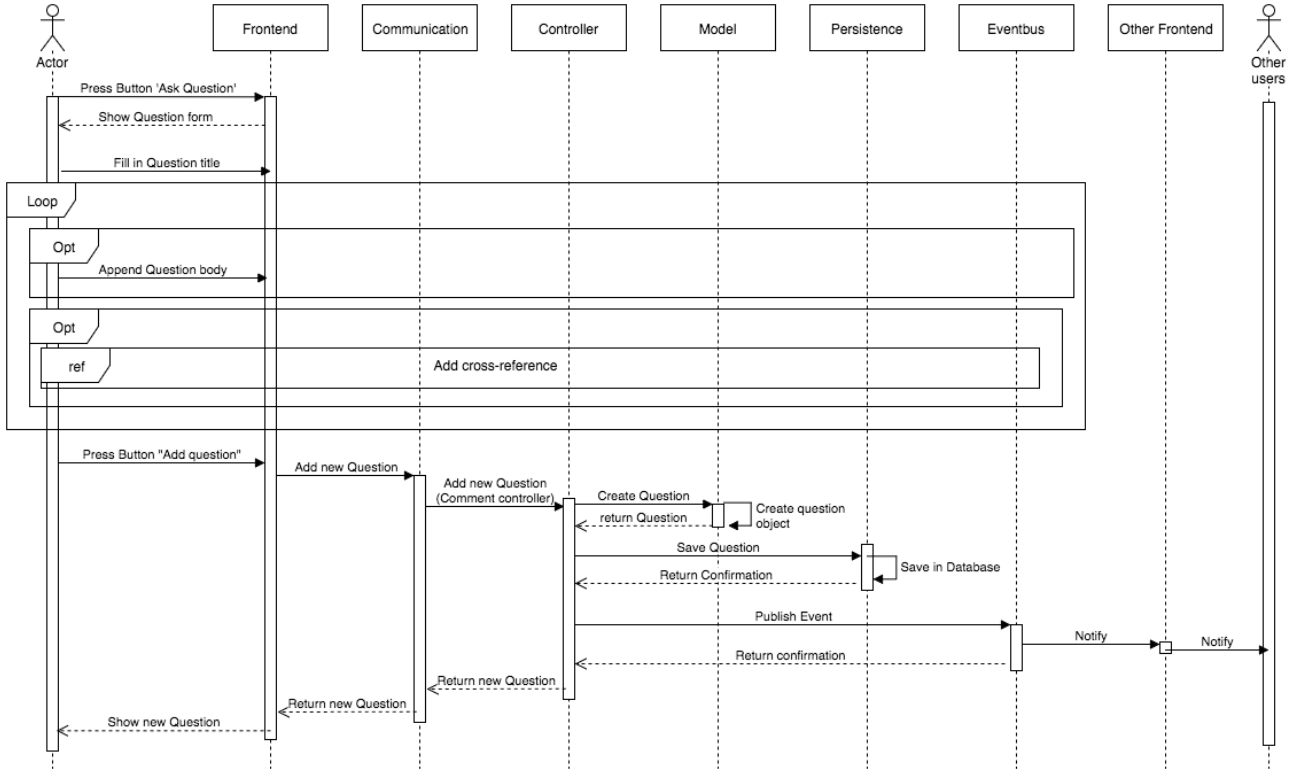


Figure 4.4: Use case 2: White-box sequence diagram (Ask question)

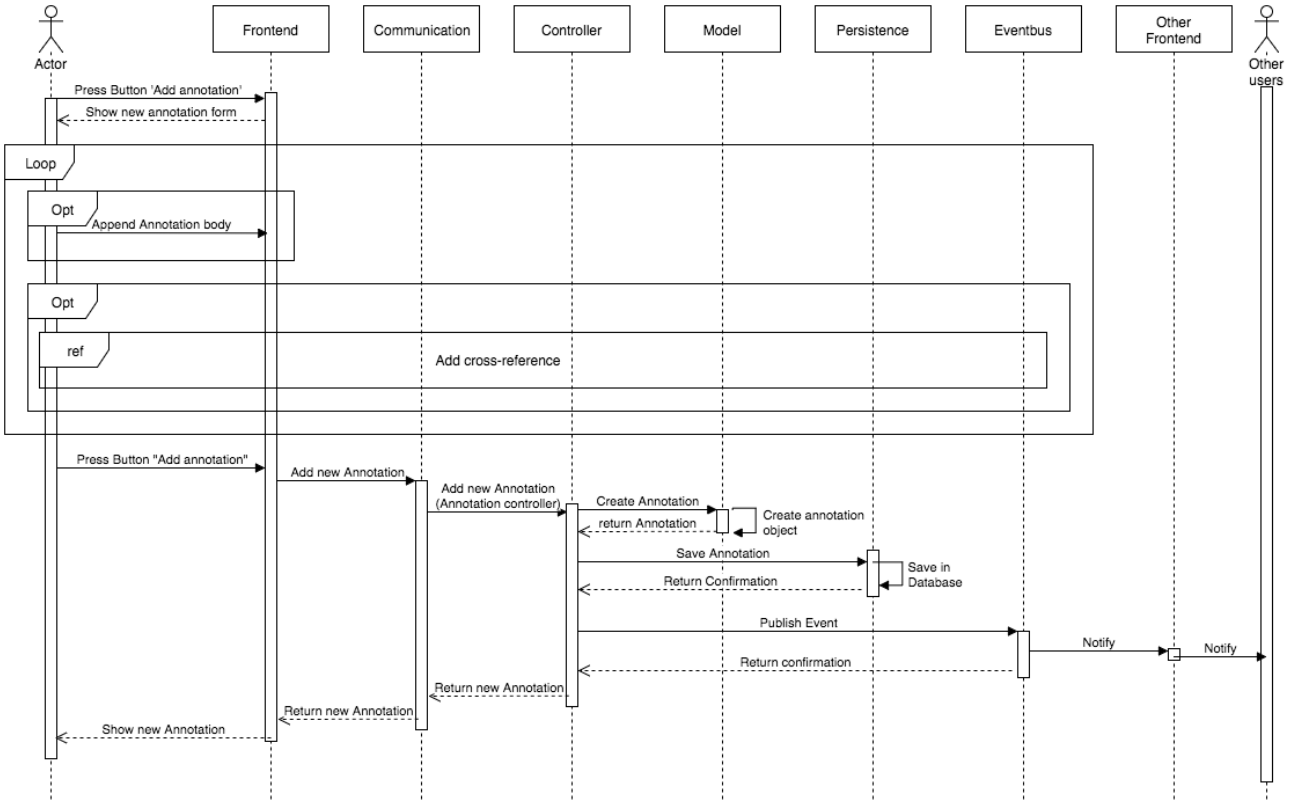


Figure 4.5: Use case 3: White-box sequence diagram (Add annotation)

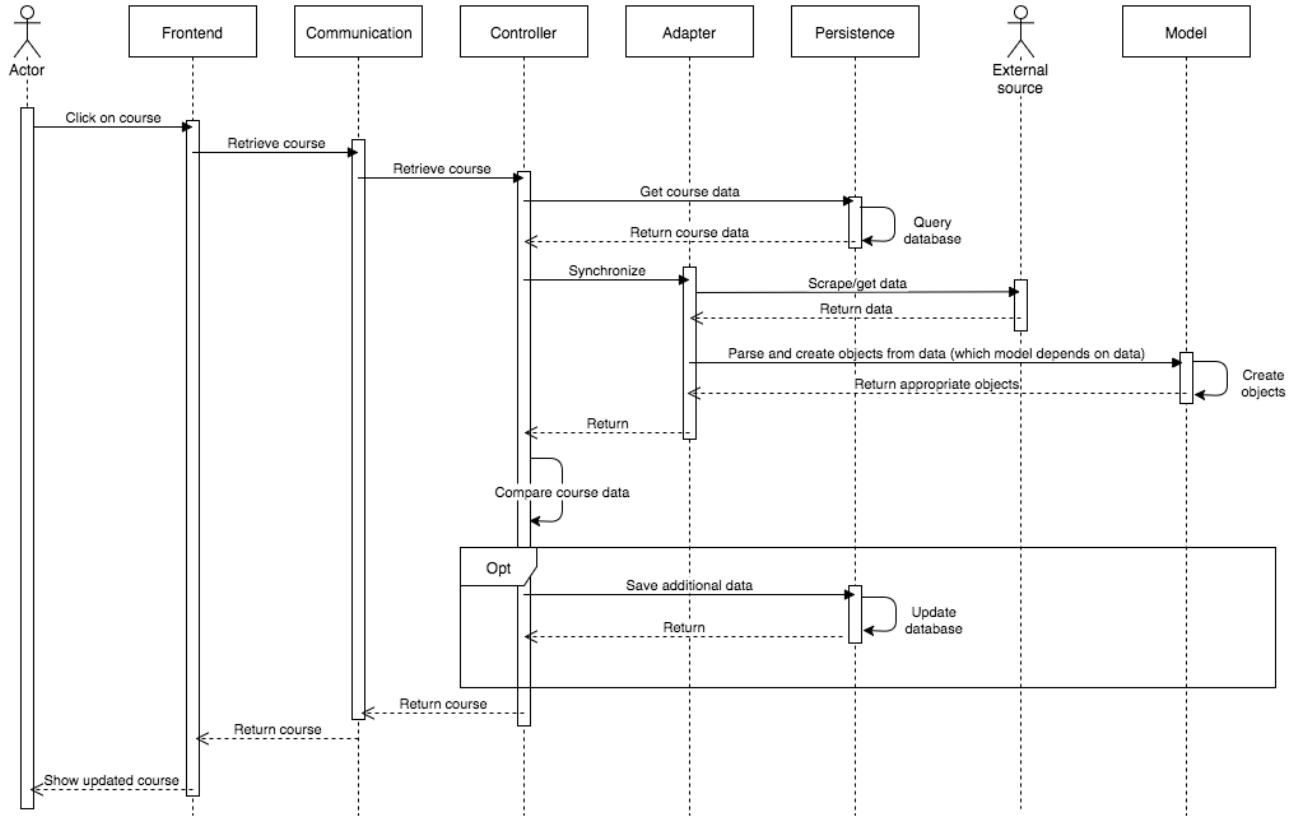


Figure 4.6: Use case 4: White-box sequence diagram (Open and synchronize course)

4.4 Deployment diagram

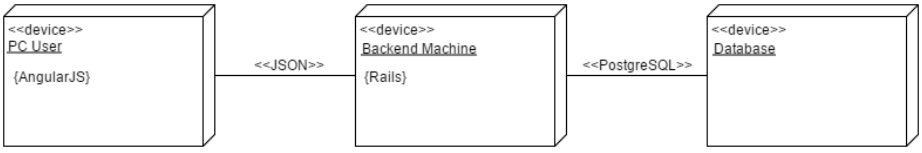


Figure 4.7: Deployment diagram

Chapter 5

Prototype

We estimate that the core of our application can definitely be implemented with a team of 8 in 12 weeks. It's likely that even our lower priority use cases can be developed in that time span. In this chapter, we will give a brief overview of what we mean with the "core" of our application and which features could be added to our application afterwards. Afterwards, we discuss the essential requisites to be able to build this prototype.

5.1 Core application

For the core application, we will focus on first developing the cross-reference, annotation and comment functionality. For this, we will need all features of the persistence, controller, model and communication component, as well as an operational front-end.

5.2 Adapters

An extension to this core is the use of adapters. This makes it possible to get data from other platforms like Minerva and IMinds X. This ensures interoperability, one of our quality attributes. Within 12 weeks, it should be possible to establish interoperability between a few systems. However, the number of adapters implemented will depend on time constraints.

5.3 Publish/subscribe

To boost interoperability, an eventbus will be added. Events can be pushed to subscribing front-ends, so they receive the most recent information. At first, this could be implemented using active polling for synchronization. This too, should be possible in 12 weeks. A later extension could be to use push notifications.

5.4 Requisites

5.4.1 Server

A basic Linux server will be needed to run the back-end and the database.

5.4.2 Minerva data

We would like some kind of access to data released on Minerva. Permission to scrape the website would be viable, however an API would be ideal.

5.4.3 IMinds X data

For courses using the IMinds X platform, we would like access to their contents using SCORM for example.

5.4.4 Recordings

To illustrate the use of timed annotations and uniform cross-references, we would like to receive several sample recordings of lectures.

5.4.5 Sample Course material

The slides and/or syllabus corresponding with these recordings would be needed to be able to clearly illustrate the use of cross-references. If these are courses we take or have taken ourselves, we can always provide this material ourselves.

Chapter 6

Conclusion

We started with the small idea that recorded lectures should be able to get used by the students to improve their way of learning and studying. The first big response we got was 'What about the lecturers?'. We don't want our application to be just another platform that lecturers have to keep up-to-date.

We want our application to be easy in use and it should just work. As a lecturer you should be able to import your course once and it should always be up-to-date. We want Classic to be an extension of a course, it should act as a bonus, not a burden.
Students should be able to just log in to our application without registering and access the courses they're taking.

After all this was decided, we concluded that we would want to build a complete, functional application instead of trying to build a huge application which we would never be able to finish. Although our application might be on the smaller side, it should be easily upgradable so that a plethora of other things can be incorporated.

Bibliography

[David Bargeron, 1999] David Bargeron, Anoop Gupta, J. G. E. S. (1999). Annotations for streaming video on the web: System design and usage studies.

[Evan F. Risko, 2013] Evan F. Risko, Tom Foulsham, S. D. A. K. (2013). The collaborative lecture annotation system (clas): A new tool for distributed learning.

[Steimle J., 2009] Steimle J., Brdiczka O., M. M. (2009). Collaborative paper-based annotation of lecture slides.