Project Experiences

Jonathan Ahlström Ossian Gewert Jacob Jönsson Simon Persson
André Roxhage Felix Sundholm

March 9, 2025

ETSN15 Requirements Engineering

Group Gamma - EasyTrip

Contents

1 Elicitation Reflection

1.1 Context diagram

1.1.1 Experiences

The context diagram was done by the whole group together during an exercise. We learned about this quick and useful elicition technique.

1.1.2 Description of the method

A context diagram was made to elicitate which stakeholders there are and what interactions can exist between actors and the product. The result is a drawing showing the product as a black box, actors and users that interact with eachother and the product. The inner and outer domain is defined by drawing a circle around actors that interact directly with the product.

1.1.3 Why this method?

The method helps to understand what actors and interactions exist around the product. Conflicts of interests between different stakeholders can also be found through this method. It also produces a diagram that can be presented to others as an overview over interactions and potential stakeholders in an understandable way. The diagram can be a useful reference for future discussion and can also be used as a requirement. The method takes relatively short time and can be done in an early state to get everyone involved on the same page. It gets people talking and misunderstandings can be sorted out early.

1.1.4 How it was

The technique showed us that we knew very little about what actors could be involved and how they would interact. It also showed us that we had different ideas about how things would work. We discovered for example that we didn't know exactly how the product could generate revenue. After some discussion, we concluded that there will be two separate sources; ads and referrals to airline companies.

We also had a healthy discussion about which actors were in direct contact with the product. We decided that travel agencies may want to use our product but that it would not be our main target user. It was also concluded that we should offer support and that they would be a stakeholder as well.

One challenge we discovered was that the team was not unified on what entities or actors are supposed to be placed in a certain state. Such as if our ad platform was supposed to be inside or outside of the inner domain or if we want to support travel agencies as well as travellers to use our product or only travellers.

Overall we thought that the method used was very useful in an early stage for elicitation.

1.1.5 What we would do differently

What we recommend for others to do is firstly shorten down your thought process and focus on one aspect at a time. For example, firstly look at direct actors inside the inner domain, and then move on one step at a time to minimize risk of missing components or misplacing entities.

1.2 Virtual window

1.2.1 Experiences

The virtual window method was done by the group together during an exercise.

1.2.2 Description of the method

The data requirements for an object (in a programming sense) are described by making a virtual window of what a form might look like that could be filled in to create any instance of the object. Example data are to be used to make it easier to understand.

1.2.3 Why this method?

The method is quite quick and gives an overview that is easy to understand for all readers. The example data can be much easier to understand than a long description such as in data dictionaries.

1.2.4 How it was

The method was found to be quick and intuitive to use. It helped us elicitate some requirements for a user such as if they would need to give consent to get emails about news from us.

1.2.5 What we would do differently

If anything we probably should have used this method more. For example we didn't delve too deeply into exactly how flight data would be handled since we would get it from the airline companies and we didn't know what that would look like. Still we could have used this method to elicitate some of the data requirements to get a better overview of what might come. The virtual window created this way could later be updated as we learned new things. This would work well since the method describes what data is needed but not how it should be implemented.

1.3 Elicitation

1.3.1 Stakeholder Analysis

The first and main elicitation made was the stakeholder analysis where we identified the stakeholders and their interests. This was done by brainstorming and discussing with the team during exercises in school. We later moved on letter two of the group members to write and deeply analyse the stakeholders.

1.3.2 Traveller Interviews

The second elicitation was of the stakeholder traveller where we interviewed two persons who travels frequently. This was done by two group members and the results were later summarised. We felt that it was a good way of getting a deeper understanding of the user and their needs.

1.3.3 Email Questions - Airlines and Travel Agencies

The third elicitation was the sending of questions by email to the stakeholders airlines and travel agencies. This was done by two group members and the results were later summarised. Happiliy we got a response from one of the airlines, easyJet, which gave us some details but not as much as we hoped for. Although, we felt that the answers were still useful and validated reflections from the stakeholder analysis that was alreadty done. We did ask for a meeting but they declined.

2 Specification Reflection

2.1 Functional requirements

2.1.1 Task descriptions

We utilized task descriptions for the tasks we identified as most essential during elicitation and our ptoject mission work (Finding a cheap flight to a general area and finding a cheap trip destination using the heatmap).

We attempted to use both Task descriptions and "Tasks & Support" (as described by Lauesen, Ch 3.8) to structure the tasks, and decided to use a hybrid of the approaches.

Finding the right level of abstraction was difficult, and we did not initially break down the processes in enough steps to find all the requirements we would later specify.

2.2 Quality Requirements

2.2.1 Importance of Quality Requirements

We noticed early on that quality requirements are just as important as functional requirements. As many know functional requirements exist to explain what the system does while quality explain how well the task should be done. By creating a quality grid we established which quality attributes are crucial for our project, these involve performance, usability, reliability and security.

2.2.2 Experiences and Insights

At the beginning of our project we understood quickly how important proper quality requirements are for our key differentiators. For example, the HeatmapSearch feature is one of our key differentiator, however this feature depends entirely on its performance and usability. From this enlightenment we had to reconsider our functional and quality requirements and reconstruct them which is reflected on further down.

Another insight we had was the gap existing between user expectations and technical implementation on features. Such as when we defined quality requirement using usability score (SUS) we learned to think outside the core product and how users would see and use the product.

2.2.3 Challenges

One major challenge was figuring out what level of detail we should use for the specific quality requirement. For example, at the start some of our requirements where to vaguely defined where there could be misinterpretation. We had some requirements at the start saying "system should show results fast" which is not well defined where different users reads the term fast as different values. We then changed these requirements to have more of a value such as "search results should be displayed within 2 seconds" which can be seen in the quality requirement FastSearchResults.

Another challenged was how we should prioritize our requirements since there is no project who has unlimited resources to make all quality aspects critical. Hence as mentioned before we created a quality grid where we discuss and prioritize what we deemed critical for our project.

2.2.4 What We Would Do Differently

For future projects we would recommend to:

- Involve stakeholders earlier, this would help create testable requirements and minimize the risk of creating requirements that is not needed for the different stakeholders. Such as saying a results should appear in 0.1 ms or 1 ms has no major impact on the user but it would be a major cost for the team to develop.
- Create a clear connection between quality requirements and features earlier on to improve aspects such as traceability and improve the workflow and understanding from developers.
- Create a quality checklist earlier on to help with validation and testing.

2.2.5 Final Reflection

In conclusion we learned that quality requirements its not something that is optional for projects but its essential factor that is needed for success of a product. By having the quality requirements well defined and constructed it creates a way to check if its competitive on the market or if it fulfills users needs and wants.

2.3 Data Requirements

For the data requirements three methods were chosen. These methods were data model, data dictionary and virtual window. The data model was chosen because it is an efficient way of describing how the user accounts should be set up in the product. The data dictionary was chosen since it describes entitites in the product in a clear way, in our product the user was described in the data dictionary which made it very clear what the user was in the system. These methods were efficient in conveying the data used in the system since you could clearly see what data and entities are used in the system and how they are used.

For this project one experience was thatmaking data requirements are more or less difficult depending on what part of the project is developed. The data requirements used were thus targeting the user account system in the product as well as the search system for the flights. These are the crucial parts that are the most important to include in early releases. Other areas such as the ad providers or how the data from airlines should be handled are harder to construct and a reason for this is a lack of knowledge in how this data would be represented from these sources, this would be improved over the release iterations in the project over time. Another point that was explored was in which level the E/R-model was placed. Initially it was placed in the design level but since data requirements usually are similar across the different levels it was chosen to be moved up to domain level instead in order to not interfere with the design level implementations.

3 Validation Reflection

3.1 Validation List

What did we learn from making the validation list? Writing a report when validating someone elses work? What feedback did we receive?

3.2 Prototyping for Validation

Prototyping can play a crucial role in validating our concept for EasyTrip, ensuring that both usability and functionality aligned with user expectations. Since we had already validated the idea during the elicitation phase, the prototyping stage focused on creating a representation of the needs and requirements identified.

3.2.1 Prototyping

First, we developed rough sketches to outline the core functionalities and user flows. These sketches helped visualize the interactions between users and system components early in the process. Our team discussions refined these ideas before transitioning into digital prototyping.

Next, we created wireframes using Figma to map out key user journeys, including searching for flights, filtering results, and accessing booking options. To simulate interactions more realistically, we wanted to build an interactive prototype incorporating dynamic elements. However, due to time constraints, it resulted in a static prototype without interactions.

Despite not being able to test features, the static prototype allowed us to evaluate the overall layout, navigation, and content structure internally. Feedback gathered from team members provided insights into information clarity and flow, helping us refine key design elements, as well as made sure the team where on the same page and has common expectations of what we were building. The prototype also enabled us to align our vision with stakeholder expectations before committing to full-scale development.

3.2.2 Future Directions and Improvements

To enhance future validation efforts, we propose incorporating interactive elements in the prototypes. One approach is to use dummy data to create a high-fidelity prototype with basic search and filter interactions, allowing for more realistic user testing.

Expanding usability testing to include a diverse group of stakeholders, such as frequent travelers, travel agencies, and airline representatives, would provide a more comprehensive understanding of user needs and potential system constraints.

Additionally, A/B testing different UI layouts, such as list-based search results versus heatmap visualizations, would further refine the interface based on user preferences.

By adopting these improvements, we can bridge the gap between conceptual validation and real-world usability testing, ensuring a more robust and user-centered development process before full-scale implementation.

4 Overall Challenges

4.1 Restructured requirements specification

During the development of our requirements specification, we initially structured the document by categorizing requirements into distinct sections such as functional, quality, and data requirements. While this approach provided a clear separation of requirement types as we learned about them in the course, we encountered difficulties in effectively contextualizing them within specific features and functionalities of the system.

To address this, we restructured our requirements specification into sections based on key system areas, such as the heatmap, security, and search functionalities. This new structure allows all relevant requirements, regardless of type, to be included within the appropriate section, making it easier to see how different levels of requirements interact within a particular feature. By organizing the document this way, we have improved traceability, making it more intuitive to understand how individual requirements contribute to the overall system design and objectives. Additionally, this restructuring has resulted in fewer cross-references, simplifying navigation and reducing the complexity of managing related requirements.

Also, we updated our requirement naming convention from numbered identifiers to camelCase IDs. This change enhances clarity and consistency across the document while improving readability and maintainability. The camelCase format ensures that requirement identifiers are more descriptive and making cross-referencing simpler when working with different system components.

These refinements have significantly enhanced the structure and readability of our requirements specification, ensuring that it better supports both development and validation processes.

4.2 Release planning and prioritization

We used three different methods for prioritazation, benefits cost analysis, priority classification and prioritazation though constraint-based release Planning in RecT. These methods were combined in order to create a iterative release plan that maximised value while costing as little as possible. To use only one of these would make the release planning more susceptible to inconsistencys. This is why it was important for us to make the techniques independent from each other.

One of the main difficulties was balancing the needs of different stakeholders while also managing dependencies between features. Many subjective estimations can be harmful for the result since the prioritaztion is not made in a systematic way. Manually handling these dependencies proved too complex which is why we decided to use the tool RecT to automate the process. The score of the releases and features that RecT gave us were somewhat ambiguous since fictitious estimates were used in benefit and cost. This made the result an abstract number which actually was negative sometimes. Even though this part of the result was not desirable, the tool provided a useful relative comparison between features. The release plan RecT generated, with realeses R4, R5 and R6, seemed realistic given our features and constraints. Looking back, we would have benefited from using RecT earlier in the project. This would have given us more time to refine our dependencies and prioritize features more effectively.

Benefits cost analysis was chosen because it provides a quantifiable way to compare features based on benefit and cost. This was a good way further analysize our stakeholders, noticing how their needs can differ and sometimes contradict each other. While travels value usability, the travel agency might prioritize safely for example. A scale of only three options, “Must-Have”, “Important” & “Nice to have”, was chosen to was chosen to avoid unnesesary complexity and ensure a clear prioritization.

Constraint based priorizitaion seemed like a logical way to make the solution space smaller which would lead us closer to a more realistic release plan. After manually connecting different types of dependencies, “Precedence”, ”Coupling” & ”Excludes”, to features, we discussed how the types of dependencies varied from each other and when each should be applied. By discussing this our understanding out our insight on how the features relate to each other got more clear. We also discussed how the complexity would be much larger in a real world scenario as every dependency would be more important to specify.

Individual Contributions

Group Member	**Contributions**
Jonathan Ahlström	Prototype. Requirements. Release plan. Requirements structure. Release plan and prioritization reflections
Ossian Gewert	Stakeholder management. Document structure. Requirements. Requirements structure.
Jacob Jönsson	Data Requirements. Data Requirements Reflections. A few Requirements.
Simon Persson	Requirements. Minor Release plan. Requirements structure. Quality grid. Quality requirements reflection, Presentation
André Roxhage	Elicitation. Prototyping. Release plan & prioritization.
Felix Sundholm	Virtual window reflection, context diagram reflection, presentation slides & presentation