Group 10 Experience Report

Achyut Madhavan gusmadac@student.gu.se
Chiranjibee Satapathy chiranjibee2012@gmail.com
Eric Erlandsson Hollgren eric.e.hollgren@gmail.com
Elin Nilsson nileli@chalmers.se
Priyanka Pentela priyanka.pentela@gmail.com
Shaphan Manipaul shaphan@chalmers.se
Tindra Järgenstedt tindra@jargenstedt.se

***Abstract*—In this report, we describe the different techniques we used when creating a requirements specification for an improved Education Learning Management System (ELMS). The report starts off by describing the different techniques used in the elicitation process. These were: defining the problems, brainstorming, clustering, creating questionnaires and taking interviews. What followed were the several specification techniques that we used to specify the requirements based on what we learned during the process of elicitation from our stakeholders such as context diagrams, use cases, quality factors etc. This gives a good insight into what our system does and what core functionalities it achieves along with many other techniques. In the next section, we give a detailed overview on how we, as a group and as individuals, evolved in the learning, communicating, understanding, implementation of not only the various techniques of elicitation and specification in a real-life scenario, but also in a dynamic group environment, where we need to do requirements engineering.**

I. INTRODUCTION

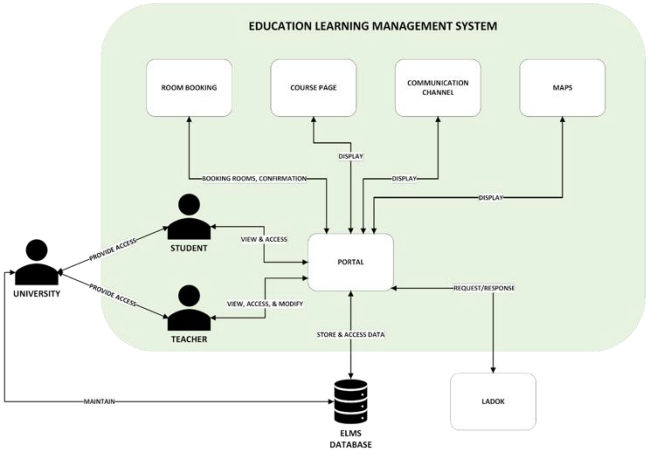

Figure 1: Context diagram for ELMS.

The system this project aims to develop is a new and improved ELMS. As of today, some universities in Sweden are using the learning platform Canvas. The platform is used to provide course material, assignments, and convey other information regarding courses to the students.

There are several problems with the way Canvas is currently operating. One of these problems is that it is confusing to find certain information on the course pages as most of the course pages have different layouts. There are also several different places where you can receive notifications and it is not clear where it is pointing us to. As of now, there is also a calendar function which is not being used by most of the teachers as they refer to time-edit instead.

Due to these problems, as well as others, we see a need for a new and improved platform where uniformity, ease of access and clarity are highlighted. This project specifies the requirements for said system.

II. TECHNIQUES

In this part, we discuss and motivate the different techniques used in the project.

A. Elicitation

1. *Defining problems:* The elicitation process started with us (as students, and therefore are classified as users) looking through Canvas and asking ourselves what we saw as problems with the platform. We discussed several things that we had experienced as problems during our time as students. In the end we narrowed the problems down to those that we could all agree on. The main points were:
   - Uniformity across all platforms so that the same functions are accessed in the same way from all different platforms.
   - Functions that students use daily should all exist in the system. Such as: schedule, maps, booking rooms, messaging system.
   - Uniform layout of course pages to make it easier for students to find the right material on the page.

2. *Brainstorming:* For the next part of our elicitation process, we conducted a brainstorming session, where we came up with new ideas that need to be implemented in our system and functionalities that had to be improved in the existing systems. We did this by writing the ideas and functionalities down on post-it notes. Some of the key ideas that we came up with during the brainstorming session were:
   - A better communication channel for the users to efficiently communicate within the system with their peers and other users.
   - A better way to group all notifications and differentiate between assignment deadlines, announcements, etc.
   - Uniformity across platforms such as Web, iOS and Android.
   - A better time-managing system where users would be able to see their schedule in the system, book rooms, see deadlines as well as be able to use a map function to get to their lectures instead of having to rely on a third-party application.

- To maintain a standard template for the course pages such as the syllabus, lecture notes, presentations and exercises, so that it would be easier for the teachers to upload and edit their course pages and make it easier for the students to efficiently find the information they need.
- To provide external links to essential third-party services like Ladok to register for exams and courses etc.
- Other small improvements to the overall system; like an option to change the language on all platforms rather than only on the website, being able to remove old courses from the dashboard, and so on.

The brainstorming activity resulted in many ideas, both good and bad, which lead to a broader scope. However, this was positive for the process since we were able to get all ideas on paper for further use and selection. We also got a clear image of all members' views and expectations of what the project would look like due to different suggestions from brainstorming. This helped us with getting on the same page regarding we could continue working on the project.

Future brainstorming sessions would be helpful; in order to add even more variety of ideas, and obvious ones. There was some obvious functionality which was missed during brainstorming, and it would have been better if we had them written down with everything else so that we do not forget about them.

3. *Clustering:* After the brainstorming session we had several ideas for functionalities. The next step was to cluster the ideas into broader categories. We did this to make it more comprehensible what functions should be implemented. We grouped three to five post-it notes into one broader category. The broader categories were:
    - Better time-management system.
    - Ease of access.
    - Better notification system.
    - Uniformity across platforms.

    The clustering technique helped us with finding some functionalities which were similar and some that were the same. The group would then either combine these or remove some to avoid duplication.

4. *Questionnaire:* To receive input from more users we used a questionnaire that was answered by students from different courses. The questions were mainly about the current system used (Canvas) to see if there is a purpose to updating the system. Also, to find out if there is a wish for additional functionalities and to receive more suggestions on what they might be.

    The results collected indicated that some users are happy with the current system, but some are unhappy. The response for our suggestions with integrating time-edit and a map to the ELMS was appreciated and 75% of the students answered 4-5 on a scale of 1-5 from “not at all” to “very much”. Similar results were shown for integrating a map system for finding rooms on campus. For this, 82.2% of the students answered 4-5 on the same scale.

    These answers confirmed the wish the students in our group had to integrate more functionality in an ELMS and was therefore further specified as requirements.

5. *Interviews*: An interview in the form of a semi-structured interview was conducted with one of our teachers in the course, which enabled us to get a perspective of how our system must perform from the teacher’s perspective as another user and stakeholder of our proposed system.

B. Specification

Based on the functionalities we came up with in the elicitation process we specified requirements. We do this to make sure that there is no ambiguity in what should be done.

1. *Context diagram:* We started the specification process by creating a context diagram, as shown in Figure 1. We did this to get an overview of the system and to easily explain its main functionalities to the stakeholders.

2. *Use Cases:* The technique used to specify the core functionality on product level as requirements was use cases. These were the first requirements specified and they are based on the ideas generated during *brainstorming* and *questionnaire*. Use cases can be traced back to the specific user and it provides a clear image of how stakeholders and users will use the application. They give a clear image of the core functionality and **what** is expected from the system without giving too much detail of **how** it will work. These use cases were further used to define functional requirements. Also, a use case diagram was used to help the stakeholders better understand the use cases.

    For the group, this helped us a lot with moving forward with defining requirements on different levels. However, a problem was defining too many use cases and cases that were not part of the core functionalities. These could later be used for describing the system on other levels.

3. *Functional Requirements:* After defining the use cases and core functionalities, we derived the functional requirements of the system from them. We were able to arrive at eight functional requirements after the elicitation process from the stakeholders and their use cases after careful consideration.

    We used the task description in order to describe the functions the software should be able to support. When we used task descriptions, we were able to break down the more general functional requirements down to segments with subtasks. The task descriptions also provided us with valuable

information regarding how often the task is used by the system, what is needed beforehand and where each subtask starts and ends.

We have documented the functional requirements in a way that each functional requirement describes its **purpose** in the system, the **pre-condition** that is to be met for the functional requirement to be **triggered**, and how often the user makes use of the requirements along with its **sub tasks** and **variants**.

4. *Quality Factors:* When the functional requirements were successfully defined, we described the non-functional requirements that are essential to reach our project goals. We defined our quality requirements using the Quality grid, where we mapped various quality requirements based on the scale of importance it plays in the functioning of the ELMS. So, we were able to introduce these "**Important**" quality factors such as Usability, Interoperability and Portability into our proposed system and remove some "**Unimportant**" quality factors like efficiency, as our system is not time critical.

   For the specific quality requirements, we used P-Language to measure our quality requirements. We measured the quality factors for our system by taking a problem faced by a user for **scale** and establish a baseline on what the system must do regarding the quality factor, and we state what we **wish** and **plan** for our system to do compared to what the system in the **past** (Canvas, in this case) did or does by measuring using the process or actions described in the **meter**.

5. *Class diagram:* For the data requirements to be illustrated we used the class diagram approach. Using the class diagram, we showed what data the system needs to accomplish the functional requirements and what functions work on that data. It helped us to clearly understand the inner working of the data.

6. *Data dictionary:* Once we collected all our data requirements, we used a data description to structure the data in a systematic view. In a data dictionary, we describe each class of data with some examples while also explaining its attributes and the methods that each class uses.

7. *UI Prototype:* We made a UI Prototype, so that it can make it easier for the stakeholders to get a better grasp of the functioning of the ELMS and to make sure that their business goals had been met. It also played an important role in validating our quality and data requirements.

8. *System Requirements:* We then defined our system requirements for three out of the eight functional requirements. The ones we specified were Book Rooms, View Schedule, and Communication Channels. It was done by breaking the requirement down into smaller steps and specifying what the system, as well as the software, should do in each of those steps. This was done to make it clear how the requirements should be implemented from a technical perspective.

C. Prioritization:

For the prioritization, we provide an overview of the techniques we used to prioritize our functional requirements for the ELMS. Since prioritization of functional requirements plays a crucial role in the development of our system, we have elicited various prioritization decisions from all our stakeholders.

This has helped us in eliminating various complex decisions regarding the functional requirements we had wanted to implement in our ELMS. We have arrived at the prioritization decisions by using the techniques that were best suited for the ELMS and its stakeholders.

We are developing the ELMS with a focus on universities as customers with its teachers and students as the users. This leads to different stakeholders having different priorities for what they want to do with the system. So, it was important not only to elicit the priorities but also combine the different priorities of various stakeholders in a way that does not compromise one stakeholder's priorities over the others.

We believe that our system is driven towards bespoke development rather than market-driven development. So, once all our functional requirements were agreed upon by the stakeholders, we used the following methods to prioritize them.

For our prioritisation process, we conducted two methods to do the prioritisation. We, as a group, assigned ourselves the roles of the stakeholders and got ourselves committed to our personas in order to do our absolute best to eliminate any bias creeping into our priorities.

1. *100-dollar test:* One of the techniques used for prioritization was the 100-dollar test. We used personas and took on the roles as University, Teacher and Student. Based on these personas, a total amount of 100 dollars each were spent on the different functional requirements of the system. The result was that all of the requirements got funding between 64 and 20 dollars total. The FR1 (Course Info) was prioritized the most by the stakeholders, while the FR2 (Room Booking) was the least prioritized. However, all of the requirements ended up a relatively small span. This can make it hard to interpret the result, since none of the requirements was significantly more important for any of the stakeholder. It also pointed towards unequal prioritization from different stakeholders.

2. *Ranking:* One of the techniques used for prioritization was the 100-dollar test. We used personas and took on the roles as University, Teacher and Student. Based on these personas, a total amount of 100 dollars each were spent on the different functional requirements of the system. The result was that all of the requirements got funding between 64 and 20 dollars total. The FR1 (Course Info) was prioritized the most by the stakeholders, while the FR2 (Room Booking) was the least

prioritized. However, all of the requirements ended up a relatively small span. This can make it hard to interpret the result, since none of the requirements was significantly more important for any of the stakeholder. It also pointed towards unequal prioritization from different stakeholders.

The second technique used was ranking the functional requirements. The process was the same as for the 100-dollar test but now the personas gave each functional requirement a unique number from 1 to 8 where 1 is the most important requirement. The ranking gave the same top prioritization as the 100-dollar test (FR1, Course Information), but the lowest ranked requirement was now FR4 (Course Page Editing). It is mentionable that FR4 was ranked as number 1 by one stakeholder (Teachers) but still ended up last.

Both techniques ended up with remarkably equivalent results. So based on the results achieved in both the 100$ test and the Ranking method, we cross analysed them and deduced that most of the functional requirements of our proposed system were prioritized by our stakeholders to the same priority level. The rest of the functional requirements were prioritized by taking into consideration other external factors and how big of an impact the requirements played on the system all agreed upon by all the stakeholders of the ELMS.

D. *Traceability:*

Traceability plays a significant role in the requirements specification. We did traceability to link all our functional requirements to its respective Business Goals, Use Cases, System, Data, and Software requirements. We do this to eliminate any confusion in our understanding of our system.

Doing this has helped us understand which artifact is related to another in the system and helps us find and eliminate weak links. We firmly believe that by having done good traceability, will give a firm helping hand, when in the case of our system facing a change in a business goal or any other artifact, we will face a need to do a change impact analysis.

E. *Validation:*

We used three *main* validation techniques when performing validation during the peer review of another group. Those were:

- Content checks – Is the entirety of the system covered in the specification?
- Structure checks – Is the specification clear and readable?
- Consistency checks – Is the specification consistent everywhere?

The three techniques were combined into a checklist that the group filled out when conducting the peer review. During the validation we looked at every requirement and assessed them based on these three points. Each group member reviewed themselves first and then we met up and compared our lists of findings. We discussed everything that we found and decided on what was high priority and what was low priority. Then we merged our lists into a single checklist and sent it to the group we were reviewing. The process was quite time intensive but in the end it we feel like we caught the most important shortcomings and let the peer reviewed group know.

III. LEARNING OUTCOME

A. *Elicitation*

Elicitation is one of the first things that need to be conducted when doing requirements engineering. The elicitations build a foundation of domain knowledge that is necessary when later deriving requirements for a system within that domain.

In our project we have employed both internal and external elicitation techniques, such as brainstorming, clustering, polls and a semi-structured interview. Since all team members had prior experience with education and learning platforms, we possessed insights into what needed to be included and what needed to be enhanced in the new system. However, we understood that relying solely on internal elicitation was insufficient for gaining a comprehensive understanding of teachers' perspectives. Therefore, we have conducted a semi structured interview with a teacher and distributed polls to fellow students to obtain a more comprehensive assessment of the strengths and weaknesses of the existing system.

Due to time constraints we only managed to do one semi-structured interview with a teacher which meant we only had one perspective from the side of the teachers. This will likely make the requirements skewed, and in a perfect world, we would like to have done more of these interviews to get a more comprehensive understanding of what teachers think of the current system.

When doing the poll for the students, we chose to only use closed questions since open questions often can be misinterpreted by the students taking the poll as well as the group misinterpreting the answers. However, we still decided to add non-required sections where students can add comments in order to get as much information out of the poll as possible.

We feel like we have learnt a lot from the workshops, lectures, and the conducting of elicitation ourselves and we as a group thought that brainstorming along with clustering of our ideas into broader categories helped us the most while eliciting our requirements.

B. *Specification*

In the course we have learnt how to use different techniques for specification. In our case we have used a context diagram, p-language, and task descriptions. Using these techniques helped us to convey our ideas in a clear and concise manner. The methods are tested and used in many cases in the real world and hopefully by taking advantage of them we too can specify our system and the relations within the system. We believe using a combination of the listed specification techniques above on different levels of the system helped us gain clarity in what we expect the system to be able to achieve.

Context diagram: For the requirements document we decided to use a context diagram. This helped us get a better grasp on the domain by simplifying and visualizing it. The context diagram helps in identifying external inputs, serves as a high-level checklist over what to develop and helps prevent scope creep. We decided to keep the students and teachers within the boundaries of the system since they are our primary users. They differ from the university actor

as the university in our perspective and understanding only interacts with the system under certain circumstances.

Use Cases: For the use cases, after thorough analysis, we made a use case diagram based on only the core functionalities that we want to introduce in the system in addition to the baseline functionality that we established as a baseline from the previous system (Canvas).

Functional Requirements: After the use cases and the core functionalities were successfully defined, we derived the functional requirements that are required for the system. And, to concisely explain our functional requirements to our stakeholders, we used task description. We arrived at Task description as the method to describe our functional requirements as it gave us the opportunity to document crucial factors like sub-tasks, variants, pre-conditions, and triggers for the respective functional requirements.

Quality Factors: Our system's non-functional requirements were derived from what we wanted in the system based on our business goals. Since our business goals are focused on uniformity, consistency, and interoperability across all platforms, we had to be concise when we derived our quality requirements. We considered the quality factors which were classified under Operation, Revision and Transition and tabulated a quality factors table on how critical each quality is for our system on a five-point scale. We also had to make sure that the quality requirements do not have an impact that will affect the functional requirements.

P-Language: The process of using P-Language was at first quite hard because we missed making most of our quality requirements measurable. This made the first keyword **scale** very hard to define. However, after we figured out our mistake, we found that P-Language was a really simple and good way to further describe our quality requirements.

Class diagram: We used the class diagram to specify our data requirements for the system. Using the class diagram helped us in identifying what data that our system needs and provides the list of attributes and methods, so that it not only helps to understand the data better but to also make it easier to trace which method corresponds to which functional requirement.

Data Dictionary: After we illustrated our data requirements using the class diagram, we used the data dictionary to provide a clear statement of each class along with its attributes and methods. Using data dictionaries, a solid knowledge of what each attribute and what role an attribute plays in a method was attained. This was tabulated with some examples.

UI Prototype: With the UI Prototype, we were able to get a better understanding of our system ourselves and helped us verify whether our system had met our initial business goals as specified by our stakeholders. Also, it helped us in arriving with some of our data requirements and system requirements.

System Requirements: The system requirements helped us get a better understanding of what parts of the functional requirements the system versus the software should deal with. It was also a way for us to get a deeper understanding of how the functional requirements would work.

While doing our specifications, we learnt a lot through doing the P-Language for our quality requirements and our System Requirements. At first, we found it very hard to find a grasp at these specification techniques but, the corresponding lectures regarding those techniques helped us gain a clear insight and implement these specification techniques.

C. Prioritization:

As previously mentioned, we decided to use the 100$ test and ranking for prioritizing our functional requirements. We realized that both the prioritization strategies had their respective strengths and weaknesses, but we feel like the combination of using them both provided us with a good prioritization. Since the project in our case is rather small when compared to industry projects, we felt like the 100$ test would provide beneficial insights as it is not as effective with a large number of requirements.

The 100$ test got us a quick overview of what each stakeholder found to be important (or in our case personas of the stakeholder due to time restrictions). The test provided a high-level understanding of what is important and from there we could use ranking to get a more fine-grained prioritization.

Due to the time restrictions we had to use personas instead of actually meeting our stakeholders meaning that the 100$ test might not be as accurate as it could be. If we were to do the prioritization again, it would be beneficial to include actual stakeholders instead of using personas, since this would give a better view on their actual needs from the system.

For the prioritisation, since we assumed personas for the stakeholders which was very insightful to prioritize based on the different stakeholder's point of view rather than having biased blindness when it comes to prioritizing the requirements ourselves.

D. Traceablity:

Doing, the traceability helped us a lot in gaining knowledge of how tracing each artifacts plays an important role in case there is a change in the business goal or a functional requirement. This also gave a better understanding in how weak links in the traceability can play a huge role in the whole requirements of a system not only from a technical point of view, but also in the business goals. This also gave us insight on how change management is done in the industry.

When we started doing the traceability, we did not have a clear vision on how to trace our requirements and artefacts without any weak links to the goals of the system. But the lecture on the traceability and the knowledge we acquired during the Documentation workshop helped us clear this hurdle and do our traceability successfully.

E. Validation

The group has performed a peer review on another group report by validating their requirement document. By defining and following a checklist the group could focus on the same parts of the document and therefore discuss and share findings within the group. This worked well and helped us with delivering a clear and structured peer review to our peers. By doing this, the group automatically started to reflect about our own document based on what we found in the peer document and based on the checklist. It was helpful to go through a document from another group and to compare structures and specifications, since this led to another perspective to use as inspiration for the project.

The group also received a peer review from another group. This helped us with finding things that were unclear

or missing in our document. When you write something it is helpful to have other people read it since it is hard to know if they interpret something the same way you do. It is also common to become blind to one's mistakes when working with the same document for too long. Therefore, the peer review was valuable, and we were able to improve the document, and also help another group with improving their document. However, this is the first time any of us is working with requirements engineering, and it is probably the same for the other groups. Because of this, we needed to be critical towards the critique we received and think about if changing our document was necessary.

F. Group Dynamics

The group had been working well. We get together and meet at least once a week in person and outside of that we book more meetings whenever needed. As a group, we aim to have as clear communication as possible, and it has been working well. The meetings in themselves did at first not follow any particular structure we just got to work in the hand-ins. However, in the group workshop it was highlighted that sometimes not everyone felt like they knew what to do. To address this, we begun the meetings we had after the workshop always with going around the group and everyone had to say what they were going to do and how it was going, like a daily stand-up in SCRUM. Some of the group members have previous experience with working in an agile way from previous courses and work which made us adapt some of it into our project. This has improved the workflow since it gives the entire group some base knowledge about what is going on and what should be prioritized for the meeting.

Being a diverse group has been very insightful, it was valuable to get the different perspectives on our project. We feel like being a diverse group only has improved a lot in how we did our project.

We did our specifications, prioritization and traceability by distributing the work among ourselves based on the individual's 'expert' knowledge about the techniques that we gained during the Documentation workshop and based on the individual's previous experience in the respective domain and each member's interest in the respective domain. For example, when choosing how to represent our data requirements we opted to go for a class diagram since some individuals in the group had previous experience creating class diagrams. This not only helped a lot in documenting our knowledge efficiently but also helped in learning a lot from each other.

Being any large group is often hard but a lot of workshops in the beginning was about getting to know each other and that helped us get insights into everyone's expectations and limitations. By respecting these expectations and limitations the overall groupwork has been smooth and easy-going.

CONCLUSION

Requirements engineering is an important process in the development cycle that sometimes can take up a lot of time and effort. We have learned about various elicitation, specification and prioritization techniques and understood the advantages and disadvantages of these techniques in software development context. Since every project is different it is important to have a good understanding of the different techniques to be able to use them. This is something we have gained during this project. We also got a broader understanding of them when doing a peer-review of another group's project and seeing how they used the techniques in different ways. With this experience we now feel confident that we could conduct further elicitation in future projects.