Examensarbete

Developing a web application: a usability approach

av

Sophie Joelsson

LIU-IDA/LITH-EX-A--15/048--SE

2015-06-30
Examensarbete

Developing a web application: a usability approach

av

Sophie Joelsson

LIU-IDA/LITH-EX-A--15/048--SE

2015-06-30

Handledare: Lars Ahrenberg
Examinator: Marco Kuhlmann
Table of Contents

Table of Contents .................................................................................................................. 3
List of figures ......................................................................................................................... 5
List of tables ....................................................................................................................... 7
Abstract .............................................................................................................................. 9
Acknowledgment ................................................................................................................ 10

Chapter 1  Introduction ...................................................................................................... 11
1.1  Goal .......................................................................................................................... 11
1.2  Purpose ..................................................................................................................... 11
1.3  Problem description ............................................................................................... 11
1.4  Limitations ............................................................................................................... 12

Chapter 2  Theoretical Background ................................................................................. 13
2.1  Translation quality assessment .............................................................................. 13
2.2  Usability development ............................................................................................ 13
2.2.1  What is usability? ............................................................................................... 13
2.2.2  Designing for usability ..................................................................................... 14
2.2.3  Prototyping ......................................................................................................... 15
2.2.4  Personas and scenarios ..................................................................................... 15
2.2.5  Use cases ............................................................................................................ 15
2.2.6  User tests ........................................................................................................... 15
2.2.7  Interviews and questionnaires ........................................................................... 16
2.2.8  Heuristic evaluation ......................................................................................... 16

Chapter 3  Method ............................................................................................................. 17
3.1  Literature ................................................................................................................. 17
3.2  Deciding on design process .................................................................................... 17
3.3  Establishing requirements ....................................................................................... 17
3.4  Prototyping ............................................................................................................... 18
3.4.1  First prototype ................................................................................................... 18
3.4.2  Second prototype .............................................................................................. 18
3.4.3  Third prototype .................................................................................................. 18
3.5  Choosing an evaluation strategy ............................................................................. 18
3.5.1  Using the DECIDE framework ....................................................................... 18
3.5.2  Determine the overall goals ............................................................................ 19
3.5.3  Explore the specific questions to be asked .................................................... 19
3.5.4  Choose the evaluation paradigm and techniques ......................................... 19
3.5.5  Identify the practical issues .............................................................................. 20
3.5.6  Decide how to deal with ethical issues ............................................................ 21
3.5.7  Evaluate, interpret and present the data ......................................................... 21
3.6  Web technologies used........................................................................................... 21
Chapter 4  Implementation............................................................... 23
  4.1  Establishing requirements....................................................... 23
    4.1.1  Requirements given by client........................................... 23
    4.1.2  Studying existing tools ............................................... 23
    4.1.3  Use cases ................................................................. 24
    4.1.4  Requirements breakdown ............................................. 29
  4.2  Prototyping ........................................................................ 30
    4.2.1  The first prototype ........................................................ 30
    4.2.2  Evaluation of the first prototype ..................................... 31
    4.2.3  The second prototype ..................................................... 32
    4.2.4  Evaluation of the second prototype .................................. 33
    4.2.5  The third prototype ........................................................ 33
    4.2.6  Evaluation of the third prototype ..................................... 38
Chapter 5  Results ....................................................................... 41
  5.1  Requirements fulfillment ........................................................ 41
  5.2  User evaluations ................................................................. 43
Chapter 6  Discussion .................................................................. 45
References  47
List of figures

Figure 3-1 An iterative design process ........................................................................................................17
Figure 4-1. Use cases diagram with available functionality ........................................................................25
Figure 4-2. Activity diagram showing the steps to upload a translation file .................................................26
Figure 4-3. Activity diagram showing the steps to upload an error taxonomy file .....................................26
Figure 4-4. Activity diagram showing the sequence to create a project ......................................................27
Figure 4-5. Activity diagram showing the sequence of evaluating the chunks of a project .........................28
Figure 4-6. Activity diagram showing the sequence to export a project to the user ..................................28
Figure 4-7 The chunk’s color becomes green when selected ........................................................................31
Figure 4-8 Several improvements were made in the second prototype .....................................................33
Figure 4-9. Overview page ..........................................................................................................................34
Figure 4-10. Project Overview Page ...........................................................................................................34
Figure 4-11. New Project Page ..................................................................................................................35
Figure 4-12. Review Page ..........................................................................................................................36
Figure 4-13. Upload Translation Page .........................................................................................................36
Figure 4-14. Upload Hierarchy Page ...........................................................................................................37
Figure 4-15. Export Page ..............................................................................................................................37
Figure 4-16. Export Project Page ................................................................................................................38
List of tables

Table 1 Requirements ........................................................................................................... 23
Table 2 Derived requirements .............................................................................................. 30
Table 3. Summary of observations for prototype 1 ................................................................. 32
Table 4 Summary of observations or prototype 3 ................................................................ 39
Table 5 Requirements fulfillment .......................................................................................... 43
Abstract

The objective of this thesis was to develop a user-friendly web application for quality annotation of text translations. Prototypes were designed in an iterative process to allow for continuous feedback from potential users. The iterative method drew attention to the frequently occurring difference between the developer’s and the user’s perception of the design, and was helpful to meet the high usability requirements. This thesis’ work included users throughout the process with user evaluation of each prototype, to ensure the usability of the final product.
Acknowledgment

I would like to start by thanking my project supervisor Lars Ahrenberg for his support and feedback during this process. I would also like to thank my examiner Marco Kuhlmann for his thoughtful input. I would also like to thank the people who volunteered for the user tests. Last but not least, I would like to thank my family and friends for their warm support during the project.
Chapter 1

Introduction

1.1 Goal

The goal of this thesis is to develop a web application to assist human annotators in the process of evaluating translation quality. The application user interface should support the user to follow a well-defined work flow for assessment of translation quality in an effective and usable manner.

1.2 Purpose

The purpose of this thesis is to investigate how to implement the design process to ensure a good usability of the application. Using the web application introduced in 1.1 as a case study, an attempt will be made to find a proper method for achieving usability. The aim is to find a way to develop a technical system that is designed primarily for human users. To be able to do this, the following questions need to be considered:

- How can human users be involved in the design process?
- How can usability be measured?
- How can the design process be broken down to keep a focus on the human user throughout the process?

In the end, the hope is that a better understanding for the human role in interface design has been reached.

1.3 Problem description

With the development of machine translation systems comes a need for evaluating the translation quality. The evaluation can be used as feedback during development and for comparing translation systems. One way of doing this is to let humans evaluate a given translation with regard to certain language aspects.

Since there is never a single correct translation, it cannot be marked as simply “correct” or “erroneous”. A more fine-grained notation is needed and a number of different taxonomies exist that can be used for this purpose. Error taxonomies are often hierarchical, i.e. they have a tree structure where an error type may have subtypes on several levels.

Research has shown that one problem with using these methods is maintaining a high degree of inter-annotator agreement, i.e. whether different annotators agree on the classification of an issue in the text (Lommel et al., 2014). For this reason, a tool used for translation evaluation should support the user in using the chosen issue taxonomy in a way that ensures high inter-annotator agreement.
There are many things to be considered when developing an evaluation system. This includes the choice of issue taxonomy and data, comparison to existing tools, and text presentation strategy. For this project, an initial request from the client was for the translation to be evaluated chunk by chunk. A chunk is a group of words that forms a sentence or part of a sentence. Ahrenberg (2014) suggests translation evaluation at chunk basis to help annotators define the scope of an error found.

This project is an attempt to develop an application for translation quality annotation that takes these factors into account. However, this thesis will focus on implementing a process mainly to ensure the usability of the application.

When discussing this project with the client, the suggestions below were expressed about the outcome, and the decision was made to focus on the design of the user interface.

1. The application should be web-based.
2. The application should present a translated text, divided into chunks, to the user.
3. It should be easy to choose a category without looking through all the levels when using a hierarchical error taxonomy.
4. The chunks should be evaluated individually and each one will be assigned a status of correct or incorrect.

1.4 Limitations

This report will not cover the technical details regarding the web technology used, more than what is needed to understand the usability development process. This report will focus on the usability of the application, and not consider performance issues like loading time and resource handling.
Chapter 2
Theoretical Background

2.1 Translation quality assessment

A variety of methods exist to assess quality of machine-translated text. But since there is never a single correct translation, various variables need to be considered, e.g. the target audience and translation purpose. According to Lommel et al. (2014), there are basically four ways to use human insight into the quality of a translation generated by a machine translation system. These are:

1. Human-generated reference translations
2. Rating of MT (machine translation) output based on perceived quality
3. Post-edits of MT output (implicit error markup)
4. Explicit error markup of MT output

It is desirable that an assessment is reproducible to be found reliable. The degree in which humans perceive the translation quality can be measured by inter-annotator agreement, i.e. whether two humans assign the same issues to a translation when using the same issue taxonomy.

When experimenting with inter-annotator agreement by letting a number of different annotators annotate the same translation, Lommel et al. (2014) found that it is common for annotators to disagree over the scope of an issue, i.e. which words it spans over.

Ahrenberg (2014) proposes a solution for better quality assessment by chunking the text prior to evaluation. A chunk is a group of words that equals to a sentence or a part of a sentence. In this way the error can be located with higher precision. Annotators do not have to define the scope of an issue themselves, but are assigning issues to a pre-defined chunk.

This chunking also helps when building statistics and each chunk can be counted as a separate unit.

One method for implementing translation quality assessment is The Multidimensional Quality Metrics (MQM). This method has been developed by the QTLaunchpad project, which was an European Commission-funded collaborative research initiative involving researchers from academia as well as industry. MQM provides issue type definitions and methods for using them (Lommel et al., 2013).

2.2 Usability development

2.2.1 What is usability?

The cognitive sciences aim to explore and explain how the human mind works. This involves several disciplines, including psychology, neuroscience, behavioral science and computer science. The human mind handles decision making, planning, intelligence among many other functions.
For this reason, when presenting information for humans to process and act upon, the mind’s cognitive functions must be taken into account. This is the purpose of usability design. Users are consulted regarding how the information and concepts should be presented to make the product usable. Human minds keep mental models about how the world is constituted and should be approached in different situations (Preece et al., 2002).

But what does it mean for a product to be usable?

Nielsen (1993) defines usability as not only one design property, but as a combination of aspects of the user’s experience:

- **Easy to learn**: The user can quickly go from not knowing the system to getting some work done with it.
- **Efficient to use**: Once the user has learned the system, a high level of productivity is possible.
- **Easy to remember**: The infrequent user is able to return to using the system after some period of not having used it, without having to learn everything all over.
- **Few errors**: Users do not make many errors during the use of the system, or if they do make errors they can easily recover from them. Also, no catastrophic errors should occur.
- **Pleasant to use**: Users are subjectively satisfied by using the system; they like it.

### 2.2.2 Designing for usability

Designing for usability is about envisioning and seeing the world from other people’s perspectives. Things and phenomena in the real world need to be represented in the product in a way that is understandable for humans. To find suitable representations, the developer or designer must try and get into the user’s head. There are many techniques for accomplishing this.

Preece et al. (2002) suggest four steps in a design process. These include:

- identifying user needs and requirements
- developing alternative designs
- building interactive versions
- evaluating the design and user experience throughout the process

A common way to carry out usability design is to use an iterative design process. When a design proposal has been developed, it is evaluated. The evaluation results are used as input to a reworked design, which is again evaluated. This cycle is repeated a number of times until the product is considered usable. Nielsen (1993) recommends an iterative design process for the reason that he considers it impossible to design an interface without any usability problems from the start.

In general an evaluation process can be described as a number of steps and issues to take into account. Preece et al. (2002) give an example of such a framework. This framework is referred to as “DECEIDE” and is based on six steps:

- Determine the overall goals
- Explore the specific questions to be asked
- Choose the evaluation paradigm and techniques
- Identify the practical issues
- Decide how to deal with ethical issues
- Evaluate, interpret and present the data
2.2.3 Prototyping

Prototyping is a key activity of usability design. According to Preece et al. (2002), a prototype is a limited representation that allows users to interact and explore the design's suitability. The purpose of the prototype is to enable testing of ideas and facilitate discussions between developers and users.

When talking about the quality of a prototype, it is done in terms of fidelity. A high-fidelity prototype refers to a prototype that has a design close to the end product, both when it comes to technology and from the look and feel of the design. A low fidelity prototype refers to a rough sketch prototype. Typically, a low-fidelity prototype is a hand-drawn sketch presenting a simple design and no other interaction possibilities.

The advantage of high-fidelity prototypes is that they allow for a high degree of interaction. When using high fidelity prototypes, users may view the design as set and are not as willing to criticize the main structure of the design, but only the details. The advantage of low-fidelity prototypes is that they enable rapid iteration. The simplicity of the prototype makes the user more willing to suggest drastic changes to the design and without being concerned about the impacts this will have for the designer (Johansson and Arvola, 2007). There is a number of tools and applications that can be used to create prototypes for web design with different levels of fidelity. These tools provide the designer with possibilities to speed up the prototype development process.

2.2.4 Personas and scenarios

It can be helpful for the designer to begin the prototype development process by envisioning the practical use of the product. This can be done by visualizing user scenarios and personas that potentially could be used in the web design.

Personas are useful to paint a picture, for example, the age and background knowledge of the average user. For example, a library search engine will likely have a higher age average than a social network app. Properties of the typical user can be shown in a persona, which is a description of an individual who might be using the product. There may also be several personas for the application. Personas shall be broadly described and it is best to restrict the number to 3-4 personas (Usability.gov, 2015c)

Scenarios are real life situations in which the product might be used. The purpose of scenarios is to put the product into a context. A scenario describes a potential situation in which a user would use the product. What reason does she have for visiting the web site, where is she, and what does she expect to achieve?

2.2.5 Use cases

Use cases are more specific descriptions of which tasks a user should be able to perform using the product. These are closer to the technical implementation of the product, than are personas and scenarios. They describe the flow of a task, which steps need to be performed to complete the task, and the pre- and post-conditions of the task.

2.2.6 User tests

User participation feedback is most commonly used when evaluating a prototype. The users are asked to do a certain task on the prototype, during which they are observed or asked about the experience. There are several methods to conduct the user participation experience, including interviews, questionnaires and by observation. It is important to determine an appropriate method for the prototype development.
When performing a user test, a potential user is given access to the prototype under supervision, and is asked to perform a specific task. Observing how the user handles the task can give valuable feedback about the prototype usability.

There are several strategies for observing the test users. One option is to record the user by video or audio. Another option is note-taking on a computer or by using paper. Mueller and Oppenheimer (2014) show that note-taking using pen and paper forces the note-taker to process the information, which promotes a more conceptual understanding.

### 2.2.7 Interviews and questionnaires

Interviews can be included as part of the user tests, or as an independent activity. Interviews can give more in-depth information of the user’s needs. Another way to ask users for their opinions is to use questionnaires.

A more specific type of questionnaire is the System Usability Scale (SUS) (Brooke, 1986). The test user is allowed to directly score the design according to a list of user-relevant criteria. The SUS consists of ten statements that can be ranked on a 5-point scale, where 1 corresponds to “Strongly disagree” and 5 corresponds to “Strongly agree”.

**System Usability Scale (SUS) statements:**

1. I think that I would like to use this system frequently.
2. I found the system unnecessarily complex.
3. I thought the system was easy to use.
4. I think that I would need the support of a technical person to be able to use this system.
5. I found the various functions in this system were well integrated.
6. I thought there was too much inconsistency in this system.
7. I would imagine that most people would learn to use this system very quickly.
8. I found the system very cumbersome to use.
9. I felt very confident using the system.
10. I needed to learn a lot of things before I could get going with this system.

### 2.2.8 Heuristic evaluation

Heuristic evaluation is a type of expert evaluation where someone with interaction knowledge scores the design according to a list of principles and issues (Nielsen, 1995).

Heuristic evaluation involves having a small set of evaluators examine the interface and judge its compliance with a set of usability principles (the “heuristics”).

Heuristic evaluation is a form of evaluation where an expert judge the application usability by recognized principles or experience. That is why it is called heuristic evaluation. Nielsen recommends in an article "How to Conduct an Heuristic Evaluation" three to five evaluators in order to catch most usability issues. The evaluators work through the application independently and fill in evaluation forms where they relate to usability principles (Nielsen, 1995).
Chapter 3

Method

3.1 Literature

A literature study was done in two areas: translation quality assessment and usability design. The literature was chosen by recommendations from my supervisor and from previous course literature. From these resources, I have used the reference lists to find further reading.

3.2 Deciding on design process

When deciding which design process to use, the suggestions from 2.2.2 were taken into account. Considering the suggestion from Preece et al. (2002) in planning a design process, the first step is to establish requirements, followed by developing prototypes and evaluating them using various forms of user participation. An iterative design was used as recommended by Preece et al. (2002), and the prototyping was repeated after evaluation.

Figure 3-1 shows a graphical representation of the workflow. The first step was defining the requirements. The prototyping and evaluation was repeated three times, after which there was a final product.

![Figure 3-1 An iterative design process](image)

3.3 Establishing requirements

Establishing requirements in the beginning of the design process is important as an agreement between developer and client, and to state what will be tested before delivery. Requirements can be either functional or nonfunctional. A functional requirement is a requirement that is possible to test, and a nonfunctional requirement is not possible to test. Good requirements should not be contradictory or ambiguous.
When establishing requirements, the first action should be to talk to the client to get an idea of what is being asked for.

Preece et al. (2002) emphasize the importance of identifying the stakeholders’ needs. It may be achieved by studying behavior and support tools from competitors’ products and earlier versions of the product under development. Given this, a study of some existing tools for translation quality assessment will be performed. These tools will be BLAST (Stymne, 2011) and translate5 (MittagQI, 2015d).

### 3.4 Prototyping

#### 3.4.1 First prototype

For the first prototype, the main focus was on two issues. Partly, to limit the scope of the interview and the following analysis, and partly to enable sufficient iterations in the prototype process within the time frame.

The two issues of concern when designing the first prototype were:

- According to the requirements from the client, it shall be possible to choose an error category from a hierarchical taxonomy, without having to scroll down the entire list of categories.
- The set of chunks under evaluation should be presented in such manner that it is clear to the user what is to be done. This includes whether the chunk has been evaluated, and if so, how it was categorized.

The first prototype was a mid-fidelity prototype made as a simple HTML-page. The reasons for choosing this level of fidelity were:
1. Give the possibility of simple interaction, to test the design for selecting an issue type from a taxonomy.
2. Lack of time to learn a specific tool for prototyping.

Walker et al. (2002) show that there is little or no significant difference in usability issues found, depending on whether the prototype is made on paper or computer in this stage.

#### 3.4.2 Second prototype

The second prototype will be of the same type as the first prototype, with some design changes following the evaluation results from the first prototype.

#### 3.4.3 Third prototype

The third prototype will be a prototype of high fidelity, developed using the same techniques as would be used for the final product.

### 3.5 Choosing an evaluation strategy

#### 3.5.1 Using the DECIDE framework

To plan the prototype evaluations, I used the DECIDE framework introduced in 2.2.2.
Different strategies were chosen for the prototypes, to cover a range of potential issues. For the first prototype, the strategy was to observe two users while they perform a task in the application. For the second prototype, the strategy was a demonstration of the web site to the client. The last prototype evaluation strategy was chosen to be more comprehensive. This strategy both included observation of the test users and a heuristic evaluation questionnaire.

### 3.5.2 Determine the overall goals

The overall goal was to find a usable design, which fulfills the requirements. Besides this, additional goals were determined to be considered for each prototype evaluation.

**Purpose for the evaluation of the first prototype:**

1. Verify with client that the project requirements have been understood correctly.
2. Test usability to help iterate a useful second prototype.
3. Test interaction design concepts.

**Purpose for the evaluation of the second prototype:**

1. Receive feedback from client to iterate the final prototype.

**Purpose for the evaluation of the final prototype:**

1. Test interaction design details.
2. Estimate the usability of the final prototype.

### 3.5.3 Explore the specific questions to be asked

This section revolves around what information is wanted from the evaluation process. These include:

- Is the presentation of the chunks clear?
- Does the user understand how to assign an issue to a certain chunk?
- Does the user understand how to perform other tasks such as file upload and project creation?
- Are there any bugs or lack of functionality?

### 3.5.4 Choose the evaluation paradigm and techniques

The evaluation techniques were planned before the project start. The three prototype evaluations were different, but some components were recurring. Relating to Nielsen (1993) definition of usability mentioned in section 2.2.1, four of the usability aspect were evaluated. These were easy to use, easy to remember, few errors and pleasant to use. Whether the application was efficient to use was not evaluated in any iteration, since it would demand involving larger tasks given to the test users, which there was not time for.

**For the first prototype,** the evaluation technique was user tests, as described in section 2.2.6. The test users tested the application and were then interviewed. There were two test users: the client, with a high understanding in the subject, and a researcher at the university, who was recommended by my client as a suitable test candidate.

**For the second prototype,** the evaluation technique was to simply demonstrate the prototype for the client, with the goal of adjusting verified design concepts.

**For the third prototype,** the evaluation technique was user tests, as described in section 2.2.6, involving two test users who were previously not familiar with translation quality annotation nor with test evaluation systems. Also, the third prototype was evaluated using the SUS scale, described in section
2.2.7. The reason for this was to perform a final evaluation to get an idea of the overall usability of the product.

3.5.4.1 **Strategy for easy to learn**

The strategy selected for evaluation of Nielsen’s criterion *easy to learn* was to observe and ask questions to the users from the first and third prototypes. This due to the reason that it was first time users evaluating the first and third prototype. I based the evaluation on a questionnaire, observation for the time it took for the user to complete the task given and questions the users asked during the task.

3.5.4.2 **Strategy for efficient to use**

The evaluation of *efficient to use* was discarded as the end user would not be using the end product during the time frame of this thesis. It can however be interesting to continue this work with this in mind, where the end user is involved.

3.5.4.3 **Strategy for easy to remember**

The strategy for evaluation of *easy to remember* was to observe the recurring user for the second prototype. She had used the first prototype and the plan was to evaluate how familiar she was with the controls to annotate a translation for the second time, compared to the first time. There were changes in the visual layout for the first and second prototype, but some functionality was the same, and that is what I based this criterion on.

3.5.4.4 **Strategy for few errors**

The strategy for evaluation of *few errors* was to continuously resolve issues during development of the prototypes that I found during my testing of features. Also, it was my ambition to resolve all issues found during the user tests as well as feedback from the users within the time frame given. Focus was on the users’ issues to remove the possibility to make mistakes with the application.

3.5.4.5 **Strategy for pleasant to use**

The strategy for evaluation of *pleasant to use* was to conduct interviews and collect general feedback from the user tests. Mainly I aimed for a more positive feedback from the third prototype compared to the first, as well as a SUS questionnaire for the last evaluation.

3.5.5 **Identify the practical issues**

There are several practical issues to take into account for the evaluation process, such as finding appropriate test users with different background knowledge. Therefore, different test users were chosen for different purposes. For the first prototype, users from the research community were included to evaluate core functionality. For the last prototype, users without direct connection to the project were chosen to evaluate the overall usability and interactive design.

Additionally, facilities, equipment, budget and time constraints were taken into account. A suitable facility for the user tests had to be found, and my supervisor helped with a room at the university. Due to practical reasons, my computer was used for the first user tests. The project’s time limitation had to be taken into account when developing the project plan. Additionally, limitations had to be continuously added throughout the project since there are endless possibilities for improvements with most projects. There was no budget for the project and test participants had to be found among friends, family and in the university community.
3.5.6 Decide how to deal with ethical issues

The only ethical issue identified is the anonymity of the users participating in the user tests. For the sake of keeping all users anonymous, the choice was to consequently use only female pronouns when referring to users in this report, regardless of what pronouns these users would choose to describe themselves.

3.5.7 Evaluate, interpret and present the data

The results from the evaluations will be presented in section 4.2.2, 4.2.4 and 4.2.6. A summary of the user tests will be provided along with questions prepared and answered about user actions and opinions. Additionally, the results for the SUS evaluation will be presented.

3.6 Web technologies used

The technical setup of the application is based on a mean (Mongo, Express, Angular, Node) stack, where the two most popular mean stacks are www.mean.io and www.meanjs.org. I started with a slim setup that I fetched from github (Angular Express Seed, 2015a).

The input and output files from the application have JSON annotation (JavaScript Object Notation). This as it is the format that is easiest to work with in Javascript.

Every change the user makes during the work will be saved in the database. This means that the application can be closed and opened at a later time without loosing any work. Exporting a project will not remove it from the database, but will export the current snapshot of the work.
Chapter 4

Implementation

4.1 Establishing requirements

4.1.1 Requirements given by client

These are the requirements given by the client:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The application should be web-based.</td>
</tr>
<tr>
<td>2</td>
<td>The application should present a translated text, divided into chunks, to the user.</td>
</tr>
<tr>
<td>3</td>
<td>It should be easy to choose a category without looking through all the levels when using a hierarchical taxonomy.</td>
</tr>
<tr>
<td>4</td>
<td>The chunks should be evaluated individually and each one will be assigned a status of correct or incorrect.</td>
</tr>
</tbody>
</table>

Table 1 Requirements

4.1.2 Studying existing tools

As previously mentioned in section 3.3, it is of value to explore alternative designs when developing the usability. This could be done by examining both competitors’ designs and your own earlier versions. Two existing alternative designs for annotating translations were examined: BLAST (Stymne, 2011) and Translate5 (MittagQI, 2015d).

4.1.2.1 BLAST

Blast is a Java application based on Swing, developed by Sara Stymne at Linköpings University for the Department of Computer and Information Science. Blast is built to handle error and support annotations where the error annotations is based on a hierarchical error typology. The tool provides three working modes, annotate, edit and search, where the default is annotate (Stymne, 2011).

When assessing the usability of the application, the drawbacks I find are that it is necessary to read the README file for using the application, and it is difficult to get an overview of the uploaded issue taxonomy.

According to Stymne (2011):

*BLAST can aid MT researchers and users in this process, by providing an easy-to-use graphical user interface. It is designed to be flexible, and can be used with any MT system, language pair, and error typology.*
BLAST is built upon modularization so each plugin will give more options to select from the user. This also makes it more difficult for the user. I limit the scope to handle only error annotations, compared to BLAST that handles support annotations as well.

The observations I made about BLAST made me come to the following conclusions regarding my application.

The application should be kept simple. I will use a minimalistic approach, and leave possible extensions for future work. To make the product as usable as possible, within the time frame, I will focus on a small set of functionality. The core functionality will be error notation of a chunked translation.

The application should not be extended to more functionality than error annotation to keep it as a specialized tool. However, it will be built using a technique that makes it extendable, so more functionality may be added “behind the scenes” without demanding action from the user.

The application should be extendable, and below functionality is desirable:

- chunking of raw documents.
- statistical evaluation of translations.

4.1.2.2 Translate5

Translate5 is an open source web application for proofreading (MittagQI, 2015d). I tried the demo version and specifically one thing stroke me as problematic in this application. It was not clear how to use it and there was a lack of embedded help provided. There is a lot of features pressed into small scope and it is difficult to get a good overview.

The main conclusion drawn from my observations of translate5, is basically the same as the conclusion from BLAST in section 4.1.2.1– to keep a simple design and avoid adding more functionality than is necessary for performing the core task.

4.1.3 Use cases

From the information given in 4.1.1 and 4.1.2, I worded these use cases as seen in Figure 4-1:
Figure 4-1. Use cases diagram with available functionality.

Each of these use cases will be described in the following subsection.

4.1.3.1 User uploads a translation document

Flow:

1. User selects a translation document to upload.
2. User clicks on the upload translation button.

Post conditions:

• The translation is uploaded to the webpage.
4.1.3.2 User uploads an error taxonomy file

Flow:
1. User selects an error taxonomy file to upload.
2. User clicks on the upload taxonomy button.

Post conditions:
- The error taxonomy is uploaded to the webpage.

4.1.3.3 User creates a project

Precondition:
- The user has uploaded a translation document.
- The user has uploaded an error taxonomy file.

Flow:
1. The user selects the document.
2. The user selects the error taxonomy.
3. The user writes down a name for the project.
4. The user clicks on the create project button.

Post conditions:

- A project is created for use in the web page.

![Activity diagram showing the sequence to create a project.](image)

**Figure 4-4.** Activity diagram showing the sequence to create a project.

### 4.1.3.4 User reviews a project

**Precondition:**
- The user has created a project.
- The project is selected for review.

**Flow:**

1. The user selects a chunk to annotate.
2. The user evaluates the chunk.
   a. The user accepts the translation of the chunk.
   b. The user reports issues to the translation of the chunk.
3. User selects another chunk, then repeats from 2

**Alternative flow:**

1. The user selects a chunk to annotate.
2. The user evaluates the chunk.
   a. The user accepts the translation of the chunk.
   b. The user reports issues to the translation of the chunk.
3. There are no more chunks to review.

**Post conditions:**

- The project is reviewed.
4.1.3.5 User exports a project

Precondition:

- There is a project available to export.

Flow:

1. The user selects the project to export.
2. The user clicks on the export button.

Alternative flow:

1. The user selects the project to export.
2. The user changes the name of the exported project.
3. The user clicks on the export button.

Post conditions:

- The project is exported as a file to the user.
4.1.4 Requirements breakdown

Four requirements were initially established (4.1.1). A fifth group of requirements has been added by me.

1. The application should be web-based.
   This is a rather broad requirement, and needs to be evaluated together with the other requirements to be able to get a clearer vision of what the customer wants. For instance, a plain HTML page with frequent reloads would not be acceptable.

2. The application should present a translated text, divided into chunks, to the user.
   This may be interpreted in many ways, and there are some questions that need to be answered. How does the text get to the webpage? Is it the same translated text all the time? How is it chunked?

3. It should be easy to choose a category without looking through all the levels when using a hierarchical issue taxonomy.
   The question that arises is what is “easy”? This is subjective and depends on the level of expertise, background, application familiarity and so on.

4. The chunks should be evaluated individually and each one will be assigned a status of correct or incorrect.
   If incorrect, reason must be given by the user.

5. I also added some additional requirements in order to make the application functional from the beginning to end. These can be seen below in Table 2.

In Table 2, the requirements are broken down. A “shall” requirement, is a requirement that must be fulfilled. A “should” requirement, is a requirement that is desirable, but not necessary, for the final product.

<table>
<thead>
<tr>
<th>Derived from</th>
<th>Lower level requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The application shall be web based.</td>
</tr>
<tr>
<td>1</td>
<td>The application shall have a responsive design.</td>
</tr>
<tr>
<td>1</td>
<td>The application shall work on PC devices supporting at least 800*600 resolution.</td>
</tr>
<tr>
<td>1</td>
<td>The application should work on mobile devices.</td>
</tr>
<tr>
<td>1</td>
<td>The application shall update asynchronously. - i.e. reload part of page without reloading the entire webpage.</td>
</tr>
<tr>
<td>2</td>
<td>The application shall show the original text to the user.</td>
</tr>
<tr>
<td>2</td>
<td>The application shall show the translated text to the user.</td>
</tr>
<tr>
<td>2</td>
<td>The text shall be divided in chunks.</td>
</tr>
<tr>
<td>2</td>
<td>The user shall be able to upload a translation to the web page.</td>
</tr>
</tbody>
</table>
The chunks shall be paired correctly with the original and translated text.

The application shall be able to pick a category from a hierarchical issue taxonomy.

The application should present categories in a dropdown menu.

The application should provide a search field for categories.

The application search field should provide auto completion.

The application should present categories from a tree view.

The application should present categories from a balloon diagram.

The application should provide popular categories to the user.

The application should present categories from a tree view.

The application should present categories from a balloon diagram.

The application should provide popular categories to the user.

The application should present categories from a tree view.

The application should learn what categories to present to the user.

The chunks shall be evaluated individually.

A chunk shall be able to have correct evaluation.

A chunk shall be able to have an incorrect evaluation.

If a chunk is incorrect, a reason shall be given by the user.

The user should be able to export an evaluated translation.

The user should be able to download an evaluated translation.

The application shall save the progress of ongoing work.

The user should be able to share an evaluated translation.

<table>
<thead>
<tr>
<th>2</th>
<th>The chunks shall be paired correctly with the original and translated text.</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>The application shall be able to pick a category from a hierarchical issue taxonomy.</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories in a dropdown menu.</td>
</tr>
<tr>
<td>3</td>
<td>The application should provide a search field for categories.</td>
</tr>
<tr>
<td>3</td>
<td>The application search field should provide auto completion.</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories from a tree view.</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories from a balloon diagram.</td>
</tr>
<tr>
<td>3</td>
<td>The application should provide popular categories to the user.</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories from a tree view.</td>
</tr>
<tr>
<td>4</td>
<td>The application should learn what categories to present to the user.</td>
</tr>
<tr>
<td>4</td>
<td>The chunks shall be evaluated individually.</td>
</tr>
<tr>
<td>4</td>
<td>A chunk shall be able to have correct evaluation.</td>
</tr>
<tr>
<td>4</td>
<td>A chunk shall be able to have an incorrect evaluation.</td>
</tr>
<tr>
<td>4</td>
<td>If a chunk is incorrect, a reason shall be given by the user.</td>
</tr>
<tr>
<td>5</td>
<td>The user should be able to export an evaluated translation.</td>
</tr>
<tr>
<td>5</td>
<td>The user should be able to download an evaluated translation.</td>
</tr>
<tr>
<td>5</td>
<td>The application shall save the progress of ongoing work.</td>
</tr>
<tr>
<td>5</td>
<td>The user should be able to share an evaluated translation.</td>
</tr>
</tbody>
</table>

Table 2 Derived requirements

### 4.2 Prototyping

#### 4.2.1 The first prototype

For the design of the first prototype, see Figure 4-7. The prototype allowed for simple interaction choosing an error category. I had manually entered a set of chunks taken from a news site (Sveriges Radio, 2015b) and translated using Google Translate. For the error taxonomy, I used MQM core that was introduced in section 2.1 (Lommel et al., 2013).

According to Krug (2000), when visiting a web-page a decision is made whether to browse or to ask (using a search field) to find what you are looking for. First time visitors are more prone to browsing, but second-time visitors want shortcuts to take them where they have been before. With this in mind, the prototype was designed to allow two ways to find an error category: either by scrolling the hierarchy tree, or by searching the hierarchy by typing in a search field.

Krug (2000) also emphasizes the importance of telling the user “You are here”, by highlighting the current location. For this reason, the prototype changes a chunk’s color to green when selected (Figure 4-7).
4.2.2 Evaluation of the first prototype

A set of questions was prepared for the evaluation. The questions were written as support for the evaluator. This helped the evaluator to know what to observe and note.

The usability tests were set in a conference room at the university. I sat next to the user in a sofa, so that I could see her actions. I kept note using pen and paper, as mentioned in section 2.2.6 as recommended by Mueller and Oppenheimer (2014).

The user sat down in front of a laptop displaying the prototype. The user was then shown a graphical representation of the taxonomy on paper, and was then asked to annotate all the chunks using the prototype. The user was also asked to think aloud how they intended to annotate each chunk, so that I could follow their actions.

The first user understood the instructions. However, she had trouble using the laptop’s mouse, and was also having problems reading the text on the screen until I adjusted the settings.

The second user started telling me how she wanted to annotate the chunks, without attempting to do this in the prototype. After a few minutes, she asked for more clear instructions which I gave her.

<table>
<thead>
<tr>
<th>Did the user understand the instructions?</th>
<th>User 1</th>
<th>User 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Did the user ask me any questions during the test?</td>
<td>Yes: Can I do multi-select?</td>
<td>Yes: Can I do multi-select? Can I unselect/regret?</td>
</tr>
<tr>
<td>Did the user use the search field?</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Did the test show that the prototype lacked any functionality</td>
<td>Yes: There is no way to select more than one category</td>
<td>Yes: There is no way to select more than one category. There is</td>
</tr>
</tbody>
</table>
needed to fulfill the assignment? no way to mark chunk as correct once an error was selected no way to mark chunk as correct once an error was selected

Other observation Had problem using keyboard and mouse.

Did the user find the chunk presentation clear? Yes Yes

Other user comments regarding hierarchy Yes: It is difficult to get an overview No

<table>
<thead>
<tr>
<th>Table 3. Summary of observations for prototype 1</th>
</tr>
</thead>
</table>

The main issues with the user interface are summarized:

- The lack of possibility to assign several issues to one chunk.
- When choosing an error category by mistake, there was no way to clear this, except to choose another error category.
- None of the users tried to use the search field.
- The scrolling list for choosing error category was too tedious to browse.

4.2.3 The second prototype

Following the evaluation of the first prototype in section 4.2.2, some conclusions were drawn about the design. This was used as input for the design of the second prototype.

Changes in the second prototype:

- The scroll lists were replaced with collapsible menus, to minimize the current visible scope to the current level only.
- The possibility to select more than one error category.
- The possibility to un-select a previously selected error category.
- An overview of selected error categories was added, with an embedded possibility to un-select previously selected error categories.
- The search field was removed, as a consequence of the replacement of the scroll lists.

These changes, except for the removed search field, can be seen in the screenshot below (Figure 4-8).
Figure 4.8 Several improvements were made in the second prototype.

### 4.2.4 Evaluation of the second prototype

The second evaluation was faster and more informal. The prototype was simply demonstrated to the client who gave feedback on the design. During the demonstration there were technical problems which made no interaction possible on the prototype. Instead, I displayed screen shots of the design, and explained verbally how the interaction was supposed to work. The planned strategy in section 3.5.4.3 for evaluating whether the design was easy to remember failed.

The main issues with the user interface are summarized:

- Whether the interface will still be clear if there are ten or more levels of issue types.
- The lack of possibility to group chunks.
- Good presentation of the text with the possibility to scroll up and down.

### 4.2.5 The third prototype

Following evaluation of the second prototype, improvements were made for the third, and last, prototype.

Improvements in the final prototype:

- Show which chunks have been annotated, and which have not.

Also, new functionality was added which include:

- Possibility to upload translation and issue taxonomy.
- Possibility to create a project that can be saved.
- Possibility to export the annotated translation.

These improvements were desirable but not implemented due to time constraints:

- Group chunks into sentences.
- Collapse parts of hierarchy vertically when showing many levels.

The final prototype was made in high fidelity. The prototype fulfilled the established requirements at the project start (4.1.2) and most of the suggestions from the prototype evaluations of the two previous versions.
Now follows a description of the design choices made in the third prototype.

The *Overview*, as seen in Figure 4-9, presents the available *Projects, Translations and Hierarchies*. The eye symbol is for viewing the current state of the document, the pencil is for editing a document and the cross is to delete it from the database. This will result in a pop-up window where the user needs to confirm this action as there is no turning back once she has confirmed deletion.

The eye and pencil icons for viewing and editing, as well as the cross for removing, are common choices when it comes to user interfaces on the web, and the user is therefore likely to recognize these and understand what will happen when pressed.

**Figure 4-9. Overview page**

The *Project*, as seen in Figure 4-10, presents available projects to the user to start editing. It presents the same options as in the overview so the user can view, edit or delete the existing projects. A project is the combination of a selected translation and hierarchy document.

**Figure 4-10. Project Overview Page**
Figure 4-11 shows the creation of a project, where you get to select from available translation documents and available hierarchy documents. The user gets to select a project name for your project which has to be unique to keep them apart. The user can however have several projects of the same document and hierarchy.

<table>
<thead>
<tr>
<th>Overview</th>
<th>Project</th>
<th>Review</th>
<th>Upload</th>
<th>Export</th>
</tr>
</thead>
<tbody>
<tr>
<td>Available Projects</td>
<td>New Project</td>
<td>Document:</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hierarch:</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Project Name:</td>
<td>Yet another project</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Project</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure 4-11. New Project Page**

The *Review* page has two options available for the user. One by clicking on the review tab in the main menu, which gives the user an example translation where you can familiarize yourself with the process. This will not be saved to the database. The second option is the review of a document that will save all information changed to the database. As seen in Figure 4-12 it will color-code the chunks depending on its status.

- Green for accepted.
- Yellow for a chunk that has issues assigned to them.
- Blue for the currently selected chunk that you work with.
- Grey for unprocessed chunks.

It has the options to either accept the translation or assign one or several issues to the selected chunk. The menu is collapsible like an accordion to avoid expanding the menus more than necessary without having to collapse manually.

Any selected issues are visible to the far right with an X next to them. This way, so the user doesn’t need to go down in the hierarchy to remove a faulty issue. In the far bottom there is a progress bar that shows the total number of chunks processed as well as the quality of the translation. Green indicates an accepted chunk and yellow indicates a chunk with an issue.
Figure 4-12. Review Page

The **Upload** view provides two options to upload files. The first option is to upload a translation document as seen in Figure 4-13. Here the user selects a file in the available upload format, followed by clicking on the **Upload Translation**.

![Image](image1.png)

**Figure 4-13. Upload Translation Page**

The second available option is to upload a hierarchy document as seen in Figure 4-14. Here the user selects a file in, for example, the provided JSON format and uploads it to the **Hierarchy** database.
Figure 4.14. Upload Hierarchy Page

The Export view is where the user can select a project according to Figure 4.15. She will then be prompted with an option to change the document name before exporting. It is seen in Figure 4.16, and this will also be the name of the file.

Figure 4.15. Export Page
4.2.6 Evaluation of the third prototype

For the final prototype, the evaluation was made with two test users. These test users will be referred to as User A and User B. User A is well-educated in the engineering sciences and uses computers in her daily work. User B is also well-educated in the social sciences and uses computers in her daily work, but does not have as advanced computer skills.

In these tests, I set up the application on the users’ personal computers, to avoid the test results to be influenced by the users being unaccustomed to the equipment. I sat next to the users observing them perform certain tasks. Afterwards, I asked some questions about their impression of the application.

The users were provided with files to be uploaded to the application: one file with a chunked translation, and one file with an issue taxonomy for translation quality assessment. The users were given these instructions:

- Upload the file with the chunked translation to the application.
- Upload the file with the taxonomy to the application.
- Create a project, using the uploaded translation and issue taxonomy.
- Annotate the chunks using the issue taxonomy.
- Export the project.

These instructions correspond to the use cases developed in 4.1.3.

The results from these user tests are shown in the table below:

<table>
<thead>
<tr>
<th></th>
<th>User A</th>
<th>User B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Did the user understand the instructions?</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Did the user ask me any questions during the test?</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Question</td>
<td>Yes</td>
<td>Yes, but it took some tries. She tried to press the “upload” button before selecting a file, which gave an unclear error message.</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------</td>
<td>-----</td>
<td>---------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Did the user understand how to upload a file?</td>
<td>Yes</td>
<td>Yes, but it took some tries. She tried to press the “upload” button before selecting a file, which gave an unclear error message.</td>
</tr>
<tr>
<td>Did the user understand how to create a project?</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Did the user understand how to export a project?</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Did the user understand how to select a project?</td>
<td>Yes, but not at first try (see comment below)</td>
<td>Yes</td>
</tr>
<tr>
<td>Did the user understand how to annotate a translation?</td>
<td>Yes</td>
<td>Understood how to select a chunk and browse issue types, but did not check the box when selecting</td>
</tr>
<tr>
<td>Did the test show that the prototype lacked any functionality needed to fulfill the assignment?</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Other comments</td>
<td>The user did not load the project she created at first, but started annotating the example project instead.</td>
<td></td>
</tr>
</tbody>
</table>

**Table 4 Summary of observations or prototype 3**

From these results, it was derived that the following changes need to be made to ensure the application’s usability:

- Make it more clear when no project is loaded, but only an example is showing. E.g. by highlighting the information using a bright color.
- Disable (grey out) “Upload” button when no file is selected.

Also, for assessing the design, both of the test users were given the SUS questionnaire mentioned in 2.2.7. The purpose of this was to get some indication of the fifth aspect of usability referring to Nielsen’s usability definition, *pleasant to use* (Nielsen, 1993). The resulting points were 90 and 55 points respectively. Commonly the average score for SUS evaluations is 68 (Brooke, 2013). The more computer-skilled user gave the higher points, which might reflect the fact that she found the application easier to use. What also can be noted is that I perceived the users equally satisfied with the design when talking informally to them after the user tests.

For SUS results to give a reliable result, at least 8 user should be given the questionnaire (Brooke, 2013). Therefore, I dare not draw any conclusions from this evaluation.
# Chapter 5

## Results

### 5.1 Requirements fulfillment

After development of the web application I compared the outcome with the requirements in chapter 4.1.4.

<table>
<thead>
<tr>
<th>Derived from</th>
<th>Lower level requirement</th>
<th>Outcome</th>
<th>Remark</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The application shall be web based</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>The application shall have responsive design</td>
<td>Fulfilled with remark</td>
<td>It was limited by the framework used. For mobile purposes the review page needs to be adjusted.</td>
</tr>
<tr>
<td>1</td>
<td>The application shall work on pc devices supporting at least 800*600 resolution</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>The application should work on mobile devices</td>
<td>Fulfilled with remark</td>
<td>The review page is too wide at the moment</td>
</tr>
<tr>
<td>1</td>
<td>The application shall save the progress of ongoing work</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>The application shall update asynchronously, i.e. reloads part of page without reloading the entire webpage.</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>The application shall show the original text to the user</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>The application shall show the translated text to the user</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>The text shall be divided in manageable subparts, so called chunks</td>
<td>Not implemented</td>
<td>The uploaded input file is expected to provide chunked texts before preview</td>
</tr>
<tr>
<td>2</td>
<td>The user shall be able to</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td></td>
<td>upload a translation to the web page</td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>---------------------------------</td>
<td>---</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>The chunks shall be paired correctly with the original and translated text.</td>
<td>Not implemented</td>
<td>The uploaded input file is expected to provide chunked texts before preview (*)</td>
</tr>
<tr>
<td>3</td>
<td>The application shall be able to pick a category from a hierarchical issue taxonomy.</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories in a dropdown menu.</td>
<td>Removed</td>
<td>Due to evaluation of prototype 1</td>
</tr>
<tr>
<td>3</td>
<td>The application should provide a search field for categories</td>
<td>Removed</td>
<td>Due to evaluation of prototype 1</td>
</tr>
<tr>
<td>3</td>
<td>The application search field should provide auto completion</td>
<td>Removed</td>
<td>Due to evaluation of prototype 1</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories from a tree view</td>
<td>Not implemented</td>
<td>Due to time limitation</td>
</tr>
<tr>
<td>3</td>
<td>The application should present categories from a balloon diagram</td>
<td>Not implemented</td>
<td>Due to limited support and Javascript documentation for dynamic creation of custom trees</td>
</tr>
<tr>
<td>3</td>
<td>The application should provide popular categories to the user</td>
<td>Not implemented</td>
<td>Due to time limitation</td>
</tr>
<tr>
<td>3</td>
<td>The application should learn which categories to present to the user</td>
<td>Not implemented</td>
<td>Due to time limitation. Also not possible as there is no individual user management implemented.</td>
</tr>
<tr>
<td>4</td>
<td>The chunks shall be evaluated individually</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>A chunk shall be able to have correct evaluation</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>A chunk shall be able to have an incorrect evaluation</td>
<td>Fulfilled</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>If a chunk is incorrect, a reason shall be given to the user</td>
<td>Fulfilled when rephrased</td>
<td>Rephrased: If a chunk is incorrect, one or several reasons shall be given to the user.</td>
</tr>
<tr>
<td>5</td>
<td>The user should be able to</td>
<td>Fulfilled</td>
<td></td>
</tr>
</tbody>
</table>
download an evaluated translation

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>The user should be able to share an evaluated translation</td>
</tr>
<tr>
<td>5</td>
<td></td>
</tr>
</tbody>
</table>

(*) Not implemented due to time limitations or technical limitations. The functionality requirement remains.

Table 5 Requirements fulfillment

These are the known issues:

- No validation of input translations is provided besides JSON validation. Nature of the mongo database also gives limited validation so its effect on invalid documents is unknown.
- No validation of input hierarchies is provided besides JSON validation. Effects of invalid documents are unknown.
- Exported documents are snapshots of the current project data, and have other fields than the input files. This means that they cannot be uploaded to create a new project.
- There is no user login or security in place. If you have access to the website you have access to all functionality as any user of the system.

5.2 User evaluations

The purpose of the user evaluations was two-fold: first of all to receive important feedback from potential users, and secondly to explore different evaluation strategies.

The user feedback during the first evaluation mainly focused on functional problems, such as the lack of possibility to assign multiple issues to a chunk. The second evaluation was a direct discussion with the project client. The client is acquainted to the field and could come with input on aspects that might be important for a professional translator as well as problems that could occur if the dataset becomes much larger. The user feedback from the third evaluation mainly highlighted smaller problems, such as misunderstandings and unclarities on how to use the application, and not so much on functional inaccuracies.

The feedback also indicated that the users with technical background had an easier time navigating the application than other test users, which is to no surprise since they have more habit and background knowledge in similar work. However, it demonstrates the importance of including test users from varying backgrounds since the developer likely will have a technical background and possibly oversees some of the concerns that might come up for others.

Three different evaluation methods were tested, one for each prototype. This was partly to explore different methods to achieve broader feedback, and partly to make sure that the evaluations could be done within the time-frame.

Evaluation strategies for the three prototypes:

1. Observation + interview
2. Demonstration to client
3. Observation + interview + SUS questionnaire
Chapter 6
Discussion

In section 1.2, an ambition to find how to put the human user in focus of the design process, was presented. This thesis has presented different methods for the developer to come closer to understanding the user’s needs. As a developer, it is impossible to fully put yourself in the position of a user. Actually watching other people being confronted with the design for the first time makes you see your work in a new light. There are aspects of a design that a developer might find intuitive, but users will struggle with.

In section 1.2, I asked the question *How can the design process be broken down to keep a focus on the human user throughout the process?* By the end of the project, it is my understanding that using the iterative design process suggested by Nielsen (1993) as well as the user tests, demos and SUS test gave a good breakdown and valuable feedback to keep focus on the human user throughout the development.

I began the project by attempting to define usability and found Nielsen (1993) definition of usability as a combination of the five aspects of usability listed in section 2.2.1. The following aspects have been tested in the prototype evaluations: *easy to learn, few errors* and *pleasant to use*. The other two aspects, *efficient to use* and *easy to remember*, were not evaluated. For the three aspects that were evaluated, the results of the evaluations showed the application’s design to be usable. To answer my question of how usability can be measured I have used these 5 questions as a guideline for the design evaluations.

The evaluation of the *easy to learn* criterion in 3.5.4.1 was done in several steps. Based on the observations of users, performing the tasks, I noticed a difference in the learning time between the first and third prototype. The users were different people for the first and third evaluation and thus all unfamiliar with the application at the testing time. Despite this, the users for the third prototype understood how to perform the task much quicker. This could indicate that prototype 3 was easier to use and hence fulfilled this requirement. However, it is important to remember that this could also be contributed to either the small set of users and thus human variation, or from the instructions being more clear later on in the project.

The evaluation of *efficient to use* is a step that needs to be done after this thesis as mentioned in 3.5.4.2, and the final application hasn’t been available to the end users. A future work or thesis would be to continue this evaluation by seeing this as a scope for continuous development of the application, where the end user is part of the feedback loop as well.

The evaluation of the *easy to remember* criterion in 3.5.4.3 was not successful due to technical issues when demonstrating the second prototype as mentioned in 4.2.4. The strategy for this was to evaluate how the user felt using the system the second time, with improvements, compared to the first time. But as the prototype interaction did not work during the demonstration, I had to demo it using only screen shots and verbal explanations, instead of observing the user annotate the document the second time.

The evaluation of the *few errors* criterion in 3.5.4.4 was evaluated during the user tests. Several errors were found, both errors that originated from misimplementation of the requirements (e.g. the user could
only select one issue type per chunk in prototype 1), and errors that were a consequence of the user performing unexpected actions (e.g. the user trying to upload an unselected file in prototype 3).

The evaluation of the pleasant to use criterion in 3.5.4.5 was done in the observation of the user tests done in prototype 1 and prototype 3. The basic idea was to listen to the overall feedback given by the users during the interviews. Also, the SUS questionnaire was used, but was done on too small a set of users to really draw any solid conclusions.

The iterative design process that was suggested by Nielsen (1993) and explained in section 2.2.2, provided valuable information about the usability of the design. The user tests for the third prototype showed fewer errors than the first prototype. The test users for the third prototype learned the design easier than the test users for the first prototype.

The evaluation strategies had different purposes and I experienced that all gave important information. The observations were valuable for understanding both the intuitive understanding and the learning speed of the user. The interviews were more focused on specific questions that I wanted an answer for. These questions both dealt with specific issues that I was concerned about, but also dealt with how the user experienced the evaluation format. The client input could provide information that the other evaluations could not. While some of the other test users are not likely use the end product, the client might be using the product on a daily basis and this input is therefore important. Last but not least, the SUS questionnaire was implemented in the evaluation process. This was to get an indication of the final result, in a similar way as in large-scale user sets. The user set in this project was too small to draw conclusions from this but was still interesting as a more quantitative comparison, as it provides a numerical score.

The user tests also turned out to teach an unsuspected, but valuable, lesson for how instructions easily can be misunderstood if you are not very clear. The test users understood the instructions for the final user tests better than the test users understood for the first prototype. This could indicate that the evaluator’s performance as a user test conductor was improved in the iteration, but due to the small number of test users it could also be from other factors.

If I were to perform user tests again, I would put more effort into the setting and into preparing the discussions for the meeting. The user tests were somewhat interrupted by the users not being used to the hardware equipment (first prototype user tests) and misunderstanding the instructions (first and third prototype user tests).

One thing that showed to be more time consuming than expected, was the time spent on processing the results from the evaluations and considering these when designing the next version. Prioritizing among the issues found was not a part of the time plan, but still something that had to be done.

Still, even though there are improvements to be made to the design, I believe this is a usable application for the task of translation quality assessment. The last user tests and questionnaires showed that the users had a positive impression of the design, and they understood how to use it rapidly. Designing for usability is an iterative process that may continue forever. Given the time frame and practical constraints, I am satisfied with the design process presented in this report.
References


På svenska

Detta dokument hålls tillgängligt på Internet – eller dess framtida ersättare – under en längre tid från publiceringsdatum under förutsättning att inga extra-ordinära omständigheter uppstår.

Tillgång till dokumentet innebär tillstånd för var och en att läsa, ladda ner, skriva ut enstaka kopior för enskilt bruk och att använda det oförändrat för ickekommersiell forskning och för undervisning. Överföring av upphovsrätten vid en senare tidpunkt kan inte upphäva detta tillstånd. All annan användning av dokumentet kräver upphovsmannens medgivande. För att garantera äktheten, säkerheten och tillgängligheten finns det lösningar av teknisk och administrativ art.

Upphovsmannens ideella rätt innefattar rätt att bli nämnd som upphovsman i den omfattning som god sed kräver vid användning av dokumentet på ovan beskrivna sätt samt skydd mot att dokumentet ändras eller presenteras i sådan form eller i sådant sammanhang som är kränkande för upphovsmannens litterära eller konstnärliga anseende eller egenart.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

In English

The publishers will keep this document online on the Internet - or its possible replacement - for a considerable time from the date of publication barring exceptional circumstances.

The online availability of the document implies a permanent permission for anyone to read, to download, to print out single copies for your own use and to use it unchanged for any non-commercial research and educational purpose. Subsequent transfers of copyright cannot revoke this permission. All other uses of the document are conditional on the consent of the copyright owner. The publisher has taken technical and administrative measures to assure authenticity, security and accessibility.

According to intellectual property law the author has the right to be mentioned when his/her work is accessed as described above and to be protected against infringement.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

© Sophie Joelsson