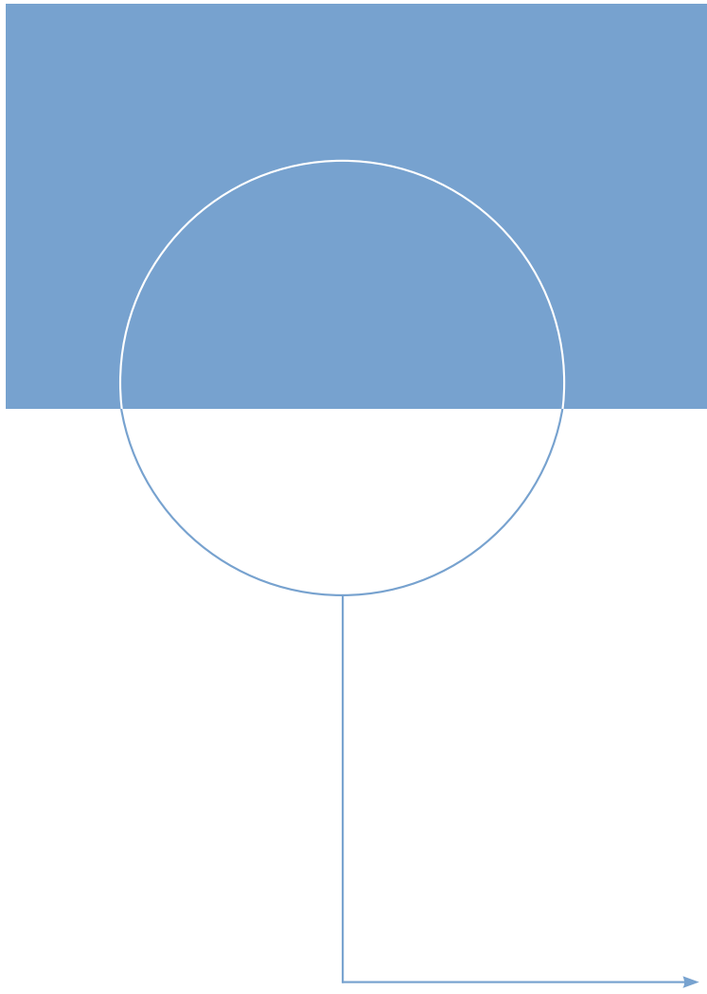


Master's thesis

2020

Christian Nyvoll

NTNU
Norwegian University of
Science and Technology
Faculty of Information Technology and Electrical
Engineering
Department of Computer Science

Christian Nyvoll

Efficient Authoring of Inline Gap Match Tasks for Digital Programming Exams

June 2020

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

NTNU
Norwegian University of
Science and Technology

Efficient Authoring of Inline Gap Match Tasks for Digital Programming Exams

Christian Nyvoll

MIT
Submission date: June 2020
Supervisor: Guttorm Sindre

Norwegian University of Science and Technology
Department of Computer Science

Abstract

Programming exams have traditionally been paper-based, but with a shift towards modernization and utilization of technology, many exams are now conducted digitally. This provides many opportunities for programming tasks to be closer to real-world usage as the computer can now be used during the exam. Some question types on digital exam platforms can be well-suited for good tasks to test programming skills and knowledge. However, the creation process for some of these tasks are not intended for code tasks and are therefore difficult to make. This thesis proposes to create an IT artefact that could make this process more user friendly and effective.

The proposed IT artefact was designed and created to be a prototype application for efficient authoring of drag-and-drop fill in missing code tasks. To test the usability compared to the current process it was compared with the question authoring system in use at NTNU, Inspera Assessment. With tasks created on the IMS Question and Test Interoperability specification (QTI) format, they are interchangeable between systems that support the same version. Therefore the IT artefact was developed to efficiently and effectively let the question author create a drag-and-drop fill in missing code task that could be exported on the QTI format. This task could then be imported into the Inspera Assessment platform and used in the same ways as a task created directly in the Inspera Assessment interface. As the IT artefact followed the QTI specification for the given version, it could also be applicable to other learning or assessment platforms that utilize the same QTI format.

To determine if the created IT artefact had a higher level of usability than the current process an experiment was conducted. Six participants that work with creating programming tasks for students were recruited to create the same drag-and-drop fill in missing code task with the IT artefact and Inspera Assessment. The results from the user testing and conducted interviews show that the IT artefact was an improvement in comparison to the current process in Inspera Assessment. It took less time and required far fewer action to complete the same question authoring process with the IT artefact, and it showed a higher level of usability in regards to the effectiveness, efficiency, and satisfaction. To summarize, the participants preferred to use the proposed IT artefact over Inspera Assessment to create the same drag-and-drop fill in missing code tasks.

Sammendrag

Programmeringseksamener har tradisjonelt vært papirbasert, men med et skifte mot modernisering og utnyttelse av teknologi, gjennomføres nå mange eksamener digitalt. Dette gir mange muligheter for at programmeringsoppgaver kan være likere bruken i den virkelige verden ettersom datamaskiner nå kan brukes under eksamen. Noen spørsmålstyper på digitale eksamensplattformer kan være godt egnet for å lage gode oppgaver som tester programmeringsevner og kunnskap i programmering. Måten man lager noen av disse oppgavene på er imidlertid ikke ment for kodeoppgaver og gjør det derfor vanskelig. Denne masteroppgaven foreslår å lage en IT-gjenstand som kan gjøre denne prosessen mer brukervennlig og effektiv.

Den foreslåtte IT-gjenstanden ble designet og laget for å være en prototype-applikasjon for effektiv forfatting av dra-og-slipp fyll inn i manglende kode oppgaver. For å teste brukervennligheten sammenlignet med den nåværende prosessen ble den sammenlignet med det digitale eksamenssystemet som er i bruk ved NTNU, Inspera Assessment. Med oppgaver som er opprettet på IMS Question and Test Interoperability specification (QTI) formatet, kan de utveksles mellom systemer som støtter den samme versjonen. Derfor ble IT-gjenstanden utviklet for å raskt og effektivt la spørsmålsforfatteren opprette en dra-og-slipp fyll inn i manglende kode oppgave som kan eksporteres på QTI-formatet. Denne oppgaven kan deretter importeres til Inspera Assessment-plattformen og brukes på samme måte som en oppgave som er opprettet direkte i Inspera Assessment-grensesnittet. Ettersom IT-gjenstanden fulgte QTI spesifikasjonen for den gitte versjonen, kan den også være aktuell for andre lærings- eller vurderingsplattformer som bruker samme QTI-format.

For å avgjøre om den opprettede IT-gjenstanden hadde høyere brukskvalitet enn den nåværende prosessen ble et eksperiment utført. Seks deltakere som jobber med å lage programmeringsoppgaver for studenter ble rekruttert for å lage den samme dra-og-slipp fyll inn i manglende kode oppgaven med IT-gjenstanden og Inspera Assessment. Resultatene fra brukertesting og de gjennomførte intervjuer viser at IT-gjenstanden var en forbedring sammenlignet med dagens prosess i Inspera Assessment. Det tok kortere tid og det krevde langt færre handlinger for å fullføre den samme oppgavelagingen med IT-gjenstanden, og den hadde en høyere grad av brukskvalitet med hensyn til hastighet, effektivitet og tilfredshet. For å oppsummere foretrakk deltakerne å bruke den foreslåtte IT-gjenstanden fremfor Inspera Assessment for å lage den samme dra-og-slipp fyll inn i manglende kode oppgaven.

Preface

This thesis is the result of the research conducted during the fall of 2019 and the spring of 2020. It concludes my five years at the Norwegian University of Science and Technology (NTNU), at the Department of Computer Science (IDI). The thesis and all other materials produced during the research period belong to this submission for the degree of Master in Informatics.

I want to thank my supervisor, Prof. Guttorm Sindre, for all feedback and guidance. Additionally, I would like to thank all the participants that completed the experiment. It was greatly appreciated that they took time out of their hectic COVID-19 affected lives to help me perform the tests and gather the results required to complete this thesis. Lastly, I would like to thank my family, who always support me, and especially my live-in partner, who motivated me throughout the whole period.

Contents

Abstract i

Sammendrag ii

Preface iii

Table of Contents ix

List of Tables xii

List of Figures xiv

Abbreviations xv

1 Introduction 1

1.1 Background 1

1.1.1 Motivation 3

1.2 Context 3

1.3 Scope 4

1.4 Research Questions 5

1.5 Report Outline 5

2 Background **7**

2.1 E-learning and e-assessment 7

2.2 Digital Programming Tasks 8

2.2.1 Parsons Problem 8

2.3 Related Work 9

2.3.1 Learning Effect of Code Completion Puzzles 9

2.3.2 Jorgensen and Kvannli 10

2.3.3 JS-Parsons 10

2.3.4 QTI.JS 11

2.4 Inspera Assessment 11

2.5 QTI 14

2.6 Task Types 15

2.6.1 Test Set and New Question 16

2.6.2 Inline Gap Match 16

3 Method **21**

3.1 Literature Review 21

3.2 Research Method for Research Question 1 23

3.2.1 Design and creation 23

3.3 Research Method for Research Question 2 24

3.3.1 Experiment 26

3.4 Data Generation Methods 31

3.4.1 Data Generation Methods for Design and Creation 31

3.4.2 Data Generation Methods for Experiment 32

3.5 Evaluation 38
3.5.1 Evaluation of Data for Research Question 1 38
3.5.2 Evaluation of Data for Research Question 2 38
3.6 System Development Methodology 41
3.6.1 Agile Development 41
3.6.2 Scrum 42
3.6.3 Extreme Programming 42
3.6.4 Kanban 43
3.6.5 Scrumban 43
3.6.6 Development and Iteration Structure 44

4 Results **47**

4.1 Design and Creation 47
4.1.1 Iteration Planning 48
4.1.2 Iteration 1 49
4.1.3 Iteration 2 51
4.1.4 Iteration 3 56
4.1.5 Iteration 4 58
4.2 Experiment 62
4.2.1 Quantitative Data 62
4.2.2 Qualitative Data 72

5 Discussion **75**

5.1 Design and Creation 75
5.2 Experiment 76
5.2.1 Quantitative Data 77

5.2.2 Qualitative Data 80
5.2.3 Research Critique 82

6 Conclusion **83**

7 Future Work **85**

Bibliography **87**

Appendix **93**

A Experiment Documents **93**

A.1 Observation Schedule 93
A.2 Test Plans 94
A.2.1 Test Plan 1 94
A.2.2 Test Plan 2 101

B Requirements **109**

B.1 Functional Requirements 109
B.2 Non-functional Requirements 110

C Transcripts **111**

C.1 Categories 111
C.2 New Features 115
C.3 Transcript Participant 1 115
C.4 Transcript Participant 2 121
C.5 Transcript Participant 3 127
C.6 Transcript Participant 4 136
C.7 Transcript Participant 5 145

C.8 Transcript Participant 6 151

List of Tables

3.1 Literature Review Record 22

3.2 Search Matrix for Digital Programming Exams Literature 23

3.3 Interview Questions 35

3.4 SUS Questionnaire 37

3.5 Scrum Attributes Compared to Kanban [50] 43

4.1 Initial Requirements 48

4.2 Relevant Requirements for Iteration 1 49

4.3 Tasks for Iteration 1 49

4.4 Relevant Requirements for Iteration 2 51

4.5 Tasks for Iteration 2 52

4.6 Relevant Requirements for Iteration 3 56

4.7 Tasks for Iteration 3 56

4.8 Relevant Requirements for Iteration 4 58

4.9 Tasks for Iteration 4 59

4.10 Bugs for Iteration 4 59

4.11 Data Points for Task Completed in Inspera 63
4.12 Data Points for Task Completed in IT artefact 63
4.13 System Usability Score for the Tested Systems 64
4.14 Wilcoxon Signed Rank Test for Time To Completion 65
4.15 Wilcoxon Signed Rank Test for Number of Actions to Complete 67
4.16 Wilcoxon Signed Rank Test for System Usability Scale 69
4.17 Positive Categories for Inspera 72
4.18 Negative Categories for Inspera 73
4.19 Positive Categories for IT artefact 73
4.20 Negative Categories for IT artefact 74

5.1 Minimum Number of Actions for Inspera 79
5.2 Minimum Number of Actions for Artefact 79

A.1 Observation Schedule 93

B.1 Functional Requirements 109
B.2 Non-functional Requirements 110

C.1 Positive Categories for Inspera 111
C.2 Negative Categories for Inspera 112
C.3 Positive Categories for IT Artefact 113
C.4 Negative Categories for IT Artefact 115
C.5 Categories for IT artefact new features 115

List of Figures

2.1 Task types that can be created in Inspera Assessment 12

2.2 Inspera Assessment Question Creation Process 1 13

2.3 Inspera Assessment Question Creation Process 2 14

2.4 QTI 2.1 Question Example 15

2.5 Task to be created 16

2.6 Initial Task Creation Interface 17

2.7 Code Pasted into Task 17

2.8 Manually Indented with Spaces 18

2.9 Gap Selected 18

2.10 Add Correct Answer 18

2.11 Complete Task 18

2.12 Task Preview 19

2.13 Preview Solved 19

3.1 Experiment Task Creation Test 28

4.1 Initial System Architecture 48

4.2 Initial Design Layout 50
4.3 Initial Gap Selection Sketch 53
4.4 Gap Selection Interface 54
4.5 Task Object Interface 55
4.6 System Architecture 57
4.7 IT Artefact Main View 1 61
4.8 IT Artefact Main View 2 61
4.9 Time to Completion with Inspera and IT artefact 66
4.10 Average Time to Completion with Inspera and IT artefact 67
4.11 Actions to Complete with Inspera and IT artefact 68
4.12 Average Actions to Complete with Inspera and IT artefact 69
4.13 SUS Scores for Inspera and IT artefact 70
4.14 Average SUS Scores for Inspera and IT artefact 71
4.15 SUS Score Adjective Ratings [70] 71

Abbreviations

BYOD	=	Bring Your Own Device
IDE	=	Integrated Development Environment
QTI	=	Question and Test Interoperability
IMS	=	Instructional Management Systems
API	=	Application Programming Interface
UML	=	Unified Modeling Language
XML	=	eXtensible Markup Language
GUI	=	Graphical User Interface
MVP	=	Minimum Viable Product
LMS	=	Learning Management System
SUS	=	System Usability Scale

Chapter 1

Introduction

This chapter provides an introduction to the thesis. First, by introducing some background information, followed by a description of the motivation behind it. Then the context and scope are presented before the research questions are posed. Lastly, a report outline that shows how the report is structured is defined.

1.1 Background

Many different occupations or fields of study require some knowledge and skills with programming. As with most anything, programming can be self-taught, but many choose to take courses, and even full studies focused on programming. These courses need to measure competency and learning in some way. There are many ways to achieve this, but the longest-standing evaluation method is the written exam. Many programming course exams are in paper format. Programming exams in paper format is an unnatural way of testing a skill that usually is utilized on a computer [1]. Digital exams have been put in place for many courses and might become the new standard in a few years because they can provide benefits compared to paper exams [2], [3]. These new digital exams raise the question; how can the latest digital exam tools be utilized to test a candidate's skill and competence.

Every digital exam will have to consider different trade-offs. One of the most important ones is between functionality and security [4]. Security in regards to how easy or likely it is that candidates can cheat. There are many tools available that can both create and let users solve programming tasks quickly. However, they are not safe enough for exam usage. The exam settings require the computer system to be locked down to disable or limit the user's access to local files, the internet, and other tools that could be considered

cheating [5]. There is also the trade-off between assisting functions, and at what point they provide too much help. Assisting functions could, e.g., be auto-completion help like most modern programming IDEs provide. They may provide "crutches" for the exam candidates that make it difficult to assess their abilities in an accurate way [6].

In Norway, there are two central digital exam systems in use at the different universities and other educational institutions. WISEFlow [7] is a digital exam and assessment system that is implemented at USN (University of South-Eastern Norway), NMBU (Norwegian University of Life Sciences), UiT (The Arctic University of Norway), HK (Kristiania University College) and others. The other system is called Inspera Assessment [8] and is used at NTNU (Norwegian University of Science and Technology), UiO (University of Oslo), UiS (University of Stavanger) and probably others.

At the Norwegian University of Science and Technology (hereafter NTNU), most digital exams are created using Inspera Assessment. There are a few different ways this tool can be used to evaluate candidates. First is the "standard" school exam with BYOD (Bring Your Own Device) [9]–[12] which locks down the user's computer with a static browser window making it impossible to use outside or third party applications during the examination. Second is the option to use NTNU-owned stationary computers, which allows for the use of third party software as well; the only limit is a max capacity of about 200 students each day. The third option comprises a lot of potential different evaluation types like a home exam, semester projects or tasks, master theses, project report, and others that should be graded by allowing the answers to these tasks to be uploaded to Inspera for grading. Since a lot of introductory programming subjects have many students, they are usually constrained to using the first, BYOD option.

The Inspera Assessment software has support for code editor like input fields with some syntax help, which in itself provides a better base than handwritten paper programming exams for both the examined and supervisor. For the student, it will be a lot easier to write on a keyboard and avoid cramps that often occur when writing a lot on paper. It is also far easier to change the written code if, e.g., an extra line has to be added in the middle. Inspera also provides syntax highlighting [13], automatic parentheses and indentation. For the examiner or grader, it is a lot easier to read machine- than handwritten code. It is also more convenient to grade each sub-task by sub-task for all candidates, instead of the entire exam candidate by candidate.

With excellent digital tools come possibilities to create interactive and better problems for the candidates to solve. E-learning and e-assessment tools with the right features can be potent and improve the current assessment processes [14]. The right types of tasks might even be able to automatically grade themselves, which would save a lot of time and resources otherwise spent on these tasks [2]. A great digital exam might even be more effective at testing intended learning outcomes [3]. While many task types will be better for students to solve in a digital version, they can take a lot longer for the lecturer to create compared to a traditional paper exam. These opportunities and challenges provide background for the project task description and the writing of this thesis.

1.1.1 Motivation

Having completed multiple programming exams, both digital and on paper, during the researchers years at NTNU, the researcher has many relevant personal experiences. The exams where code had to be written on paper were challenging, and they did not test how the researcher's skills would be used in a real-life scenario. The digital exams greatly simplified things like writing a lot of text, which was painfully slow without the keyboard we are growing accustomed to outside of the examination halls. However, for programming, they have not been a substantial immediate improvement. Being able to improve the programming exams hugely motivated the researcher. The researcher wanted to be able to contribute to, and possibly change, the programming exams for the students at NTNU and potentially other universities the next years.

1.2 Context

The (translated) task description of the master thesis is as follows:

The transition from paper-based exams to digital exams gives potential benefits with more effective organizations of the exam and more effective grading. In return, it can require more effort to create an exam. For most task types, it will be much faster to type questions in a normal word-processor than to input them in Inspera (what NTNU uses for digital exams). Digital exams provide us with opportunities for new, exciting task types like for example drag-and-drop, but again, it can be a problem that these are relatively time-consuming to enter into the system.

Two master students have already made an application (delivered spring 2019) to make it easier to create drag-and-drop-tasks to use in Inspera. Here the solution code can be typed (or pasted) in to generate a drag-and-drop question on the QTI 2.1 format, which can be loaded into Inspera (QTI is an international standard for exchange of exams or tests and questions in those, see `http://www.imsglobal.org/question/qtiv2p2/imsqti_v2p2_oview.html`*).*

A new master thesis on the subject could be based upon the already created application and try to develop it further with additional functionality. This could be:

- *support for other task types than drag-and-drop*
- *More advanced support for drag-and-drop-tasks*
- *Research whether it is possible to change the application to a plug-in directly integrated with Inspera trough its API, rather than it being a standalone application that must be run separately*

The application developed for this master theses builds upon the work done in a previous

master thesis within the same area. It is important to note that the authors of this thesis, Jørgensen and Kvannli [15], also based some of their work on a previously created proof-of-concept application. The need for an application like this stems from the Department of Computer Science (IDI) at NTNU. However, other departments and even universities could likely utilize a tool to generate tasks if they use the QTI format in their e-learning or e-assessment platforms.

1.3 Scope

When using an online question or exam creation tool like Inspera, it can be difficult and very time consuming to create good programming exercises. There are many different task types, but some might be better suited to test programming skills. The scope for this master thesis will be to conduct research and create an IT artefact that can improve upon the current process of programming task creation. The IT artefact will provide the user with a for-purpose-made interface that outputs tasks on the QTI format that can be imported into the user's assessment platform. For this thesis, the focus will be on making tasks that can be used in the Inspera Assessment system. However, many other systems (WiseFlow [7], Canvas [16], Moodle [17], Blackboard [18]) use the QTI format and could potentially benefit from the same IT artefact and the research done in this thesis.

Considering some of the task types that exist, the scope for the thesis and the IT artefact is to create an application that makes it easier to design programming tasks on the QTI standard. The task types that could be relevant because of their ability to test programming skills or knowledge include; fill in missing text, fill inn missing text with drop-down options, fill in missing text with drag-and-drop and drag-and-drop. In Inspera Assessment these tasks are called *Text Entry*, *Inline Choice*, *Inline Gap Match* and *Drag and drop*. As the *Inline Gap Match* task type can achieve the same functionality as *Drag and drop* if the drop areas are placed on distinct lines it was chosen as the task type to be part of the scope for this thesis. When working with code and programming tasks the distinct lines rule is a wanted feature as this adheres to the code syntax of most programming languages. The *Inline Gap Match* task type can have similar functionality as the *Inline Choice* tasks, and requires little change to adapt to the *Text Entry* task. Explicitly worded the scope for the thesis regarding the IT artefact is therefore as follows:

Create an IT artefact that lets the question author create fill in missing text with drag-and-drop (*Inline Gap Match*) tasks on the QTI format that can be exported from the IT artefact, and then imported into an assessment platform.

It is also part of the scope to make the IT artefact easy for others to develop it further and extend the functionalities. The requirements and goals of the application are further explained in chapter 2 and chapter 3.

1.4 Research Questions

This thesis and research process has the goal to create a useful IT artefact that improves a task creation process. To create the IT artefact and decide if it is an enhancement, the researcher poses the following research questions:

RQ1: How can one design and create an IT artefact that can support effective authoring of tasks on the QTI format?

RQ2: What improvements does the IT artefact give compared to using the authoring tool included in Inspera Assessment?

1.5 Report Outline

Following the thesis introduction, the rest of the paper structure is as follows; the background for the thesis is detailed in chapter 2. Chapter 3 explains the process of selecting the research methods and how the research was planned. All the results gathered from the research will then be presented in chapter 4 before they are discussed in chapter 5. The research conclusion will be laid out in chapter 6. Lastly, any work that remains to be done or possibilities for future work relating to this thesis will be found in chapter 7.

Chapter 2

Background

This chapter presents relevant research material and theory related to the thesis topic. First, it looks at important definitions of e-learning and e-assessment. Secondly, it details the task type and its relevance for the thesis before examining related work. It talks about the previous work done by Jorgensen and Kvannli, and how this thesis will build upon their efforts. An explanation of Inspera Assessment and the definitions of the QTI format is also included. Some more details on the task type that is part of the scope (see section 1.3) will also be provided.

2.1 E-learning and e-assessment

In this thesis, the focus will be on improving the digital exams process specifically for programming courses and programming tasks. These improvements for the digital exam can be used for learning and assessment throughout a course, not only as a final examination. A digital exam is a tool under the commonly used terms of e-learning and e-assessment.

E-learning serves as a label for an extensive collection of uses of information- and communication technologies to distribute, present, manage and support individual or group learning activities, typically in a computer-based and connected networking context [19, p. 35]. Some definitions include any activity that uses a technological item in some way for learning [20].

There are a lot of different elements regarding the assessment process that can be improved upon with the usage of information technologies. This can include pre- and post-testing, diagnostic analysis, student tracking, rubric use, the support and delivery of authentic assessment through project-based learning, artifact collection, and data aggregation and

analysis [21]. The usage of e-learning and information technologies in the assessment process is referred to as e-assessment. For this thesis, the area of study pertains specifically to the usage of the e-learning and e-assessment tools used to assess a students learning and skills in a final examination.

2.2 Digital Programming Tasks

In addition to the improvements regarding the assessment process, as explained in the previous section, e-learning and e-assessment tools provide opportunities to enhance the tests themselves. Digital applications can be used to test in different, and maybe better ways, than the traditional paper-based tests [1]–[3] as mentioned in section 1.1. This section provides insight into the types of tasks that are relevant for this thesis, and that can be created and completed digitally to improve the examination.

2.2.1 Parsons Problem

As explained in section 1.3, one of the goals for the thesis is to create an IT artefact that can create *Inline Gap Match* tasks. An *Inline Gap Match* task is a task where the examinee is presented with "gaps" containing characters, words, or sentences. These gaps have to be drag-and-dropped into their correct position in an empty matching gap in the task area. This could, for example, be to fill in a missing word to complete a sentence, place the correct mathematical operator between two numbers to create the correct result, or in this case, place a piece of code correctly to make the function work as expected.

One type of task that can be created using the *Inline Gap Match* task type is Parsons Problems [22]. Parsons Problems are drag-and-drop tasks where one must arrange blocks of scrambled code to produce the correct output or completed code. A feature of the Parsons Problems that are often included to make the task more difficult is called *distractors*. A distractor in a Parsons Problem is a code block that is meant to distract. It is not part of the correct answer, and should therefore not be used if the candidate wants to achieve a full score. A good distractor is usually created in a way that it is difficult for the candidate to know whether it is a correct answer or just a distractor. This could, for example, be done by having it closely resemble the correct answer.

Another variant or addition to the Parson Problem is the two-dimensional Parsons Problem [23]. In this version, the code blocks must be correctly placed in two dimensions, meaning that the indentation levels also matter. With the *Inline Gap Match* task type in Inspera, it is possible to have distractors and create multiple gaps in a two-dimensional order. This makes it a good task type for the creation of Parsons Problems and an alternative to the type used by Jorgensen and Kvannli [15], as will be explained in the following section 2.3.2. The complete process for creating a *Inline Gap Match* task in Inspera is described in section 2.6.2

Parsons Problems can be used to make more engaging code completion tasks that teach or test syntactic and semantic language constructs [23]. Traditionally the usage of repetitive tasks to foster learning has proved effective, but they are also tedious and boring [22]. With Parsons Problems, the tasks can be more efficient, effective, and require less cognitive load, while still providing the same learning outcome as the alternative task [23]. The alternative task would be to fix and write code from scratch, while a Parsons Problem can have the errors as distractors and let the user drag-and-drop all the correct code blocks to complete the code.

The usage of distractors can provide many benefits. They can be used to make the task more difficult in regards to certain angles that the test administer wants to examine. If the professor wants to test the student's ability to determine the correct syntax of e.g., a for loop, he could add common mistakes as distractors. For a task concerning the for loop syntax in the Java programming language [24] the distractor could be *for (int i = 0; int i < 5; int i++)* and the correct answer *for (int i = 0; i < 5; i++)* . Having multiple different distractors for the same task could enable the professor to create versions of the task that are not like each other, but test the same knowledge. It is important to choose distractors that are equal so as not to make the task more difficult for one student than another. If this is achieved, the tasks can reduce cheating by giving away less information when students peek at each other's tasks. A task with two distractors chosen from a pool of ten total distractors would make it possible to create 45 unique tasks.

2.3 Related Work

This section provides some insight into the work that is related to this thesis. It means both the research that is conducted that can be linked to this thesis as well as other systems that are similar to the proposed IT artefact.

2.3.1 Learning Effect of Code Completion Puzzles

Some research has been done on the learning effect of code completion puzzles. For this thesis, it is relevant to examine if there is a use case for creating code completion tasks instead of utilizing only the standard code input tasks. Code input tasks refer to the traditional way where a student writes code on a piece of paper, or for a digital task, writes it into a text or code input field. Two common task types for this input type are code tracing and code writing [25]. Code writing could be further sectioned into code completion and code generation [26]. An alternative to these types of tasks is code completion puzzles like one could create using the *Inline Gap Match* task type, which is the focus for this thesis. Some research has shown that Parsons Problems, as detailed in section 2.2.1 above, can provide benefits compared to the common code writing tasks [25].

Other research shows that there are a good learning effect and other benefits from us-

ing an alternative way of assessing a student's programming skills. *Code mangler* tasks where the code is scrambled, and the student must piece it correctly back together is one example. These task types require less effort to grade, provides a higher level of confidence in the grading while also correlating strongly with the student's abilities, just like in the traditional question style [27]. A part-complete solution method where the student is presented with some parts of the code and has to fill in the missing parts has also been researched. The results show that it can provide sound learning effects, but also that there are differences between the various methods one could implement to fill in the missing parts [28].

2.3.2 Jorgensen and Kvannli

Jorgensen and Kvannli [15] researched the possibility to design and create a prototype to streamline the generation of drag-and-drop Parson Problems for digital programming exams on the Inspera Assessment platform. They evaluated the effect of their prototype in regards to usability compared to the usual manual process of creating tasks directly in the Inspera Assessment interface. Their proposed system aims to automate parts of the process to make it more effective and easier to use for the end-users.

Their prototype was designed to be able to create drag-and-drop tasks on the IMS Question and Test Interoperability specification (QTI) format in version 2.1 [29]. This format lets the user import externally created tasks into Inspera and is a standardized format to accommodate interoperability between systems. This meant that the results of their thesis potentially could be utilized by other systems than just Inspera if those platforms also support QTI 2.1.

The results from their user tests showed that the prototype that was created presented significant improvements in regards to usability compared to the Inspera process. Especially in regards to effectiveness, efficiency, and satisfaction among the test subjects. Their prototype was the preferred method to create drag and drop Parsons problem tasks for Inspera.

2.3.3 JS-Parsons

JS-parsons [30] is a JavaScript library to construct Parsons Problems as described in section 2.2.1. It has support for distractors, indentation levels for two-dimensional problems and variables inside statements. Two different modes are available to either let the user rearrange already placed lines or drag-and-drop lines without placement. It is free to use and open-source so that it can be reused or changed by volunteer contributors as well as the creators.

2.3.4 QTI.JS

QTI.JS [31] is a JavaScript-based tool that supports the QTI 2.2 version (see section 2.5 for an explanation of QTI). It supports all the 21 question types and can be used to create, import, export, or exchange tasks in the correct format. QTI.JS is server-less, requires no configuration, and is fully themeable [31]. While it was scheduled to release in 2019, it is currently not yet completed (January 2020).

2.4 Inspera Assessment

Inspera Assessment (or just Inspera) is a platform for online assessment. Inspera's explanation of their system is as follows: *Inspera Assessment is a cloud-based assessment platform supporting the entire examination process, including planning, designing, delivering, invigilating, marking & annotating, sharing, and improving* [8]. Inspera Assessment can be used to assess many different task types, be it exercises, projects, tasks, thesis's, practical or oral exams. There is also specific support for tasks within certain genres, like mathematics formulas and code formatting. With the support for design, creation, cooperation, communication, collaboration, and delivery, it can be used for e-learning according to the definition explained in section 2.1.

In Inspera Assessment the question types are divided into three categories; *automatically evaluated*, *manually evaluated* and *not evaluated* [32] . All the different task types can be seen below in figure 2.1. In the user interface, automatic marks correspond to automatically evaluated, manually marked to manually evaluated, and not marked to not evaluated.

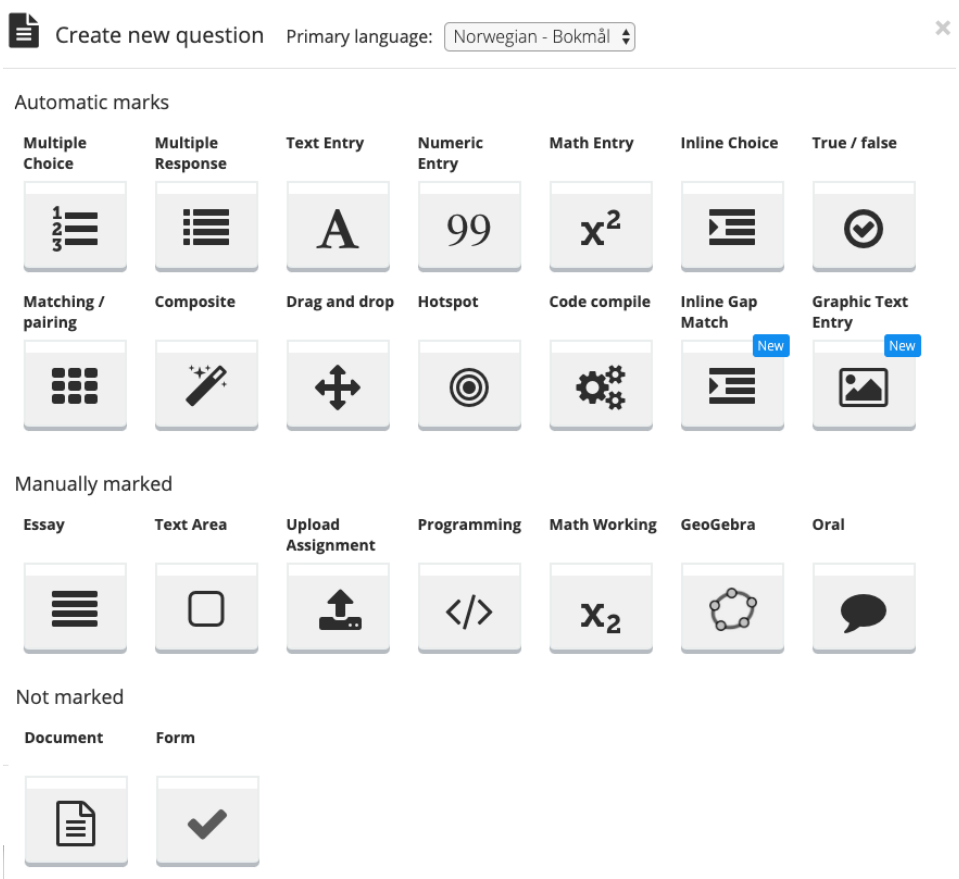


Figure 2.1: Task types that can be created in Inspera Assessment

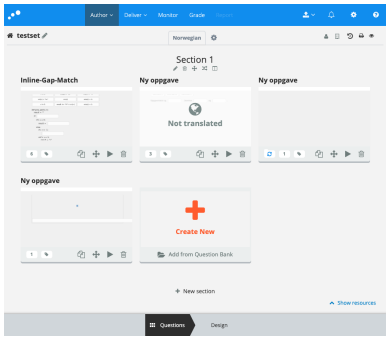
Question types like multiple choice, hotspot, and text entry can be automatically evaluated. This means that Inspera Assessment can automatically determine whether the answer provided by the candidate is correct and give a score. The manually evaluated question types like essay, text area, and GeoGebra will need to be marked by a person manually looking trough the answered question to determine a score. Finally, the last category, not evaluated, contains the document and form types. These types are only used to provide extra information to a question set and should themselves not be graded or receive a score.

Marking and grading are time-consuming tasks for teachers and professors. In addition to the automatic and manual marking tool for each question explained in the paragraph above, Inspera Assessment has a grading tool. It is highly configurable and can provide support for almost any workflow. While it needs to be configured, it can fit each user's needs and therefore improve efficiency. Some features it includes are flexible learner feedback, facilitation of discussion between markers, complete candidate overview, complete questions overview, and easy marking. Utilizing the marking and grading tool can improve

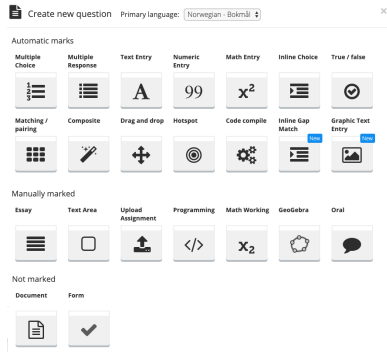
the marking process to make it easier, faster, and at the same time, more reliable.

Inspera has open APIs to facilitate interoperability and utilizes the IMS QTI 2.1 [29] specification to achieve this. More information on the specifics of this specification can be read in section 2.5. With the usage of this standard, they can both export and import questions and question sets from other assessment platforms or question databases. The APIs follow assessment technology standards [8] that allows for functional integration with other Student Information Systems and Learning Management Systems as well as the possibility to create custom question types.

The process of creating a new question, or adding a question to a complete question set can be seen below in figure 2.2 and figure 2.3. First step is to create a new question, or add a new question to a set (see figure 2.2a). The user will be prompted to select from the list of available task types (see figure 2.2b). Depending on what question type is selected, the process and view will vary some. However, they are created similarly, and the controls have many similarities across the different tasks. An example of a *Text Entry* task can be seen in figure 2.3a, and the preview button lets the user see what the task will look and perform like for the students (see figure 2.3b).

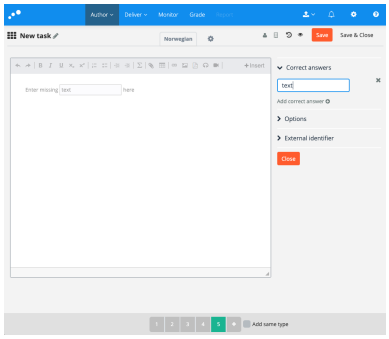


(a) Question Set

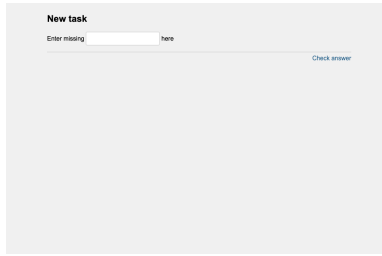


(b) Select Task Type

Figure 2.2: Inspera Assessment Question Creation Process 1



(a) Task Creation



(b) Task Preview

Figure 2.3: Inspera Assessment Question Creation Process 2

2.5 QTI

Assessments and tests are essential tools for education and learning. The creation of good tests can also be very time consuming and require many resources. To save costs, it is beneficial if the tests are reusable. Being able to reuse the tests ensures sustainability as well as the ability to preserve investments and intellectual assets [33]. If one is also able to reuse each question of the tests, it is possible to create unique tests by selecting, maybe randomly, from a "bank" of questions. One such proposed standard is the IMS Question, and Test Interoperability (QTI) specification, their description of the specification is as follows:

The IMS Question & Test Interoperability (QTI®) specification enables the exchange of item and test content and results data between authoring tools, item banks, test construction tools, learning platforms, assessment delivery systems, and scoring/analytics engines [34].

More specifically, the QTI specification describes a data model that lets the user represent questions, test data, results, and reports. The specification enables these types to be interoperable and reusable between different tools. It could be assessment systems, learning platforms, question banks, or authoring tools. The data model is an abstract description written in the Unified Modeling Language (UML) to support a wide range of data modeling tools and programming languages. However, the interchange between systems is facilitated by the widely used eXtensible Markup Language (XML) [34].

By utilizing the standardized format and specification, different platforms and tools can create and then import, export, or exchange the same questions. The data model defines a set of interaction types that can be used to create a lot of different question types, some of which can be seen in Insperas implementation in figure 2.1. With a standard like QTI, a university could use multiple different systems,e.g., Inspera for exams and Blackboard

[18] or Canvas [16] as Learning Management Systems, LMS, and reuse questions or tests between them as long as they support the same QTI version. Additionally, if the university should ever drop the usage of a system, they could avoid losing all the questions or tasks created if the new system supports the same standard. One could also use the standard to share and exchange questions or tests with other, even foreign, universities.

As Piotrowski [33] discussed, there are limitations to the QTI specification. There are "breaking changes" between the version numbers, which does not allow for interoperability and means that if a platform uses an older version, it might not work when updating the data model to a newer version. Even the same version number specification can be implemented differently on one platform compared to another, and make the exchange of tasks between them incompatible. The underlying XML structure is also different because the different versions are built upon different data models. Figure 2.4 below shows an example XML file on the QTI 2.1 specification that is used by Inspera. As Inspera uses this version, and the thesis uses Inspera for comparison, this version is also the main focus of this thesis.

```xml
<?xml version="1.0" encoding="utf-8"?>
<assessmentItem identifier="VE-IP-01" title="QTI v2.1 Entry Profile Single T/F Item Test Instance" adaptive="false" timeDependent="false"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.imsglobal.org/xsd/imsqti_v2p1" xsi:schemaLocation="http://www.imsglobal.org/xsd/imsqti_v2p1 http://www.imsglobal.org/xsd/qti/qtiv2p1/imsqti_v2p1p1.xsd">
  <responseDeclaration identifier="RESPONSE" baseType="identifier" cardinality="single">
    <correctResponse>
      <value>true</value>
    </correctResponse>
  </responseDeclaration>
  <outcomeDeclaration identifier="SCORE" cardinality="single" baseType="float" normalMaximum="1" normalMinimum="0">
    <defaultValue>
      <value>0</value>
    </defaultValue>
  </outcomeDeclaration>
  <itemBody id="content5">
    <p id="content2">Answer the following question.</p>
    <choiceInteraction maxChoices="1" minChoices="0" id="content1" shuffle="false" responseIdentifier="RESPONSE">
      <prompt id="content6">Sigmund Freud and Carl Jung both belong to the psychoanalytic school of psychology.</prompt>
      <simpleChoice id="content8" fixed="true" showHide="show" identifier="true">
        <p id="content7">True</p>
      </simpleChoice>
      <simpleChoice id="content4" fixed="true" showHide="show" identifier="false">
        <p id="content3">False</p>
      </simpleChoice>
    </choiceInteraction>
  </itemBody>
  <responseProcessing template="http://www.imsglobal.org/question/qti_v2p1/rptemplates/match_correct"/>
</assessmentItem>
```

Figure 2.4: QTI 2.1 Question Example

2.6 Task Types

This section will present a more detailed explanation of the task that was chosen to focus on in this thesis. It describes how the task is created in Inspera Assessment, which could give some insight into the improvements that the proposed IT-artefact can achieve. The design decisions and shortcomings of Inspera Assessment when it comes to creating programming tasks will be highlighted. This can show where the proposed IT-artefact can improve the process, usability, and effectiveness of the task creation workflow.

To explain the process of the task creation, let us assume we have a piece of code that we want to use in a task. The piece of code can be seen below in figure 2.5, and the red squares are the parts of the code that the task solver should fill out.

```
def poly_part(c, n):
    result = ""
    if c != 0:
        if n == 0:
            result = str(c)
        else:
            if c == -1:
                result = "-x"
            elif c == 1:
                result = "x"
            elif abs(c) > 1:
                result = str(c) + "*x"
            if n > 1:
                result += "**" + str(n)
    return result
```

Figure 2.5: Task to be created

2.6.1 Test Set and New Question

For each question or task type, there is a basic setup process in the beginning before the selection of which task type to create. To be able to create a new task or question, one can either just create a single new question or add a new question to a question set. A question set is what eventually is created as a complete exam that can be solved by examinees in the Safe Exam Browser [35]. When a question set is created, it is possible to add questions to it. Inspera Assessment has a lot of different task/question types to choose from as explained in section 2.4 and seen in figure 2.1.

2.6.2 Inline Gap Match

After selecting the *Inline Gap Match* task type in the create new question interface (see figure 2.1), the user is presented with a new task example as can bee seen in figure 2.6 below. The newly initialized task contains a task description text, some task text with three gaps, and three correct answers that correspond to one gap each. To begin the creation of the task with the code piece in figure 2.7, the user needs to either manually type in the code or paste it in the task text area.

In figure 2.6 below the code is pasted into the task text area. Notice that there are no

indentations even though the code was copied and pasted with indentations and without the remove formatting option. In both cases, typed manually or pasted, if there is a need for indentations, they will have to be added manually, and this text area does not support the use of the tab-key. This means that the user will have to make indentations with spaces. It can be cumbersome and difficult to create the correct amount of spaces on each indentation level, but it is important to make the task appear correctly. One trick that can help speed up this process is to copy the number of spaces you want as an indentation (e.g., four space equals one indentation level) and paste it once for each indentation level needed.

At the late stages of the project Inspera changed the QTI version as explained in iteration 4 of the development process (see section 4.1.5). This also changed the formatting of the task text area so that it sometimes kept the formatting when code was pasted. The researcher could not determine why it only worked most of the time, but for most users it means that they do not have to do the cumbersome task of inserting the correct indentations themselves.

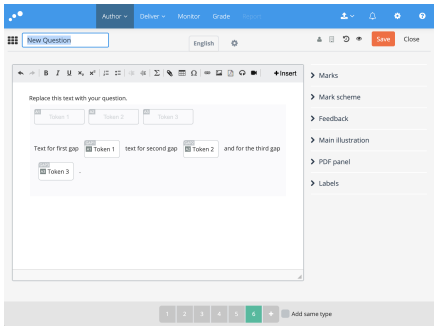


Figure 2.6: Initial Task Creation Interface

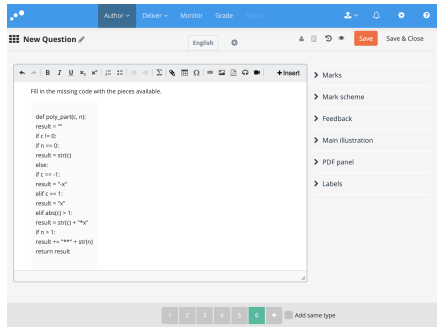


Figure 2.7: Code Pasted into Task

In figure 2.8 below, the code has been indented manually with spaces on all lines that require it. The next step is to create the gaps that the examinee will need to fill. This has to be done by clicking the *+Insert* button. When the button is clicked, it will insert a gap at the currently selected position in the task text field. If some text is selected, the button press will replace the selected text with a gap. In figure 2.9 below the part that should become a gap (as seen in figure 2.7) is highlighted. Since the text is replaced, it can save the user some time by copying the text to the clipboard first, before replacing it with a gap. In the separate field for inputting the correct answer, this text can then be pasted instead of having to type it in again.

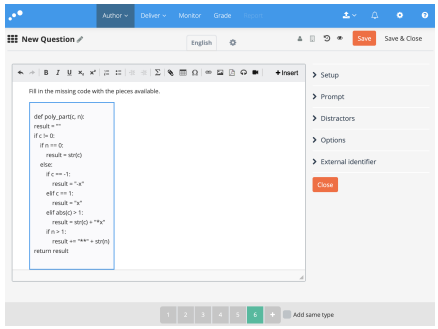


Figure 2.8: Manually Indented with Spaces

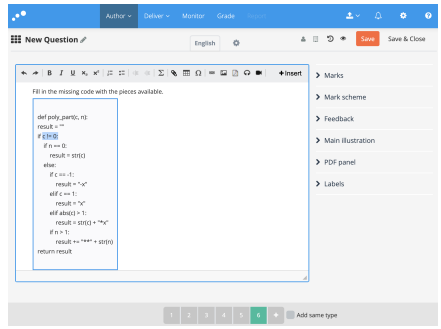


Figure 2.9: Gap Selected

After creating or inserting a gap, the user will need to click the new gap two times to bring up the side panel menu for that gap. This side panel menu can be seen to the right in 2.10 below. In the drop-down menu called *Correct answers*, the user can click *Add correct answer* to add an answer that should be regarded as correct when drag-and-dropped into this gap. This is where it could be more efficient to have copied or cut the correct answer before inserting the gap, as explained at the end of the last paragraph, because you could then just paste it in this field.

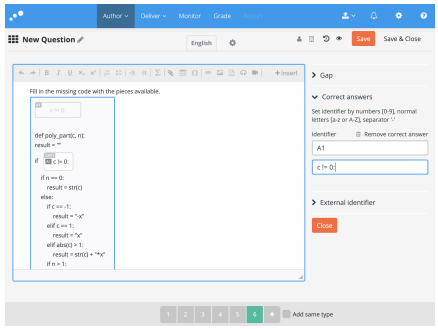


Figure 2.10: Add Correct Answer

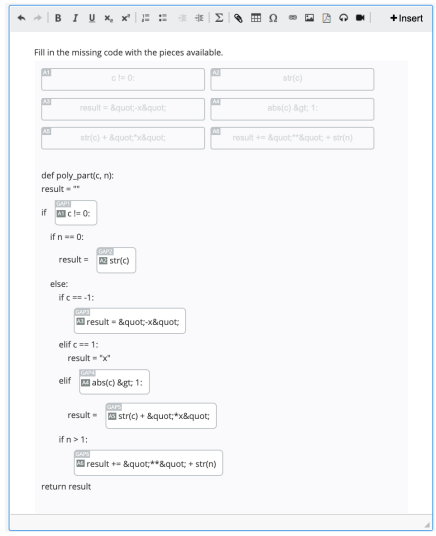


Figure 2.11: Complete Task

When this process of creating gaps and setting their corresponding answers is done for every code piece marked in figure 2.7, the task is completed. The completed task in the task creation interface can be seen in figure 2.11 above. There is also an option to add distractors in the main side panel, which lets the user add optional additional wrong answers

to make the task harder for the examination. The other setting that can be made are:

- Positioning of the possible answers to be drag-and-dropped: Top, right or bottom
- Order of the possible answers: Random or ascending
- Reuse of possible answers: Allowed or not allowed

During the creation of a task and after it is saved or completed, it is possible to preview the task. The user will then, in a new tab or window, be presented with the task as it will look for the examinee during an exam. This task preview window for the task created during all the steps explained in this section can be seen in figure 2.12 below. In figure 2.13, some of the possible answers are drag-and-dropped into incorrect gaps, one moved into the correct gap, and the rest (two possible answers) are not placed at all. The result can be seen in the bottom right corner.

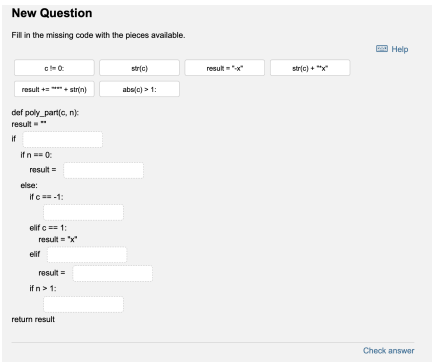


Figure 2.12: Task Preview

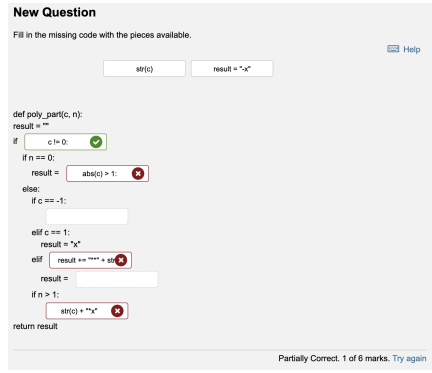


Figure 2.13: Preview Solved

Chapter 3

Method

This chapter will use the research questions defined in section 1.4 to discuss the different research methods it is possible to use to examine these questions. The strengths, weaknesses, and trade-offs will be presented and discussed. Of the possible research methods evaluated, a conclusion will be drawn on what method is most suitable for this thesis. The selected research methods will then be explained in detail how they were used to conduct this research and answer the research questions.

3.1 Literature Review

A literature review was important first to define the motivation, scope, and research questions for the thesis. Secondly, the literature review was used throughout the thesis to support the research, results, discussion, and conclusion. Together with the experiences of the researcher,the background and motivations explained in section 1.1 and 1.1.1 the literature review formed the basis for this thesis. The supervisor provided some of the literature that was reviewed, and some were even authored or co-authored by him. This is because the supervisor is actively doing research within the same area and has extensive experience with the available research material. Every piece of literature that was considered important after analyzing the abstract was assessed, read, critically evaluated, and recorded [36, pp. 83-85]. An example of the records for some of the initial material provided by the researcher can be seen below in table 3.1.

Title	Bibliography	Summary	Evaluation	Relevance
What Good Can Digital Exams do for Constructive Alignment	G. Sindre	The paper discusses whether digital exams can increase the validity of exam tasks and thereby improve how well the assignments test the intended learning outcomes. It looks at how this can be done with digital exams compared to old paper-based exams. There is however a limitation on currently available systems that make the paper suggested improvements not possible currently	The paper brings up important aspects of intended learning outcomes (ILO), and how different assessments fail to cover all ILO's. Most points are referenced. Basically a good evaluation of how ILOs are covered or not covered, and how it differs between pen&paper and digital exams. Also, how both ways have different ways to tackle the difference and advantages in the same regard comparatively.	Digital exam >paper exam. Can be constructed more easily, and therefore more often to cover more of the ILO's
E-exams and exam process improvement	G.Sindre and A. Chirumamilla	Explores advantages on digital exams over paper-based exams. Deeply looks at the processes involved in the examinations and how the can be improved upon, Explains the three goals of an exam, high reliability, high validity and low cost and how the impact each other.	Few references comparatively considering the length of the paper. Goes into mostly how processes can be changed with digital tools to make the exams more effective.	Has a lot of sources on why digital >paper. Also how digital exams can be good in many ways. Wants open and well documented APIs so that it can be extended with extra services.
E-assesment in programming courses: Towards a digital ecosystem supporting diverse needs?	G.Sindre and A. Chirumamilla	Discusses wants and actual features of e-learning and e-assesment applications for programming tasks. Specifically code writing and parson problem tasks. It then tries to examine how applications or systems like this can improve e-assesments	Very well laid-out and explained with a lot of good sources. The research questions to be answered are more of a discovery answer than definite answers to a specific problem. Decisions taken are justified. Focuses quite heavily on personal experiences with only BB/Inspera, other programs are mentioned but not explored at depth (lack of access)	Talks about parsons problems and tool support for good programming tasks. Also different types of programming tasks for assignments/exams
E-exams versus paper exams: A comparative analysis of cheating-related security threats and countermeasures	G.Sindre and A. Chirumamilla	Discusses how e-exams have advantages over paper-based exams. At the same time, they open for new ways to be exploited or cheated on. It compares the attack angles difference between the e-exam and paper exams. It finds that although they are different, neither has a clear advantage or disadvantage from a security perspective	It is good at narrowing the scope for what is actually discussed and evaluated. A paper with a different claim (basically saying that BYOD e-exams are by definition less secure) than they are trying to make is "debunked". However, without clearly stating how the assumption was made in the referenced paper (it might not have been).	Some relevance regarding cheating and how the BYOD alternative creates new attack angles. Might be tied into how it also provides many opportunities regarding using familiar development environments or other tools. Also cost saving

Table 3.1: Literature Review Record

To explore, search for, and find other relevant literature to review, the researcher used a search matrix. Keywords were defined to build a search query using relevant alternative terms [36, pp. 80-81]. If the single most relevant keyword provided too many results, the researcher tried to limit the number of hits by combining multiple words using the Boolean operators *AND*, *OR*, and *NOT*. For example "programming *OR* code *AND* tasks". An example of a search matrix used to find additional relevant literature to the digital programming exams can be seen below in table 3.2.

Search	Date	Keywords	Constraints	No. of results	Read abstracts	Chosen articles
S1	25.09	"digital exam" OR "computer exam"		2660		
S2	25.09	"programming task" OR "programming assignment"		17200		
S3	25.09	S1 AND S2		10	3	0
S4	25.09	S1 AND effect OR performance OR learning		1870		
S5	25.09	S4 AND "programming"	2018-	50	4	Coding by hand or on the computer? Evaluating the effect of assessment mode on performance of students learning programming

Table 3.2: Search Matrix for Digital Programming Exams Literature

3.2 Research Method for Research Question 1

As described in section 1.4, the first research question, *RQ1*, asked about how one could design and create an IT artefact to create tasks on the QTI-format. Therefore a natural choice for the research strategy could be the design and creation research strategy [36]. The other research strategies described by Oates [36, p. 33], e.g., survey, case study, or action research, would not directly lead to the creation of an IT artefact, which is a prerequisite for the possibility to answer *RQ2*. Therefore, when this is considered, the most fitting choice is to use the design and creation research strategy to answer *RQ1*.

3.2.1 Design and creation

The design and creation research strategy focuses on developing new IT products, also called artefacts [36, p. 108]. There are different types of IT artefacts. The one that is to be developed for this thesis falls into the "instantiations" category because it will be an IT-system that showcases how the task model and creation methods can be implemented in a new computer-based system [36, p. 108].

For a design and creation project to be considered research, it should provide some new knowledge. It should also showcase academic qualities like argument, justification, analysis, explanation, and critical evaluation [36, p. 109]. There is also an important distinction to be made between "normal" design and creation, and design and creation research. For it to be design and creation research, the process should allow for something to be learned. Unlike industry run projects where time and resource limits make backtracking and design changes unwanted, for a research process, this reiteration is precisely what could provide useful knowledge [36, p. 114].

When developing an IT artefact with the design and creation research strategy, the process

should follow the established principles of system development [36, p. 111]. The system development methodology chosen for this thesis to explore *RQ1* and develop the IT artefact is described in section 3.6.

After the IT artefact was developed, it had to be evaluated [36, p. 115]. The reason why the IT artefact was created (see section 1.3) had to be put under test to research if it fulfilled its intended purpose. One of the main criteria was the usability and functionality because the goal of the IT artefact and the research question (*RQ1*) pertained to whether it is possible to design and create a system that can perform certain operations, most importantly create tasks on the QTI format. Furthermore, many of the IT artefacts functions and usability was tested trough the exploration of *RQ2*. This was because *RQ2* relates to *RQ1* and the usage of the created IT artefact. The research method and evaluation of *RQ2* can be seen in section 3.3.

The IT artefact is evaluated in use to be able to establish "proof by demonstration" [36, p. 116]. The task that will be used to test the IT artefact is one that could be called a typical task used in a real-world scenario. It is, however, likely that a real exam would contain more than just one task like this. It is also the goal to test the IT artefact on the real potential end-users, in this case, professors, lecturers or others that usually create programming tasks for digital exams. These two approximations allow the IT artefact to be tested in a somewhat close to "real-world evaluation" situation.

Ethics

There are multiple points of unethical and unlawful acts that have to be considered for any design and creation project. For this thesis, some temptations have to be accounted for, and the usage of the created system should only be used in ethical ways [36, p. 63]. Access and data copying will be mitigated by storing only the user's temporary data, and only in the users own browser-session. This ethical point will have to be further accounted for if this IT artefact is extended upon in the future with implementations that provide access and data storage.

Privacy and anonymity are kept intact by the same metrics. There is no sign on or user data storage. The system users could input private data in the IT artefact even though this is not its intended use, nor part of the workflow in any way, but it would only be available to him- or herself. Additionally, no data is gathered or observed without informed consent.

3.3 Research Method for Research Question 2

As described in section 1.4, the second research question, *RQ2*, asked about what improvements the IT artefact can provide compared to using the assessment platforms' interface. When choosing a research strategy to answer this research question, it is essential to con-

sider how one best could gain insight into the comparison between the usage of the created IT artefact and the assessment platform. Many of the research strategies described by Oates [36, p. 33], e.g., survey, experiment, case study, and ethnography, could be relevant. The survey strategy does not allow testing the created IT artefact or the Inspera Assessment interface properly. The requirement of a decent sampling size [36, p. 94] is also difficult considering the sampling frame is quite small already, being only professors or teachers using a digital platform (preferably Inspera Assessment) to create programming tasks.

The experiment research strategy could be conducted with a purpose to test whether using the IT artefact enables more effective and user-friendly task creation than Insperas interface. However the need for *repeatability* [36, p. 127] is difficult to maintain because of the quite large focus area. Many factors might influence the experiment, and some of those factors are hard to control or remove. An experiment is based on a hypothesis to be tested [36, p. 127]. For this thesis, the *RQ2* is posed almost as a hypothesis. Predictions for the hypothesis could be, e.g., "When the IT artefact is used, it takes less time to produce a task" or "using the IT artefact provides a higher level of usability."

In a case study, there is usually focus on one thing; in this case, an information system. This one thing is then studied in-depth with a lot of different data generation methods (interviews, observation, document analysis, and questionnaires) to get detailed information on how this thing works. It must not be controlled in the same manner as an experiment as it measures and allows for the complexities of the real world to influence the case study. It is fitting if boundaries between phenomenon and context are not evident [36, p. 142]. Natural setting makes it difficult since there will be no instance where anyone uninvited uses the created IT artefact.

Ethnography could be similar and more realistic than using the experiment research strategy. It is, however, also more challenging to implement because of the time-frame required, and almost impossible considering the ongoing COVID-19 pandemic [37]. Getting useful data on real-world usage might provide insight outside what is gathered in a controlled lab environment. However, the usage of the systems that are the target of this research question (*RQ2*) is usually limited to only a few days before the exam draft delivery due date. This makes it difficult to time the natural observation of the processes that need to be evaluated to answer this research question.

Considering all the strengths and weaknesses highlighted in the paragraphs above, this researcher chose the experiment research strategy to answer *RQ2*. Oates highlights the importance of measuring something before the introduction of the new method, in this case, the IT artefact, and again afterward, while accounting for the other environmental factors that might affect the measurements [36, p. 134]. If this is not conducted correctly, the researcher will be unable to determine whether the IT artefact has caused a change. While case study and action research could be very relevant to identify what happens when the IT artefact is introduced, it is important to showcase that it is better to use than the current process, not only what happens if it is used. This is because if it is not better, more efficient, or has better usability, there is really not that much of a reason to create a

new IT artefact like the one proposed in this thesis.

3.3.1 Experiment

Hypothesis

The experiment research strategy investigates cause and effect relationships, seeking to prove or disprove a causal link between a factor and an observed outcome [36, p. 127]. A hypothesis statement is formed, which is then tested to either prove or disprove it. It is performed in a controlled manner that removes all factors except the one factor that should cause the wanted outcome. When the research experiment is repeated many times to provide the same results, by the researcher and others, the hypothesis can be said to be proven.

Hypothesis: *The IT artefact provides a higher level of usability and is more effective at supporting a question author in creating Inline Gap Match tasks for Inspera, compared to using Inspera Assessment's own interface*

Tests

To test the hypothesis, the researcher devised two tests to compare the effectiveness and level of usability between the IT artefact and Inspera. The *control* test was to create a *Inline Gap Match* task with Inspera. With the second test the IT artefact was used as the *treatment* to complete the exact same task [36, p. 135]. For the test with the IT artefact it was decided to include the steps of exporting and importing the created task into Inspera Assessment. One could argue that this is not part of creating the task with the IT artefact, and this would take a longer time. However, it was important to show that the IT artefact could provide benefits compared to using the current system, Inspera. Since the IT artefact does not have an interface for students to be able to solve the task, it must be imported into a platform that does for it to be as complete as the same task created in Inspera. The export and import step of the process would be of diminishing effect the more tasks that are created at the same time as they could all be imported in the same operation. One could, therefore, call this the "worst-case" test. If it is more effective at creating just one task, one could assume it would likely be even more effective for multiple tasks.

A test plan for each test was designed to guide the experiment participants. Half of the test participants started with one test, while the other half started with the other. This was done to prevent any effects of learning between the tests. The complete test plans that were handed to the participants can be seen in appendix A.2. While the test participants were free to solve the tasks as they wanted, the general approach they would use for each system are as follows:

For Inspera:

1. Choose new Inline Gap Match task
2. Copy and paste code into task area
3. Remove placeholder text
4. Repeat process to create gaps:
   (a) Select text or code that should become a gap
   (b) Copy or cut text to clipboard
   (c) Click on the *+Insert* button
   (d) Click on the created gap twice
   (e) Click on *Correct answers*
   (f) Click on *Add correct answer*
   (g) Select the *Alternative* text and replace it with clipboard content
5. Add distractor
6. Preview or save complete task

For IT artefact:

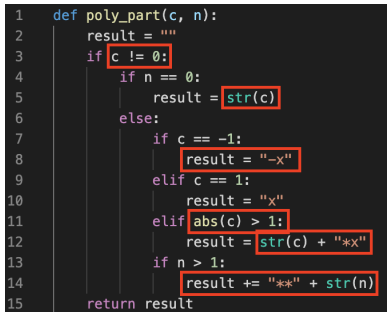
1. Create a new task
2. Copy and paste code into task area
3. Navigate to the second tab
4. Repeat the process to create gaps:
   (a) Select text or code that should become a gap
   (b) Click the crop icon
5. Add distractor
6. Export task
7. Import task into Inspera
8. Preview or save complete task

In a more detailed description, the test plan was to let the participants manually create the task in the Inspera Assessment question creation interface. The process was recorded from the video-conferencing tool with the screen of the subject shared and visible. This was done to enable the researcher to follow the action and thought process of the subjects, as well as be able to review the video later to gain more insight and data. It was essential to not share any of the test metrics and measurements with the experiment participants to

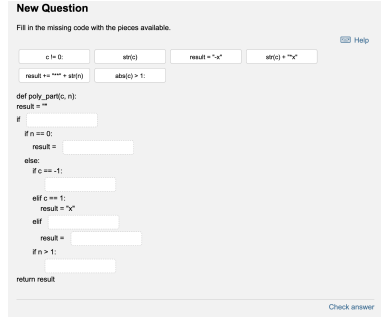
avoid any behavioral changes as a result. After the test was completed the subject was given a questionnaire to answer about the usability of the Inspera Assessment interface for task creation. A more detailed description of the questionnaire can be seen in section 3.4.2.

For the second test, the independent variable, the created IT artefact was introduced to the subject. They then had to create the same task again, but this time using the IT artefact. After the task was completely created, they were once again given the (almost) same questionnaire. In the end, there was conducted a short interview to gather more experiences and relevant feedback for the researcher. More information about the conducted questionnaires, observation process, the interview, and measurements can be read about in section 3.4.2.

The task to be created can be seen in figure 3.1 below. For both tests, the code to create the task from can be seen in figure 3.1a below. This is a code that has previously been used in an exam with the *Inline Gap Match* question type (see figure 2.1), making it relevant for the research to be as close to a real-world use case as possible. The outcome of the tests should be the same complete task, as seen in the preview in figure 3.1b below.



(a) Code to Make Task



(b) Task Preview

Figure 3.1: Experiment Task Creation Test

Independent and dependent variables

Dependent and independent variables [36, p. 129] for this experiment research strategy focus on the change to the task creation process. The newly introduced independent variable is the IT artefact that is the outcome of this thesis. The dependent variables that will be affected include e.g., time to completion, satisfaction, usability and effectivity. It is the recorded *effect* the introduction of the independent variable has on the dependent variables.

Controls

Controls of all the factors were handled as best as possible to ensure that only the independent variables caused a change in the dependent variables [36, p. 130]. The potential subject group is quite small, relatively speaking, being only professors or other staff tasked with creating digital programming exams or tasks. It is the researcher's opinion that using a control group might allow for factors regarding each individual to affect the results more than by not using a control group. This is because the group sizes will be small since it is hard to find enough participating subjects, and therefore individual differences can be quite impacting. The researcher assumes that the control will be best if each subject receives the same manipulation of the independent variables.

Observation and measurement

There are many observations and measurements that can be used in a experiment research strategy [36, p. 130]. For this research the following were implemented:

1. **Project data:** Time to completion, Number of bugs or errors, Number of actions needed
2. **Self-report responses:** Questionnaire and feedback to determine usability, effectiveness and satisfaction.
3. **Behavioural counts:** Number of times subjects asked for help or found help outside of the system in use, number of system and user errors as well as unintentional actions.

Pre- and post-test

To accurately measure the change, a pre- and post-test were conducted [36, p. 131]. The pre-test and post-test were exactly the same to measure the *causes* of the IT artefact. They consisted of the measurements done during the observations as explained in section 3.3.1 above and the SUS questionnaire detailed later in section 3.4.2.

Internal validity

To maximize and ensure proper internal validity, some threats will be mitigated in the following ways [36, p. 131]; no control group will avoid differences between the experimental and control group. The tests with both Inspera and the IT artefact will be completed right after each other. This was done to avoid any history or time between to influence the results, and the same measurement will also mitigate the changes that could be caused by

maturity. Instrumentation will be controlled by video-recording each test, using the exact same questionnaires for all subjects, and a script for the interviews if the interview results are to be used as data.

Following are some factors and conditions that were changed or removed to improve the internal validity of the experiment:

- **Task description:** The need for a task description is present in a real-world example. However, for the experiment, the process would be the same, whether using the IT artefact, the independent variable or not. Thus the need to create a task description was removed.

- **Time measurement:** To better measure the time used for the factors that are changed by the independent variable, it was decided to control the time-tracking window. The start-point of the tracking was set to when the subject started the creation of a new task, to avoid having the variable and irrelevant time to initialize a new test set or question to influence the variables. It was done to remove the navigation times needed in Inspera Assessment, especially if the user was unfamiliar with the interface and because it does not relate to what the IT artefact aims to change. The stop-time was when the task was completed and ready to preview for both tests.

- **Test system:** For a real-life like setting while still controlling the factors in a laboratory-like manner, the subjects completed the tests on their own systems. The control was maintained by having the tasks pre-defined and using the same system for both tests. This should allow for the chosen system to not influence the changes by the independent variable. A bonus was that the IT artefact was tested to be working on multiple different systems as well.

- **Test objective:** The test subjects were given the final task preview image to know what the task should look like when completed. This was done to avoid confusion and remove the need for any extra time or questions used on uncontrolled factors.

Experimental Mortality, Reactivity, and Experimenter Effects

Experimental mortality will be considered when collecting and processing the data if it occurs. Reactivity and experimenter effects will hopefully be minimized by the subject's experience with participation in research projects and wish not to alter their normal behaviour. It could also be beneficial that the subjects hold a higher position than the researcher who is a student, such that they will be critical towards the work done instead of wanting to help the researcher look good.

External validity

External validity for this research project could mean that the results are also valid for other tasks that can be created, or even other assessment systems than Inspera Assessment that support tasks on the QTI format (see section 2.5). To ensure external validity, the experiment will be conducted on subjects that are typical users of the system, and like those found elsewhere [36, p. 133]. It might be a weakness that the experiment relies on volunteering participants as it may not be generalizable to all the potential end-users. Too few participants are also of high risk for this experiment as the pool is small, and the potential subjects have minimal incentive to want to participate. A representative test case was to be used to ensure that it is a typical test that could have been used in a real-life situation.

Quasi- or field experiments

As explained in the previous paragraphs, the experiment was conducted as accurately as possible. Quasi- and field experiments would provide insight into the real-life settings, but the results would not be conclusive as there would be many factors that could have been an influence. An uncontrolled trial was also avoided by making sure to measure both before and after introducing the new IT artefact [36, p. 134]. The chosen true experiment design was "one group, pre-test, and post-test." This was done by having the participants complete a task in Inspera Assessment while their performance was recorded and measured. Afterward, the same was done with the new IT artefact. By comparing the results before and after, the researcher was able to draw conclusions regarding the effect of the IT artefact. Usually, the researcher cannot know whether the participants have been affected by the passing of time, but this was mitigated almost entirely by having the tests completed in immediate succession.

3.4 Data Generation Methods

This section describes the data generation methods chosen for the research methods as described in section 3.2 and section 3.3 above.

3.4.1 Data Generation Methods for Design and Creation

For the design and creation research method, the data generation methods chosen were mainly interviews, but also documents and observations. The main document containing relevant data for the design and creation of the proposed IT artefact for this thesis was the feedback and data recorded by Jørgensen and Kvannli [15].

The other document was the task description provided for the thesis as it provided some context on what should be developed. See section 1.2 for a complete overview of the task description. Some interviews were also done with the master thesis supervisor because he has a lot of domain knowledge and experience.

For the first part of the design and creation process, the observations made by the researcher of the current digital programming exam process, the testing of the IT artefact by Jørgensen and Kvannli, and testing of the Inspera Assessment interface provided useful data.

All the data collected, as explained in the above paragraphs, were used to gather the initial requirements for the design and creation process for the proposed IT artefact. The initial requirements can be seen in section 4.1.1 table 4.1. It was not placed any more importance on doing further interviews or gathering more initial data as an agile development process was selected, as can be seen in section 3.6. This was done to be able to refine the requirements and adapt the design and creation method during the project after testing and receiving feedback.

3.4.2 Data Generation Methods for Experiment

The main goal of the experiment research method chosen for the answering of *RQ2* is to test the hypothesis defined in section 3.3.1. This section describes the data generation methods used to research if the developed IT artefact increased the level of usability and made the process of creating programming tasks in Inspera Assessment more effective.

Usability is according to ISO [38] defined as *"The extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use"*. The specified goals were tested separately because effectiveness was a goal in itself beside usability and because it is difficult to represent all the goals as a single value [39]. The usability dimensions were measured in the following ways as defined in the ISO 9241 standard [38]:

- **Effectiveness:** Tasks completed, objectives achieved, errors in a task, tasks with errors, task error intensity
- **Efficiency:** Task time, time efficiency, cost-effectiveness, productive time ratio, unnecessary actions, fatigue
- **Satisfaction:** Overall satisfaction, satisfaction with features, discretionary usage, feature utilisation, proportions of users complaining, proportions of users complaining about a particular feature, user trust, user pleasure, physical comfort

The number of test persons that the researcher wanted to test on was around five as [40] Nielsen and Landauer mathematical model shows that will uncover 80% of all usability

problems. That aligns well with the goal of the research question (*RQ2*) that the experiment strategy aimed to answer. The data needed to evaluate the research question and measure the dimensions of usability was gathered with the data generation methods detailed in the following sections.

To be able to do statistical analysis and determine statistically significant improvements, one would need a much larger sample size [41]. This was not the primary goal of the thesis, as it was mainly conducted to prove that a concept (the IT artefact) could be beneficial by being more effective and have a higher level of usability. Gathering a larger group of relevant test participants was also difficult because the potential end-users are a limited number of people. There are not that many people among the general population that have to create programming tasks regularly. Additionally, the restrictions put in place because of COVID-19 [37] made it impossible to meet physically. Lastly, because of the change in the teaching situation, the professors and lecturers were also to busy to participate.

Observation

The observation data generation method was conducted in an overt research [36, p. 203]. The subjects were aware of the observations being conducted, which could lead to them behave unnaturally (known as the *Hawthorn Effect* [36, p. 204]). However, advantages include the requirement of consent and ethical research. Additionally, it let the researcher ask questions to gather specific knowledge, data, and feedback regarding the usage of the IT artefact to be able to answer the research questions (see section 1.4).

To better evaluate the effectiveness and efficiency, a systematic observation was planned to be able to gather quantitative data [36, p. 205]. The created schedule to observe consisted of the following metrics (the complete observation table can be seen in appendix A.1):

- time to completion
- number of actions
- number of errors
- number of questions asked or assistance required

To be able to compare the created IT artefact with the Inspera Assessment interface, the observations were noted for both parts of the experiment strategy, as explained in section 3.3.1. When many people are observed, the effect is the same as conducting a survey [36, p. 208]. The sampling size was determined as described in the introduction of this section (see section 3.4.2). A set time frame of about one hour was used to avoid any effect of learning between the tests.

Because of the COVID-19 pandemic [37], the observations had to be done using digital means as human contact had to be kept at a minimum. This meant that some factors, e.g.,

behavior, and emotions, were more challenging to capture. To get a scenario as close as possible to one in a laboratory, the experiment was conducted with video-conferencing tools like Skype or Zoom. These tools provided the opportunity for the participants to share their screen and the session to be recorded for later analyzing and data-gathering purposes.

Sampling was determined to provide the most real-life like results if the participants were experienced realistically potential end-users of the system. That means professors, lecturers, or others tasked with creating programming tasks on the QTI format as described in section 3.2.1. Since this is a minimal subset of the general population, a referral sampling method [42] was used to reach the suitable candidates. Most of the participants were recruited through recommendation by the thesis supervisor because he is a potential end-user and has many colleagues that fall into the same category.

As the sample candidates are considered well-versed in the field of creating programming tasks, the verbal protocol-analysis [36, p. 213] also known as "think aloud" or "talk about what they are doing" methods were encouraged for all the experiment participants. Thinking aloud made it easier to understand the reasoning behind actions taken and the thought process of the test subjects to analyze the results better. It also allowed the researcher to gain insight into the user's expectations, misinterpretations, and eventual misunderstandings. All the results of the observations are gathered and evaluated in section 3.5 below.

Interviews

Interviews were chosen as an additional data generation method to be conducted alongside the observations and questionnaires. It was mostly considered an addition to the questionnaires to gather more detailed information. The interview allowed for more complex and open-ended questions to facilitate a discussion about topics or nuances that could not be conveyed through the conducted questionnaires.

Trough the interviews, it was possible to explore the participant's emotions and feelings regarding the usage of the tested systems. This was the main objective of the interview, along with the intent to gather feedback on whether or not the potential end-users found the created IT artefact valuable. It was also useful to be able to collect detailed descriptions of what was good, bad and could be improved regarding the task creation process as well as what could be more explicit in the graphical user interface, GUI. The interview was also planned to be able to gather information on what additional requirements the subjects would like or need in order to be able to utilize a new system like the proposed IT artefact most effectively.

The interview was planned and conducted as a semi-structured interview in a one-to-one situation [36, p. 188]. In the table 3.3 below the questions to be asked can be seen. They were only considered guiding questions to be able to cover the wanted topics, but the interview was not controlled. If the participant asked questions or strayed off from the original question, the conversation continued to allow all ideas to be explored. If this

exploration leads to one of the other questions being answered, this was, of course, not repeated. It enabled the subjects to raise new views, angels, and perspective the researcher had not considered and would be lost with a set line of questioning.

Nr	Question
1	What are your thoughts and opinions regarding the new system?
2	What was positive, and what was negative about creating the task in Inspera?
3	What was positive, and what was negative about creating the task in the new system?
4	What are the benefits and drawbacks of creating the task in Inspera compared to creating it in the new system?
5	What effect did the new system have on your productivity with regards to creating the task?
6	What impressions did you get from the user interface in the new system?
7	What functionalities are required or need to be improved in order for the new system to be usable by you or others that are responsible for creating tasks?
8	What functionalities do you wish the new system had?
9	Do you have any suggestions that could make the system even better and more effective?
10	Do you have any other thoughts, suggestions, or opinions you want to share in regards to the new system?

Table 3.3: Interview Questions

As explained in the observation section 3.4.2 above, the experiment had to be conducted digitally. This meant using video-conferencing tools like Skype or Zoom. It is important to conduct the interviews somewhere the interviewees feel comfortable [36, p. 190], and this was difficult to control. The interviewees were mostly working and participating in the experiment from their own homes. On the one hand, this might be more comfortable than an office setting. On the other hand, it could also be more stressful if they, for example, have a family or kids at home that could be a disturbing factor. An advantage of the remote experiment might be that a video-tool is already required, making the recording process more straightforward and also perhaps not as noticeable and distracting for the interviewee. It could also be negatively affecting the whole experiment because it feels unnatural to participate and communicate only through a screen with a camera.

After each interview, the video recording was used to transcribe it. This was a time-consuming process, but essential to better capture and systematize the recorded responses and discussion [36, p. 193]. The participants were offered the transcript of their interview to be able to correct statements and check the intent. All data gathered and how it was used can be seen in section 3.5.2, and the results they lead to can be read about in section 4.2.2.

Questionnaires

Self-administered questionnaires were created and conducted as part of the experiment research strategy to gather information and standardized data [36, pp. 219-220]. To avoid the risks of creating an untested questionnaire with regards to uncertainty, effect, and how to evaluate it was decided to use a tried and tested method that fits the strategy and research questions (see section 3.2 and 1.4).

The system usability scale, SUS, is a 10 points scale that aims to give a single score for the subjective assessment of a given systems usability [43]. For each point, the respondent has to mark on a 5-options scale, where one is the least and five is the most, to what degree they agree with the given statement, just like a Likert scale [43][36, p. 223]. It is well-tested and often used to compare the usability of two systems [44]. The SUS provides a template for a questionnaire. For this research, the template what modified slightly to make it clearer for the participants what part of the system they were evaluating. All the modified statements or questions for the SUS questionnaire regarding the tests can be seen below in table 3.4.

SUS Template	**SUS for Inspera**	**SUS for IT artefact (*this system*)**
I think that I would like to use this system frequently.	I think that I would like to use Inspera frequently to create Inline-Gap-Match tasks.	I think that I would like to use this website frequently.
I found the system unnecessarily complex.	I found Inspera unnecessarily complex.	I found this website unnecessarily complex.
I thought the system was easy to use.	I thought Inspera was easy to use.	I thought this website was easy to use.
I think that I would need the support of a technical person to be able to use this system.	I think that I would need assistance to be able to use Inspera.	I think that I would need assistance to be able to use this website.
I found the various functions in this system were well integrated.	I found the various functions in Inspera were well integrated.	I found the various functions in this website were well integrated.
I thought there was too much inconsistency in this system.	I thought there was too much inconsistency in Inspera.	I thought there was too much inconsistency in this website.
I would imagine that most people would learn to use this system very quickly.	I would imagine that most people would learn to use Inspera very quickly.	I would imagine that most people would learn to use this website very quickly.
I found the system very cumbersome to use.	I found Inspera very cumbersome or awkward to use.	I found this website very cumbersome or awkward to use.
I felt very confident using the system.	I felt very confident using Inspera.	I felt very confident using this website.
I needed to learn a lot of things before I could get going with this system.	I needed to learn a lot of things before I could get going with Inspera.	I needed to learn a lot of things before I could get going with this website.

Table 3.4: SUS Questionnaire

SUS score is a number within the range of 0 to 100. To calculate the score, the contributions for each statement or question is as follows [43]:

- The odd-numbered questions contribute their scale position minus 1
- The even-numbered questions contribute their scale position subtracted from 5.
- Multiple the contribution totals by 2.5

3.5 Evaluation

This section describes how the data gathered in the previously described processes were collected, analyzed, and evaluated. It does so in regards to the different research strategies that were chosen to explore each of the research questions (see section 3.2, 3.3 and 1.4).

3.5.1 Evaluation of Data for Research Question 1

As described in section 3.4.1 regarding the data generation methods for the design and creation research strategy, the data was not overly analyzed initially. The main evaluation of the IT artefact pertains to *RQ2* and is, therefore, more deeply discussed and evaluated in the following section. However, some data was analyzed at the project beginning to develop the initial requirements and during the development phase to alter, improve, and re-iterate on the requirements and design choices. This process is described in section 4.1.

3.5.2 Evaluation of Data for Research Question 2

In this section, the data relating to *RQ2* is explained in terms of how it was used to analyze and evaluate the IT artefact. As explained in section 3.4.2 about the data generation method for the experiment, the researcher was interested in measuring effectiveness, efficiency, and satisfaction.

Effectiveness was measured by the number of actions needed, numbers of errors made, and the number of questions asked or assistance required. Efficiency was measured by time used to complete the task creation and comparing the time used between the two different systems. The satisfaction was measured mainly with the SUS questionnaire, but additional data regarding the participant's feelings of satisfaction were gathered through the interview. The evaluation done was split into the two categories of research data, qualitative and quantitative.

Quantitative Data

The quantitative data is the numbers based data that was gathered during the research process [36, p. 245]. The following data points were recorded for both tests to be analyzed and compared in section 4.2.1

Time to completion

The time to completion was controlled to be measured only on the comparable parts of the task creation process in both systems, Inspera and the IT artefact. This data is ratio data because there is a true zero, the starting time of 0 minutes, and 0 seconds [36, p. 248]. The intervals and ratio between the data points for different participants can, therefore, be compared in multiple ways by using mathematical operators.

Participants were not informed that they were timed on task completion. They were, however, encouraged to think out loud and ask questions if they got stuck. This sometimes lead to participants taking longer pauses to explain their thought process or even provide feedback. To avoid uneven pauses to influence the results greatly the researcher encouraged to subjects to complete the tasks in an as normal way as possible and save the discussion until after the task was completed. One could argue that such pauses, questions or explanation times would be equal for both tests, but that assumes that there are just as many pauses or questions for each system. That is unlikely to be accurate, and it is probably more likely that an extended test time would mean exponentially more time used on questions, clarifications, and feedback. For that reason, the participants were encouraged not to use more time than needed on these points that could potentially influence the total time.

Number of actions

This data point is the number of actions required to complete the given task in the tests. An action was defined as a mouse-click or selection, navigation or selecting keyboard input, or text entry. One instance of text entry was counted for the entirety of that input field or session, not once for every character. While 0 actions might be impossible to obtain, it is the definite starting point for any way of counting the actions, and therefore, this data is also classified as ratio data [36, p. 248].

Number of errors

The numbers of errors were divided into two categories and counted. 0 is the true zero, and there can never be negative amounts of errors. The relative scale and consistent intervals of e.g., four and five errors are comparable to five and six errors; the difference is one error in both cases. Therefore the number of errors was also considered as ratio data [36]. Nuances and varying degrees of severity for each error were not captured as quantitative data, but it is something to consider and was recorded as qualitative data. One error might be more important or damaging than another and could potentially influence the process more than three small errors would. For the comparison between the tests, only the numbers were considered as data, and any variation was ignored.

- **User Errors:** User errors are the errors or mistakes caused by the user. A mistake

could, for example, be not inserting a correct alternative for a gap, placing a gap in the wrong location, or placing a new gap instead of a distractor.

- **System Errors:** System errors are errors or mistakes caused by the system. These errors are commonly referred to as bugs for software systems. A bug might be a failure, crash, unwanted behavior, or wrong results and outputs. It was counted as a system error if the user reasonably expected the system to behave in a certain way, and it did not. E.g., keeping the formatting of pasted content.

Number of Questions Asked or Assistance Required

The number of questions asked or assistance required is comparable to the measure of numbers of errors in that it has the same true zero, and the assistance or questions can be of varying levels. It is therefore also ratio data [36, p. 248]. Questions were clarified to the participant only to be asked if stuck. Therefore assistance was only given it the subject was on the wrong track for a long time without themselves asking for assistance. The only exception to this rule was on the infrequent occasions that either of the systems experienced a random and intermediate bug that confused the subject.

If a test participant asked leading and confirming questions like "this was what the test plan wanted me to do, right?" the researcher either let the participant carry on without answering or confirmed if the participant waited for an answer. If it was confirmed, it was not counted as a question asked as it pertained more to the understanding of the test than the usage of the test system.

System Usability Scale

System usability was measured using the System Usability Scale, SUS, as described in section 3.4.2. The single overall score to rate and compare usability between the two systems in each test is a "quick and dirty" [43] method. It is, therefore, important to consider that a score from one user of e.g., 80 might not mean any actual difference in how usable a system is to other users with a SUS score of 70.

Qualitative Data

Most of the qualitative data in this research project come from the observations during the test and the following interview. The test and interview were recorded and later transcribed to textual data [36, p. 267]. As the context is clearly defined, the contents of the transcript were divided into only two main themes; relevant segments and irrelevant segments, where the relevant segments have some relevance to the research questions for this thesis.

The data from the transcripts marked as relevant were then categorized. In appendix C the

full transcripts can be seen as well as the categories in section C.1. Similar occurrences within the same category were counted and placed in a table to numerate it like quantitative data. This was done to be able to weight the importance of different parts of the qualitative data. Since the interview and observations related to the usability and effectiveness of the different systems, they were used as a complement to the system usability scale to deeper explore the individual experiences and differences regarding these factors. The process of analyzing the textual data to show what it says was concerning the usability, and the research questions, are described in section 4.2.2.

In addition to transcribing the video into textual data, the tape was examined by the researcher. The researcher made a note of any unexpected or noteworthy events. This provided insight into data points that could not be captured by the transcript because no words were spoken about it, or no actions was taken that could be added to the transcriptions. An example of this is users looking for a button or actionable element in an expected location, but not finding it, or finding it elsewhere. Occurrences like these provided the researcher with knowledge about user behavior and how to implement improvements to the IT artefact. The drawback being that this is a highly subjective data collection method as what is interesting or noteworthy, can differ a lot between different people.

As described in the previous section (see section 3.5.2), the events or actions taken during the test observation were recorded and counted. A higher-level description of actions taken during the test was also noted and recorded together with the textual transcriptions as qualitative data. The qualitative data bank then consisted of a textual transcription of the entire test with action descriptions alongside it. To analyze the qualitative data, all irrelevant parts were stripped, and the remaining relevant parts separated into units. The units were decided by the researcher so that each unit belonged to one quantified action of the test or one interview question answered. Then all the units were analyzed for themes and patterns using an inductive approach [36, p. 269] and open coding [36, p. 275].

3.6 System Development Methodology

To develop the IT artefact that should answer *RQ1*, a system development methodology was chosen. This section discusses the different system development methods that were considered, which were chosen, why they were chosen, and how they were used.

3.6.1 Agile Development

A simple process model like the Waterfall Model [45] was considered for the system development methodology. It requires one phase of the project to be fully completed before one can move on to the next, just like the steps in a waterfall. A potential drawback is that it requires everything to be planned out ahead, and it is not designed to be changed at later stages. Testing is also placed at the end of the process, which makes it difficult to know if

everything works as excepted or receive feedback along the way. To be able to test early and adapt to changing requirements, an agile development process [46] was chosen as the systems development methodology.

Agile development is an iterative approach to software development. In contrast to the Waterfall Model with an agile process, the work is done in small increments, building piece by piece, testing along the way, and allowing for changes to be made. It is often a process consisting of planning, implementation, and testing, which is repeated until a satisfactory product is completed. There are many software development methods within the category of agile development; the following subsections describe the ones evaluated for this thesis, which were chosen and how they were used.

3.6.2 Scrum

Scrum is probably the most known agile development method and is based around fixed-length iterations called sprints [47]. Each sprint is usually around 1-3 weeks in duration and has a pre-planned list of tasks that need to be completed. These tasks are evaluated and fitted into the sprint, depending on their estimations. There are many techniques used for estimation, and Scrum itself does not demand anyone particular. Examples of ways to estimate include time it will take to complete, story points, or planning poker.

Considering Scrum is a team-based agile development method with meetings and quite some planning required, it was not regarded as an optimal fit for this project. There was only one team member, the researcher himself, and, therefore, not a natural team or group to discuss sprint plans and retrospectives with. Additionally, the requirements were difficult to estimate in both importance directly and especially time used as both the task, domain and technologies were new for the researcher. The lack of experience would make planning the scrum-based workflow difficult for the researcher, and the required overhead could take away from valuable time spent developing, prototyping, and testing.

3.6.3 Extreme Programming

Extreme Programming is another agile development method that focuses heavily on customer satisfaction [48]. A key component of Extreme Programming is pair programming and team cooperation. This was impossible to utilize in this project as the only developer was the researcher. The method is also supposed to be used in close collaboration with the customer, which for this project are the potential end-users. Since the "customers" were not available enough for this level of commitment throughout the project, the researcher had to make most decisions on his own, making Extreme Programming an unfit development method for this thesis.

3.6.4 Kanban

Kanban is another framework for implementing agile software development. The main principle of Kanban is to match the amount of work the team currently has, with the team capacity. Simply put, it is a sort of queue-system, where one cannot start a new task before the previous task is complete. This gives room for flexible planning and transparency throughout the team and process. Another main benefit of Kanban is that it allows the team to start working with almost no overhead, as it requires less planning, time estimation, and meetings compared to methods like Scrum [49]. In table 3.5 below, some of the properties of Kanban and Scrum are compared. Kanban also tries to limit the number of roles within the team, and if possible, there should be none, which is beneficial for this thesis, where the researcher is the only developer.

	SCRUM	**KANBAN**
Cadence	Regular fixed length sprints (e.g 2 weeks)	Continuous flow
Release methodology	At the end of each sprint if approved by the product owner	Continuous delivery or at the teams discretion
Roles	Product owner, scrum master, development team	No roles if possible
Key metrics	Velocity	Cycle time
Change philosophy	Teams should strive to not make changes to the sprint forecast during the sprint. Doing so compromises learning around estimation.	Change can happen at any time

Table 3.5: Scrum Attributes Compared to Kanban [50]

The core feature of Kanban is the Kanban-board. This is a tool for visualizing the remaining tasks and workflow of the project. A limit is placed on the number of tasks that can currently be worked on at the same time to mitigate tasks being started on but taking a long time to be fully completed. There is no planned or estimation done on time required with Kanban. Tasks are usually ranked by their importance, and the most crucial task is to be picked and completed before the work on the next task can be initialized. This would allow the researcher to reduce time spent on planning and estimating task efforts.

3.6.5 Scrumban

Scrumban is, like the name suggests, a combination of Scrum and Kanban [51]. It takes parts of each method and combines them into one development method. Specifically, it takes the structure of Scrum, and the flow-based methods from Kanban, and combines them into Scrumban. It has iterations and intervals, like sprints in Scrum, and tasks are prioritizes based on the importance of each task. Sprints are based on how much the team thinks they can accomplish in the set amount of time, like Scrum, but WIP-limits can

be added to limit how many tasks can be worked on simultaneously. It also integrates continuous workflow, attempting to do more just-in-time planning, rather than planning everything at the beginning of the sprint. This mix and match of two well-known development methods seemed to provide the researcher with the most well-suited tool. Some of the reasons were because it would remove the need to plan and figure out correct estimates before each sprint, but still allowing for some definition of workload into set periods with some iteration cycles.

3.6.6 Development and Iteration Structure

With Scrumban chosen as the development method as described in the previous section, the development structure was planned. The process from idea to finished product or in this case, a working prototype, was structured as follows:

1. Identify initial requirements
2. Continuously develop a prototype using the chosen development method and iteration structure
3. Test with end-users
4. Release final product

As described in section 3.4.1, the initial requirements (see table 4.1) were gathered from the previous work done by Jørgensen and Kvannli [15], the thesis supervisor and the researcher himself. This was done to have a list of wanted features to work towards from the start. It was not considered a complete and refined list of requirements. With an agile development process, this allowed for more or changing requirements to be explored along the way when experience by trial was gained.

The iteration structure was implemented to be able to split up the required work on the IT artefact into smaller parts. It made it easier to work towards smaller or shorter goals and keep to a schedule. For every iteration, a goal was set, and the requirements needed to reach that goal were set as milestones for that iteration. After every iteration, the requirements were tested. Any small error, bug or missing functionality pertaining to the implemented requirements were fixed during testing, and if they would require more effort noted as a new task and put into the next iteration. Every iteration and their outcome can be seen in section 4.1. The iteration structure was as follows:

1. Define overall iteration goal
2. Select requirements to meet the goal
3. Break requirements down to development tasks

4. Implement functionality in prototype
5. Test functionality
6. Evaluate prototype, fix small bugs or add to tasks for next iteration
7. Repeat

Chapter 4

Results

This chapter presents the results of the research described in the previous chapter (see chapter 3). First, the results from the design and creation strategy are explained with regards to the iterations, design, and technology choices that led to the IT artefact. Then the results from the experiment research strategy are detailed in regards to their qualitative or quantitative data.

4.1 Design and Creation

The design and creation strategy was chosen to answer *RQ1* by creating an IT artefact that can support question authors in making program completion tasks on the QTI format. In this section, the results gathered from this strategy are presented. Every iteration of the development process that was chosen (see section 3.6) is outlined in the following subsections. In addition, every design, technology, and other decision made in regards to the IT artefact is explained. Before the iteration process began, the gathered information had to be broken down and structured into an iteration plan. Following the description of this planning, the iterations were structured in the following way:

- **Goal:** The overall goal of the current iteration as well as a list of requirements to be implemented to reach the iteration goal.
- **Implementation:** How the requirements, tasks, and features were implemented to reach the iteration goal
- **Evaluation:** The testing and evaluation of the implemented requirements, tasks and features for the iteration

4.1.1 Iteration Planning

Before the iterations began, there was a need to break down all the information gathered into an actual application to be developed. This was done by creating a list of initial requirements that would then be implemented throughout the iteration process, as described in the following sections. A general overview of the system architecture was also created to better envision the cooperation of the different parts that would make out the IT artefact. The selection of actual technologies was made during the iterations where these choices needed to be done. In figure 4.1 below is an initial draft of a proposed system architecture for the IT artefact.

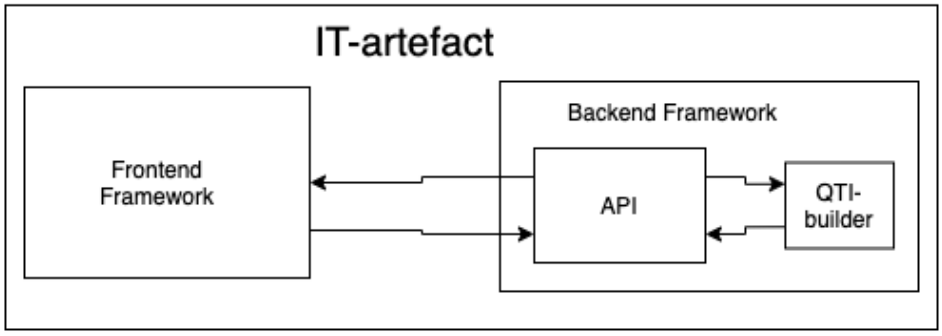


Figure 4.1: Initial System Architecture

As described in section 3.4.1, some different data was collected to be able to provide some initial requirements for the development of the IT artefact. The initial requirements can be seen below in table 4.1.

Number	Requirement Description
R01	The IT artefact should provide the user with a graphical user interface
R02	The IT artefact should support code/text pasting
R03	The IT artefact should let the user create *gaps* from selected parts of the code/text
R04	The IT artefact should let the user add distractors
R05	The IT artefact should be able to generate a *Inline Gap Match* task QTI-file with the users text, selected gaps and distractors
R06	The IT artefact should be able to generate a ZIP-file with the packaged QTI-file and a manifest-file to upload to Inspera
R07	The IT artefact should let the user download the generated ZIP-file

Table 4.1: Initial Requirements

Each iteration had a set duration of two weeks, making the total development time, excluding the iteration planning, eight weeks. The iterations were planned and carried out, as described in the following sections.

4.1.2 Iteration 1

Goal

The overall goal for this iteration was to choose a frontend framework and create effective GUI for pasting of text or code. Requirements relating to this goal, and therefore relevant to this iteration can be seen in figure 4.2 below. This goal was set as the first because the researcher knew that the QTI generation of a task created in another system or program was possible, from the research done by Jørgensen and Kvannli [15] and research on the QTI format by the researcher himself. Therefore the most important thing to first explore was whether or not it would be possible to create an interface better suited than Inspera's own process for creating *Inline Gap Match* tasks. The goals of the first and second iteration were set to reach a MVP, minimum viable product, of a GUI that could give the researcher an indication on whether or not the IT artefact potentially could be valuable compared to just using Inspera itself.

Number	Requirement Description
R01	The IT artefact should provide the user with a graphical user interface
R02	The IT artefact should support code/text pasting

Table 4.2: Relevant Requirements for Iteration 1

These requirements and overall goals were broken into tasks that needed to be done as can be seen below in table 4.3:

Number	Task Description
T01	Research and choose appropriate frontend framework
T02	Create initial and extendable design sketches
T03	Build application skeleton using design sketch
T04	Create code/text input field
T05	Add code/text formatting and/or syntax highlighting

Table 4.3: Tasks for Iteration 1

Implementation

Firstly, an initial design layout was created, as can be seen below in figure 4.2. The layout was created with it being expendable as a key feature because this was one of the goals of

the IT artefact as explained in section 1.3. With a task-selection bar to keep track of tasks that are created and navigate between them, it should be able to handle multiple tasks of different types, if the artefact should support that in the future. The top tab-selection bar would let the content be split up into more tabs if necessary, either for more functionality or just better usage and viewing. It could minimize the amount of scrolling needed to use the needed functions by the user.

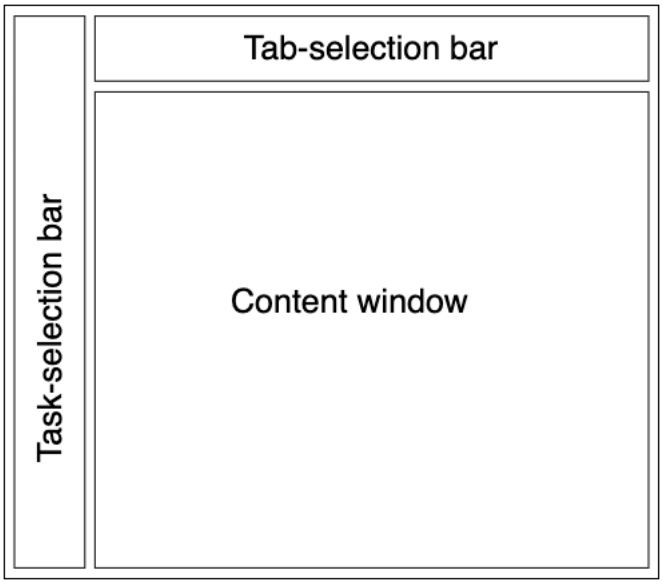


Figure 4.2: Initial Design Layout

As the scope of this thesis and the IT artefact was to be able to be changed and expanded in the future, it was important to choose and develop using a modular frontend framework. The content window could then easily be exchanged with different modules to create different task types. As a modular framework, the researcher chose the component-based modern and widely used web framework called React [52]. There are many other modular frameworks to chose between, but React was picked because the researcher had some previous experience with using it, because it is well-documented, and because it has access to the most extensive collection of premade components. This made it so that the researcher did not have to build every basic component from scratch like the simple buttons, text-fields, navigation bars, and modals. Instead, time could be spent piecing these basic components into higher-order components with domain-specific functionalities that could be reused throughout the application.

Most web development frameworks are based on JavaScript, as it is the standard web programming language [53]. React is no exception, but it also has support for TypeScript, which is a typed superset of JavaScript that compiles into plain JavaScript [54]. Being typed means that the type of every variable is explicitly declared, in contrast to plain

JavaScript where any variable can be of any type. This does take a little longer to write and handle at every definition and usage of a variable. However, a big advantage is that it is a lot simpler to know what type of variables a particular component deep in a component tree expects because it is typed. This makes the code easier to read for the developer, and especially if other developers than the researcher are to be continuing the development further down the line.

Code/text pasting was a highly wanted feature from Jørgensen and Kvannli's gathered research and requirements [15]. Programmers are accustomed to having good IDEs when working with code with support for syntax highlighting making the code more human-readable. Therefore this was a priority for this application as well. To add a code field with support for syntax highlighting, the researcher found a React TypeScript converted version of CodeMirror [55] that he was able to implement. CodeMirror is probably the most used text editor for the browser with over 600 000 downloads every week on npmjs, the world's largest software registry [56].

Evaluation

At the end of the first iteration, all the tasks were completed in a minimal way. The researcher was satisfied with the layout, and the components were structured in a way that made them easy to read, understand, and build upon. In other words, ready for the next iteration and further functionalities.

4.1.3 Iteration 2

Goal

To further develop towards the goal of a MVP GUI as explained in the goal section of the first iteration (see section 4.1.2) the goal of the second iteration was set as follows; Make it easy to select gaps and save the users task input in sensible data-formats to be sent to the backend.

Number	Requirement Description
R03	The IT artefact should let the user create "gaps" from selected parts of the code/text
R04	The IT artefact should let the user add distractors

Table 4.4: Relevant Requirements for Iteration 2

These requirements and overall goals were broken into tasks that needed to be done as can be seen below in table 4.5.

Number	Task Description
T06	Research and choose between different methods of gap selection
T07	Create design sketch for the gap selection interface
T08	Build gap selection interface
T09	Make the user able to select gaps in their code or text
T10	Make the user able to create distractors
T11	Structure the data in a suitable and extendable format

Table 4.5: Tasks for Iteration 2

Implementation

The researcher did a lot of brainstorming and iteration around how to solve the task of selecting the wanted gaps. Many different options were though possible, e.g., click on the text and enter in a popup-window what should become a gap or drag-and-drop a resizable gap over the wanted gap area. The selected method had to preferably be both more usable and effective than the flow in Inspera. Therefore the researcher landed on a way of choosing gaps by selecting the wanted gap text with the browser supported text marking feature and clicking a button to make this text the gap with the selected text as the correct answer. Although it did not seem very intuitive to select text or code in this way to create gaps, it was very fast and seemed easy once one understood the workflow. To explain the not so intuitive action, the researcher added a tool-tip explaining how to perform the gap selection.

After the way of selecting gaps was decided, the researcher created a draft sketch to visualize what it could look like. The sketch can be seen below in figure 4.3. It was divided into two main sections; the leftmost section contained the text or code, one container for each line with the line number, and a button to "send" the selected text over into the other main section. The rightmost main section contained the selected gaps.

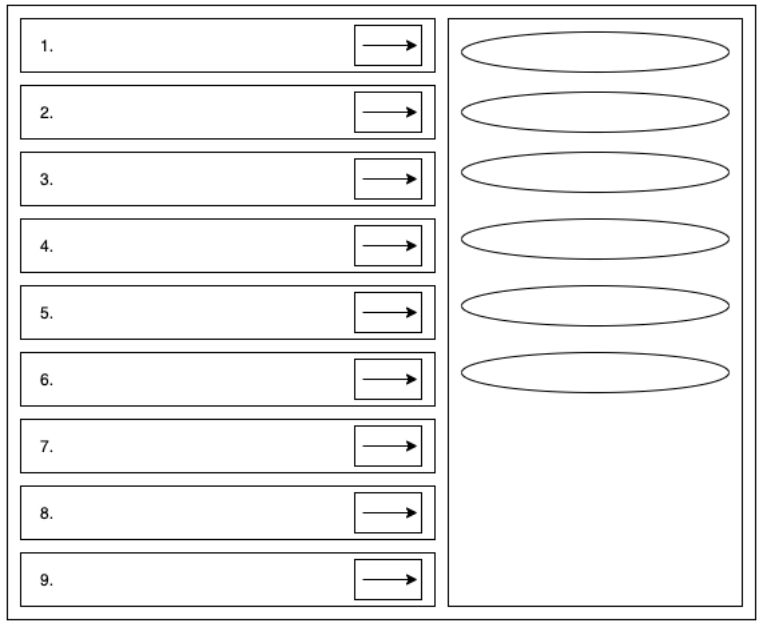


Figure 4.3: Initial Gap Selection Sketch

While prototyping the design of the gap selection, the researcher found the need for line numbers in the view obsolete, at least in the current version as it is unlikely that anyone would create a very long *Inline Gap Match* task. He also discovered the possibility to let the interface look more like how the completed task would look in Inspera, almost like a preview. Since the components were created modular, it was only a matter of switching the components around to make the view layout a little bit different and similar to the finished task look for the student's view when doing an exam. The new and finished layout of the gap selection interface can be seen below in figure 4.4. The arrow icon was also changed to a crop-icon to portray better that the selected text would be cut out. As mentioned in the first paragraph, a helper text was also added even though it was not a part of the initial design sketch for the gap selection interface.

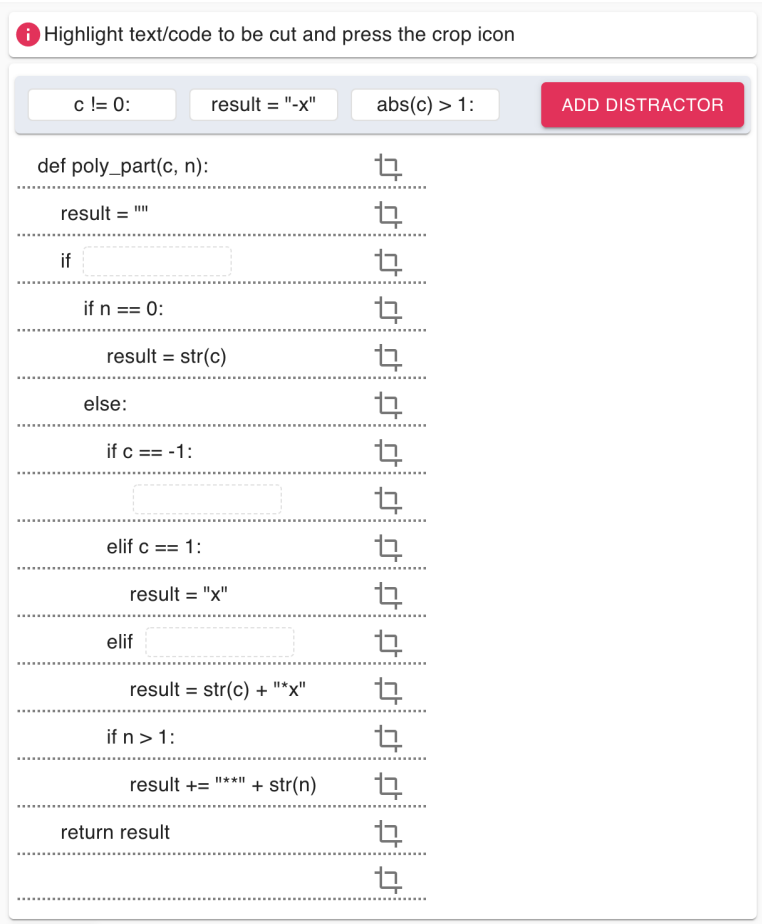


Figure 4.4: Gap Selection Interface

When it comes to the second part of this iteration goal regarding the data-format, the researcher went through some different options. The researcher had to take into account that the data should be passed on to a backend for processing and generation of the task on the QTI format. To utilize the features of the selected TypeScript language as described in section 4.1.2, the researcher chose to structure the data in an object defined by an interface. The *Task*-object definition can be seen below in figure 4.5. As one can see, the types of variables that make up the *Task* object are clearly defined, making it easy to read and understand for the researcher and other developers. Additionally, the object can then easily be changed or added upon because the usage is explicit in any component that uses the *task* object. For example, one could want to add a *taskType* field if the program should later support multiple task types.

```
export interface Task {
  label: string;
  code: string;
  splitCode: ReactNodeArray[];
  distractors: string[];
  language: string;
}
```

Figure 4.5: Task Object Interface

The *Task*-objects created are saved in an array containing all the tasks that are added or changed by the user. This array resides in the top-level parent component (App.tsx), which allows all child components that might need to know about the task or change it able to import and manipulate it. This is done by manipulating the App state by using React (useState) hooks [57].

```
const [tasks, setTasks] = React.useState<Task[]>([]);
```

A helper function was created to use the *setTasks* method to either change, add to or delete *Task* objects from the *tasks* array. This function was then passed down to child components that needed to be able to change the *tasks* array where they could be used. Making the function available in the child components only requires them to define the function as a *prop*, which means a property for the component. The function can then be used by the child component without having to write i again by calling the function name and passing it the correct (with correct type, thanks to TypeScript) parameters.

Evaluation

The only deviation from the plan for this iteration was the change in the actual interface layout for the gap selection process. At the end of the iteration, the first prototype of the GUI was also complete, and the researcher could evaluate the usability and effectiveness subjectively to be better than Insperas. This was of course, to be tested further in relation to the answering of *RQ2* (see section 4.2). With the chosen data structure, it was easy to extend the object when the researcher had to add the distractor's functionality. As the next iteration was planned to produce a MVP, the researcher decided to wait until a more completed artefact was created to receive feedback from the supervisor. The next iteration would mostly focus on creating the backend anyways. To summarize, the iteration was considered a success by the researcher, but he awaited feedback from the supervisor to better determine if the implementation was satisfactory.

4.1.4 Iteration 3

Goal

As an overall goal for the third iteration, it was set as a milestone to create a MVP of the IT artefact. To do this, the following things had to be implemented; select backend framework, make a connection between frontend and backend and build a QTI-file from the *Task*-object sent from the frontend. The related requirements for these goals can be seen below in table 4.6.

Number	Requirement Description
R05	The IT artefact should be able to generate a *Inline Gap Match* task QTI-file with the users text, selected gaps and distractors

Table 4.6: Relevant Requirements for Iteration 3

These requirements and overall goals were broken into tasks that needed to be done as can be seen below in table 4.7.

Number	Task Description
T12	Research and choose appropriate backend framework
T13	Create system architecture plan
T14	Define API structure
T15	Create endpoint for receiving task from frontend
T16	Parse object and build QTI-file with the task object

Table 4.7: Tasks for Iteration 3

Implementation

Contrary to the frontend, the framework for the backend was not as important. This was because the focus of the research was on the usability and effectiveness of the interface. It was, however, somewhat related to the goal of keeping the complete IT artefact modifiable and easily extendable. The researcher, therefore, wanted to pick a well-known and highly used backend framework as well as create an API following the REST architectural constraints [58]. The reasoning being that REST is increasingly becoming the standard for web APIs [59], and the chosen frontend framework, React, comes with support for REST and JSON [60] out of the box. To fit these requirements, the developer chose to use Spring Boot [61].

The default programming language for Spring Boot is Java, but the researcher wanted to try Kotlin [62]. It can make the code more readable by explicitly stating the parameter types in the functions parameter field. Additionally it removes the needs for *getters* and *setters*. These two features and more make it easier to extend the code by others in the same

way TypeScript does for the frontend framework. Other than that, it is also becoming increasingly popular and will be supported for a long time as Google made the largest mobile operating system, Android, "Kotlin-first" [63].

To establish a connection between the frontend and backend, the data was sent as JSON over the HTTP-protocol using the RESTful [58] methods. On the backend side, the server was set up to run a controller with API-endpoints that expect and handle a request body gathered from the requests sent from the frontend. The parsing or handling of the request body was further handled by a connected *TaskBuilder* class that builds the QTI-file. From the frontend a request was made with the following code;

```
fetch("/api/task", {
      method: "POST",
      headers: {
        "Content-Type": "application/json"
      },
      body: JSON.stringify(tasks[0])
    }
```

This method uses the POST-method to call the server controller listening on the */api/tasks* endpoint. The method or request body is the task object created in the frontend by the user parsed as JSON. When the backend receives this data, it maps the values to its own representation of the *Task*-object, which was then passed to the *TaskBuilder* class. With a QTI-skeleton, the *TaskBuilder* builds the required parts of the QTI-file and inserts them into the appropriate places in the skeleton. Finally, the controller takes the QTI-file generated by the *TaskBuilder* and returns it to the frontend as a response. The flow of and between the applications with a focus on the backend can be seen below in figure 4.6. It shows the system architecture that was created for this iteration with task T13.

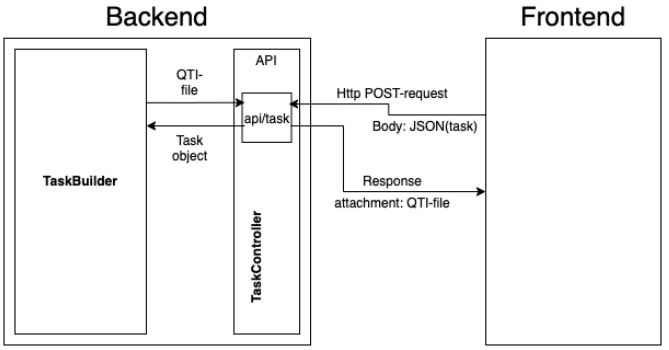


Figure 4.6: System Architecture

With this separation of concerns, the API can easily be extended in the future. The Spring

Boot framework allows for a data layer that can be configured as a database link. In this layer, the type of database can be changed easily, and all required queries are handled by the framework as long as the model is properly defined. One could imagine this being used in a later version of the application to provide a task or question bank trough the API at e.g., */api/taskBank/**. For more details on what could be possible future directions for the IT artefact see chapter 7. Another benefit is the ease of extending the API with the controller. A new task type could utilize a new endpoint e.g., */api/newTaskType*, and call many of the already created functions in the *TaskBuilder* class to create the QTI-file for that task type, which saves time and effort for the developers.

Evaluation

After the completion of the third iteration, the researcher did a walk-trough and evaluation with the supervisor of the IT artefact. The supervisor was pleased with the current interface and features, concluding that it was ready for a user-test if the researcher wanted to. He had some points that could be improved upon and some small bugs that would need to be fixed before the testing. With all the tasks completed, the IT artefact was a MVP that let the user create a *Inline Gap Match* task and export it as a QTI-file. It proved that the workflow was possible, and the user tests would show if it were also considered more usable and effective than Inspera (see the results of the user tests in section 4.2). The researcher was pleased with the working application and started more heavily testing and noting down bugs that would need to be fixed before the user tests.

4.1.5 Iteration 4

Goal

As the overall goal for the fourth iteration, the IT artefact should become a polished application ready for use and research with the experiment strategy. To achieve a working application protoype, the sub-goals were set to complete backend functionality with the export of ZIP-file, make the frontend handle ZIP-file download, and evaluate or test if the prototype is ready for use and research purposes. The requirements that are relevant to these goals and iteration can be seen in table 4.8 below.

Number	Requirement Description
R06	The IT artefact should be able to generate a ZIP-file with the packaged QTI-file and a manifest-file to upload to Inspera
R07	The IT artefact should let the user download the generated ZIP-file

Table 4.8: Relevant Requirements for Iteration 4

These requirements and overall goals were broken into tasks that needed to be done, as can be seen below in table 4.9.

Number	Task Description
T17	Create manifest builder
T18	Create function to package and compress manifest and QTI-file into a ZIP-file bundle
T19	Change response to handle and transfer ZIP-files
T20	Make frontend receive and handle ZIP-file response
T21	Test and bug-fix every part of the artefact in preparation for user testing

Table 4.9: Tasks for Iteration 4

Additionally, some bugs had emerged while the researcher showed the artefact to the supervisor as well as throughout the development process. All the operation critical bugs would have to be fixed before the user tests, and some smaller bugs were also essential to repair to avoid unwanted user actions and program responses. The list of bugs was changed and added upon during the iteration as well, as things were finalized and thoroughly tested by the researcher. However, the initial list at the start of the iteration was as follows in table 4.10 below.

Number	Bug Description
B01	Long gap line gets wrapped
B02	Task cannot be exported if not split on the first line
B03	Text is removed from other lines than the one selected if it matches
B04	Marking direction makes a difference when it should not
B05	It is not possible to make a gap of only one character
B06	A broken task gets exported if no task is created
B07	The QTI identifier is wrong

Table 4.10: Bugs for Iteration 4

Implementation

The implementations for this iteration was focused on polishing all the features added in the previous iteration. To finalize the artefact the researcher also had to add the ability to generate and download ZIP-files. This was because, for the use case of the experiment, the user would have to upload the QTI-file with Inspera manifest packages as a ZIP-file. To achieve this, the backend was extended to create an Inspera manifest file, take both the manifest and the generated QTI-file, and compress them together to a ZIP-file. The ZIP-file was then sent as a response in the same way the QTI-file was after the previous iteration, a HttpServletResponse.

For the frontend to be able to handle the ZIP-file response correctly in all browsers and file systems, there was a need to implement a file-saver dependency. This file-saver library [64] uses the browser or system default method of saving files from the browser. It could

be automatically downloaded into the user's download folder, or they could be prompted to select a download location depending on their browser choice, version, and settings.

As a test of how modifiable and extendable the system was a critical change had to be handled during this last iteration. Without notice, Inspera changed the QTI version from 2.1 to 2.2, which made the artefact misbehave and create the wrong output when imported to Inspera. This had to be fixed before the experiment could be conducted. With the separation of frontend and backend, only the backend needed to be changed to fix this problem as the data required from the frontend did not need any changing. With the experience gained from researching the QTI standards (see section 2.5) and working with creating the QTI-file in the backend, as well as every part of the artifact being built with modifiability in mind the researcher was able to adapt to the new standard in just two days. Many parts of the QTI definition received breaking changes between the versions, but the researcher found that for the *Inline Gap Match* task, the version update broke the text and line formatting. The QTI builder class structure had to be adapted for the parts that handled the visible text and gaps placement. Previously every new line was wrapped in a paragraph *p*-tag, but with the version change this tag was ignored, and explicit newlines were used instead. The most time-consuming part of the change was reading up on the changes and figuring out which parts of the IT artefact were affected, when that was done, the actual code changes only took a few hours to implement and test.

Evaluation

Considering the unexpected and potentially hugely impacting change in the supported QTI version by Inspera, the researcher was very pleased with the results of the final iteration. The product was completed, and any bugs that were predefined or discovered underway were fixed in time for the experiment. To see the results of the experiment see section 4.2.1 and section 4.2.2. These results provide insight into an evaluation by the researcher supported by the experiment done with real potential end-users of the created IT artefact.

Screenshots of the artefact after the completion of the fourth iteration can be seen below in figure 4.7 and figure 4.8. They represent the main views of the application. In the first view (figure 4.7), the interface that is visible to the user after creating a new task with the *ADD TASK* button is shown. In this interface, the user can write or paste their content into the black input area. Different formatting options are available from the drop-down menu where the default option is *Python* code formatting, as can be seen in the example where a Python code snippet is pasted.

In the second view (figure 4.8), which the user can see after navigating using the *STEP TWO* top bar or *NEXT* button, the gap selection process is presented. Here the user can select text on a single line using the system selection method with the mouse. After some text is selected, it can be transformed into a gap by pressing the crop icon. This process takes the selected text and moves it into the top card, where all gaps are displayed together with the *ADD DISTRACTOR* button. Where the text used to be an empty gap will replace it to indicate that it will become a drop-able area in the completed task.

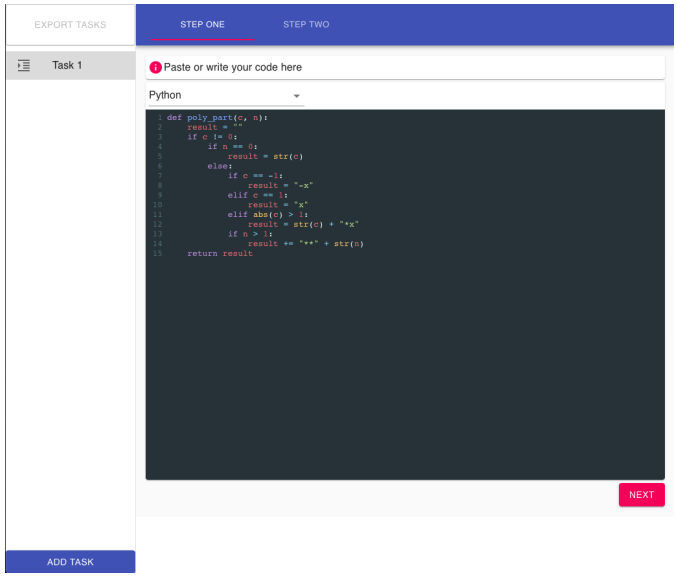


Figure 4.7: IT Artefact Main View 1

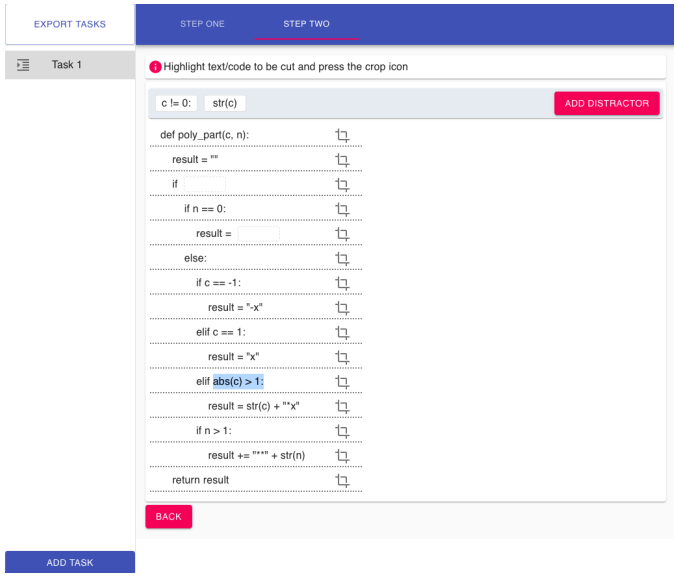


Figure 4.8: IT Artefact Main View 2

4.2 Experiment

In this section, the results from the experiment research strategy chosen to answer *RQ2* will be presented in the two categories quantitative and qualitative data.

4.2.1 Quantitative Data

With this section, an overview of the quantitative data gathered using the experiment research strategy will be detailed. As explained in section 3.4.2, the data generation was done by performing user-tests with real potential end-user of the created IT artefact, having them answer two SUS questionnaires and an interview. The observations were done during the user-tests, and the questionnaires provided the data that will be presented in this section as the quantitative data, while the interview is presented in the following section as qualitative data.

In the table 4.11 and table 4.12 below the raw quantitative data that was gathered during the observation of the experiment can be seen. The experiment was conducted with six participants that completed the test with both Inspera and the IT artefact. As explained in section 3.4.2, which test was done first for each participant was split so that half started with Inspera, and the other half started with the IT artefact. However, the results were gathered together in one table for each system, and the order of completion is not included in the results. The recorded data points are the quantitative points discussed in section 3.5.2.

There were a few special occurrences that are not visible in the results tables but could potentially have impacted the results. Firstly participant 1 had to make the task again with Inspera as a user error lead to a system error that was unnoticed until the task was almost completed the first time. The same thing happened for participant 6, but with the IT artefact and the error was discovered at an earlier time. Both incidents were recorded as user- and system errors in the tables below. The time was not altered but kept on being recorded until the task was redone and completed. However, the data is only presented as results in this section and will be discussed in chapter 5.

Participant	Time to Completion (seconds)	Actions	User Errors	System Errors	Questions Asked or Assistance Needed
1	1167	137	2	3	0
2	562	88	1	0	9
3	1080	167	2	1	3
4	422	100	0	0	1
5	705	99	1	0	3
6	627	164	2	1	0

Table 4.11: Data Points for Task Completed in Inspera

Participant	Time to Completion (seconds)	Actions	User Errors	System Errors	Questions Asked or Assistance Needed
1	405	37	0	0	0
2	350	33	0	0	2
3	482	45	0	0	0
4	369	38	0	0	0
5	631	39	0	0	2
6	652	43	1	1	0

Table 4.12: Data Points for Task Completed in IT artefact

While the two previous tables show the data gathered during the observation, table 4.13 below shows the computed scores from the SUS questionnaires. Each participant answered a questionnaire related to the usage of the system directly after creating a task with the system. This was done for both the tests and resulted in points on a scale from one to five regarding the system usability. The SUS questionnaire and the method of calculating the scores are previously explained in section 3.4.2. In addition to the scores for both tests for each participant the table shows the computed average and standard deviation for each systems SUS.

Participant Number	SUS Inspera	SUS IT artefact
1	45	92.5
2	7.5	95
3	22.5	85
4	50	70
5	20	82.5
6	32.5	82.5
Average	**29.6**	**84.6**
Standard Deviation	**16.1**	**8.9**

Table 4.13: System Usability Score for the Tested Systems

The Wilcoxon Signed Rank test [65] is a common alternative to the dependent samples t-test [66] and can be used to test the statistical hypothesis. Since the dependent samples t-test assumes that the dependent variable is approximately normally distributed [67], it is not suited for analyzing these results. Only six data points are too few to assume normality or confidently remove any outliers. To provide a valid result with the Wilcoxon Signed Rank test, the data and study must meet three assumptions [68]:

1. The dependent variable should be measured at the ordinal or continuous level

2. The independent variable should consist of two categorical related groups or matched pairs

3. The distribution of the differences between the two related groups need to be symmetrical in shape

For the first assumption, we have a Likert-like scale with the SUS, ratio data from the time to completion, and the number of actions needed. Secondly, the groups are related because the same subject did both tests, and the data is recorded for each. They are measured on two occasions on the same dependent variables. Lastly, for the third point, our distribution is symmetric as there is the same number of participants and, therefore, scores for both the SUS, time to completion, and actions needed. In the following sections, the Wilcoxon Signed Rank test will be applied to the data gathered about *Time To Completion*, *Number of Actions to Complete* and the *System Usability Score*.

Time To Completion

Participant	Time Inspera	Time IT artefact	Difference	Rank	Sign	Signed Rank
6	627	652	25	1	-	-1
4	422	369	53	2	+	+2
5	705	631	74	3	+	+3
2	562	350	212	4	+	+4
3	1080	482	598	5	+	+5
1	1167	405	762	6	+	+6

Table 4.14: Wilcoxon Signed Rank Test for Time To Completion

In table 4.14 above the time used by the participants with both Inspera and the IT artefact can be seen. They are compared by computing the absolute value of the difference between them. Figure 4.9 below uses a graph to illustrate the completion times for both tests for every participant. In the second figure, figure 4.10 the average or mean time is visualized in the same graph type. With the following formulas and calculations, the Wilcoxon Signed Rank test was applied to find the p-value and statistical significance.

We would like to test the following hypotheses, where H_0 is a null hypothesis and H_a is an alternate hypothesis:

- H_0: Time used has the same distribution between Inspera and the IT artefact.
- H_a: Time used is systematically higher for Inspera.

The test statistics is the sum of the ranks of the positive differences. That is the sum of all the positively signed ranks in the rightmost column of table 4.14 above. This value is called the *Wilcoxon signed rank statistics* and the value here is $W^+ = 2 + 3 + 4 + 5 + 6 = 20$.

We can then find the mean of W^+ with formula 4.1:

$$\mu_{W^+} = \frac{n(n+1)}{4} \tag{4.1}$$

Where n is the number of observation pairs, in this case six, so $n = 6$ and we get:

$$\mu_{W^+} = 10.5 \tag{4.2}$$

The standard deviation of W^+ is given by formula 4.3 as follows:

$$\sigma_{W^+} = \sqrt{\frac{n(n+1)(2n+1)}{24}} \tag{4.3}$$

Which we can compute by filling in the same value for n as for the mean, $n = 6$ in equation 4.4:

$$\sigma_{W^+} \approx 4.77 \tag{4.4}$$

We have a *one-tailed* test in one direction because we only care if the change is shorter time used with the IT artefact than with Inspera. Further we can find the Z score measure of the standard deviation with the following formula 4.5:

$$z = \frac{W^+ - \mu_{W^+}}{\sigma_{W^+}} \tag{4.5}$$

Inputting the values found in equation 4.2 and equation 4.4 we get:

$$z = 1.99174119 \tag{4.6}$$

Using a standard normal table [69] or a statistical program we get the p-value of $p = 0.02330$ which means the result is significant at $p < .05$ and the null hypothesis can be rejected.

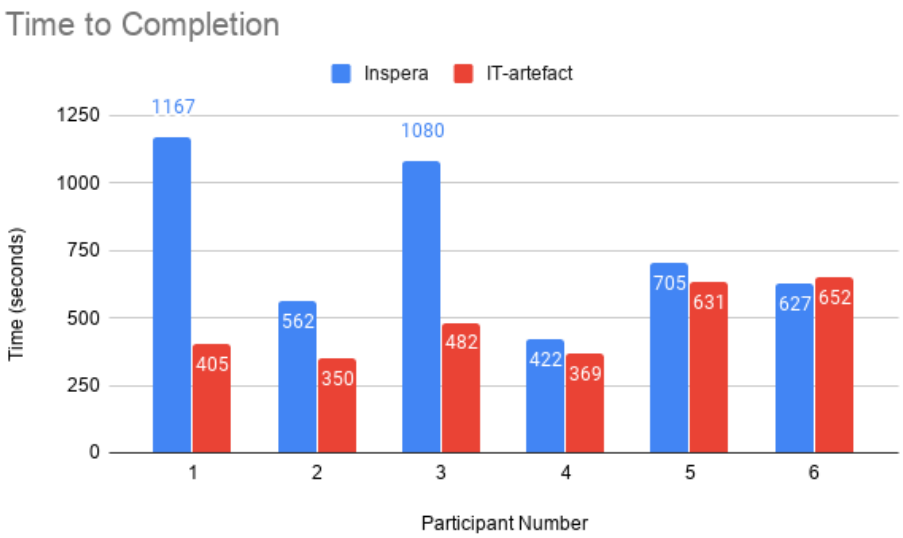


Figure 4.9: Time to Completion with Inspera and IT artefact

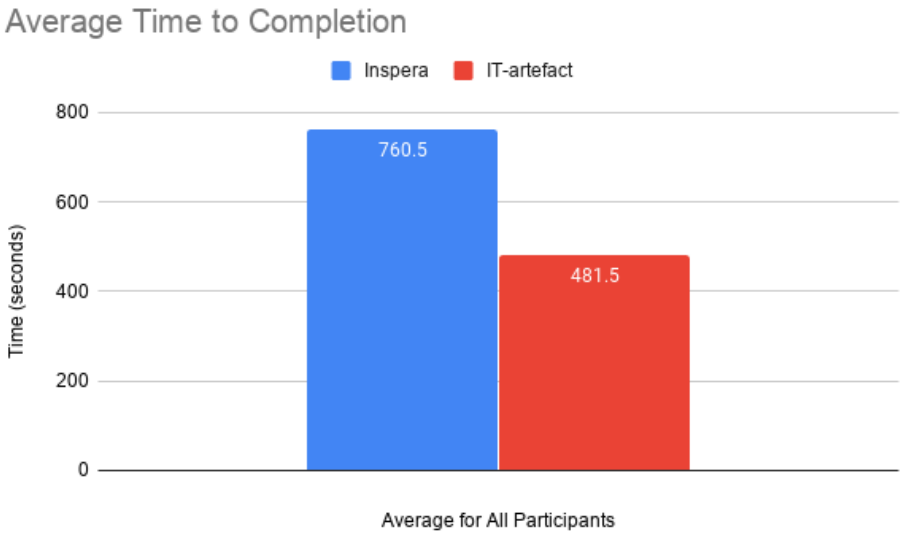


Figure 4.10: Average Time to Completion with Inspera and IT artefact

Number of Actions to Complete

Participant	Actions Inspera	Actions IT artefact	Difference	Rank	Sign	Signed Rank
2	88	33	55	1	+	+1
5	99	39	60	2	+	+2
4	100	38	62	3	+	+3
1	137	37	100	4	+	+4
6	164	43	121	5	+	+5
3	167	45	122	6	+	+6

Table 4.15: Wilcoxon Signed Rank Test for Number of Actions to Complete

In table 4.15 above the number of actions required to complete the tasks by the participants with both Inspera and the IT artefact can be seen. They are compared by computing the absolute value of the difference between them. Figure 4.11 below uses a graph to illustrate the number of actions used for each test for every participant. In the second figure, figure 4.12 the average or mean number of actions is visualized in the same graph type. With the following formulas and calculations, the Wilcoxon Signed Rank test was applied to find the p-value and statistical significance.

We would like to test the following hypotheses, where H_0 is a null hypothesis and H_a is an alternate hypothesis:

- H_0: Number of actions has the same distribution between Inspera and the IT artefact.
- H_a: Number of actions used is systematically higher for Inspera.

Using the same method as for the *Time To Completion* as described above in section 4.2.1 we get the Wilcoxon signed rank statistics value $W^+ = 1 + 2 + 3 + 4 + 5 + 6 = 21$. Because we have six pairs of observations for the number of actions used as well the value for n is the same, $n = 6$. The value for the mean and standard deviation of the ranks will therefore be the same by using the formula 4.1 and formula 4.3, where the results are given by equation 4.2 and equation 4.4 respectively.

We have a *one-tailed* test in one direction because we only care if the change is less actions used with the IT artefact than with Inspera. Further we can find the Z score measure of the standard deviation with the same formula 4.5 as used for the *Time To Completion* in section 4.2.1 by just replacing the different value for W^+. The result is seen in equation 4.7:

$$z = 2.201398157 \tag{4.7}$$

Using a standard normal table [69] or a statistical program we get the p-value of $p = 0.0138539$ which means the result is significant at $p < .05$ and the null hypothesis can be rejected.

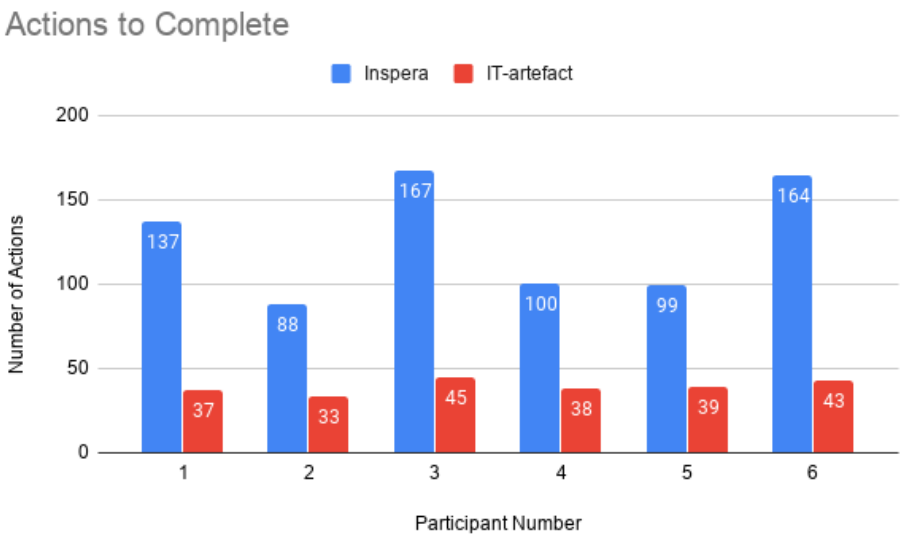


Figure 4.11: Actions to Complete with Inspera and IT artefact

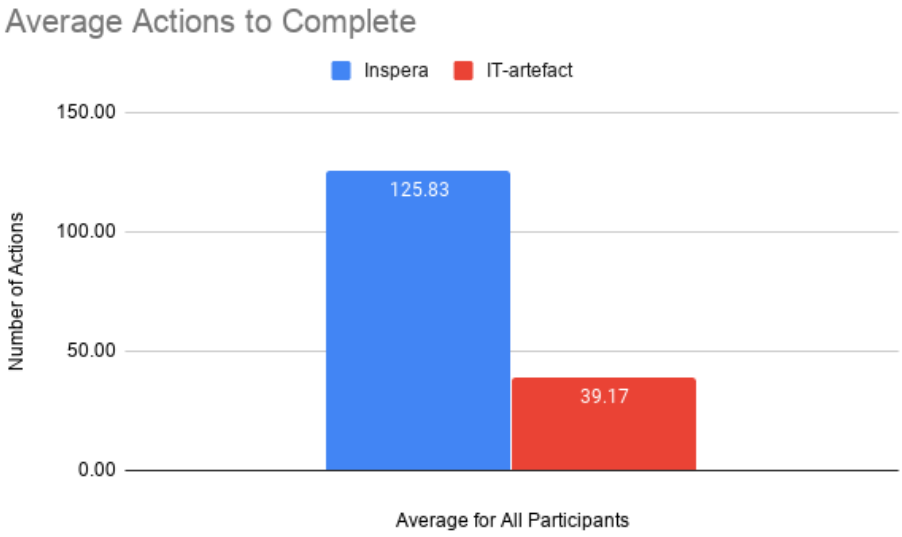


Figure 4.12: Average Actions to Complete with Inspera and IT artefact

System Usability Scale

Participant	SUS Inspera	SUS IT artefact	Difference	Rank	Sign	Signed Rank
4	50	70	20	1	+	+1
1	45	92.5	47.5	2	+	+2
6	32.5	82.5	50	3	+	+3
3	22.5	85	62.5	4	+	+4
5	20	82.5	62.5	5	+	+5
2	7.5	95	87.5	6	+	+6

Table 4.16: Wilcoxon Signed Rank Test for System Usability Scale

In table 4.16 above recorded SUS scores for the participants with both Inspera and the IT artefact can be seen. They are compared by computing the absolute value of the difference between them. Figure 4.13 below uses a graph to illustrate the SUS score for the tests for every participant. In the second figure, figure 4.14, the average or mean SUS score is visualized in the same graph type. With the following formulas and calculations, the Wilcoxon Signed Rank test was applied to find the p-value and statistical significance.

We would like to test the following hypotheses, where H_0 is a null hypothesis and H_a is an alternate hypothesis:

- H_0: SUS Score has the same distribution between Inspera and the IT artefact.

- H_a: SUS Score is systematically higher for IT artefact.

As the Wilcoxon signed rank statistics value, $W^+ = 1 + 2 + 3 + 4 + 5 + 6 = 21$, and the number of observation pairs are the same ($n = 6$) as for the *Number of Actions* in the previous section 4.2.1 the results will be the same. In this case we also only care about the increase in SUS Score between Inspera and the IT artefact, so we have a *one-tailed* test.

The Z score measure is the same, $z = 2.201398157$. Using a standard normal table [69] or a statistical program we get the p-value of $p = 0.0138539$ which means the result is significant at $p < .05$ and the null hypothesis can be rejected.

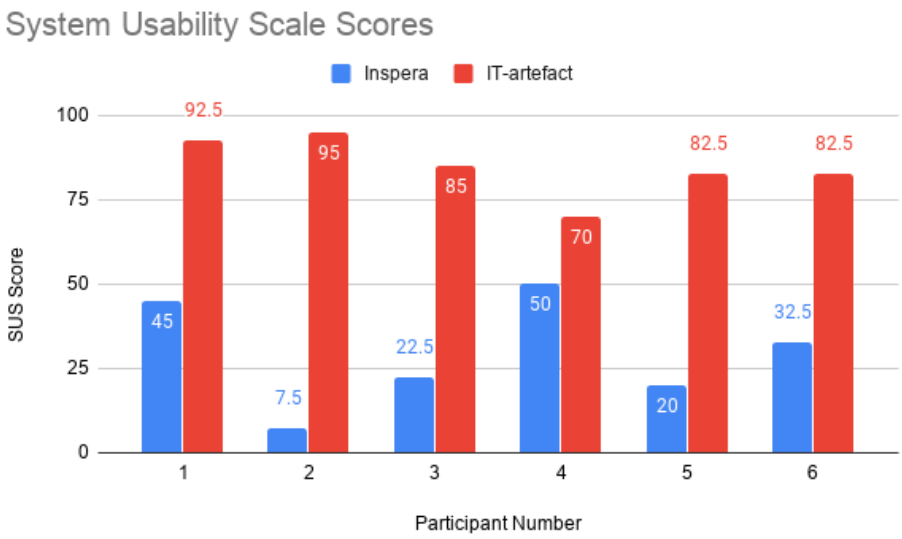


Figure 4.13: SUS Scores for Inspera and IT artefact

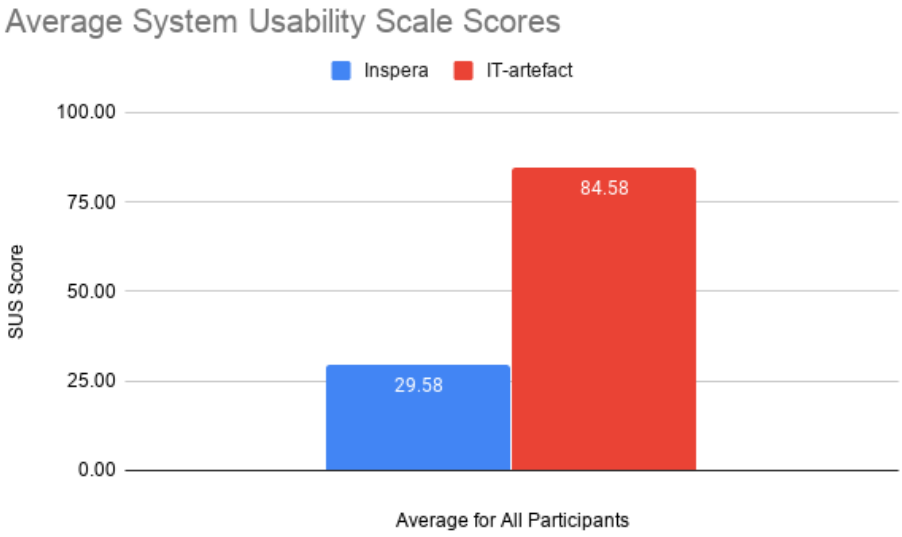


Figure 4.14: Average SUS Scores for Inspera and IT artefact

Although the SUS score is represented on a scale from 0 to 100, it is not the percentage of usability. The average SUS score is 68, so the comparisons and rankings need to consider the scores differently. Using the SUS Score adjective rating and acceptability ranges from figure 4.15 below the scores have the following ranges of acceptability; Every SUS Score for Inspera was below 50, thus placing it in the *NOT ACCEPTABLE* range. On the other hand, the SUS Scores for the IT artefact were 70 or higher, meaning every score is *ACCEPTABLE*. The mean or average SUS scores were 29.6 for Inspera and 84.6 for the IT artefact as seen in table 4.13. For Inspera this means the score of 29.6 is between *WORST IMAGINABLE* and *POOR*, and closer to *WORST IMAGINABLE* than the latter. For the IT artefact the score of 84.6 is between *GOOD* and *EXCELLENT*, and only .4 points away from the *EXCELLENT* marker.

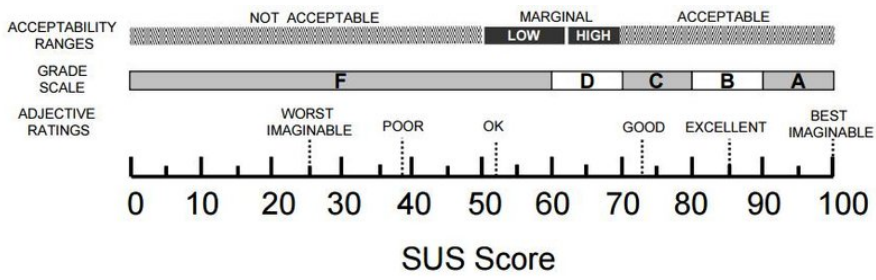


Figure 4.15: SUS Score Adjective Ratings [70]

4.2.2 Qualitative Data

In this section, the qualitative data that was gathered during the experiment research strategy will be explained. The data generation was done by recording feedback during the observations of the tests as well as the concluding interview. When the testing was completed, the recorded footage of all the six participants totaled over seven hours. To better understand, work with and visualize the qualitative data in these recordings, the interactions were transcribed. Every spoken interaction was transcribed, and data about important visual cues were added where they would provide needed context. To be able to reference different parts of the transcript, every line was numbered. The complete transcripts of every participant recorded partaking in the experiment can be found in appendix C.

After the recordings were transcribed, the researcher did some data analysis [36, p. 268] to compress and categorize the transcript. Firstly from every part of the transcript, one or more categories were applied to the participant's recorded words using an inductive approach [36, p. 269]. When deciding the categories, the researcher extracted the parts of the transcript that were relevant to answering the research questions (see section 1.4), in particular *RQ2*, and that could say anything about the usability, effectiveness, efficiency, and satisfaction. Then the categories were analyzed for themes and patterns between them. Similar categories were cut and pasted together. The researcher then decided to split the transcript into two main parts, where part one was everything directly related to Inspera, and the other part contained categories relevant to the IT artefact.

When the data was stripped, categorized, and split into two main parts, the next step was to organize it. To be able to order the categories, a count of their frequency was used. The frequency was of course not a sure way to determine importance or weight as they could be heavily influenced by the line of questioning during the interview, but it does provide some insight into the number of times a particular topic was discussed. The line of questioning might also have naturally lead to there being a clear difference in positive and negative categories. Either way, this was used to split the transcribed and categorized data in the Inspera part further. The positive categories for Inspera can be seen below in figure 4.17 and the negative categories in 4.18. The tables only show the frequency numbers of the subcategories within the overall category, each subcategory that was counted can be seen in the full tables in appendix C.1.

Number	Category	Frequency
01	Everything in one system	5
02	Other good things about Inspera	4
03	Good user interface	1

Table 4.17: Positive Categories for Inspera

Number	Category	Frequency
04	Bad user interface	13
05	Bad workflow	8
06	Bad process for gap creation	7
07	Should handle code formatting or syntax highlighting	3

Table 4.18: Negative Categories for Inspera

The second main part that contained categories relevant to the IT artefact was also further split. As for Inspera, the questioning in the interview lead to there being a distinct separation between positive and negative categories. Additionally, one of the interview questions led to the discovery of some wanted features or improvements that were out of the scope of this thesis. These categories were used to compile a list of wanted features for a potential new version of the IT artefact which can be seen in appendix C.2. The positive categories for the IT artefact can be seen below in figure 4.19 and the negative categories in 4.20. The tables only show the frequency numbers of the subcategories within the overall category, each subcategory that was counted can be seen in the full tables in appendix C.1.

Number	Category	Frequency
08	Good workflow	10
09	Good user interface	8
10	Good process for gap creation	6
11	Sees usefulness of new system	5
12	Good that it includes code formatting and syntax highlighting	3
13	Can be used for more than code or programming tasks	2
14	Other feedback	2

Table 4.19: Positive Categories for IT artefact

Number	Category	Frequency
15	Bad process for gap creation	15
16	Bad user interface	6
17	Need to be able to undo	5
18	Difficult to import to Inspera	4
19	Need to use two systems	4
20	Export button placed wrong or indistinct	4
21	Step one is selected even though it is not the first that should be clicked	3
22	Bad workflow	3
23	Bad security	2
24	Bad helper texts	2
25	Code in step two looks like normal text	2
26	Bad text/code field	2
27	Other buttons badly placed	2
28	Bad error handling	1

Table 4.20: Negative Categories for IT artefact

It is important to note that while the categories are called positive and negative, that does not mean everything was meant as either directly positive or negative. Many of the categories in the negative section were meant as constructive criticism and feedback. One would need to read the entire transcript or listen to the recordings to gain full insight into the participant's positive or negative feelings and the importance placed with each statement.

Chapter 5

Discussion

This chapter contains a discussion of the results gathered throughout the project. First, the process of the design and creation research strategy will be considered. Then the experiment and its results will be reviewed. To examine these processes, they will be conferred about in relation to the research questions defined in section 1.4. Where relevant, the results will be compared and talked about in relation to other similar or relevant research.

5.1 Design and Creation

To answer *RQ1*; *How can one design and create an IT artefact that can support effective authoring of tasks on the QTI format?*, the design and creation method was chosen as explained in section 3.2. The results of this research method were detailed in section 4.1, and this section will discuss those results.

The overall goal of the design and creation strategy was to create an IT artefact that could be used to make tasks on the QTI format. This is a goal that might be beneficial for others that want to develop something similar or use the proposed system as is. To test the IT artefact, the goal had to be specified to accommodate importing QTI files to Inspera explicitly. This was because it was the platform used by the researcher's faculty and, therefore, the platform that would be most practical to perform the testing on. While it was not tested, the goal was for the IT artefact to be able to create valid tasks on the QTI format that can be imported and reused in any system or platform that supports the same QTI version.

While the design and creation research strategy was chosen to answer *RQ1*, *RQ2* also had to be considered during the development process. This was because to make the IT

artefact able to answer *RQ2* the usability, effectiveness, and satisfaction had to be good. To achieve this these measures had to be a part of the plan and results of the design and creation strategy. If the IT artefact was able to attain these measures and answer *RQ2* or not will be discussed in section 5.2 because it was researched using the experiment strategy.

In the first part of the design and creation strategy, the user interface of the proposed IT artefact was planned, designed, and created. To be able to create a task on the QTI format, it was important that the users were presented with an understandable graphical interface that provided an easy way to create a task. Additionally, the sub-goal of making the IT artefact easy to extend and rework, as described in 1.3 was achieved by creating small components that were reused throughout the application. A well defined and extendable *Task* object as shown in figure 4.5 also contributed towards this goal.

For the second half of the design and creation process, the equally important task of creating the QTI file was the top priority. Although this was already proven to be possible by others with, e.g., QTI.JS (see section 2.3.4) and specifically for Inspera by Jørgensen and Kvannli [15] it was necessary functionality. Without it, the IT artefact would not be able to let the users create tasks on the QTI-format and, therefore, not be able to answer *RQ1*. After connecting the frontend to the backend to generate the QTI file, the IT artefact was a MVP that proved its ability to take user input and form a complete task on the QTI-format.

During the last part of the design and creation process, the QTI version was changed by Inspera without notice, as explained in section 4.1.5. Although an upgrade meant changes had to be made, the researcher managed to implement the necessary code alterations. At the same time, the ZIP-file generation was handled and thus completing the first prototype of the IT artefact that should be able to let the user create tasks on the QTI-format.

As long as QTI stays relevant as the current standard for task interoperability [34] the IT artefact could be valuable. However, as explained in section 2.5 Piotrowski [33] has done research on the shortcomings and limitations of QTI as a standard. In particular regarding the "breaking changes" between versions, which did impact this thesis as explained in the above paragraph. This is hard to predict, and should it ever change, the research, development, the IT artefact and results of this thesis could still be relevant for anyone that wants to explore anything related.

5.2 Experiment

This section will discuss the process and results of the experiment research strategy. As the strategy was chosen to answer *RQ2* it will be examined with regards to that. As the experiment strategy gathered both qualitative and quantitative data, this section will talk about the data gathered and what it means in the following sections.

5.2.1 Quantitative Data

The quantitative data was collected using observation and SUS questionnaires as explained in section 3.4.2 and presented as results in section 4.2.1. This section reviews the data in regards to the research question, *RQ2*, and the researcher's opinions regarding the findings.

While the observations gathered data points regarding time to completion, the number of actions, user errors, system errors, and questions asked or assistance needed, the first two will be focused on in the following sections. This is because they are more suited and better used to substantiate any claims regarding the comparison of the results between each system. User errors, system errors, and questions asked or assistance needed could be highly afflicted by chance or the individual performing the test. That said, they could, of course, be said to have an impact on the usability and effectiveness of the system if the user is slowed down or confused by recurring system errors, thus having to ask questions or seek assistance, which could further lead to ineffectiveness and a poor experience regarding usability.

There is a clear difference between the two tested systems in regards to the number of user errors, system errors, and questions asked or assistance needed to be recorded in the result. This can be seen in table 4.11 and 4.12. For the test done using Inspera, the total number of user errors was eight, while for the IT artefact, on the other hand, only one participant made an error. The same participant was also the only one that encountered a system error with the IT artefact, while the sum of system errors for the tests with Inspera was five. Lastly, the total number of questions asked or assistance needed was only four for the IT artefact, while 16 for Inspera. These differences could probably have an impact on the usability and satisfaction of the systems as there is a relationship between user errors and perceived usability [71].

Time to Completion

To be able to examine the effectiveness of each system, the time to completion was measured. This was a measurement of how long it took the participant to finish creating the given task in each system, as explained in section 3.4.2. The researcher wanted the participants to perform the task creation in a natural way as possible. Therefore the participants were not informed that they were timed on their task creation. A possible shortcoming of this method is that the participant felt no need to try and complete the task in a timely manner, taking their time, and discussing things that interested them along the way. This would then not really make it as natural as the researcher had hoped. In some instances, this did happen, so to avoid it the researcher reiterated that comments or discussions were best left for after the test, as he wanted to observe the difference in usage between the systems without distractions. It is impossible for the researcher to determine if this had a noticeable impact or not, and if so, for what system it would have affected the time the most.

Time to completion was measured from when a new task was created in the respective system until the task was preview-able as a complete task in Inspera. This meant that for the IT artefact the participants would finish creating a task, and then spend some more time uploading the QTI file to Inspera to preview it there, thus completing the task and stopping the timer. As observed by the researcher, and noted by many of the participants (see e.g., line 96-98 in the transcript for participant 4 in appendix C.6 or line 47-61 in the transcript for participant 3 in appendix C.5), the process of uploading in Inspera was quite hidden and not well explained by the IT artefact. While the time used to upload the file was not measured explicitly the researcher noted that every participant spent quite some time at this stage of the test and it was even noted by participant 1; (freely translated, see line 19 in the transcript for participant 1 in appendix C.3) "*So what I spent time on was the functionality in Inspera to upload*". This extra, but needed, time to upload the task somewhere would also have less of an impact on the total time if one would create multiple tasks at once and then only upload them all at once at the end.

Despite the influence the above-mentioned factors could have on the time to completion the researcher would argue that they could be called diminishing variables and equally affecting each system. If not possibly more so negatively for the time of the IT artefact because the time here was including the upload to Inspera as detailed in the above paragraph. Looking at the numbers, only one participant used less time with Inspera than the IT artefact, and the difference was only 25 seconds. On average, the difference was 279 seconds more spent on creating the task with Inspera. That is an increase of 58% more time used with Inspera than the IT artefact. On the other hand, looking at the graph in figure 4.9, it seems two participants used significantly more time with Inspera than the others, which could just be unlucky outliers that potentially affect the average more than if the test was performed with a more significant number of participants. With the Wilcoxon Signed Rank test, the null hypothesis that there is no difference in time to completion between the systems was rejected, and the results computed to be statistically significant.

Number of Actions to Complete

Number of actions to complete might be a better measure of effectiveness for this project. Firstly, because four of the participants had previous experience with using Inspera, which could have to lead them to perform quicker in this part of the test. Secondly, because the number of actions required to perform a task will always put a limit on how quickly it is possible to perform a given task. Having to click a button three times instead of one will take longer given that the training and experience are equal, meaning the other factors like finding the button will be eliminated. Lastly, the number of actions required to complete the given tasks is significantly higher with Inspera for every participant.

On average, the participants needed 125 actions to complete the task with Inspera and only 39 to do the same task with the IT artefact. That is more than three times as many actions, and an increase of 220% or 86 more actions. Even the participant that used the least amount of actions to complete the task with Inspera used 50 more actions than with

the IT artefact for a total of 88 actions. Because every difference was positive in favor of the IT artefact the null hypothesis that there was no difference in the number of actions required had to be rejected. As with the *Time to Completion* the results were statistically significant, as shown using the Wilcoxon Signed Rank test in section 4.2.1.

The researcher also did a subjective measurement of the minimum number of actions required to complete the task given in the experiment in each system. Given that the researcher worked extensively with both systems for a long time creating multiple tasks in each, he got the opportunity to deeply explore the possibilities and different imaginable interactions. One could, of course, never be sure that there doesn't exist a more effective way to do the given task, but it could at least provide an interesting perspective. The minimum amount of actions the researcher found needed for Inspera was 66, and the minimum for the IT artefact was only 29. The typical user actions are described in the lists 3.3.1. In table 5.1 and table 5.2 below the researcher assumptions on the number of actions required for each step of the task creation process are listed. Of the 29 actions needed for the IT artefact, seven of them were necessary to import the QTI file in Inspera. The largest difference was the creation of gaps, where each gap required nine actions to be created in Inspera and only two in the IT artefact. This difference would increase drastically if more gaps were made. For the test given in the experiment, only six gaps had to be created, which meant 12 actions in total to create all the gaps with the IT artefact, and on the other hand, it was necessary to perform 54 to do the same in Inspera.

Action	Number of actions
Copy code	2
Paste code	2
Remove placeholder text	2
Create gaps	$9 \bullet 6gaps = 54$
Add distractor	6
SUM	66

Table 5.1: Minimum Number of Actions for Inspera

Action	Number of actions
Copy code	2
Paste code	2
Navigate to next view	1
Create gaps	$2 \bullet 6gaps = 12$
Add distractor	3
Export task	2
Import task	7
SUM	29

Table 5.2: Minimum Number of Actions for Artefact

System Usability Scale

System Usability Scale scoring showed the perceived usability of the participants. The scores can be seen in table 4.13, and the difference in score between Inspera and the IT artefact are calculated in table 4.16. Every participant except one had a significantly higher score for the usability of the IT artefact. The difference for the average score between the systems was 55, which means they are more than half the scale apart from each other.

As none of the participants awarded Inspera with a high score of usability according to the administered SUS questionnaire, it is difficult to argue it could have a higher level of usability than the IT artefact. This was also substantiated by the responses gathered from the interviews, as can be seen in the following section regarding the qualitative data. The null hypothesis that the SUS score was the same for Inspera and the IT artefact was discarded by using the Wilcoxon Signed Rank test, which also showed a statistical significance.

5.2.2 Qualitative Data

Interviews were added as an additional data generation method to be able to deeper explore the thoughts and feelings of the participants regarding the usage of both systems. It was also used to gather feedback on features that were missing, or they would want to be added to the IT artefact in potential future versions of it. The transcripts and categorization of the interviews provide qualitative data along with the observations done during the tests of the experiment. This data might not provide enough objective meaning on their own, but can at least be used to support the quantitative data further as discussed in the previous section 5.2.1.

As shown in table 4.17, table 4.18, table 4.19 and table 4.20 the number of times participants mentioned anything that was recorded as data and categorized varied very much between the systems. The line of questioning was likely the main factor as to why there are a lot more recordings for the IT artefact as many of the questions were worded to gather feedback regarding the usage of the IT artefact (see the questions in table 3.3). This is by design as the researcher wanted suggestions and responses regarding how the IT artefact could be further improved and what it would require for the user to use it regularly. While the researcher has control over the development of the IT artefact, there is very little he could do to influence the way Inspera works, meaning this data is off less importance.

While there are 15 data points from the interviews that were placed into the *bad process for creating gaps* category with the IT artefact the participants were mostly pleased with the way it was done. This is because suggestions for making it even better like ”the line could have been a little less light gray” (translated from the transcript for participant 4, line 52-53) were put in this category although the same participant said this in the following sentences: ”And it is incredibly easy because the explanation is right there. It cannot be simpler than that. So it was actually really easy to create gaps” (translated from the transcript for participant 4, line 53-55). Another example from a different participant was

the inclusion of the desire to be able to revert the creation of a gap (from the transcript for participant 6, line 30-32). The same participant stated this about the gap selection process: ”It was very satisfying, I found this to be very fun (shows the selection and creation of a gap in the IT artefact). I would like to use this to create small quizzes in courses quite regularly” (translated from the transcript for participant 6, line 46-49).

On the other hand, multiple participants noted that the process of creating gaps in Inspera was cumbersome. For example, from the transcript of participant 6 (translated, line 12-13): ”I thought it was very cumbersome in Inspera to click and create (gaps), and that it was not formatted correctly was actually very frustrating”. Participant 1 also noted the following (translated from the transcript for participant 1, line 49-59): ”It was quite a lot more clumsy in Inspera, and another thing was purely the visuals because the gap-boxes became so large”. That same participant also noted something about the number of actions required for the gap creation process (translated from the transcript for participant 1, line 31-35) ”Even for the Gap-Match it is a bit more cumbersome in Inspera because it requires quite a bit more button presses or mouse clicks to be able to do stuff, because you first have to paste, and then you need to mark what you want to remove, cut it out, and then you need to navigate up to the insert button, and then get a gap that you have to click on twice, and then you have to go to the right of the screen to be able to insert the answer”.

Four of the six participants commented on the fact that Inspera required a lot of actions, some specifically for the gap creation process. Two of the participants commented that the process was cumbersome in addition to requiring many actions, and all of the others mentioned negative things about the process required to create a gap. This difference in opinions of the gap creation process between the systems could be related to *RQ2* in that it might be an improvement of the overall usage of the IT artefact compared to the assessment platforms’ own interface, which in this experiment was Inspera. At the very least, one might be able to substantiate the findings regarding the difference in effectiveness between the systems, as discussed in section 5.2.1.

In regards to the usability, some of the data points from the interview support the results of the SUS scores calculated from the questionnaires. Over half the participants noted that they would like to use the IT artefact in its current state and that they would prefer using it over Inspera. This is in line with the quite a lot higher average score using the SUS for the IT artefact than Inspera.

When it comes to satisfaction, four of the six participants specifically said that the IT artefact was fun. Every participant also commented positively on the user interface and the overall system workflow, which could be both an indicator of good usability as well as the user’s satisfaction with the system. Some of the participants commented directly on the usability of the IT artefact compared to Inspera, e.g., (translated from the transcript for participant 2, line 2-5) ”It is dramatically better than what is built into Inspera, there is no doubt about that. It is also a lot simpler to use, and therefore it is no doubt that if I were to make this type of tasks, and that I probably will, then I would definitely prefer to use this system (the IT artefact) over Insperas built-in”. This could substantiate a claim that the IT artefact has better usability than Inspera, and in turn, help answer *RQ2*.

5.2.3 Research Critique

This section discusses some of the points of critique the researcher has for the conducted research strategies and results. As this was the researcher's first major research work, the inexperience might be of critique itself, in addition to making it difficult for the researcher to have insight into the critique-worthy parts of his own work.

While the researcher did manage to find significant results with the number of participants recruited, it could be a weakness. Having a larger sample size would make it possible to have more trust in the results. Another factor includes the limited variation in participants, which was mainly due to the sampling technique used and the fact that it is difficult to get into contact with the relevant group of potential candidates. All the participants were male, belonged to the same institution at NTNU, and had many experiences working with technical systems. On the one hand, this might limit the validity of the results, while on the other hand, one could argue that this is probably a decent sample of what would be the typical user of the systems tested in this thesis.

There was also a limited amount of time to do the testing with each participant. A more extensive test could even out the differences that were due to unforeseen events or bugs with each system. On the other hand, it could also make the differences in experience become more of a factor. Some of the participants had much experience with creating tasks in Inspera already, while some had only a little to no experience. It could be interesting to see the difference between a novice user and an experienced user, but it would require significantly more time for the participants to become experienced. That was just not possible given how busy all the potential end-users were because of the changes due to the COVID-19 [37] pandemic.

Another point for critique is in regards to the conducted interviews and questionnaires. As the participants knew that the researcher had created the IT artefact they could be inclined towards wanting to be positive and nice, and in that way, give feedback differently than what they would have done otherwise. The potential points of weakness and critique also include every countermeasure mentioned in section 3.3.1 e.g, the *controls*, *experimenter effects* and *external validity*.

Chapter 6

Conclusion

As explained in chapter 1, the goal of this thesis and project was to create an IT artefact and improve the task creation process. To be able to evaluate this goal, the research set up two research questions, as listed in section 1.4. The study leads to a literature review, gathering and analyzing requirements, a design process, selection of development process, implementation, prototyping, testing, and the conduction of an experiment. This chapter presents the conclusion of the research done throughout this project and the thesis.

In order to answer the research questions of this thesis, the IT artefact was created, and an experiment was conducted with six participants. The participants were split into two groups, where one group tested the creation of a given task with Inspera first and then with the IT artefact. For the other group, the exact same task was done with the same systems, only they started with making it with the IT artefact first. Both qualitative and quantitative data were collected through the observations, questionnaires, and interviews done for the experiment research strategy. With this data, two overall conclusions were drawn to answer their respective research questions as posed in section 1.4:

RQ1: How can one design and create an IT artefact that can support effective authoring of tasks on the QTI format?

One can design and create an IT artefact that lets a user effectively create tasks on the QTI format by correctly implementing the QTI standard with a user-friendly graphical user interface, which answers *RQ1*. To achieve this it is essential to study the given QTI version specification and beneficial to follow a systems development methodology as well as adhere to a proper coding structure. Following modern industry standards and best practices could also be very helpful. The created prototype of the IT artefact is targeted towards programming or code solving tasks, but it can be used to effectively author *Inline Gap Match* tasks for other purposes as well.

RQ2: What improvements does the IT artefact give compared to using the authoring tool included in Inspera Assessment?

With a system specialized for creating programming tasks on the QTI standard, one gains a higher level of usability, effectiveness, and satisfaction than when using the assessment platforms' own interface. In this thesis, this was proven using the created IT artefact and Inspera, respectively, to create the same task. The time used, the number of actions required, and a SUS score was calculated and compared for both systems, showing the improvements related to the usability, effectiveness and satisfaction. Data gathered from the interviews also support these findings. Additionally, the interviews revealed that the IT artefact is an overall improvement to the question authoring of *Inline Gap Match* fill inn missing code tasks compared to Inspera.

Chapter 7

Future Work

This chapter discusses the things that could be worked with on future versions of the IT artefact and any additional research that could be done. These are things that were not done during this project or were out of the scope. With future research and development of the IT artefact it could become a complete product to improve the question authoring process on the QTI format. Different and new directions for future research within the same domain could build upon this thesis or look at new angles. This could further the knowledge on, e.g., question authoring systems, the QTI standard, and learning outcomes from digital programming tasks.

Regarding the research method that can be improved upon for future studies, the researcher has some suggestions. The first thing to do would be to examine the points mentioned in the research critique (see section 5.2.3) to see if they can be eliminated in follow-up research. This includes research done with many more people on an improved version of the artefact. An improved version of the IT artefact could become a complete product to replace more of the task authoring process, for programming specifically, or for digital tasks in general. One could also consider testing the IT artefact against an assessment system where the user creates their own task instead of following a test plan or script.

Any new features for the IT artefact would prompt new research to be done on the effect and outcomes of the changes done. The following are some suggestions that could be done in the future with a new or improved version of the IT artefact. It could generate other task types. As mentioned in section 1.3 there are task types closely related to the *Inline Gap Match* task type that could easily be implemented with the current IT artefact like *Text Entry* and *Inline Choice*.

If the IT artefact should become a staple tool for the question authors some more quality assurance would be beneficial. When the tasks are used in an exam setting, it is essential

that they function as intended and that there are no errors that could affect the results. More testing in the form of both user tests and system tests could ensure that every task is valid and will not fail once it is live in a critical setting. Another important part to ensure this could be to improve the validation done in the artefact and have it provide user feedback along the way to allow the user to fix the issues during the authoring process.

One of the goals of the IT artefact was to make the question authoring process more effective. This could be further improved by adding more automation features to the artefact. Generate task description, either manually in IT artefact or automatically based on the task type, points system, and other inputs.

Cheating is a problem that was discussed in section 1.1 and it is crucial to consider when changing the task process. The developed IT artefact can help mitigate cheating by making it easier for the authors to create unique tasks. However, this currently manual process could be automated to generate different tasks from the same material to make it harder to cheat by peeking at other student's screens during digital exams. Within the same category, one could also let the artefact generate tasks from a template.

A question bank with good *Inline Gap Match* fill inn missing code tasks could be useful both as examples and inspiration, but also to let users share their work. To enable this, the artefact would need to be able to save to and retrieve tasks from a database. Saving the tasks the user is currently working on could be an important feature either way for a second iteration on the artefact. From the user-test observations and interviews, some requests for potential future features of the artefact were collected, they can be seen in appendix C.2.

Bibliography

[1] J. Bennedsen and M. E. Caspersen, "Assessing process and product: A practical lab exam for an introductory programming course", *Innovation in Teaching and Learning in Information and Computer Sciences*, vol. 6, no. 4, pp. 183–202, 2007.

[2] G. Sindre and A. Vegendla, "E-exams and exam process improvement.", in *NIK*, 2015.

[3] G. Sindre, "What good can digital exams do for constructive alignment?", in *MNT-conference*, Nordic Journal of STEM Education, 2019, pp. 85–90. [Online]. Available: `https://www.ntnu.no/ojs/index.php/njse/article/view/2992/2918` (visited on 01/17/2012).

[4] G. Sindre and A. Vegendla, "E-exams versus paper exams: A comparative analysis of cheating-related security threats and countermeasures", in *Norwegian Information Security Conference (NISK)*, vol. 8, 2015, pp. 34–45.

[5] A. Heintz, "Cheating at digital exams-vulnerabilities and countermeasures", Master's thesis, NTNU, 2017.

[6] V. Lappalainen, A.-J. Lakanen, and H. Hogmander, "Paper-based vs computer-based exams in cs1", in *Proceedings of the 16th Koli calling international conference on computing education research*, ACM, 2016, pp. 172–173.

[7] WISEFlow. (2019). Wiseflow - more than paperless, [Online]. Available: `https://europe.wiseflow.net/` (visited on 09/26/2019).

[8] Inspera. (2019). Sikker og digital tentamen | inspera, [Online]. Available: `http://www.inspera.com/no/` (visited on 09/26/2019).

[9] M. Hillier and A. Fluck, "Transforming exams-how it works for byod e-exams", in *Proceedings ASCILITE2017: 34th International Conference on Innovation, Practice and Research in the Use of Educational Technologies in Tertiary Education*, University of Southern Queensland, 2017, pp. 100–105.

[10] T. M. Sogaard, "Mitigation of cheating threats in digital byod exams", Master's thesis, NTNU, 2016.

[11] P. Dawson, “Five ways to hack and cheat with bring-your-own-device electronic examinations”, *British Journal of Educational Technology*, vol. 47, no. 4, pp. 592–600, 2016.

[12] B. Kuppers and U. Schroeder, “Bring your own device for e-assessment-a review”, in *International Conference on Education and New Learning Technologies, EDULEARN proceedings*, 2016, pp. 8770–8776.

[13] A. Sarkar, “The impact of syntax colouring on program comprehension.”, in *PPIG*, 2015, p. 8.

[14] A. Chirumamilla and G. Sindre, “E-assessment in programming courses: Towards a digital ecosystem supporting diverse needs?”, in *Conference on e-Business, e-Services and e-Society*, Springer, 2019, pp. 585–596.

[15] J. W. H. Jorgensen and S. Kvannli, “Efficient generation of parsons problems for digital programming exams in inspera”, Master’s thesis, NTNU, 2019.

[16] Instructure. (2020). Mer tid til undervisning og læring | Modernde LMS | Canvas, [Online]. Available: `https://www.instructure.com/canvas/nor-no/schools` (visited on 04/05/2020).

[17] Moodle. (2020). Moodle: Online learning with the world’s most popular LMS, [Online]. Available: `https://moodle.com/` (visited on 04/05/2020).

[18] Blackboard. (2020). North america | blackboard.com, [Online]. Available: `https://www.blackboard.com/` (visited on 04/05/2020).

[19] J. Handke and A. M. Schafer, *E-Learning, E-Teaching und E-Assessment in der Hochschullehre: Eine Anleitung*. Walter de Gruyter, 2012.

[20] A. Sangra, D. Vlachopoulos, and N. Cabrera, “Building an inclusive definition of e-learning: An approach to the conceptual framework”, *The International Review of Research in Open and Distributed Learning*, vol. 13, no. 2, pp. 145–159, 2012.

[21] N. A. Buzzetto-More and A. J. Alade, “Best practices in e-assessment”, *Journal of Information Technology Education: Research*, vol. 5, no. 1, pp. 251–269, 2006.

[22] D. Parsons and P. Haden, “Parson’s programming puzzles: A fun and effective learning tool for first programming courses”, in *Proceedings of the 8th Australasian Conference on Computing Education-Volume 52*, 2006, pp. 157–163.

[23] B. J. Ericson, L. E. Margulieux, and J. Rick, “Solving parsons problems versus fixing and writing code”, in *Proceedings of the 17th Koli Calling International Conference on Computing Education Research*, 2017, pp. 20–29.

[24] Oracle. (2020). Java programming language, [Online]. Available: `https://docs.oracle.com/javase/8/docs/technotes/guides/language/index.html` (visited on 04/05/2020).

[25] P. Denny, A. Luxton-Reilly, and B. Simon, “Evaluating a new exam question: Parsons problems”, in *Proceedings of the fourth international workshop on computing education research*, 2008, pp. 113–124.

[26] J. J. Van Merriënboer and M. B. De Croock, “Strategies for computer-based programming instruction: Program completion vs. program generation”, *Journal of Educational Computing Research*, vol. 8, no. 3, pp. 365–394, 1992.

[27] N. Cheng and B. Harrington, “The code mangler: Evaluating coding ability without writing any code”, in *Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education*, 2017, pp. 123–128.

[28] S. Garner, “An exploration of how a technology-facilitated part-complete solution method supports the learning of computer programming.”, *Issues in Informing Science & Information Technology*, vol. 4, 2007.

[29] I. G. L. Consortium. (2012). Ims question & test interoperability (qti) specification overview | ims global learning consortium, [Online]. Available: http://www.imsglobal.org/question/index.html#version2.1 (visited on 03/24/2020).

[30] (2020). Js-parsons - JavaScript library for Parson’s Problems, [Online]. Available: https://js-parsons.github.io/ (visited on 04/05/2020).

[31] B. Mottershead. (2020). Qtihub.org, [Online]. Available: http://qtihub.org/ (visited on 04/05/2020).

[32] Inspera. (2017). Knowledge base - inspera, [Online]. Available: https://inspera.atlassian.net/wiki/spaces/KB/overview (visited on 03/25/2020).

[33] M. Piotrowski, “Qti: A failed e-learning standard?”, in *Handbook of Research on E-Learning Standards and Interoperability: Frameworks and Issues*, IGI Global, 2011, pp. 59–82.

[34] I. G. L. Consortium. (2019). Ims question & test interoperability (qti) specification overview | ims global learning consortium, [Online]. Available: http://www.imsglobal.org/question/index.html (visited on 03/26/2020).

[35] T. S. Consortium. (2020). Safe exam browser - about, [Online]. Available: https://safeexambrowser.org/about_overview_en.html (visited on 04/05/2020).

[36] B. J. Oates, *Researching information systems and computing*. Sage, 2005.

[37] Helsedirektoratet. (2020). Helsedirektoratet stenger alle barnehager og skoler, [Online]. Available: https://www.helsedirektoratet.no/nyheter/helsedirektoratet-stenger-alle-barnehager-og-skoler (visited on 05/05/2020).

[38] ISO. 9241-11:2018(en), “Ergonomics of human-system interaction”, International Organization for Standardization, Standard 11, 2018.

[39] E. Frokjaer, M. Hertzum, and K. Hornbaek, “Measuring usability: Are effectiveness, efficiency, and satisfaction really correlated?”, in *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, 2000, pp. 345–352.

[40] J. Nielsen and T. K. Landauer, “A mathematical model of the finding of usability problems”, in *Proceedings of the INTERACT’93 and CHI’93 conference on Human factors in computing systems*, 1993, pp. 206–213.

[41] J. Cohen, “A power primer.”, *Psychological bulletin*, vol. 112, no. 1, p. 155, 1992.

[42] J. Penrod, D. B. Preston, R. E. Cain, and M. T. Starks, "A discussion of chain referral as a method of sampling hard-to-reach populations", *Journal of Transcultural nursing*, vol. 14, no. 2, pp. 100–107, 2003.

[43] J. Brooke *et al.*, "Sus-a quick and dirty usability scale", *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[44] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale", *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.

[45] tutorialspoint. (2020). Waterfall model, [Online]. Available: `https://www.tutorialspoint.com/sdlc/sdlc_waterfall_model.htm` (visited on 01/31/2020).

[46] D. Wells. (2009). Agile software development: A gentle introduction, [Online]. Available: `http://www.agile-process.org/` (visited on 01/31/2020).

[47] C. Drumond. (2018). Scrum - what it is, how it works, and why its awesome, [Online]. Available: `https://www.atlassian.com/agile/scrum` (visited on 01/31/2020).

[48] D. Wells. (2013). Extreme programming: A gentle introduction, [Online]. Available: `http://www.extremeprogramming.org/` (visited on 01/31/2020).

[49] A. Salazar, "So long scrum, hello kanban", *Stormpath*, 2014. [Online]. Available: `https://stormpath.com/blog/so-long-scrum-hello-kanban` (visited on 01/31/2020).

[50] W. Wallace, *IT Governance: Policies and Procedures, 2020 Edition*. Wolters Kluwer Law & Business, 2019, ISBN: 9781543810998. [Online]. Available: `https://books.google.no/books?id=kG%5C_JDwAAQBAJ`.

[51] R. Lynn. (2020). What is Scrumban, [Online]. Available: `https://leankit.com/learn/agile/what-is-scrumban/` (visited on 01/31/2020).

[52] Facebook. (2020). React - a javascript library for building user interfaces, [Online]. Available: `https://reactjs.org/` (visited on 04/15/2020).

[53] S. H. Jensen, A. Moller, and P. Thiemann, "Type analysis for javascript", in *International Static Analysis Symposium*, Springer, 2009, pp. 238–255.

[54] Microsoft. (2020). Typescript - javascript that scales, [Online]. Available: `https://www.typescriptlang.org/` (visited on 04/15/2020).

[55] CodeMirror. (2020). Codemirror, [Online]. Available: `https://codemirror.net/` (visited on 04/15/2020).

[56] npm. (2020). Npm - build amazing things, [Online]. Available: `https://www.npmjs.com/` (visited on 04/15/2020).

[57] Facebook. (2020). Hooks api reference - react, [Online]. Available: `https://reactjs.org/docs/hooks-reference.html` (visited on 04/15/2020).

[58] R. T. Fielding, "Rest: Architectural styles and the design of network-based software architectures", *Doctoral dissertation, University of California*, 2000.

[59] G. Levin. (2015). The rise of rest api, [Online]. Available: https://blog.restcase.com/the-rise-of-rest-api/ (visited on 04/20/2020).

[60] JSON. (2020). Json, [Online]. Available: https://www.json.org/json-en.html (visited on 04/20/2020).

[61] VMware. (2020). Spring boot, [Online]. Available: https://spring.io/projects/spring-boot (visited on 04/20/2020).

[62] K. Foundation. (2020). Kotlin programming language, [Online]. Available: https://kotlinlang.org/ (visited on 04/20/2020).

[63] D. Ramel, "Google goes kotlin-first for android mobile development", May 7, 2019. [Online]. Available: https://adtmag.com/articles/2019/05/07/kotlin-android.aspx.

[64] E. Grey. (2020). File-saver - npm, [Online]. Available: https://www.npmjs.com/package/file-saver (visited on 04/20/2020).

[65] F. Wilcoxon, "Probability tables for individual comparisons by ranking methods", *Biometrics*, vol. 3, no. 3, pp. 119–122, 1947.

[66] S. Solutions. (2020). How to conduct the wilcoxon sign test - statistics solutions, [Online]. Available: https://www.statisticssolutions.com/how-to-conduct-the-wilcox-sign-test/ (visited on 05/05/2020).

[67] ——, (2020). Paired sample t-test - statistics solutions, [Online]. Available: https://www.statisticssolutions.com/manova-analysis-paired-sample-t-test/ (visited on 05/05/2020).

[68] L. Research. (2018). Wilcoxon signed-rank test using spss statistics, [Online]. Available: https://statistics.laerd.com/spss-tutorials/wilcoxon-signed-rank-test-using-spss-statistics.php (visited on 05/05/2020).

[69] ztableblog. (2020). Z table, [Online]. Available: https://www.ztable.net/ (visited on 05/05/2020).

[70] A. Bangor, P. Kortum, and J. Miller, "Determining what individual sus scores mean: Adding an adjective rating scale", *Journal of usability studies*, vol. 4, no. 3, pp. 114–123, 2009.

[71] A. Oulasvirta, K.-P. Engelbrecht, A. Jameson, and S. Möller, "The relationship between user errors and perceived usability of a spoken dialogue system", *ISCA/DEGA*, 2006.

Appendix A

Experiment Documents

A.1 Observation Schedule

Participant Role	Time to completion	Number of actions	Number of errors	Number of questions or assistance required

Table A.1: Observation Schedule

A.2 Test Plans

A.2.1 Test Plan 1

Takk for at du tar deg tid til å gjennomføre denne testen! Instruksjoner følger i dette dokumentet så les igjennom før du begynner.

Testen består av 5 deler, på hver sin side i dette dokumentet:

1. Lage oppgave i Inspera
2. Svare på spørreskjema om Inspera
3. Lage oppgave i nytt system
4. Svare på spørreskjema om nytt system
5. Kort intervju

DEL 1 - Lage oppgave i Inspera

(Helt samme oppgave skal lages i begge systemer)
Navigering i Inspera er ikke en del av testen. Testleder vil veilede deg til riktig oppgavetype om det trengs.

1. Lag en ny “Inline Gap Match” / “Plasser i tekst” -oppgave
2. Koden du skal lage oppgave av finnes her: https://raw.githubusercontent.com/Chr1stian/masterapp/master/task.py
3. Kodebitene markert med rød firkant skal byttes ut med en “Gap” (hint: **+Insert** eller **+Sett inn** knappen)
4. Det som står inne i firkanten skal så settes som et korrekt svar for det “Gap”et
5. Legg til en distractor: “str(x)”
6. Når alle gaps er laget med tilhørende korrekt svar kan du trykke på øyet for å få en forhåndsvisning av den ferdige oppgaven
7. Den ferdige oppgaven skal se ut som bildet til høyre

```
1   def poly_part(c, n):
2       result = ""
3       if c != 0:
4           if n == 0:
5               result = str(c)
6           else:
7               if c == -1:
8                   result = "-x"
9               elif c == 1:
10                  result = "x"
11              elif abs(c) > 1:
12                  result = str(c) + "*x"
13              if n > 1:
14                  result += "**" + str(n)
15      return result
```

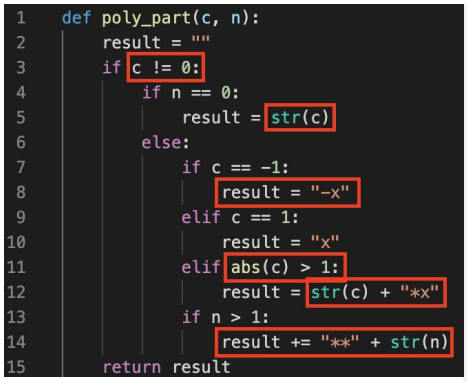
DEL 2 - Spørreskjema om Inspera

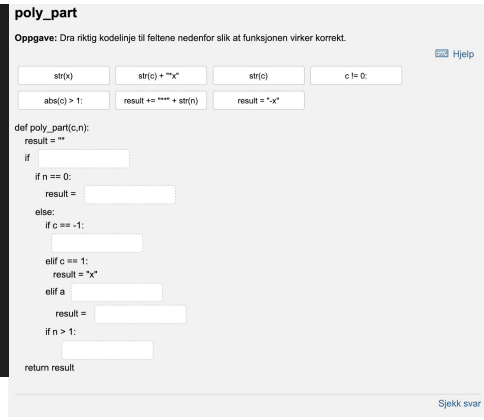
Gjennomfør spørreskjema på linken under når testleder ber deg om dette.
https://forms.gle/pAybb3bEzJWvMUDL7

DEL 3 - Lage oppgave i nytt system

(Helt samme oppgave skal lages i begge systemer)

1. Naviger til det nye systemet på: https://master.preference.no/
2. Koden du skal lage oppgave av finnes her: https://raw.githubusercontent.com/Chr1stian/masterapp/master/task.py
3. Kodebitene markert med rød firkant skal byttes ut med en “Gap” (hint: marker koden og trykk -ikonet)
4. Legg til en distractor: “str(x)”
5. Når alle gaps er laget skal du eksportere oppgaven med “EXPORT TASKS”-knappen og laste opp .zip-filen til Inspera. Følg instruksene i systemet.
6. Den ferdige oppgaven skal se ut som bildet til høyre





DEL 4 - Spørreskjema om nytt system

Gjennomfør spørreskjema på linken under når testleder ber deg om dette.
https://forms.gle/Yo6d1VyrZvzQ5xxPA

DEL 5 - Intervju

Svar på spørsmål, diskuter og kom med innspill om du ønsker. Takk for at du deltok!

A.2.2 Test Plan 2

Takk for at du tar deg tid til å gjennomføre denne testen! Instruksjoner følger i dette dokumentet så les igjennom før du begynner.

Testen består av 5 deler, på hver sin side i dette dokumentet:

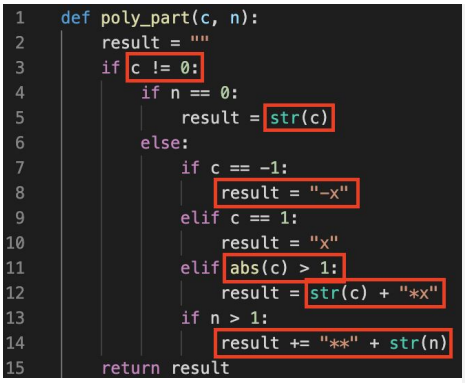
1. Lage oppgave i nytt system
2. Svare på spørreskjema om nytt system
3. Lage oppgave i Inspera
4. Svare på spørreskjema om Inspera
5. Kort intervju

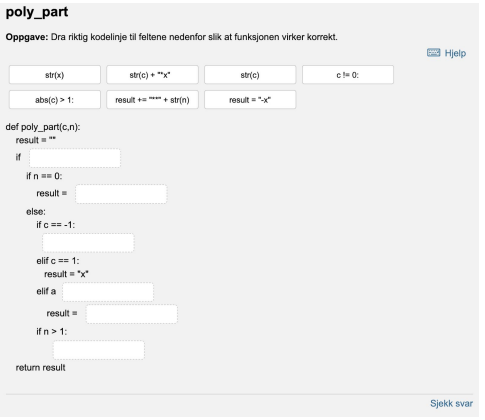
DEL 1 - Lage oppgave i nytt system

(Helt samme oppgave skal lages i begge systemer)

➢ **NB: Begynn med å logge inn i Inspera først**

1. Naviger til det nye systemet på: https://master.preference.no/
2. Koden du skal lage oppgave av finnes her: https://raw.githubusercontent.com/Chr1stian/masterapp/master/task.py
3. Kodebitene markert med rød firkant skal byttes ut med en “Gap” (hint: marker koden og trykk -ikonet)
4. Legg til en distractor: “str(x)”
5. Når alle gaps er laget skal du eksportere oppgaven med “EXPORT TASKS”-knappen og laste opp .zip-filen til Inspera. Følg instruksene i systemet.
6. Den ferdige oppgaven skal se ut som bildet til høyre





DEL 2 - Spørreskjema om nytt system

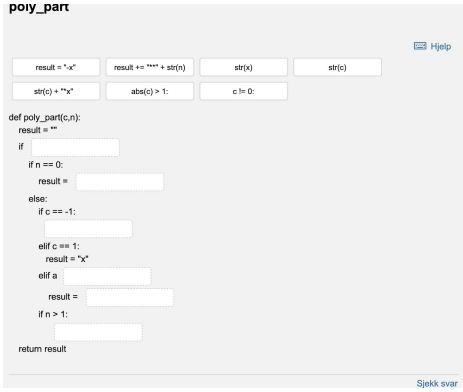
Gjennomfør spørreskjema på linken under når testleder ber deg om dette.
https://forms.gle/Yo6d1VyrZvzQ5xxPA

DEL 3 - Lage oppgave i Inspera

(Helt samme oppgave skal lages i begge systemer)
Navigering i Inspera er ikke en del av testen. Testleder vil veilede deg til riktig oppgavetype om det trengs.

1. Lag en ny “Inline Gap Match” / “Plasser i tekst” -oppgave
2. Koden du skal lage oppgave av finnes her: https://raw.githubusercontent.com/Chr1stian/masterapp/master/task.py
3. Kodebitene markert med rød firkant skal byttes ut med en “Gap” (hint: **+Insert** eller **+Sett inn** knappen)
4. Det som står inne i firkanten skal så settes som et riktig svar for det “Gap”et
5. Legg til en distractor: “str(x)”
6. Når alle gaps er laget med tilhørende korrekt svar kan du trykke på øyet for å få en forhåndsvisning av den ferdige oppgaven
7. Den ferdige oppgaven skal se ut som bildet til høyre

```
def poly_part(c, n):
    result = ""
    if c != 0:
        if n == 0:
            result = str(c)
        else:
            if c == -1:
                result = "-x"
            elif c == 1:
                result = "x"
            elif abs(c) > 1:
                result = str(c) + "*x"
            if n > 1:
                result += "**" + str(n)
    return result
```



DEL 4 - Spørreskjema om Inspera

Gjennomfør spørreskjema på linken under når testleder ber deg om dette.
https://forms.gle/pAybb3bEzJWvMUDL7

DEL 5 - Intervju

Svar på spørsmål, diskuter og kom med innspill om du ønsker. Takk for at du deltok!

Appendix B

Requirements

B.1 Functional Requirements

Number	Requirement Description
R01	The IT artefact should provide the user with a graphical user interface
R02	The IT artefact should support code/text pasting
R03	The IT artefact should let the user create *gaps* from selected parts of the code/text
R04	The IT artefact should let the user add distractors
R05	The IT artefact should be able to generate a *Inline Gap Match* task QTI-file with the users text, selected gaps and distractors
R06	The IT artefact should be able to generate a ZIP-file with the packaged QTI-file and a manifest-file to upload to Inspera
R07	The IT artefact should let the user download the generated ZIP-file
R08	The IT artefact should provide tool-tips to guide and help the user
R09	The IT artefact should provide the user with error, warning or information messages. E.g., when a wrong actions is performed or before anything is deleted permanently
R10	The IT artefact should temporarily store the users work as long as the current session remains active
R11	The IT artefact should support the change and editing of the pasted code/text

Table B.1: Functional Requirements

B.2 Non-functional Requirements

Number	Requirement Description
NR01	Every action should be performed immediately without any delay
NR02	The IT artefact should be available on every operating system
NR03	The IT artefact should be available from common browser (IE, Safari, Chrome, Firefox)
NR04	The usability (efficiency, effectiveness and satisfaction) of the IT artefact should be higher than Inspera for the creation of *Inline Gap Match* tasks
NR05	The IT artefact should be easy to learn and use
NR06	The design of the IT artefact should be simple and pleasant
NR07	The code should be structured split into components to be readable, modifiable and extendable

Table B.2: Non-functional Requirements

Appendix C

Transcripts

C.1 Categories

Number	Category	Frequency	Subcategory
01	Alt i et system	5	Alt på et sted, i et system, Bra at alt er i samme verktøy, Alt på samme sted, Alt på samme sted, Alt i samme system
02	Andre gode ting med Inspera	4	Bra for å rette eksamen, Bra på strukturering av eksamen, Har oppgavebank, Bra at oppgavetypen finnes
03	Bra brukergrensesnitt	1	Tydelig fargebruk på knapper

Table C.1: Positive Categories for Inspera

Number	Category	Frequency	Subcategory
04	Dårlig brukergrens-esnitt	13	Flytte fokus, Sett inn knapp dårlig plassert, Komplekst, Forvirrende menylinjer, Skjult funksjonalitet, For mange valg i grens-esnitt, Rotete interface, ikke tilpasset hver oppgavetype, Inspera er komplis-ert, Mangler beksrivelse av hva opp-gaven er, Ikke intuitivt grensesnitt, Mange vinduer/arbeidsflater, Veldig uoversiktlig, Dårlig hjelpetekst
05	Dårlig workflow	8	Krever flere aksjoner, Mer tungvint, Flere aksjoner, Krever mye hjelp, Lite intuitivt og dårlig med veiledning for hva som kan gjøres, Krever mange aksjoner, Vanskelig å bruke, Mange aksjoner
06	Dårlig prosess for laging av gaps	7	Rotete pga størrelse på gaps, Tungvint å lage Inline-Gap-Match oppgave, For store gaps, Identifikator som er unødvendig, Gaps kan ikke fjernes enkelt, Dårlig pros-ess for å lage gaps, Gaps kan ikke fjernes med backspace, med mindre de markeres
07	Burde håndtere kode-format/syntax high-lighting	3	Burde ha kodeeditor, Mangler syntax high-lighting, Feil formattering

Table C.2: Negative Categories for Inspera

Number	Category	Frequency	Subcategory
08	God workflow	10	Bra workflow, Enkel workflow, Enkel prosess, Går fort å lage oppgave, Krever få aksjoner, Få aksjoner, God workflow, Effektivt å lage oppgave, Er mer brukbar en Inspera, Gode forklaringer
09	Godt brukergrensesnitt	8	Enkelt brukergrensesnitt, Lett å lære, Enklere å bruke, Forståelig brukergrensesnitt, Krever lite hjelp, Inuitivt interface, Ser bra ut, Gøy interface å bruke
10	Bra måte å lage gaps på	6	Gaps er oversiktlige, Kul måte å lage gaps på, Enkelt å lage gaps, Gøy å lage gaps, Enkelt å legge til distraktorer, Linjer under kode er hjelpsomt
11	Ser nytteverdi av systemet	5	Ser nytteverdien av systemet, Ser nytteverdien av systemet, Ser nytteverdien i et slikt system (ville brukt det over Inspera), Ser nytteverdien av systemet, Ser nyttverdien av systemet
12	Bra med kodeformatering / syntax highlighting	3	Bra med kodefelt, Bra med syntax highlight / kodeeditor, Bra med kode formatering
13	Kan brukes til mer en kodeoppgaver	2	Kan brukes til mer en programmeringsoppgaver, Burde også kunne brukes til annet en kode
14	Andre tilbakemeldinger	2	Fungerer også på mobil, Enkelt å eksportere

Table C.3: Positive Categories for IT Artefact

Number	Category	Frequency	Subcategory
15	Dårlig prosess for laging av gaps	15	Uklart om man skal markere først, eller trykke crop-ikon først, Add distraktor knapp ikke synlig på liten skjerm, Vis (riktig) innhold i gap, Flytte add distraktor knappen til under koden, Velge/markere flere gaps av gangen, Gaps kan være mer synlige, Gaps kan være mer synlig, Bytte crop-ikon til saks-ikon, Kan ha bildeeksempel av lage gap prosessen, Ikke intiutivt å markere ren tekst (i step two), Enkelt å gjøre feil i lage gap prosessen, Gaps kan vise riktig svar, Ikke så intuitiv måte å lage gaps på, Syntax highlight når man velger gaps også, Linjene som skiller i step two støyer
16	Dårlig brukergrensesnitt	6	Kan gjøres enklere, Mangler beskrivelse av hva oppgaven er, Litt vel minimalistisk grensesnitt, Uklart hvilke aksjoner som skal eller kan gjøres, Flere og bedre forklaringer, Må bruke samme begreper som i Inspera
17	Må kunne angre	5	Burde ha angrefunksjon, Bør kunne angre, Muligheten til å angre, Ingen angremulighet, Må kunne angre
18	Vanskelig å importe til Inspera	4	Vanskelig å laste opp i Inspera, Brukte mest tid på import til Inspera, Kan ha bedre veiledning for eksport av task, Kan ha tydeligere forklaring av export/import
19	Må bruke to systemer	4	Brukere må lære seg et ekstra system, Dumt å måtte bruke ekstra verktøy, Ekstra system å bruke, Må bruke et ekstra system
20	Eksport knapp feil plassert eller utydelig	4	Eksport knapp nede i høyre hjørne, Alle aksjoner/knapper samlet på ett sted, Export tasks knappen ser ikke ut som en knapp, Eksport tasks knappen dårlig plassert
21	Step one markert selv om det ikke er første som skal trykkes på	3	Step 1 markert initielt, selv om det ikke er første steg, Step one markert er missvisende og kunne vært fjernet, Step One for synlig og markert
22	Dårlig workflow	3	Start rett i en oppgavelaging, Flere eller bedre hjelpetekster for å forklare flyt, Vanskelig å forstå hva man skal gjøre

Number	Category	Frequency	Subcategory
23	Dårlig sikkerhet	2	Ikke fullgod sikkerthet at fil blir generert, Anbefale å slette generert fil etter importering
24	Dårlig hjelpetekst	2	Paste or write your code here (below), Paste or write your code here (below)
25	Kode i step two ser ut som vanlig tekst	2	Kode ser ut som vanlig plain-text, Linjenummer kan gjøre koden enda tydeligere
26	Dårlig tekst/kodefelt	2	Text/kodefelt bør ha fokus og vises tydeligere, Annen metode for valg av inndata
27	Andre knapper dårlig plassert	2	Add Task knapp ikke synlig nok, Next knapp bør være synlig hele tiden (kan være grået ut)
28	Dårlig feilhåndtering	1	Feilmeldinger ved markering i forskjellige linjer stemmer ikke

Table C.4: Negative Categories for IT Artefact

C.2 New Features

Kunne lage oppgavesett
Være integrert som en del av Inspera
Randomisere oppgaven mer
Mulighet til å legg inn poengscore
Autogenerere tekst som forteller om poenggivning på norsk og engelsk
Mulighet for å skrive inn oppgavetekst på norsk/engelsk
Mulighet til å lage andre oppgavetyper
Foreslå distraktorer
Et brukergrensesnitt til å gjøre en laget oppgave

Table C.5: Categories for IT artefact new features

C.3 Transcript Participant 1

R: Hva er dine tanker om det nye systemet jeg har laget?

P1: Jeg synes det så veldig greit ut, det var jo et veldig enkelt og greit rett frem brukergrensesnitt der det bare var å lime inn koden som er den naturlige måten å starte på siden koden vil man jo foretrekker å kjøre i en program editor slik at man vet at den kjører og fungere. Så man vil jo starte med å lime den inn. Det var veldig enkelt og intuitivt at man velger det som skal skjules og så trykker man den crop-tasten på samme linje. Man er da sikker på at det er riktig fasit som går inn automatisk som går inn i systemet siden det er akkurat det som blir fjernet som havner i samme. Altså man oppnår jo det samme i Inspera hvis man passer på å bruke Ctrl + X og så paster igjen, men det blir jo et ekstra tastetrykk fordi du må først ta Ctrl + X for å klippe og så må du trykke deg inn i gappet i Inspera og så klikke deg bort på lime inn svaralternativ så det blir jo en del ekstra brukergrensesnitt-aksjoner, mens det er mye smudere i systemet ditt med bare marker + klikk, marker + klikk. Så det går fortere hvis du har en kodesnutt med en 8-10 ting du skal skjule så er det ganske fort gjort likevel. Og så var det jo veldig lett å få lagt til distraktorer da det stod veldig synlig som en rød knapp der, "legg til distraktor". Ikke at det er så veldig vanskelig i Inspera heller, men jeg måtte se litt siden det var lenge siden sist jeg har brukt det

P1: Så det som jeg husker det var det veldig lett å få eksportert fra programmer, det var bare å trykke på Download ZIP-knappen og så kom det en ZIP-fil i downloads com man da etterpå lett kan laste opp i Inspera. Så det jeg brukte tid på var jo funksjonaliteten i INspera for å laste opp, det var jo ikke funksjonaliteten i din app egentlig. Det var også ett resultat av at det var lenge siden siste jeg lastet opp et resultat i Inspera, så jeg brukte litt tid på å huske eller finne ut av hvordan jeg gjorde det. Hvis det var kort tid siden sist jeg hadde gjort det hadde jeg jo visst med en gang at jeg skulle trykke på den lille pila der

R: Det hadde gått med en gang da ja tenker jeg også

R: Har du noen positive eller negative ting å si om det å lage inline-gap-match oppgave i Inspera

P1: I Inspera så var det jo litt mer tungvint, men det er ikke en så forferdelig lidelse å gjøre det i Inspera heller. På tross av at INline-gap-match er mye enklere å lage enn de her todimensjonale drag-and-drop oppgavene som jeg lagde før der en har en sånn grid av slipp-felt fordi at der kommer det i tillegg at man må hånd justere alle de elementene fordi det finnes ingen rask måte å lage et todimensjonalt grid på. Selv med Gap-Match er det litt mer tungvint i Inspera med at det blir en del tastetrykk og eller museklikk for å få gjort ting fordi at først paster du og så må markere det du skal fjerne, klippe bort, så må du opp å klikke på den her sett inn knappen og så får en inn en gap og så må en klikke seg inn i gappen og så må man bort til høyre i skjermbildet for å få lagt inn svaretet. Det kan du jo skrive i oppgaven for å sammenligne i tillegg til eksperiment med personer så kan du telle antall brukergrensesnitt-aksjoner man trenger for å få gjort noe. Det er nok større i Inspera, både flere aksjoner og du må flytte fokuset ditt fra ett sted på skjermen til et annet fordi du har oppgaven der og så må ut til høyre for å

gjøre noe med fasiten og sånt da. Så er det selvfølgelig også sånn i Inspera at du har mulighet til å gjøre feil, f.eks der jeg først feilaktig rimte inn i labelen istedenfor det som skulle være innholdet på svaret, pluss at jeg feilaktig pasta inn alt for mye da siden jeg pastet hele koden. Og selvfølgelig hvis en lager mange oppgaver så vil jo det stort sett forsvinne de typiske samme feil man gjør om det er en stund siden sist man gjorde det. Men nå er jo eksamensoppgaver man typisk lager en gang i halvåret, så hver gang man starter igjen så vil en vanlig bruker som ikke lager masse oppgaver hele tiden vil ha det problemet hver gang. Man vil typisk havne i den situasjonen hvert halvår hvor man ikke husker helt hvordan man gjorde ting, særlig siden det er mange forskjellige sjangre på oppgaver. Så hver sesong lager man kanskje bare 2 inline-gap-match oppgaver, 2 fill-inn-blanks, ett par multiple choice oppgaver et par vanlige programmeringsoppgaver og sånt så man blir ikke veldig dreven i hver enkelt oppgave sjanger. Så var en del mer knølete i Inspera, og så en annen ting var det rent visuelle fordi gap-boksene ble så digre. At til slutt blir det vanskelig å se sammenhengen i koden, og særlig hvis man sitter på en litt liten skjerm som jeg gjør nå så ble det jo ikke plass til hele koden i skjermbildet så da må man begynne å scrolle tilbake igjen. Og den her sett inn knappen blir jo da ekstra tungvint. Alt det her slipper man med løsningen din. Her igjen var det jo sånn at det jeg brukte mest tid på var importen etter at jeg hadde eksportert ZIP-filen og der selvfølgelig er det jo sånn at hvis man lager mange oppgaver på rappen og lager en ZIP-fil og importerer så vil man jo slippe unna med en import-aksjon, men kan ha laget fem oppgaver. Så hvis det her var et eksamenssett jeg skulle lage, og la oss si at jeg hadde tenkt til å bruke tre eller fire sånne inline-gap-match oppgaver i det eksamenssettet så ville jeg jo hvis jeg skulle gjort det effektivt lagd alt på en gang og så eksportert ZIP-filen og så importert i Inspera. Da hadde det jo brukergrensesnitt messig gått enda fortere per oppgave fordi jeg ville sluppet unna med en eksport/import aksjon.

R: Har du noe negative tanker rundt å lage Inline-Gap-Match oppgaver i det nye systemet mitt?

P1: Nei det var ikke noe jeg så som jeg var misfornøyd med egentlig. Kunne tenkt seg å muligens ha download knappen helt nede i høyre hjørne. Muligens kunne man også hatt alle aksjoner man kunne gjøre samlet ett sted på skjermen. Det kan hende det hadde vært mer intuitivt. Det kan jo hende at brukerne overseer, hmm- er det noe nede i høyre hjørne. Gjerne hvis man ikke sitter med appen i fullscreen, men har flere vinduer oppe samtidig så kan det være at man ikke ser det som man har nede i høyre hjørne. Det har jeg ikke prøvd da. Det kan jo være en ting å tenke på, er det best om den står nede der eller kan den stå oppe der da og at man så mer sammen hva er de forskjellige aksjonene du kan gjøre nå. På den andre siden så er det ikke vits i å eksportere noe før oppgaven er ferdig heller da på en måte da, så det kan jo være greit at den stod der da.

R: Knappen blir ikke brukbar før du har fullført oppgaven til en viss grad. Kan jo også få eksporter oppgave knappen til å dukke opp i det andre viewet når man har gjort en viss del av oppgaven

P1: Det som jeg var litt inne på var jo at det finnes jo selvfølgelig mulig ekstra funksjonalitet. En brukergrensesnitt messig ting som kunne vært enda mer effektivt enn det som står nå,

istedenfor å merke av en ting som skal fjernes, trykke crop, en ny ting som skal fjernes, trykke crop. Om man kunne man merke en ting, den blir stående, merke enda en som også blir stående, så man kan merke 5-6-7 ting som blir stående og så kan man trykke en knapp som gjør at alle blir gjort til gaps. Det er fordeler og ulemper ved det og jeg vet ikke om det er noen stor fordel med det. For da ser du fortsatt hele sammenhengen i koden da før noe har forsvunnet på en måte. Jeg merker de her 6, kanskje skal jeg merke noen andre istedenfor. For det kan være noe man tenker på litt underveis hva er det som er lurt å skjule eller ikke. Med denne testen her blir det sånn at man har bestemt seg allerede for hva som skal skjules. Det kan jo være et scenario hvor en bruker har noe kode og så lurte han på hva er det som er lurt å skjule her og ikke, så det kan jo være en mulig ting å vurdere. Vil det være bedre enn sånn som det er nå, vil det være lettere å merke av mange ting uten at det første man merket av forsvinner, og så til slutt trykke en knapp for det her var det jeg ville skulle for å lage en 8 gaps samtidig på en måte. Og så det her med om det var mulig med en brukerdialog for å legge inn poengscore, da kan man jo se litt på hvordan det er i Inspera, er det lett å få det til på en mer effektiv måte i din app, for da kan det jo være en ting som lønner seg, for da får en jo alt en trenger med oppgaven på samme sted. Istedenfor som nå hvor du må inn i Inspera for å si noe om hvor mye poeng du får per riktig plasserte (gap). La oss si at du har en sånn Inline-Gap-Match oppgave der det er 5 ting som er skjult så teller oppgaven 5 prosent på eksamen så vil en typisk sånn scoring-måte være at hver av de teller ett poeng. Så kan det jo selvfølgelig være at man ønsker å gi minuspoeng om man har feil plasserte gaps eller ikke. Så kan man tenke seg at ut i fra de valgene så blir det autogenerert norsk og engelsk tekst som forklarer "på denne oppgaven får du et pluss-poeng for hver riktig plasser, og minus ett halvt for hver feil plasser" alt etter hva brukerne har valgt i disse poeng valgene. Du må jo se hva som er gjort i Inspera, er det tungvint, ser jeg en enklere måte å gjøre det på i min app. For hvis jeg ser en vesentlig enklere måte å gjøre det på så vil jo det kunne lønnse seg for da vil brukeren kunne gjøre alt i appen og så bare importere den når oppgaven er helt ferdig på en måte versus om man må gjøre litt i appen og så gjøre litt i Inspera.

P1: Og en annen ting relatert er selvfølge den norske/engelske oppgaveteksten i forkant. Vil det være hensiktsmessig om man også kan ha den i appen, at man paster den inn der, eller er det bedre at folk gjør det i Inspera. Der er det kanskje ikke så masse å tjene på å slippe å gjøre det i INspera, hvis det i din app også blir bare å paste inn. Det blir ikke hensiktsmessig for deg om du skal lage noe mer avansert tekst-editor som gir fin formatering eller sånt, fordi i Inspera er teksteditoren veldig bra, men den har jo i det minste sånn bold og sånne forskjellige ting som man kan få til som det ikke er hensiktsmessig om du skal drive å fikle med i din prototype fordi hovedpoenget i din prototype er jo å få til det som skal gjøres med din prototype i forhold til koden.

R: Har ikke sett så nøye på det med poenggivning, men det er jo en default som gir ett poeng per rette svar. Når det gjelder oppgavetekst så var det noe jeg hørte fra Simon og de at det var noe de brukte veldig mye tid på som egentlig ikke ga så mye gevinst, fordi det var veldig begrenset hvilke muligheter man hadde.

P1: Så jeg tenker at det er lavere prioriteret enn. Det med poengene tenker jeg det kan være mer poeng å gjøre noe med enn oppgaveteksten. Oppgaveteksten er sånn sett ganske enkelt å få til i Inspera, der tar man jo bare å paster inn den. Men poengene må en jo drive å klikke mer ute til høyre.

P1: Andre utvidelser du kan ha er f.eks hvis du først har et grensesnitt der du kan markere masse ting så er det jo ganske kjapp å lage muligheten for at du kan velge vil du ha en Inline-Gap-match oppgave eller vil du ha en Fill-In-Blanks oppgave. Det kan du jo på direkten bare velge og få generert. Og Inline-Choice er jo ganske lett bortsett fra at da må man jo ha distraktorer til hver enkelt ting man fjerner da nødvendigvis.

R: Ja det har jeg på listen fra sist vi snakket sammen. Det er relativt enkelt å endre i frontend, men for QTI-koden er det litt mer enn å bare endre en knapp, men det burde ikke ta så mye mer enn en uke å få til den funksjonaliteten der.

P1: Du får vurdere hvor mye tid du får på slutten.

R: Det står høyt på prioriteringslista for det vil gi veldig mye gevinst om du kan lage den type oppgaver også

P1: Særlig for Inline-Choice hvis en også automatisk foreslår noen distraktorer og de faktisk var bra nok til at faglærer ville velge å bruke de. Ellers må man jo i Inspera klikke legg til distraktor, skrive inn den, legg til distraktor, skriv inn den slik at du får en del fikling i brukergrensesnitt for å lage de distraktoreren. Da blir det et litt mer tricky problemer med å foreslå gode distraktorer, da må en jo ha hva som er typiske distraktorer en kunne finne på å bruke og så kan jo systemet foreslå mer enn man trenger. La oss si at man tenker å ha tre eller fire svaralternativ på hver Inline-Choice så kunne jo systemet godt foreslå ti distraktorer for den del og så klikker bare faglærer på de tre man har lyst til å bruke. Da behøver ikke programmer å være så intelligent i hva det foreslår for da kan man håpe at av de 10 den foreslår så vil tre være brukbare, og selv om bare en er brukbar så kan faglærer velge den og så må han skrive resten selv.

R: Jeg ser for meg at det blir enten gjetting, eller så må den være veldig smart. Det blir vanskelig da det er så mye som kan variere mtp f.eks programmeringsspråk, oppgavetype osv. Distraktorene burde jo være litt sannsynlige svaraltentativer gjerne

P1: Det kan og tenkes at brukergrensesnitt messig om faglæreren må skrive inn alle distraktorene, hvis man bare har ett felt under hverandre linje for linje hvor han kan skrive så går det mye kjappere enn man kan gjøre det i Inspera. Så vidt jeg husker må man der trykke på legg til distraktor, legg til distraktor (inaudible..)

R: Istedenfor de gapsene du får når du trykker crop nå kan du istedenfor få en dropdown meny hvor du kan skrive inn alternativer og velge det riktige alternativet der. Da slipper du så mye

trykking, da kan du bare skrive og trykke enter for å komme til neste linje. Så er det første det riktige svaret, men så blir de random når de kommer inn i Inspera

P1: Hvis du har pastet inn koden din og så markere hva som skal fjernes så vet en jo allerede hva som er det riktige alternativet, så det er bare distraktorene man trenger å skrive inn i tillegg egentlig

R: Så du noen ulemper i mitt program i forhold til Inspera? Var det noe som var bedre i Inspera?

P1: Nei, det kan jeg vel egentlig ikke si i det hele tatt. En kan alltid være bekymret for når man bruker ett annet system, blir det her riktig i Inspera nå, men straks man importere så ser man jo det, og straks man trykker på øyet i Inspera så ser man jo at det virker. Så egentlig så jeg ingen ulempe i det hele tatt. Det eneste man kan si som ulempe er at brukerne da må lære seg to forskjellige system istedenfor ett. Hvis man bruker bare Inspera trenger man bare å lære seg det, og bruke det konsekvent. Men i å med at dette systemet var såpass lett å lære så vil ikke jeg betrakte det som en ulempe, men det kan være en vilkårlig bruker som ville eller måtte lære seg færrest mulig system ville foretrekke å bare gjøre det i Inspera. En mulig ulempe hvis en skulle komme med noe, kan gå på bekymringer rundt sikkerheten. La oss si hvis du tenker på filene i Inspera er forhåpentligvis ganske sikkert lagret siden man trenger passord for å logge seg inn, og man kan se hvem som har vært inne på forskjellige oppgaver og sånt, men den her ZIP-filen man generer blir jo bare en fil i downloads-katalogen til den aktuelle faglærerer, og selvfølgelig skal jo ikke folk heller vite passordet til den faglæreren. Så det er jo ikke slik at hvem som helst skal ha tilgang til å lese den downloads folderen, men det kan jo hende at noen har bekymringer til - "hmm, vil den filen kanskje være lettere å få tak i enn en hacker, enn oppgavefilen i Inspera". Men det er jo bare en hypotetisk bekymring, jeg tror ikke at det reelt sett er noen veldig stor risiko. En kan selvfølgelig være en anbefaling om at man bør slette alle de filene straks man har fått lastet de opp til Inspera slik at ikke oppgaven ligger flere forskjellige steder.

R: Mhm, det er jo smart og vil unngå mye av det problemet

R: Hva slags effekt hadde det nye systemet i forbindelse med effektiviteten ved å lage oppgaven?

P1: Jeg synes jo det ble bedre. Så kan sikkert du si mer om det etterpå om du har tatt tiden på det og sånt, men jeg følte jo at det gikk vesentlig kjappere og selv om jeg måtte gjøre det ekstra steget med å eksportere og importere, men det å lage selve oppgaven med å merke av hva som skulle fjernes og få lagt inn distraktorer og sånt gikk mye kjappere da. Så den her import aksjonen og var det jo bare at jeg måtte inn i hjelpefunksjonen hvis jeg ikke hadde måttet det hadde det jo gått veldig kjapt.

R: Da har vi egentlig vært innom alle spørsmålene, så da takker jeg for at du var med på intervjuet også.

C.4 Transcript Participant 2

R: Hva er dine tanker og meninger om det nye systemet jeg har laget?

P2: Det er jo dramatisk mye bedre enn det som er innebygd i Inspera, det er det ikke noe tvil om. Og mye enklere å bruke og så det er jo ikke noe tvil at hvis jeg skal lage den type oppgaver og det skal man vel sikkert så vil jeg absolutt foretrekke å bruke det systemet foran Inspera sitt innebygde.

R: Takk, det var jo fin tilbakemelding. Hvis man skal dra det litt tilbake til Inspera, hva synes du var positivt og negativt med å lage oppgaven der?

P2: Inspera er jo komplekst. Nå er det jo klart at det er en enklere oppgave for deg som har lagt oppgave for en spesifikk oppgave-type, mens Inspera har jo lagd masse forskjellig. Altså du blir jo litt forvirret fordi du vet jo ikke hva slags oppgave du skal velge for å lage den type oppgaver, jeg vet ikke du har jo typ 15 forskjellige typer å velge mellom. I tillegg har du jo den menylinjen oppe som egentlig ikke sier deg så mye, så det er veldig mye som i Inspera skjuler det du egentlig skal gjøre, fordi du har for mange valgmuligheter som egentlig ikke er relevante i forhold til den oppgaven, det du jobber med. Dermed så ser du alle trærne, men du ser ikke noe skog.

R: Rett og slett for mange valg?

P2: Ja, jeg synes det er viktig når man lager eller jobber med noe at de valgene som faktisk er relevante for deg, det er de valgene som skal være tydelige. I mange situasjoner så har du andre ting som kan være relevante i enkelte sammenhenger, men de skal ikke være like tydelige som de tingene du faktisk bruker, har mest bruk for. Det er jo selvfølgelig veldig vanskelig å vite akkurat hva det er når du skal utvikle, men det er jo det som er vanskelig med design, å vite hva du skal kutte ut.
I utgangspunktet synes jeg at Inspera gjør ting unødvendig tungvint. Jeg har ikke følelsen av at man virkelig har jobbet med å designe noe der.

R: Alt synlig for brukeren, de kunne kanskje vunnet litt på å skjule det som ikke brukes så ofte og la det mest nyttige være litt tydeligere?

P2: Ja, og så er det vel akkurat det at uansett hva slags oppgave du skal lage i Inspera så jobber du i samme “view”, og så har du menyer på forskjellige steder, øverst så har du en rad med ting du kan klikke på som det øyet osv, og så når du skal lage selve oppgaven skal du holde deg innenfor den ramma, men når du skal legge til alternativer skal du utenfor den ramma og bruke den menyen på høyre siden, det er spredt litt over alt, du kan ikke fokusere på akkurat de tingene du trenger å jobbe med.

R: Hvis vi går over litt til mitt system igjen, hva synes du var positivt og negativt med det?

P2: Fordelene med det er jo veldig fokusert på akkurat den type oppgaver slik at du slipper alt det formelle rundt det, siden du er veldig fokusert på det.

Det var to ting, det første er at når du kommer inn og skal begynne så har du bare et stort hvitt felt og så står det “Step 1” markert og så er det ikke sånn, nå er det jo bare tre steder du kan trykke, og trykker du på to av de skjer det ingenting, trykker du på den tredje da kommer du i gang så det er jo ikke noe stort problem, men mangler litt informasjon til å begynne med.

Det andre var, det var jo ikke noe stort problem. Den linja jeg fant er spørsmålet, skal jeg markere før jeg velger crop-verktøyet eller skal jeg velge crop-verktøyet først, før jeg marker. Men jeg prøvde å velge crop først, og så velge etterpå, da skjedde det ingenting. Da gjør du det jo på andre måten og da fungerte det, så. Jeg vet ikke, det er ikke noe stort problem, fordi du har to alternativer og gjør du på den ene måten så skjer det ingenting så da prøver du på den andre måten og da funker det så det er jo ikke noe stort problem. Alternativet kunne jo være at disse crop-symbolene var dimmet ned, og når du valgte noe i en linje så ble akkurat det crop-symbolet det ble tydelig igjen.

R: Det var en veldig smart ide, takk!

P2: Men det løser jo bare det problemet.

R: Det gjør det også lettere hvis du har flere linjer med kode å trykke akkurat det ikonet du ønsker. God tilbakemelding. Sånn ellers synes du brukergrensesnittet var greit, var det noe mer som var uklart?

P2: Nei synes det var veldig rett frem når du først hadde sett hvordan du skulle gjøre akkurat den biten så gikk det jo veldig greit.

R: Det kan virke som om man synes det er greit å bruke når man skjønner hvordan, men det kanskje skulle vært presisert litt bedre hva man skulle gjøre for å komme i gang.

P2: Ja, det andre, det siste var jo det at den der knappen “add distractor” på min skjerm så ble jo den. Akkurat den knappen jeg bare så så vidt at det var noe rødt der, jeg tenkte kanskje jeg skulle scrolle bortover der for å se knappen. Så den bør kanskje være plassert på et annet sted der det er garantert at du ser den på samme uansett om du må scrolle lite. Jeg vet ikke hvordan det oppfører seg i andre nettlesere.

R: Nei akkurat det var litt rart fordi da testperson 1 brukte det i Safari (samme nettleser) så holder programmet bredden til vinduet og gjør at knappen alltid er synlig uansett, men det er uansett et godt poeng og knappen kan stå på venstre side, som løser det problemet.

P2: Jeg ser at når jeg skalerer vinduet så blir det, bruker du frames eller ett eller annet sånt her?

R: Nei jeg bruker, hva heter det, flex. Så den skal egentlig dyttes rundt om kring etter hvor stort vinduet er, og dermed også være mobilvennlig som den til en viss grad er.

P2: Det er nettopp det, det er litt rart at den knappen forsvinner, jeg ser det at det gråe feltet som inneholder alternativene det forsvinner også ut. Og jo bredere vinduet blir, jo mer forsvinner den knappen faktisk når jeg gjør vinduet smalt så ser jeg knappen veldig tydelig.

R: Det skal jeg notere og sjekke.

P2: Det kan være noe med oppsettet mitt det sikkert, men i hvert fall et potensielt problem.

R: Skal se om jeg klarer å gjenskape det.

P2: Det løses veldig enkelt ved å plassere knappen et annet sted. Det vil jo forsåvidt så ville det ikke være noe galt ved å ha den knappen “add distractor” rett under den koden du sitter å jobber med når du jobber med å legge inn disse alternativene så jobber du jo med den koden og når du da skal legge inn ugyldige alternativer så er det jo faktisk. Det er jo i hvert fall min sånn umiddelbare opplevelse at det ville være en naturlig plassering.

R: Det skal jeg skrive ned og ta med videre. Er det noen andre egenskaper er forbedringer du tenker at systemet kunne hatt nytte av for at det skal kunne brukes av deg eller eventuelt andre som lager sånne type oppgaver?

P2: Ikke sånn på sittende bak. Som sagt når du fant ut hvilken knapp jeg skulle trykke på for å komme i gang, og når jeg fant ut hvilken rekkefølge jeg skulle bruke på crop og select verktøyet da, så fungerte det jo egentlig veldig greit. Oversikten over alternativene der oppe blir jo tydelig og fin så, nei jeg har ikke noe sånn forslag til endringer her og nå for å få det til å funke bedre for mitt vedkommende.

R: Det kunne for eksempel også vært tilleggsfunksjonalitet, om det er noe du skulle ønske var der i tillegg for å gjøre det lettere eller bedre å bruke.

P2: For denne type oppgave så er vel dette egentlig veldig greit, du har kode, du har kode og så har du alternativer og så gjelder det å få det inn på rett sted. Nei, jeg synes til denne type oppgaver så burde dette fungere veldig godt.

R: Da har vi egentlig gått igjennom de fleste spørsmålene jeg hadde. Har du noen spørsmål, forslag, tanker eller meninger du vil dele?

P2: Nei, ikke spesielt, men du kan si at jeg opplever jo gjerne Inspera på den måten at det er mange typer oppgaver man kan ha lyst til å lage og som sikkert er mulig å lage i Inspera, men på grunn av at Inspera liksom skal være alt for alle så blir det såpass kronglete og så mange muligheter og ting å lage, at alt blir såpass komplisert at du vegrer deg litt fra å gjøre det fordi at

det er ikke noe problem å lære seg akkurat den type oppgave som du tenker å lage og så lærer du deg det til to eksamener, men så finner du ut at det går greit du kan lage flere sånne oppgaver, men så går det et halvt år eller ett år til neste gang du skal gjøre det og så er det borte, hvordan kan du gjøre det igjen fordi det blir for komplekst.

R: For du bruker det bare en gang hvert semester når det skal lages ny eksamen.

P2: Ja ikke sant, mens dette er jo i langt større grad selv forklarende synes jeg.

R: Sånn oppsummert så ser du nytteverdien i et sånt program?

P2: Så absolutt, hvis jeg skulle laget denne type oppgaver så er det ikke tvil om at jeg ville brukt dette systemet mye heller enn det som finnes i Inspera.

R: Takk, det setter jeg pris på. Har du da noen spørsmål til meg, noe du lurer på?

P2: Nei, egentlig ikke, men er det noen mulighet for at dette kommer til å bli implementert slik at vi kan bruke det når vi lager eksamen, eller kommer det bare til å være en sånn oppgave som havner i en skuff og som du bruker for å få deg jobb.

R: Nei jobb hadde jeg fått før jeg begynte på oppgaven

P2: Haha, ja det er vel det man får når man tar en master i IT/datatek

R: Hva som skjer med oppgaven videre vet jeg ikke helt, det må du nesten spørre veileder om. Oppgaven min er en videreførelse av en oppgave fra i fjor, så den døde jo ikke på ett år.

P2: Det stemmer det jeg husker jeg var faktisk med å testet da det ble gjort en oppgave i fjor også. Jeg husker ikke helt hvilken oppgavetype det var men det var også mye bedre system enn det som var (i Inspera)

R: Det var sikkert den 2-D Parson drag-and-drop oppgaven

P2: Ja riktig, stemmer det.

R: Min oppgave fortsetter da siden Inline-Gap-Match dukket opp som ny oppgavetype, som kanskje kunne være enda bedre, men selve programmet lever jo foreløpig på den nettadressen sikkert en stund til, ellers har jeg det som en ferdig pakket .jar fil om du ønsker å ha programmet i nåværende versjon.

P2: Ja, men veileder har vel tilgang på det uansett så.

R: Det får han selvfølgelig, og hvis du ønsker det så er det bare å sende en mail.

P2: Og hvis vi skal lage oppgaver så kommer vi til å få tilgang på de verktøyene gjennom han så det er ikke noe problem.

R: Nei

P2: Nei jeg synes det så veldig greit ut, dette vil gjøre det mye enklere å lage denne type oppgaver enn det som allerede finnes i Inspera

R: Det er flott, det var jo målet.

P2: Ja ikke sant, og du så jo hvor mye jeg måtte spørre deg om hjelp for å komme i gang med Inspera, mens her så var det vel to spørsmål, hvilken knapp skal jeg trykke for å komme i gang og så hvor er distractor

R: Da vil jeg bare si tusen takk for du ble med T, det setter jeg veldig pris på. Takk for tilbakemeldingene, gode tips og forslag.

P2: Ja, okey greit.

R: Ha en fortsatt fin mandag og uke.

P2: Du også, fest med måte.

R: Haha, ja, ha det bra.

P2: Ha det bra

C.5 Transcript Participant 3

R: Hva er dine tanker og meninger om det nye systemet som jeg har laget?

P3: Du vet du hva, for det første så tenker jeg at den type oppgave kan være ganske interessante å prøve mot studenter. Abolutt, når det gjelder programmering og også andre ting så kan. Husk på vi tenker gjerne programmering, i en sånn situasjon, men vi må tenke lenger enn det. Det sitter drøssevis av folk som tar botanikk og andre fag, kjemi alt mulig, de har en mulighet til å bruke de samme typer oppgaver som det her. Og det du gjør er å forenkle en sånn prosess veldig, du dummer ned den prosessen veldig. Du kan med en del små grep gjøre det enda enklere, sant, og jeg har forsøkt noen av dem. Det er noen flere også, men det jeg tenker er at. Håpet mitt er at du skal lage noe som paniske forelesere og eksamenslagere kan si “oki jeg drar hit og så får jeg hjelp til å lage disse oppgavene på en mye bedre måte enn sånn det er i Inspera”, sant. Og jeg synes definitivt du har fått til det på en veldig god måte, du har gjort det veldig mye enklere å forstå prosessen enn det den var i Inspera. Det har du gjort, det synes jeg. Og så er det en del sånne ting som du kan tenke på. Eller ting jeg ville tenkt på hvis det var et ferdig produkt.

(Viser til eksport-task popupen) Frykten min var jo at det her skulle bli borte, nå var det heldigvis bare å trykke på knappen en gang til. Jeg vet ikke hva målet ditt med oppgaven er ut over det du skal gjøre. Jeg har ikke lyst til å bruke lang tid på å hjelpe deg med ting som ikke er innenfor din masteroppgave, men som ville vært noe hvis det var noe som var et faktisk brukt produkt. Hva er det du har tid til å gjøre? Hva blir du evaluert på?

R: Jeg har bare en måned igjen nå slik at det blir litt begrenset hva jeg rekker å gjøre i tillegg (til å skrive masteroppgaven)

P3: Nettopp, men de her småtingene jeg tenker på, det er sannsynligvis en liten ting. Det med å faktisk vise hva det gapet er, snarere enn å ha en hvit firkant. En sånn liten detalj som muligens er en forholdsvis enkel fiks. En annen ting er jo hva er det du ønsker å gjøre. Ønsker du å bruke tiden til å skrive masteren din og tenker at du er ferdig nå, hva er det du vil ha ut av kommentarene mine.

R: Jeg setter veldig pris på alle ting som gjelder brukbarhet, for hele poenget er jo at det skal være mer brukbart enn Inspera.

P3: Og det er det jo definitivt.

R: Jeg samler inn en del resultater nå og så skal jeg skrive en del på opgpaven min, men jeg har veldig lyst til å fikse og forbedre alt før jeg er ferdig, men jeg har jo ikke så mye tid igjen, det er det eneste.

P3: Jeg skjønner og det er jo en. Altså virkelig altså, jeg ser jo at du har tatt utgangspunkt i at det her er kode (viser i “step one” vinduet i Artefact) og det vel og bra, for det er på en måte det

domenet vi lever i her, men husk på det når du skal skrive i denne masteren din. Ikke bare skriv at det for kodeoppgaver, husk at det faktisk er for så mye annet. Sant.

R: Ja veileder har jo prøvd å si at jeg ikke må begrense det til bare QTI i Inspera, men hvis jeg kan si at det ikke bare er til programmeringsoppgaver heller så er det vel bare positivt for oppgaven min.

P3: Ja for det er ikke det, der er så mange andre forskjellige ting det her kan brukes til.

R: Sånn som det er nå kan det jo også brukes til andre ting enn kode, det er kanskje bare litt rart å lime det inn i kode-editoren.

P3: Det du kan gjøre er jo å lime inn en tekst. Hvis jeg da går til step two kan jeg velge den (velger en bit i teksten og trykker crop for å klippe ut). Strålende for da får vi "hvilke planteareter er det" "denne blomsten er en" eller ikke sant. Her kan du bruke sånt.

R: Hva var positivt og hva var negativt med å lage oppgaven i Inspera?

P3: (gjentar spørsmål) Altså det å lage, når du sitter å lager en eksamen så det å måtte gå over å bruke et annet verktøy for å lage en type oppgave er ikke nødvendigvis i seg selv en positiv ting. I den situasjonen her så er det jo selvfølgelig på et vis bedre ikke sant, men husk her at for å få din løsning til å fungere så skal jeg for det første vite at det eksistere, og så skal jeg gå til den nettsida og så lage det, og så får jeg den ZIP-fila som jeg så må gå inn igjen og lages som en oppgave, sant. Så den prosessen i seg selv det er en del red-tape som ligger, en del ekstra rundt den her prosessen, som du får ekstra hvis du må gå på utsiden. Nå er det ikke veldig mange andre ting jeg synes er positivt ved å gjøre det i Inspera, egentlig. Som jeg kan komme på. Det er det rett og slett ikke, fordi jeg - vi har hatt våre kamper Inspera og jeg. Hahaha, så det er ikke veldig mange, jeg sa tidligere her at Inspera har en god del positive ting, og det har det helt klart. Det er mye bra ved det, som for eksempel å rette eksamen i Inspera. Det å ligge med i touchskjerm og skulle rette oppgave 2b for alle 900 stundene i Inspera der du trykker neste student, karakter 9 poeng, neste student, 8 poeng, neste studen - det fungerer kjempefint rett og slett. Det er en veldig god måte å gjøre det, men når det gjelder å lage eksamensoppgaver så, altså det er en god del ved Inspera som er positivt når det gjelder eksamensoppgaver.

(navigerer til et tidligere laget oppgavesett i Inspera)
Altså det Inspera har som er ganske fint er den måten man kan strukturere en eksamen på. I forskjellige kategorier, men at her har du en del som er om teori, en som er om.. Det fungerer forholdsvis bra når du har lært deg det, det gjør det absolutt.

R: Det ser jo relativt ryddig ut det der, jeg har ikke gått så nøye gjennom mulighetene for det.

P3: Man har muligheten til å se og dra ting rundt. Flytte opp og ned og sånne ting, hvis du skal importere den oppgavene som du nå har laget, du har kjørt den import saken via ZIP-filene så

kan du “lag ny fra oppgavebank” da kommer du inn i oppgavene som er blitt levert. Så har du den oppgaven her.

R: Men litt spesifikt i forbindelse med å lage en slik Inline-Gap-Match oppgave da.

P3: Den Inline-Gap-Match oppgaven der har for det første ikke jeg noe mye erfaring med å gjøre akkurat det og for det andre så synes jeg at det var unødvendig keitete å gjøre det, jeg synes at det var veldig lite i flyten som forklarte meg hva jeg skulle gjøre. Det du har gjort på et vis er nå resonnerte jeg meg frem til det, men det du gjør er at du deler denne i to. Det du sier er først, limer du inn alt, og så gjør du den jobben med å opprette linje for linje, og så kan jeg markere biter i teksten og si lag gap her. Den prosessen i Inspera var, synes jeg, kjip. Rett og slett.

R: Er det den mest negative tingen med å lage denne oppgaven?

P3: (navigerer til Inline-Gap-Match oppgaven i Inspera) Jeg skjønner jo i og for seg, men jeg hoppet litt kjapt inn i det. Jeg ser ikke her hva er, jeg vet ikke egentlig hva de her er (peker på forhåndsutfylte oppgavetekster og gaps), det er ikke intuitivt for meg hva disse alternativene her er for noe. Det er det kanskje ikke heller i din (navigerer tilbake til artefact), du har også bare liggende, det står ikke noe her hva disse. Ingen av disse grensesnittene har en beskrivelse av hva dette er. “Alternativer:” eller. (legger til distraktor i Inspera) Det naturlige for meg hadde vært her at vi for eksempel kunne fargekode at dette var en distraktor, dette er en, ikke sant fordi det står ingenting her om at det er en distraktor, det står bare at det er en A4 og så må jeg gå inn og se at okey A1, A2, A3. A4 er ikke her altså er det distraktoren. Det jeg vil frem til er at grensesnittet her var kanskje litt, det er ikke fullt så intuitivt hvordan jeg skal gjøre det da.

R: Hvis vi da går litt over til mitt system, hvordan synes du grensesnitt er der?

P3: (navigerer til Artefact og starter på nytt) Grensesnittet ditt er, du er minimalist, og kanskje litt vel minimalist, for jeg vet ikke hva som er det man skal starte med her. Ikke sant, jeg vet ikke i grensesnittet ditt om hvis jeg nå legger på “step one”, okey, hvorfor skjer det ikke noe? Hvorfor er det ikke noe her i step one? Jeg må se at, ja jeg kan lage en oppgave, i det jeg trykker på den.

R: Tror du man hadde unngått det hvis “step one” ikke var synlig før du har addet en task?

P3: Altså det her kan du gjøre på forskjellige måter, du kan enten forsøke å fjerne andre ting, for å gjøre det til det eneste alternative, eller så kan du på en eller annen måte markere at det er her du skal starte, ikke sant. Noen grensesnitt så gjør man det med å ha noe som viser ned til plassen du skal starte, er det sånn å forstå at hver eneste man kommer til dette grensesnittet så må du trykke på add task?

R: Ja

P3: Hvis jeg skulle effektivisert dette her så ville jeg tenkt, og jeg sier ikke at du skal gjøre det, hvis det alltid er sånn at man skal trykke på add task, kan man ikke bare automatisere det med en gang. I den teksten her, okey si at vi lager første oppgave, hva er det oppgaven skal være, sånn og sånn. Og nå kan du komme med at men jeg har noen andre tanker om at sånn og sånn skal man gjøre senere, ikke sant. Så vi lar den bare bli med det.

R: Det kan jo hende det ser litt annerledes ut hvis jeg legger inn lagring av de oppgavene man holder på med og flere oppgavetyper

P3: Nettopp, helt klart, og da vil grensesnittet være annerledes. Så kanskje den her (peker på "step one") ikke burde være der.

R: Jeg tror egentlig jeg hadde tenkt til å få den bort, jeg vet ikke hvor den ble av i task-boardet mitt.

P3: Den ("step one") er viktigere i grensesnittet enn den ("add task" knappen) for den synes ikke. Det første jeg tenker er den noe ligger langt nede til venstre det er ikke veldig synlig da.

P3: Det er viktig at du har det samme begrepsapparatet som i Inspera. Nå bruker jeg norsk i Inspera og dermed kjente jeg ikke igjen hva det er du mener med en task her (popup-meny når man har trykket på add task).

P3: "Paste or write your code here" hehe

R: Det burde vært "below" kanskje?

P3: Ja, ikke sant. Så kan du godt tenke Python, det er ikke sånn at du må, det å ha informasjon eller hjelpetekst på et grensesnitt alltid er galt. Det kan være at det er positivt at du istedenfor å bare at det står Python, da vil folk hvis de er kjemikere eller botanikere så er det noe som ikke passer helt, da kan du heller kanskje ha en "velg", finn på ett eller annet du, du skjønner hva jeg mener

R: Velg inndata, velg kodespråk

P3: Ja ikke sant, format da eller en eller annen sånn sak, der en av dem er ren tekst. Skal bare kopiere noe inn igjen for å se om det var noe mer jeg kunne komme på.

P3: Her ble jeg litt usikker på hva jeg skulle gjøre videre (etter å ha limet inn i "step one") og hva jeg blir usikker på er jeg usikker på, ikke sant. Jeg kommer ifra Inspera der jeg nå er vant til å skulle tagge, jeg kommer hit med det formålet også, så det du må vurdere er, er dette nok. Kan du ved å gjøre noen veldig kjappe endringer, forklare meg som bruker at det du skal gjøre her nå er å legge inn teksten, i neste steg, så skal du markere. Fordi det gjør at jeg mye lettere

forstår hvordan det her foregår. Man tenker “ja men nå er jeg ferdig” da går vi videre, okey. Dette, nå leste aldri jeg den her (viser til hjelpeteksten i step two), det er min feil, fordi det er jo her du har valgt å skrive informasjon så jeg burde ha lest det her, men jeg kjenner igjen i hvert fall et crop-ikon og tenkte at det naturlige hadde vært at jeg kunne markere her. Det som gjør det litt vanskelig for meg og lett å misforstå er at; du har prikkene under her til å hjelpe deg, det er sant, men det at dette ser på mange måter ut som en vanlig tekst. Nå kan du egentlig argumentere med at det at det at det er linjer eller prikker under her gjør at jeg oppfatter det som noe annet, men hvis det ser ut som en vanlig nettside tekst så blir det plutselig veldig vanskelig å tenke at, okey er det naturlig at jeg da skal markere der (markerer i tekst) og trykke på den (trykker på crop-ikonet). Ikke sant? Men det at du har de strekene gjør det forståelig og så står det jo faktisk her oppe også altså.

R: Det var det jeg måtte ta en vurdering på, er det intuitivt å måtte markere ren tekst som ser ut som om den står som det og ikke tekstfelt, den er bare ren tekst. Men jeg fant ingen bedre måte, og jeg var i diskusjon med veileder også om det var noen bedre måte å gjøre det på enn å forklare tydelig at det var det som måtte gjøres. Det som var tanken var at det er enkelt og intuitivt når man begynner å bruke det.

P3: Absolutt. Også tenker jeg det at du har de prikkene under der, det gjør det hakket mer forståelig at det ser ut som om det er et sett med linjer nedover som har en funksjon.

R: Jeg ser også på det som en utfordring hvor man tok flere vurderinger og avveininger på om man skulle ha ett kode lignende-felt osv, men da igjen hadde det vært vanskelig å se linjer og sånne ting, jeg vet ikke.

P3: Ja, jeg aner ikke det finnes jo noen kollegaer av meg som jobber med grensesnitt. Jeg hadde tenkt at skulle det vært noe, hvis du har behov for å gjøre det enda mer tydelig så måtte det vært at det var noe linjenummer foran eller ett eller annet sånt, for å virkelig få frem at her er det. Nå gjør vi noe med teksten, men når jeg tenker meg om så er det med at du har linjer under her, det er en god ting altså.

R: Jeg lurte litt på om det er noe du ser som trengs eller må forbedres for at du eller eventuelt kollegaene dine skal kunne bruke det systemet slik det er tenkt å brukes nå?

P3: Altså jeg hadde jo, hvis jeg skulle laget Python eksamen og hadde lyst til å bruke en sånn oppgave, så hadde jeg brukt grensesnittet ditt, nå. Selv med de feilene det har, fordi det er bedre enn alternativet, definitivt bedre enn alternativet.

R: Takk

P3: Men det er det.

R: Hadde jo håpet at det skulle bli sånn, så jeg er glad for å ha fått det til.

P3: Nei artig, det må ha vært et morsomt prosjekt for du har fått prøvd deg meg mye sånn React og sånt også. Kult da.

R: og Kotlin backend, og ja det har vært gøy

P3: Kotlin aa, nettopp. Nei vi snakker jo om vi skulle, mobber annen faglærer litt om vi skulle gått til Kotlin i objektorientering kurset, men jeg tror ikke det blir hahaha.

R: Haha, jeg likte det veldig godt. Håper jeg får jobbe med det når jeg begynner i fast jobb.

P3: Har du fått deg eller?

R: Ja

P3: Ja, fikk vel for et år siden er det ikke det de fleste har gjort

R: Ja det stemmer ganske bra det

P3: Begynner med fjernjobbing med en gang.. Hvor er det du skal jobbe da?

R: Hos selskap.

P3: Aaahh, i

R: Oslo

P3: Ahh, i selskap i Oslo ja

R: Ja

P3: Jo jeg kjenner jo litt folk i selskap, studerte med en mens jeg forsket i annet selskap, smidig programvareutvikling, fornavn etternavn, har du hatt noe kontakt med hen?

R: Nei tror ikke det

P3: Nei samma det

R: Da har vi vært igjennom de fleste spørsmål og ting jeg lurer på, har fått svar på veldig mye og mye ekstra. Jeg lurer på om det er noe ekstra du har å tilføye?

P3: Nei, egentlig ikke. Nei. Rett og slett

R: Veldig bra, takk for hjelpen.

P3: Jeg har ikke noe mer å legge til altså, jeg må bare prøve en liten ting. (prøver å croppe fra en annen linje enn den som er markert i Artefact)

R: Vet ikke hvorfor du fikk den feilmeldingen, men du fikk jo i hvert fall en feilmelding.

P3: “You can only split the text once per line”, ja for det var feil.

R: Det finnes en annen feilmelding hvis du prøver å, (B markerer en gang til på en linje som allerede har gap), ja hvis du gjør sånn, den skal komme opp hvis du gjør det. Hvis du nå trykker crop ikonet på en annen linje så skal du få beskjed om at du har trykket feil ikon

P3: (trykker og får riktig feilmelding) Ja, men grunnen er at det første du gjør når knappen blir trykket inn er at du sjekker om det finnes fra før, hvilken event er det som dukker opp først.

R: Ja det vel en logisk brist i if/else løkken min der.

P3: Ja, for den dukker opp hvis du trykker på en tom, for da kicker ikke den der inn

R: Selvfølgelig

P3: Ja, hehehe. Jepp, det er morsomt

R: Tusen takk for hjelpen.

P3: Du vet du hva det skulle bare mangle, jeg synes det er så bra at veileder gir dere sånne, eller ville valgt den helt selv også, at du har jobbet med en sånn oppgave som kan gi oss noe. Rett og slett. Så tusen takk skal du ha.

R: Kan gi dere noe og så kanskje studentene får bedre oppgaver det er jo win-win.

P3: Absolutt. Kult!

R: Du får si ifra hvis du vil ha en kopi av programmet mitt så er det bare å bruke det.

P3: Jaaa, nettopp, du har det ikke liggende på nettaddressen i all evigheter? hehe

R: Det hender jeg gjør andre ting med serveren min, men det blir sikkert det i forseelig fremtid.

P3: Hehehehe. Nettadresse.no, du verden. Var den ledig?

R: Ja

P3: Ja men da så, jeg fikk knabbet etternavnet mitt, etternavn, men inntil videre brukes det bare som Minecraft-server hahahahha.

R: Det er bra bruk det da, jeg prøvde å ta mitt etternavn, men det var det noen som hadde tatt allerede, da måtte det bli .net istedenfor

P3: Nei det funker ikke, det blir ikke det samme

R: Nei det blir ikke det

P3: Nei, men vet du hva, det var artig å være med.

R: Det er godt du synes det. Tusen takk.

P3: Da får vi bare prates etterhvert hvis det skulle være noe.

R: Jada, hvis jeg rekker å forbedre mye på magisk vis i Mai så kanskje du får teste det på nytt

P3: Jaja, helt klart. Okey, greit du!

R: Ha en fortsatt fin mandag

P3: Takk det samme, ha det så godt!

C.6 Transcript Participant 4

R: Hva er dine tanker og meninger om det nye systemet jeg har laget?

P4: jeg skjønner jo at noen vil ha en enklere applikasjon, fordi ja, hvertfall når det er hvis man skal lage en gap oppgave så synes jeg jo din var mere spot on da. Det var jo marker og trykk og that's it. Det kan vel egentlig ikke bli enklere enn det. Jeg husker ikke hvor godt de gappene var markert der når man trykket klipp ut, det hadde kanskje vært greit om det var på en måte et passe stort gap. Ja, eller skjønner du hva jeg mener?

R: Ja, at det blir synlig gap der hvor du klippet ut i teksten?

P4: Ja, det tror jeg hadde vært fint

R: Det blir det altså.

P4: Ja okey. For grunn til at jeg reagerte på det var at de gappene i Inspera ble voldsomme, da var dine litt mer smooth i hvert fall.

R: Ja det skjønner jeg ikke, det kom en endring i Inspera i løpet av de siste to ukene som gjorde at det plutselig ble sånn. Det var ikke sånn for to uker siden.

P4: Det må jo være en bug

R: Det tror jeg også

P4: Jeg tror ikke de har lagt opp til at den skal være så stygg

R: De var ikke det før som sagt, men samtidig så ble det lettere å lime inn kode, det gikk ikke før for da forsvant indenteringen. Så tydeligvis kan man ikke få alt

P4: Ja, for jeg skjønte ikke helt. For andre plasser i Inspera kan man velge kode og da blir det både fargelegging og automatisk innrykk og sånt, det skulle jeg jo gjerne ha sett på den her også, at det faktisk så ut som Python kode i en fargerik editor. Det hadde vel ikke du heller?

R: Nei det går jo ikke ettersom jeg er begrenset av de samme muligheten som finnes i Inspera egentlig

P4: Jo men jeg tenkte på når jeg pasta koden i systemet ditt.

R: Joda, der ble koden indentert og formatter, og highlightet.

P4: (navigerer til systemet)

R: Så den har mulighet til formatering, du må bare trykk add task først. Har mulighet for å velge andre kodespråk enn Python også og få formatering.

P4: Ja så nettopp. Det synes jeg jo kanskje Inspera skulle gitt oss også, for det betyr mye i lesbarhet.

R: Helt enig i det. Hva synes du om var positivt og negativ med å lage den oppgaven i Inspera da?

P4: Det har jeg sagt litt om, det som var positivt er jo at det er i prøven og du kan trykk øyet og det blir akkurat som det ser ut for studentene. Det negative var jo manglene da, ikke syntax highlight, ikke fornuftig størrelse på gaps, identifikatorer som jeg ikke skjønner hvorfor må være synlig. Kunne ikke tenkt meg at det blir noe annet en trøbbel om noen begynner å rote med de på egenhånd.

R: Ja det har skjedd. Synes du også det krevde mange “brukeraksjoner” for å få gjort ting i Inspera.

P4: Jeg fikk jo inn “fingertrykkingen” etterhvert.

R: Ja du fant et bra system ganske fort

P4: Men det er jo mye klikking da, det er jo det. Og så irriterte det meg at jeg måtte trykk to ganger på det gappet før den faktisk ble valgt. Det må også være en bug, jeg kan ikke skjønne noe annet enn det.

R: Tja, det virker ikke som om den automatisk highlighter gappen man akkurat har laget. Veldig rart.

P4: Ledende spørsmål, men ja

R: Med mitt system da, hva synes du var positivt og negativt med å lage Inline-Gap-Match oppgaven med det?

P4: Jeg snakket om det også, syntax. Jeg reagerte ikke på at det var der, men det betyr jo at det funker. Altså jeg reagerte jo på det når det ikke var der (i Inspera). Gap som jeg ikke husker hvor synlig faktisk var. (lager gap i step two) Den kunne godt vært litt mindre lysegrå den stipla linja der (som viser gappet). Og så er det jo utrolig enkelt da, for hele forklaringen står jo der. “Highlight text/code to be cut and press the crop icon”, hvis du i tillegg skriver det med, jo, det kan jo ikke bli enklere enn det. Så det var jo veldig enkelt å lage gaps faktisk. Vanligvis så bruker man saks da, når man klipper ut ting som skal limes inn et annet sted. Kanskje en saks er enda mer

R: Jeg tror jeg prøvde å finne en saks, men dette er Google sine ikoner og de har tydeligvis byttet ut saks med den greia der (crop-ikonet), men jeg kan jo alltids finne en saks

R: Ser du noe direkte fordeler og ulemper med Inspera vs mitt system eller motsatt? Du nevnte det med preview i hvert fall, at du kan se direkte hvordan oppgaven blir seende ut i Inspera

P4: Ja, men det har du jo. Ja nettopp, du har jo prøvd på det her, det var jo nok til at jeg så at jeg manglet en distraktor, bare ved å telle 6 i forhold til 7. Det stod ikke at det her var en preview da kanskje, så

R: Nei, men det er jo laget for å ligne på det du eventuelt får ut i Inspera selvfølgelig, men det står ikke spesifikt nei.

R: Hva synes du om brukergrensesnittet her da som jeg har laget?

P4: Det funker jo, det er jo få trykk. Knappen nevnte jeg vel, det har du notert (Export Tasks knappen ser ikke ut som en knapp)

R: Det har jeg vet du

P4: Den knappen må nesten se, der har jeg blitt solgt av Inspera, alt som lager ting er merket med rødt og alt annet er ikke merket i rødt. Så back vil typisk ikke være rød i inspera, mens både add og generer er typisk røde.

R: Ja, det er smart

R: Noen andre tilbakemeldinger bortsett fra knappen, så var det greit system på tabs og knapper? Det virket som om du skjønte at det var meningen å gå til step two, for å si det sånn, selv om det også dukker opp en next knapp på det første viewet etter at man har skrevet inn litt kode.

P4: Det tenkte ikke jeg på.

R: Kanskje litt langt unna

P4: Ja altså den add task, er det universell utforming man kaller det, at det skal vel helst være like enkelt på mobilen og. Det her funker sikkert på mobilen også tenker jeg

R: Det gjør faktisk det, til en viss grad

P4: Ja, eller tab (tablet), men på en stor skjerm så blir det veldig sånn i hjørnene.

R: Ja, den funker helt OK på mobil og hvis man hadde fått den menyen der til å poppe inn og ut (task-menyen) så hadde den fungert helt perfekt på mobil. Det er ikke så mye som skal til, men det er ikke laget for det med vilje foreløpig

P4: Ja, det tror jeg er mhm

R: Er det noen ting som måtte endres eller forbedres eller legges til slik at det her systemet skulle kunne bli brukt av deg eller andre som har ansvar for å lage sånn type oppgaver?

P4: Det jeg satt og tenkte på var URLen da. En ting er jo, og så next, click (export task). Og så er det jo nyttig at man får den her forklaringen her for her er det jo en del ting man må igjennom også. Den der for eksempel at man trykker på pil ned (den lille pilen i den lille knappen i Inspera for å importere QTI), for meg så virker det, man skal først create new, skal man det?

R: Det var kanskje litt misvisende ja.

P4: Nei først pil ja, nei nettopp, så den burde jo også stått, hvis alle hadde stått til venstre kanskje sånn som eneren (tallene som indikerer rekkefølge på hva som skal trykkes for å laste opp i Inspera).

R: Ja det der er jo et problem i Inspera, da jeg skulle finne knappen først så tok det litt tid før jeg skjønte at man i det hele tatt skulle trykke på den lille pilen, men det kan jeg jo helt klart forklare bedre i mitt system, det er bare å flytte de tallene og så kanskje legge på litt piler og sånt det kan jo hjelpe.

P4: Ja, minst mulig eller okey, her må du ha pil for du kan ikke sette rød på rød (indikerer på step 3.). Enten til venstre for eller oppå ja. (Flytte tallene). Men når det, den genererte ZIPen fint og den gikk jo rett inn i Inspera fint. Så det var vel også laget den veien så kort som den kan bli, tror jeg. Det er jo bra.

R: Er det noe som mangler for at du skulle kunne brukt det her for å lage en sånn type eksamensoppgave?

P4: For det andre intervjuet jeg var i der laget, hva var det gikk på, han drev jo å da, ja det var jo mer sånn han parset oppgaven og så kunne man si at her skal det være en betingelse kanskje og så vart det laget forskjellige oppgaver da hvor noen fikk, nettopp. Så fokuset der var mer sånn at hver student får en unik oppgave, men malen er lik for alle. Her sånn får alle studentene samme oppgave.

R: Ja, det gjør de, men rekkefølgen på gapsene det er det eneste som blir random da. Så det blir jo ikke veldig unikt for de skal jo plasseres på samme sted.

P4: Så det er litt lettere å jukse på den her nå da siden alle sitter hjemme nå da, bare screenshot til bestevennen.

R: Så du skulle gjerne sett at det var litt mer mulighet for å automatisk gjøre det mer random?

P4: Ja, kanskje det. Det var i hvert fall det han andre studenten kikket på, det ble jo plutselig litt akutt viktig akkurat nå i år da.

R: Plutselig veldig relevant. Smart, da var han tidlig ut

P4: Hehe

P4: Men spørsmålet var om jeg kunne bruke det, og det tror jeg jo at jeg kan så lenge URLen er oppe.

R: Ja meningen var jo egentlig at dere skulle fått programmet i en .jar fil som kjøres bare for sikkerhetens skyld så kjører man det bare lokalt. Men det viste seg at det var ikke alle som hadde java og sånne ting, så jeg la den bare opp på serveren min.

P4: Ja sånn ja, det er jo veldig godt poeng at det her er jo ting man kanskje ikke vil dele med en tilfeldig student.

R: Sånn som det er nå så lagres jo ingenting, det eneste som lagres er hvis du eksporterer ZIP-filen, hvis du glemmer å gjøre det så forsvinner alt.

P4: Så hvis du glemmer å eksportere ZIP-filen så ligger det en kopi av den ett sted?

R: Ja hvis du glemmer å slette den ja, fra systemet ditt så.

P4: Ja sånn ja

R: Den lagres ikke på serveren eller noe som helst

P4: Så det vil si at hvis jeg glemmer eller blir avbrutt så kan jeg kanskje ikke nødvendigvis komme tilbake hit i morgen

R: Nei det kan du ikke som det er nå

P4: Nei og det er jo bra fra et sikkerhetsperspektiv, men ikke hvis du har brukt masse tid på å lage gaps

R: Men det er bare en ekstra kodelinje for meg så blir det lagret som localstate, men jeg har ikke valgt å ta det med foreløpig. Det finnes jo ingen måte nå å angre på om du har gjort noe feil.

P4: Åja, jeg kan ikke sette tilbake dette her (viser til en gap i artefact) i koden nei.

R: Nei, så det er mindre mulighet for å gjøre feil enn i Inspera, der kan du prøve på nytt, her må du gå tilbake og gjøre en liten endring i koden for å kunne endre i gapsene igjen

P4: Så hvis jeg gjør sånn (legger inn ny kode i step one, og navigerer til step two), da nullstiller det seg, alle gaps da.

R: Ja det gjør den.

P4: Er det vanskelig å, hvis du har både saks og pil tilbake?

R: Neida, det er nok ikke så vanskelig å legge til mer funksjonalitet. Det er jo også derfor den sidebaren er det for at du skal kunne legge til flere tasks og også andre task-typer. Det er ikke vanskelig å gjøre det her med, hva heter det, dropdown-meny oppgaver og fyll-inn-tekst oppgaver. Det er meningen at man skal kunne generere det ganske enkelt med det her også, og også da lagre underveis og så til slutt eksportere flere tasks, det er derfor det står export tasks. Så man kan si at interfacet er klar for å utvides med mye funksjonalitet.

P4: Da er man på det prinsipielle igjen da om man vil ha det i Inspera eller om man vil gjøre det her. For da altså, hver gang Inspera gjør en liten ting så må også det systemet her oppdateres kanskje. Eller er det hver gang QTI folka finner på noe nytt bare

R: Det skal jo være det QTI formatet da, men nå har de jo oppdatert det fra 2.1 til 2.2 i løpet av oppgaven min og det var en veldig minimal endring for meg å gjøre, men det er jo ting som må stemme ja, det er klart. Det er derfor jeg har valgt å separere det litt. Den frontenden her lager bare sånne tasks-objekter og så er det backenden som tar seg av QTI-formatering. Så da de endret det så trengte jeg ikke å endre noe i det du ser, men jeg måtte endre litt i genereringen av QTI-filen bare

P4: Jeg vet ikke hvor ofte man trenger undo jeg, men det er jo når du sier det utrolig nyttig funksjon

R: Men meningen er jo bare å teste om det å kunne lage en sånn enkel oppgave, om det er nyttig og det er noe man ser nytteverdien av og om man eventuelt vil utvide det senere med all funksjonalitet som da gjør det enda bedre. Og det er jo positivt at du sier du kunne brukt det, det sier jo at det er visst behov for det i hvert fall

P4: Ja, jeg vil tro hvis jeg skulle hatt noen sånne her oppgaver, så hadde jeg kanskje satt mer pris på muligheten for å kunne lage dem i farge editor med enkel trykking (artefact), at det kanskje er verdt mer enn kostnaden ved å måtte laste opp QTI

R: Man vil jo også få mer gevinst når man etterhvert lager flere oppgaver og laster opp flere samtidig

P4: Og så lurer jeg på i Inspera, vi snakket jo om undo. Hvis man først har kommet hit ett sted (en ferdig oppgave som er lagret eller forhåndsvist), hvordan angrer man her?

R: Da må du slette boksen og lime inn koden på nytt, så det er vel ikke akkurat angreknapp, jeg vet ikke om Ctrl + Z fungerer når man har previewet allerede, men det er mer mulihget for å manipulerer det som blir stående enn det jeg har foreløpig. Du kan slette de gapsene og så skrive inn den vanlige koden igjen

P4: Ikke sant så hvis jeg skulle laget det her (fjernet et gap og satt koden tilbake). Hvordan fjerner jeg den boksen her (gappet)

R: Trykker backspace er det vel eller delete på tastaturet

P4: Tydeligvis ikke

R: Huhh, jeg får det til med backspace

P4: Men det kan jo hende at når man først har kjørt preview at man

R: At det låses litt og lagres? Så kanskje det ikke er så mye bedre i Inspera likevel

P4: Nei det var jo det jeg tenkte jeg skulle finne ut av da. Det her virker jo litt “fånatta” da. Hvis man markerer da (får slettet når gap markeres “over”). Ja, det var ikke enkelt å bli kvitt den, da forsvant det jo mye mer igjen.

R: Da har jeg egentlig fått svar på ganske mye, er det noen flere ting du ønsker å nevne, om du har noen tanker eller forslag eller meninger?

P4: (gjesper) Er det noe jeg ikke har sagt? Ja, hvordan blir det med kommersialisering skulle jeg til å si. Hvor lenge lever systemet videre.

R: Det aner jeg ingenting om, det lever vel så lenge jeg holder på med masteroppgaven min frem til 2. Juni, så får vi se. Men hvis du vil ha en kopi av programmer så må du bare si ifra og veileder håper jeg har lyst til å ta det videre hvis det går bra. Dette er jo allerede en videreførelse av en oppgave veileder hadde i fjor.

P4: Ja nettopp, send gjerne en kopi av .jar med en kort tutorial

R: Ja, det kan jeg gjøre. Som .jar'en er akkurat nå så fungerer det ganske bra. Så da sender jeg bare det er bare java -jar snapshot så kjører den

P4: Da kjører den lokalt?

R: Da kjører den på localhost:8080

P4: Ja

R: Det kan du spesifisere også i java -jar kommandoen også hvilken port den skal kjøre på, du kan endre 8080 til noe annet

P4: Bra, kult! Veldig bra!

R: Tusen takk for at du deltok!

P4: Jo, takk for demo, lykke til med innspurt.

R: Jo takk, ha en fortsatt fin dag.

P4: I lige måte, ha det godt du

R: Ha det

C.7 Transcript Participant 5

R: Nå på slutten så har jeg satt av tid til et slags intervju, men vi kan også bare snakke løst om hva du synes om oppgaven, programmet og sånne ting

P5: Jeg synes det var gøy jeg

R: Hvilken del var gøy?

P5: Nei altså det å lage oppgaven var morsommere i din

R: Det var gøy å bare kunne trykke på ting?

P5: Ja altså egentlig, det var litt mere intuitivt. Jeg føler at Inspera har prøvd å lage det så mest kronglete som det går ann da. Det var så mye knapper som ikke var der de burde være i Inspera som gjorde at jeg vet ikke helt hva, de har jo åpenbart ikke ansatt en designer i det hele tatt. Det her er det utviklere som har laget et utvikler system. Så det var litt bedre å komme over til ditt system. Litt mere intuitivt. Det er en del forbedring her også. Det er ikke noen designer som har vært her, hehe.

R: Nei, haha, det er bare meg

P5: Men altså jeg tror det vil hjelpe for folk flest da, å ha en sånn step-by-step funksjonalitet sånn som det var nå (viser Artefact) spesielt der den faktisk formaterer (koden) etter det du limer inn.

R: Det gjør jo Inspera også, men bare av og til.

P5: Det inne i Inspera var veldig, når man limer inn noe så limte man det inn i et vindu, i vinduet, i vinduet.

R: Du tenker på gapen?

P5: Det var på en måte den hvite boksen, og så hadde du en grå boks, og så opprettet jeg koden og da var det en ny boksen. Her (i Artefact) skal du ikke håndtere det i det hele tatt.

R: Man må jo da senere gå inn i Inspera å legge inn oppgavetekst, og det kan jo være en negativ ting da

P5: Ja det vet jeg ikke

R: Hvilken effekt synes du det nye systemet hadde på produktiviteten i forbindelse med å lage selve oppgaven?

P5: Jeg følte jo at det gikk mye fortere. Det var liksom sånn, det var et klikk på en måte, som regel istedenfor to eller tre-fire det var i å for seg på Inspera. For eksempel det når du lagde alternativer, det var superenkelt og alternativet ble laget med en gang basert på hva du klippet ut. Super enkelt, det sparte jo 4-5 klikk og masse skriving. Så det var dritnice.

R: Er det noen ting du ser som er bedre eller mer en fordel med å lage oppgaver i Inspera over mitt system?

P5: Det er vel bare det du nevnte at du kan lage oppgaveteksten og gjøre alt samme sted da.

R: Veldig ledende spørsmål i sted kanskje, alt på samme sted er jo en god egenskap da

P5: Det kan jo fort bli litt tungvint å drive å måtte bytte mellom to programmer, når du uansett må mest sannsynlig bare for å finne oppgaveteksten altså du lager jo ikke oppgaveteksten den har man jo som regel laget før og så bytter man ut ting. Altså sånn man bare velger hva som skal være svaralternativer da, så da ender du opp med å være innom tre-fire programmer da istedenfor bare to da.

R: Hva opplevde du brukergrensesnittet i det nye systemet jeg har laget?

P5: Altså det er jo som jeg sier da, flyten i først i starten der, den er litt sånn vanskelig å forstå. Jeg vet ikke om det er fordi jeg sitter med sollys bakfra at det er litt dårlig skjerm her, men som jeg sa da at “paste or write your code here”, av en eller annen grunn forventet jeg at det var et input-felt.

R: Det er litt kjipt siden jeg har fått tilbakemelding på det før at det burde stått below, men jeg kan jo ikke endre på det i mellom testene nå da

P5: Nei

R: Det er helt greit at du kommer med kritiske tilbakemeldinger, jeg setter pris på det. Jeg har jo ikke hatt noen til å vurdere objektivt. Jeg har jo bare smekket sammen for å si det sånn

P5: Altså det ser veldig bra, det kommer veldig mye i gang da når du på en måte har kommet i gang og fått lagt inn ting, så ser det ganske bra ut, men sånn før man på en måte ha lagt inn kode så er det sånn det er vanskelig å vite at man skulle legge inn kode her. Man kunne enten brukt placeholder, og den burde vært i fokus, det antar jeg du vet hva menes med.

R: Jajaja. Ja det kunne vært noe placeholder, noe som “paste your code here”

P5: Et eller annet. Eventuelt at den er i fokus helt og blinker gult sånn at du vet at det her er et tekstfelt.

R: Det var smarte tips

P5: Videre så er det ikke så mye annet å si på akkurat den delen. Next knappen burde kanskje være der hele tiden, eller kanskje være grået ut. Jeg vet ikke om den var der hele tiden, men jeg følte at det bare poppet opp plutselig.

R: Ja den poppet opp plutselig. Det har du rett i

P5: Mange ganger kan det være mer behagelig å gjøre alt i samme view, men her så forstår jeg at du hadde to da. Siden du på en måte prosesserer dataene i mellom.

P5: Den delen her var veldig grei (step two, markere og croppe gaps). Det eneste var at man sleit litt med å forstå bare highlight tekst. Fordi det er ikke noe som er intuitivt for folk å gjøre enda.

R: Åja, fordi det står som plain-text på en vanlig nettside, man forventer ikke at man skal gjøre noe med den?

P5: Nei, det er ingenting. Det er sånn det å markere den det gjør jeg hvis jeg skal kopiere ting, jeg forventet ikke at det skulle fungere å bare markere å trykke.

R: har du noen forslag til hvordan det kunne vært gjort mere synlig?

P5: Jeg vet egentlig ikke helt

R: Nei det er det jeg og veileder kom frem til, at vi vet egentlig ikke helt

P5: Eneste er eventuelt å legge til et bilde-eksempel, det er det eneste jeg ser for meg

R: Ja det kan jo være lurt. Det er masse plass

P5: Jeg hadde en kjapp ting ting til. Man kan ikke gjøre noe tilbakesteg.

R: Nei, det er ingen angre-knapp

P5: Så ting er liksom fixed.

R: Ja det er det, tanken er litt det at det ikke så viktig å få til siden det er såpass kjapt, så hvis du gjør en feil så er det bare å gjøre en endring i koden og få det tilbake

P5: Det er veldig enkelt å gjøre en feil her å da

R: Ja det er fort gjort å markere forbi eller for langt eller markere med knappen og sånt

P5: Ellers så synes jeg det er ganske kult. Export tasks er av en eller annen grunn i hodet mitt, skal den aldri være oppe i venstre hjørne.
R: Si du skulle laget en sånn type programmeringsoppgave. Hva mangler eller hva må forbedres for at du skal kunne bruke programmet mitt for å lage en sånn oppgave?

P5: Jeg tror ikke det så mye som må forbedres, jeg skal være helt ærlig der. Bare litt mere forklaringer og litt bedre forklaringer hvis det er mulig å si det sånn. Og egentlig muligheten til å angre enkelt da. Nå vet jeg ikke hvordan det er i Inspera per dags dato da, det prøvde jeg ikke

R: Nei man kan i hvert fall redigere litt

P5: For det her blir man veldig låst da

R: Ja man gjør det, man kan ikke endre seg og ombestemme seg underveis, man må nesten vite på forhånd hva man skal lage ja

P5: Mhm

R: Er det noen ekstra features du skulle ønske at systemet hadde, noe som hadde vært kult eller nyttig?

P5: Altså det som hadde vært litt kult tror jeg hadde vært å kunne lage oppgavesett.

R: Mhm

P5: Sånn at man ikke må inn å eksportere ett og ett, fordi det kan bli litt irriterende

R: Man skal jo egentlig kunne gjøre det, det er jo derfor knappen heter Export Tasks og man kan legge til flere tasks, men det fungerer jo ikke foreløpig, men alt er klart til at det skal gå an

P5: Nei men det tror jeg kunne vært hovedgreia

R: Så egentlig bare å kunne lage flere på en gang? Av samme type eller andre oppgaver også, hadde det vært noe?

P5: Altså, det beste hadde jo vært om systemet her (artefact) var noe sånt Inspera brukte i utgangspunktet, du vil jo egentlig ikke, altså du vil jo lage alle oppgaver samme sted, og aller helst i Inspera.

R: Så en kul egengenskap hadde vært om mitt system var inkludert i Inspera når man skulle lage en sånn type oppgave?

P5: Ja

R: Skjønner
P5: Ja, fordi du vil ikke bruke to sider for å lage en ting

R: Greit, da er vi nesten på slutten da, har du noen andre forslag, tanker eller meninger du vil dele, forbindelse med det nye systemet, eller oppgavelagingen i Inspera?

P5: Nei altså det er jo bare det du sa selv, om at det burde vært noe placeholder i utklipps-feltene (gaps i artefact). Det var veldig kult å dra sånn (viser markering og kutting av gaps)

R: Hehe. Nei, men takk for at du ble med på test opplegget.

P5: Det var gøy, jeg likte veldig godt den der, det var veldig lite intuitivt, men det er veldig gøy når du får brukt den der markeringen

R: Takk igjen for at du deltok!

P5: Ikke noe problem

C.8 Transcript Participant 6

R: Hva var positivt og negativt med å lage den oppgaven i Inspera?

P6: synes ikke det var så, altså det er jo kult at de har den funksjonaliteten da, den type oppgave er jo bra, og det er bra at man kan lage den digitalt and all that. Jeg synes det var veldig uoversiktlig, selv default teksten som var der så var det uklar for meg om det var teksten på oppgaven eller der du skulle gjøre (noe). Nå tror jeg har skjønt det sånt at jeg måtte være inne i den boksen for at gaps og sånt skulle fungere.

R: Så den hjelpeteksten som dukket opp ved en ny oppgave vår ikke egentlig så veldig hjelpsom?

P6: Den var kanskje det når jeg først skjønte den. Jeg tror det ville vært verre uten den, men den hjalp ikke helt på en måte.

R: Skjønner

P6: Jeg synes det var veldig cumbersome i Inspera å trykke og lage, og det at det ikke var formatert riktig var rett og slett veldig frustrerende

R: Ja, noen ganger så nekter den å beholde formateringen av koden også, så noen som har testet har hatt litt uflaks og det har rett og slett ikke være mulig

P6: Ja det var jo en gang da jeg kopierte det fra utsiden og inn i boksen og da forsvant alt

R: Da forsvant formatteringen ja. Tab fungerer heller ikke

P6: Det fikk jeg med meg ja da jeg prøvde det en gang. Gaps kan ikke fjernes med backspace, men jeg kan markere de og så bruke backspace. Så jeg kan ikke trykke på de og så trykke backspace

R: Med mitt system da, hva synes du var positivt og negativt med å lage oppgave i det?

P6: Jeg synes det var veldig mye lettere å gjøre liksom the core loop av det du skal gjøre da. Det var veldig naturlig å markere noe og trykk på den knappen og så ble liksom det som var forventet ble gjort. Det tok meg jo liksom to sekunder å lage oppgaven som tok meg ett kvarter i Inspera virket det som.

R: Du følte det sånn?

P6: Ja mesteparten av tiden jeg brukte var brukt på å se på der du hadde sagt, fasiten liksom, hvilke biter. Altså finne hvilke seksjoner av teksten som skulle bort

R: Ja

P6: Det var det som tok tid, interaksjonen med systemet var veldig bra. Men når det er sagt så er det meget dumt at det ikke er noen back-mulighet for å krysse ut noen av de her (viser lagde gaps)

R: Ingen angre mulighet?

P6: Ja jeg for jeg kan forestille meg at hvis jeg skulle lage mer en en oppgave hvis jeg ikke hadde det veldig tydelig, en ting var jo det vi snakket om i sted hvis jeg skal leke meg i det UIet for design oppgaven, men det andre er også bare hvis jeg skal lage mange av de her så ville jeg forestille meg at jeg kommer til å gjøre feil ganske regelmessig, og da må man kunne backe upp. Okey, du sa at jeg kunne redigere den teksten her og så var alt tilbake (tilbake i step one), men jeg vil mistenke at de fleste vil ikke tenke det, og da finner folk forskjellige måter å resette på og den naturligste hardeste reseten er å slette hele oppgaven og begynne på nytt, og hvis du da allerede har gjort ganske mye så er det veldig frustrerende.

R: Ja, man kan vel egentlig si det sånn at den oppgavene som er gitt gir et visst bruksmønster som ikke inneholder muligheten til å utforske alle normale hendelser som kan veldig sannsynlig komme til å skje når man bruker det programmet da, litt lenger

P6: Mhm, men selv i den oppgaven her så gjorde jeg jo en feil på en måte, og da ønsker jeg gå tilbake, så ser jeg også at det er ikke noen måte å fjerne distraktoren på heller. Men det var veldig satisfying, det her synes jeg var gøy liksom (viser å markere og lage gap i Artefact). Jeg kunne gjerne tenke meg å bruke dette for å lage små quizer i “fag” med dette liksom, ganske regelmessig

R: Det er innenfor et av spørsmålene så vi kan ta den litt videre, hva skal til eller hva mangler for at du skal kunne bruke det systemet her for å kunne lage sånne type oppgaver?

P6: Den ene tingen er jo at den kan åpenbart importeres i Inspera da, men hvis jeg skulle bruke det bare i “fag” liksom, jeg kunne forestille meg at det kunne være nyttig å introdusere i kurs og i øvingsforelesninger og sånne ting og da vil jeg bare, da måtte jeg ha et brukergrensesnitt for brukeren da som ikke var Inspera, en måte å la folk svare på oppgaven

R: Har de ikke det da? Enten BlackBoard eller Inspera, går det ikke an å bruke det i fagene?

P6: Jojojo, jeg bare vet ikke hvor lett det er for meg å sette opp Inspera, med en oppgave i Inspera som de kan bruke som ikke er en eksamen, det har jeg ingen peiling på

R: Det vet ikke jeg heller, men jeg tror det skal gå an. Hvis ikke så skal denne oppgaven sannsynligvis gå an å importeres til BlackBoard også fordi de også bruker QTI formatet så vidt jeg vet

P6: Ja, okey, det hadde jo vært smud liksom, det er den ene tingen. Altså så klart jeg som ikke er en som lager eksamener, men kunne jeg brukt den enkelt ville jeg brukt den regelmessig tror jeg. Den aller viktigste tingen tror jeg er det med å kunne undoe. Det hadde vært nice å se, det hadde vært smud å ha farge formatering og sånt mens jeg, hvis jeg kunne gjort det i den versjonen her liksom (i step one kode input field) og bare hatt de x-ene (crop-ikon) her liksom (på høyre side av koden) mens det ser sånn her ut. Greit nok at den lastes eller lagres eller noe sånt noe, men formateringen hadde vært smud.

R: Det gjør ikke noe at linje skillene ikke blir like tydelig?

P6: Jeg synes ikke at linje skillene gjør noe her ass, for meg så føles det ut som om de er, jeg har jo ikke sett det uten linje skillene da, men for meg så føles det litt ut som om de er mer støy, enn hva som trengs. Det kommer jo litt an på om teksten kanskje, altså hvis, si at den her var mye lenger (lager en lang linje). Da kunne det hende at du måtte ha noe sånn linje skille greie

R: Okey. Ser du noen fordeler med å lage oppgaven i Inspera i forhold til mitt system?

P6: Ehh, det er vel eneste jeg kan tenke meg er at direkte kobles med Inspera, mest sannsynlig er jo målet ditt å laste opp dette i Inspera, og det slipper du jo i Inspera. Så den delen må jo være så seamless som mulig da, men gitt at den er - så det er den eneste fordelen jeg ser Inspera har, der trenger du ikke trykke for å laste opp oppgaven

R: Så basically at man gjør og er alltid i samme interface?

P6: Ja

R: Da har jeg egentlig fått mange bra tilbakemeldinger og svar, om det ikke er noen flere ting du tenker på, har noen tilbakemeldinger eller innspill, eller forslag til det nye systemet?

P6: Nei jeg tror ikke det

R: Nei, men hvis det ikke var noe mer da så sier jeg tusen takk for at du ble med. Det var veldig nice.

P6: Så jo kjempekult ut da, har du laget det fra bunnen av?

R: Ja jeg har jo det da, eller jeg har jo brukt material komponenter da, men det ser du sikkert, men ja det er jeg som har laget dette og så har jeg en backend i Kotlin som parser det der og lager den QTI-filen som lages og importeres i Inspera

P6: Ja, fett, kult! Kult! Det er stilig system da

R: Ja det har vært gøy å jobbe med, skulle ønske jeg bare kunne fortsette å jobbe med det istedenfor å skrive på rapporten

P6: Ja ikke sant