

MASTER THESIS

A Concept for Improving ICT Tools as Support for
Morning Meetings in the Oil and Gas Industry

Amund Lågbu

01.10.2015

Master in Applied Computer Science
Faculty of Computer Sciences



EGENERKLÆRINGSSKJEMA

Alle studenter plikter å kjenne til hvilke regler som gjelder i tilknytning til bruk av hjelpemidler/kilder til den enkelte eksamen/avsluttende vurdering og andre studentarbeider (f.eks. arbeidskrav), og hva som anses som fusk/forsøk på fusk i tilknytning til bruk av hjelpemidler/kilder. Det vises til universitets- og høyskoleloven § 4-7 (1) og § 4-8 (3) og til forskrift om eksamen og studierett ved Høgskolen i Østfold § 12.

Som fusk regnes uredelige handlinger eller forhold som tar sikte på å skaffe studenten et uberettiget fortrinn ved eksamen eller arbeidskrav. Som fusk regnes også uredelige handlinger eller forhold som tar sikte på å gi studenten uberettiget tilgang til eksamen eller vitnemål.

Det vil foreligge brudd på bestemmelsene om fusk i forbindelse med eksamener og andre studentarbeider (arbeidskrav o.a.) i blant annet følgende tilfelle:

Urettmessig tilgang til oppgavetekst og lignende før gjennomføring av eksamen eller ved andre studentarbeider. Det samme gjelder uretmessig tilgang etter at det skriftlige arbeidet er innlevert.

Bruk eller besittelse av andre hjelpemidler enn det som er oppgitt på eksamensoppgaven etter at eksamen er iverksatt.

Ved hjemmeeksamen og andre skriftlige vurderingsformer vil plagiering og avskrift av faglitteratur og andre skriftlige arbeider uten korrekt bruk av referanser/kilder bli vurdert som fusk.


Innlevering av oppgaver som er helt eller delvis identiske regnes som fusk. Dette kan også gjelde dersom besvarelsen er preget av manglende selvstendighet, svært lik besvarelse som studenten har benyttet ved annen eksamen ved HiØ eller annen utdanningsinstitusjon. Det samme gjelder dersom besvarelsen er svært lik en annen/andre besvarelser, eller den åpenbart er utarbeidet av andre.

Egenerklæringen gjelder eksamen i følgende emne:

Emnekode	Emnenavn	Sett kryss:	
IT140614	Masteroppgave	Individuell eksamen	<input checked="" type="checkbox"/>
		Gruppeeksamen	<input type="checkbox"/>

Ved gruppearbeid er alle i gruppen gjensidig ansvarlig for innholdet i besvarelsen.

Jeg/vi erklærer å være kjent med ovenstående regler for fusk ved Høgskolen i Østfold:


Dato:	Studentens navn (blokkbokstaver):	Signatur:
30/9-18	AMUND LÅGBU	
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:
Dato:	Studentens navn (blokkbokstaver):	Signatur:

Godkjenning av bruk/publisering av masteravhandling Ja Nei

 Godkjenning av bruk/publisering av bacheloroppgave

Etternavn LÅGBU	Fornavn AMUND
Fødselsnr. (11 siffer) 03 05 83 44 361	Studentnummer 022073
Adresse ROALD AMUNDSSENS VEI 33D	Postnr. / Poststed 1654 SELLEBAKK
Telefon 922 23 903	Studium / Avdeling Master i anvendt informatikk

Studium: Master i anvendt informatikk
Tittel på avhandling/oppgave: "A concept for Improving ICT Tools as Support for Morning Meetings in the Oil and Gas Industry"

<ul style="list-style-type: none"> • Undertegnede samtykker i at ovennevnte avhandling/oppgave stilles til rådighet som eksempel for andre studenter ved Høgskolen i Østfold. • Undertegnede er kjent med at oppgaven arkiveres i høgskolens biblioteks magasin etter fagområde. • Undertegnede samtykker i at masteravhandlingen, dersom den får karakter C eller bedre, tilgjengeliggjøres elektronisk i Brage (HiØs åpne institusjonelle arkiv) 		
Dato: 30/9-15	Sted Halden	Studentens underskrift 

Foreword and Acknowledgements

This project started as an idea promoted by senior research scientist at Institute for Energy Technology (IFE), Sizarta Sarshar. Sarshar's suggestion was to examine current software used in morning meetings in the oil and gas industry, and propose changes for better supporting the decision-making process. The thesis is the embodiment of this idea.

IFE has been an important collaborator during the whole study. Different personnel from the organization have participated during the requirement definition phase and in several tests of the developed concept - just to mention a few.

First and foremost I want to thank the project's supervisor and contracting authority, Sizarta Sarshar, for his contributions to the project. He has been involved in all stages of the work, and have guided the development in the right direction. He was also willing to continue this work, although the project stretched out in time, and the right for guidance lapsed.

Both Sarshar and principal scientist at IFE, Grete Rindahl, have also taken part in writing a research article during the project. I am grateful for their help, without it the article wouldn't be realized. Sarshar was also willing to present this work at the ESREL 2015 conference in Zürich, when I did not have the opportunity to participate, something I am highly grateful for.

Further I want to thank my better half, Ann-Helen Ottum, for her limitless patience by looking after our three children in weekends and afternoons while the work was in progress.

Lastly, but not least, I want to thank my brother, Yngve Lågbu, for important guidance during the process of writing the thesis.

Abstract

The main goal of this thesis is to find if software tools may better support decisions in morning meetings, and how. Additionally the thesis aims for studying how such software may promote important elements in the Integrated Operations concept, which is fundamental in modern oil and gas industry.

The need for conducting such a study was also defined in a research article written during the project work. "Improving ICT Tools as Support for Morning Meetings in the Oil and Gas Industry" by Lågbu et al. [18] was published at the ESREL 2015 conference in Zürich. The article describes an early concept based on requirements defined in a workshop, and evaluations of currently available systems and prototypes. The research article is presented in Appendix A.

The process of experimenting and gathering results for answering the research questions, defined in the thesis, have been twofold. The first part of this project revolved around gathering requirements for a future morning meeting system, and to evaluate current solutions based on these findings. The goal of this process was to define some ideal requirements for a software to be used in future morning meetings, and to find if there is a need for such software.

It was concluded that the evaluated systems did not satisfy all the requirements. Therefore a concept was developed in nine iterations. This concept was tested with different techniques, to find if the proposed solution could be used to support decision making in morning meetings. It was also tested if the solution could promote elements in the integrated operations concept.

The results are however ambiguous. While the evaluation of current solutions pointed towards a need for new software, the developed concept is perhaps not the ideal solution. Tests performed of the concept gave inconclusive results, whether required features were at place, or not. But a final usability inspection concluded that the development of software, designed for supporting a work practice for meetings, is unique - making the tool better adapted to current collaboration meetings. The proposed design was therefore considered a step in the right direction for supporting decision making in today's oil and gas industry.

Table of Contents

1	Introduction	13
1.1	Background	13
1.1.1	A Brief Introduction	13
1.1.2	Risk and Cost Challenges	13
1.1.3	Integrated Operations	14
1.1.4	Meetings in Onshore/Offshore Organizations	15
1.1.5	Collaboration Technology in IO Meetings	15
1.2	Problem Statement	16
1.2.1	Research Questions	16
1.2.2	The Research Questions' Words and Phrases	16
1.2.3	Scope and Limitations	17
1.2.4	Brief Overview of Research Method	17
1.2.5	Hypothesis	18
1.3	Project Structure	19
1.4	List of Abbreviations	20
1.5	Related Literature	21
1.5.1	Morning Meetings in an IO context	21
1.5.2	The Structure and Scope of Collaboration Sessions	23
2	Theory	26
2.1	Decision Support Tools in IO organizations	26
2.1.1	A Study of an Email Platform and Other Decision Tools in the Oil and Gas Industry	26
2.2	Systems Designed for Meetings in the Oil and Gas industry	28
2.2.1	Epsis Teambox	28
2.2.2	ARKit	28
2.2.3	Protosphere	28
2.2.4	Email platform	29
2.2.5	IO-MAP	29
2.2.6	Wisio	30
2.3	System Development Methodology	31
2.3.1	Traditional methods	31
2.3.2	Rapid Application Development	33
2.3.3	Agile methods	35
2.4	Requirement Methodologies	37
2.4.1	Requirement Definition Methodology	37
2.4.2	MoSCoW Prioritization of Requirements	38
2.5	Prototyping Techniques	39

2.5.1	Offline and Online Prototypes	39
2.5.2	Protoyping Strategies	39
2.6	Testing	41
2.6.1	Usability Testing	41
2.6.2	Experts Reviews	41
2.7	Visualization	43
2.7.1	Map Visualizations	43
2.8	User Interface Design	45
2.8.1	The Eight Golden Rules of Interface Design	45
2.8.2	Interaction Styles - Advantages and Disadvantages	46
3	Method	47
3.1	Methodology Selection	47
3.1.1	Development Methodology	47
3.1.2	Requirement Definition Methodology	49
3.1.3	Complete Project Methodology	49
3.2	The initial phase - the analysis	52
3.2.1	MoSCoW Ranking of Requirements	52
3.2.2	Evaluation of Current Systems	52
3.3	Quick design	53
3.4	Prototyping Cycles	54
3.4.1	Prototype Fundamentals	54
3.4.2	Iterative Development	55
3.5	Testing	56
4	Observations and Experiments Prior to Concept Development	57
4.1	The Studied Email Platform	58
4.1.1	Findings from Studying an Email Platform Document	58
4.2	Defining Requirements for Future Software Supporting Morning Meetings	61
4.2.1	Defining Key Functionality	61
4.2.2	Importance of Functionality in the Future System	63
4.2.3	Requirements the Reviewers Defined as Important	64
4.3	Evaluation of Existing Software's Functions	65
4.3.1	Current Implementations of Key Functions	65
4.3.2	Functionality Based on Other Systems' Solutions	66
5	Concept Development	70
5.1	Quick Design - Paper Prototyping	72
5.2	Prototyping Cycles - Horizontal Prototyping	73
5.2.1	Iteration I	73
5.2.2	Iteration II	77
5.2.3	Iteration III	83
5.3	Prototyping Cycles - Vertical Prototyping	87
5.3.1	Iteration IV	87
5.3.2	Iteration V	92
5.3.3	Iteration VI	93
5.3.4	Review	95
5.4	Prototyping Cycles - Adjustments to the Horizontal Prototype	96
5.4.1	Iteration VII	96

5.4.2	Iteration VIII	99
5.4.3	Iteration IX - Final Changes	101
5.5	Requirements and Features Thought Implemented when Developing the Horizontal and Vertical Prototypes	103
6	Concept Evaluation Results	105
6.1	Testing of the Developed Concept	105
6.2	Cognitive Walk-through	107
6.2.1	Pre-meeting tasks	107
6.2.2	In-meeting tasks	108
6.2.3	Post-meeting tasks	109
6.3	Heuristic Evaluation	110
6.3.1	Evaluation of the Concept Based on the Requirements	110
6.3.2	An Evaluation Based on the Eight Golden Rules of Interface Design	112
6.3.3	Comparison of the Reviewers Ratings	113
6.3.4	Usability Inspection	113
7	Discussion	117
7.1	How May Software Promote Overall Decision Support in the Oil and Gas Industry?	118
7.2	Defining Requirements for Decision Support in Morning Meetings	120
7.3	Testing Current Solutions	123
7.4	A Concept Made for Promoting Decision Support in Morning Meetings	125
7.4.1	Using Features Discovered in Other Software	125
7.4.2	Major Changes to the Design	126
7.4.3	Removing Functionality	128
7.4.4	Features not Tested	129
7.5	Testing of the Developed Concept	130
7.5.1	Cognitive Walk-Through	130
7.5.2	Heuristic Evaluation	131
7.5.3	The Design's Ability to Provide Decision Support with Respect to the Test Results	134
8	Conclusion	136
	References	137
A	Research Article Written During the Project Work	140
B	Requirements	150
B.1	Requirements Defined for the Project	150
B.2	Requirements Based on a Company's Guidelines	177
C	Design	186
D	Test of Developed Concept	194
E	Test of Current Systems' Features	231

F XML DTD Schemas	239
F.1 Menu DTD	239
F.2 Map DTD	239

List of Figures

1.1	A redrawn version of the DCS model described by Skjerve et al. [34]	24
1.2	A simplified model of distributed collaboration structures, described by Skjerve et al. [34]	25
1.3	A simplified model made of the main collaboration sessions, described by Skjerve et al. [34]	25
2.1	This illustration shows how a software development methodology may be selected, on the basis of a project's complexity and uncertainty [8].	33
2.2	An illustration of the RAD development process, as described by Mediapshere [20].	34
3.1	An illustration of the combination of the customized requirement definition and the RAD methodology used in this project. [20].	51
5.1	The concept was changed in nine iterations. The gradually evolution of the design is illustrated in this figure. The design in the fourth, fifth and sixth iterations were developed in code.	70
5.2	A screen made for the initial paper prototype.	72
5.3	The initial digital prototype's main screen.	73
5.4	Selection of topics in the initial prototype.	74
5.5	A pop-p window containing participants in a collaboration session.	74
5.6	A pop-up window containing notifications in a collaboration session.	75
5.7	A pop-p window containing a whiteboard.	75
5.8	The second prototype - alternative 1.	77
5.9	The second prototype - alternative 2.	78
5.10	The second prototype - alternative 3.	78
5.11	The second prototype.	78
5.12	The second prototype - collapsed.	79
5.13	Magnifying glasses to compensate for lack of readability control.	79
5.14	Notifications alert symbol.	80
5.15	A pop-up window displaying system settings.	80
5.16	A menu organized as a schedule used during a morning meeting.	81
5.17	Adding documents/whiteboards to specific item in menu.	81
5.18	Functionality for "playing" through a meeting's topics.	82
5.19	A simple map displaying an oil platform, with tasks connected to several topics. Notice that the tasks that are not directly relevant are gray.	84
5.20	Comparison of visualization techniques.	85

5.21	The initial prototype version's user interface.	87
5.22	The interactable menu - version I.	88
5.23	The first version of the functional map visualization.	89
5.24	Adding an installation background to the map surface.	93
5.25	Adding images of other software systems.	93
5.26	Interaction with a map surface in the last, vertical prototype. . .	94
5.27	The overall prototype was changed during vertical work with specific system parts. A collapsible comments window was firstly added.	96
5.28	A new comment option was added to the prototype developed vertically.	97
5.29	This is a slide illustrating how the outer part of the system could be made, for handling several meetings.	98
5.30	This illustration presents the design generated for the meeting application during the eighth prototyping iteration.	100
5.31	An excerpt of the design displaying the new buttons replacing the previous tools menu. In Figure 5.30 this menu is mostly hidden below the "New action"-window.	100
5.32	A slide presenting the report with action and decisions shown without the need of scrolling.	101
5.33	An excerpt of a design slide presenting possibillited for adding a deadline to actions and decisions.	102
5.34	An excerpt of a slide presenting information about reports, without the yellow warning sign used in the previous iteration. . . .	102
6.1	A bar diagram presenting how the reviewers scored the requirements thought to be implemented in the design.	114
6.2	The percentage of points given by each of the reviewers, with regard to the total number of points given in the heuristic evaluation.	114
6.3	Composition of points given by reviewer 1.	115
6.4	Composition of points given by reviewer 2.	115
6.5	Average scores given in the different parts of the heuristics. . . .	116

List of Tables

4.1	Main requirement categories	63
4.2	Implementation of requirement categories in evaluated systems. .	67
4.3	This table presents functionality found in evaluated prototypes and systems, which could be implemented into the future meeting system.	69
5.1	Function Suggestions for External Software	86
5.2	A table presenting functionality thought implemented in the prototypes of the future system.	104
6.1	A table presenting the results of the heuristic evaluation. The reviewers individual answers are shown in the columns to the right. The fulfillment of requirements are ranked on a scale from 1-5, where 1 means not fulfilled, and 5 stands for fulfilled. Additional comments are written in the same table cells.	112
6.2	A table presenting the results of the heuristic evaluation in accordance to fulfillment of the Eight Golden Rules of Interface Design.	113

Chapter 1

Introduction

1.1 Background

1.1.1 A Brief Introduction

Oil production on the Norwegian continental shelf origins from the finding of the Ekofisk reservoir in the autumn of 1969. This was a major breakthrough for the country's oil business, and could be considered the starting point of the Norwegian offshore industry¹.

Oil and gas production in the North Sea has since then continued uninterrupted. Today it is a major industry by Norwegian standards. In 2012 oil exports alone accounted for more than 50 percent of the country's total exports. The industry's revenues have also been saved in a fund worth 5.000 billion NOK (in 2014)². These large impacts on Norwegian economy have nicknamed the offshore industry "the Norwegian oil adventure".

But despite the well-sounding wording, the oil and gas production has not only been a fairy-tale story.

1.1.2 Risk and Cost Challenges

The business has been plagued with several minor accidents and precursors, and some major incidents with several fatalities. The most extensive of these (on the Norwegian continental shelf) happened on 27 of Mars 1980, when the Alexander L. Kielland platform capsized, claiming the lives of 123 people [22].

Precursors and minor accidents, on the other hand, happen on a yearly basis. Vinnem et al. [40] mapped the causes of precursors and accidents in the Norwegian offshore petroleum industry in the period 2003-2008. They found that the most frequent incidents involved vessels on collision course, on average 33 incidents each year. The second most frequent were not ignited hydrocarbon leaks, annually half as many as the most frequent precursor category.

Another challenging aspect regarding the industry has been, and still is, the cost development. This is illustrated in an article published by McKinsey & Company [19] - describing the offshore cost development in the North Sea in

¹<https://www.regjeringen.no/en/topics/energy/oil-and-gas/norways-oil-history-in-5-minutes/id440538/>

²https://snl.no/Norsk_oljehistorie

the period from 2003 to 2013. The authors point out that the annual inflation rates increased much more for the oil and gas industry than the average rise of the consumer price index (CPI) in both Norway and the United Kingdom. They found that annual inflation rates for operational costs, and business expenses, associated with future operations, yearly rose with 10% and 16%, respectively. In comparison CPI increased on average with 1.8% in Norway and 2.7% in the United Kingdom.

According to Bjørnland [6] the increased cost level have made the industry vulnerable to market fluctuations. This has been noticeable since the steep reduction in oil prices following the summer of 2014. At the time of writing the value of European crude oil has declined with more than 50% since then. This development could according to Bjørnland result in further reduced activity in the petroleum industry.

1.1.3 Integrated Operations

For more than a decade a concept called Integrated Operations (IO) have gradually been implemented in the offshore industry. Some of the main goals of IO is to meet the challenges mentioned in Section 1.1.2. This may be supported by the article "Benefits of Integrated Operations"³, published on the Kongsberg Groups's websites. According to the company the introduction of IO has several benefits; reduced operating costs, increased oil and gas recovery, accelerated production, longer life-spans, extended field-life and higher safety levels.

Generally speaking IO may be described as a concept for integrating people, organizations, work processes and information and communication technology (ICT). However, it depends on global access to real time data and collaboration technology - making it possible for all involved parties to communicate across disciplines, organizations and geographical borders.

In the oil and gas industry IO typically entails cooperation between reservoir management, drilling, production optimization, operation and maintenance, logistics and health safety and environment (HSE). In such an organization IO also implies cooperation between organizational parts situated at various geographical locations. Examples may include virtual communication between offshore and onshore personnel, and communication between the oil company and external vendors⁴.

Oil and gas IO organization is therefore a network of offshore installations and onshore support centers communicating with each other and external vendors with the use of collaboration technology.

Norwegian Oil and Gas (NOG)⁵, formerly known as the Norwegian Oil Industry Association (OLF), distinguishes between first and second generation of integrated operations. While the first generation focused on real time collaboration in virtual rooms, and inter alia the establishment of onshore operation centers - supporting offshore operation, the second generation of IO will be more profound. According to NOG this generation will entail much more automation of processes in the IO organization. NOG also foresees that the offshore facilities

³<http://www.kongsberg.com/en/kog/news/featurestories/benefitsofintegratedoperations/>

⁴<http://www.iocenter.no/info/what-integrated-operations>

⁵<http://www.norskoljeoggass.no/PageFiles/14295/070919%20IO%20and%20Ontology%20-%20Brosjyre.pdf?epslanguage=no>

themselves will be radically changed - the traditional offshore platforms will be replaced by intelligent and more self-propelled facilities.

1.1.4 Meetings in Onshore/Offshore Organizations

One of the important components of an IO-environment is plans and meetings to discuss plans. Sarshar et al [30] point out that since the introduction of IO, several planning tasks have been reorganized, and moved to the main land - hence increasing the need for communication between offshore and onshore personnel. Several forms of meetings are therefore conducted to discuss plans, stretching from several years to 24 hours plans.

One kind of meeting, specifically studied in this project, is morning meetings. Ose and Steiro [26] describe these meetings as activities which are held every day and involve both onshore and offshore personnel at individual rigs. All the personnel supporting the rigs are attending the meetings, and communicate by phone or video conferencing.

Usually topics as upcoming work and changes involving already planned work may be discussed in morning meetings. Wahl et al. [41] describe a concrete example where a plan had to be updated in real-time during such a meeting. For this task several decision support tools had to be accessed and used. These tools may therefore be essential as support in morning meetings.

1.1.5 Collaboration Technology in IO Meetings

In a modern IO organization, where meetings have to take place between geographically dispersed personnel, collaboration technology plays an important role. This is emphasized by Albrechtsen [2], who has described the necessary steps to implement IO in an organization, meanwhile achieving a high degree of value creation. The steps are parts of an IO stack containing elements as data capture, communication infrastructure, information access and, among others, information visualization.

Both how and what data that is presented to a meeting's team members could therefore be considered important.

1.2 Problem Statement

This project was given as a task by Institute for Energy technology (IFE) in Halden, and investigates the demands for creating new software specifically designed for activities as morning meetings. Some studies indicate that there may exist needs for such software, or a review of the current solutions. Such findings are mentioned in Section 1.5.

The studied meeting type will involve collaboration between offshore and onshore personnel in virtual rooms. It is important that the prototype system is adapted to this context.

Further the system should be customized for the relevant IO concept. Secondary goals should therefore imply the prototype being made in accordance to important IO elements. Solutions for minimizing risks of accidents and precursors, as well as cost effective choices, should be taken when possible.

The following research questions have been selected:

1.2.1 Research Questions

1. *Can software tools better support decisions in morning meetings in oil and gas organizations, and how?*
2. *How may software, supporting morning meetings, promote important elements of the Integrated Operations concept?*

1.2.2 The Research Questions' Words and Phrases

- *Concept*: A concept is a logical, and not mental, entity ⁶.
- *Decision support*: According to Kaarstad and Rindahl [16] decision support helps people making decisions. Decision systems and decision theories are components of decision support. The decision systems are software made for helping users achieving decision making and to solve problems.
- *Integrated Operations*: According to the Center for Integrated Operations in the Petroleum Industry⁷ Integrated Operations may be summarized as an integration of people, organizations, work processes and ICT to make better decisions.
- *Morning meetings*: In this context morning meetings refers to a specific kind of meeting type held in the oil and gas industry. This activity is described by Ose and Steiro [26] as meetings that are held every day, and involve onshore and offshore personnel at individual rigs - communicating by phone or video conferencing.
- *Oil and gas organizations*: Oil and gas companies may be defined as businesses involving production, exploration, refinement and/or distribution of oil and gas⁸. But in this project's context the phrase is limited to companies directly engaged in the offshore oil and gas production.
- *Software*: Software may be defined as programs that are used by computers and similar products containing logic circuitry⁹.

⁶<http://global.britannica.com/topic/concept>

⁷<http://www.iocenter.no/info/what-integrated-operations>

⁸<http://www.investopedia.com/terms/i/integrated-oil-gas-company.asp>

⁹<http://www.linfo.org/software.html>

1.2.3 Scope and Limitations

The scope of this thesis is not to analyze how all types of meetings are conducted offshore. This would have been an enormous task, demanding much more time and resources than available. The thesis has therefore been limited to one kind of meetings - daily morning meetings between offshore and onshore personnel.

Development and design of a prototype of a support-tool could be considered a timely process. The requirement analysis was therefore formed without observations and interviews with offshore and onshore personnel in an oil company. This would be a task requiring too much time. Instead a concept was developed in close cooperation with available expertise at IFE in Halden. These experts were also interviewed during the development process. Further the student decided to analyze meeting transcripts, and existing meeting systems and prototypes.

The goal of this thesis is to develop a software concept suited for morning meeting participant. The software should support decision making in an IO context. But the creation of a fully developed computer system was never a goal of the thesis. It would be unrealistic to have such expectations. Instead small proof of concept modules was made for test purposes only.

1.2.4 Brief Overview of Research Method

The work with the thesis has been done in two, major stages. The first part consisted of mapping requirements for a future meeting system, and an evaluation of existing solutions on whether they fulfill these requirements, or not. The second part consisted of the actual development and testing stage.

The project could be considered as two logical parts since the development stage was solely dependent on the initial design of requirements and testing of other systems. If the tests concluded that current systems had implemented a sufficient amount of functionality, satisfying requirement needs, there would have been no reason for developing a prototype system. But needs were discovered, and these will be described later in this document.

However, since the project contained two, major stages it was considered if it was appropriate to let each state have it's own research methodology. It was also seen as important that these methodologies could be linked together. Deciding upon which to choose was therefore done in an early stage of the thesis work, to prevent incompatible selections later on.

The methodology selected for the work with the requirements in the first stage was a requirement definition methodology. A procedure for selecting a suitable software methodology, Burns and Dennis' [8] matrix (presented in Section 2.3), was used for finding the second methodology. The choice fell on RAD (Rapid Application Development). This could be combined with the initially selected methodology, and was also considered appropriate with regard to Burn and Dennis' matrix. The selection of methodology is however firmly described in Chapter 3.

To ensure that the requirements for the meeting system were of high quality, experts in the current research field at IFE were invited to take part in the creation of software requirements. They also played an important role when selecting and ranking these requirements.

Employees at IFE were also highly involved with the creation of the prototype system, especially in the early stage of this process, and involved in evaluation of it.

1.2.5 Hypothesis

It was difficult to foresee the outcome of this project, and if a new meeting system should be developed. As mentioned in Section 1.2.4 the project could logically be divided into two parts - the initial study of needs, and how existing software fulfill those needs, plus the development of a software concept supporting IO meetings. The second phase was however dependent on the outcome of the first phase. If there were no need for new software, the work with the second phase could have ceased.

But there were indications pointing in the direction of needs for concept development. Researchers in several studies found weaknesses while studying current software implementations in the oil and gas sector - such as solutions threatening information integrity, and causing unnecessary use of resources. Other studies indicated that currently implemented and complex solutions were exaggerated, and doesn't have to work as well as the simpler ones. Such research is presented in Section 1.5.

If there were weaknesses connected to the meeting software evaluated in this project, could not be told before the analysis was conducted. However, studies have proved that current software have a potential for improvement.

Another indication of needs for new morning meeting software is the continuing implementation of IO in the oil and gas sector. As mentioned earlier in the introduction, in Section 1.1.3, IO is gradually changing the oil and gas business. After implementing the first generation of IO, companies may evaluate the needs for integration of the second generation. If experts are to define requirements for a meeting software, for a business in transition, it is possible that currently implemented software and suggested prototypes are obsolete in some ways. In that case a new concept, supporting decision making in morning meetings, could be needed.

Large parts of this project has been conducted in close cooperation with some of IFE's experts on software development for oil and gas organizations. Cooperating with researchers, possessing extensive knowledge about the field, may also increase the chances of fulfilling the second research question - to promote important elements of the IO concept. They may also suggest needs, later formulated as requirements, for decision support based on their firsthand knowledge about the industry.

1.3 Project Structure

The project is divided into several chapters, each with more specific sections and sub sections. The following list briefly describes the project's chapters, without detailed depth information about underlying sections:

1. *Introduction*: In the introductory chapter a brief background of the selected topic is presented, before the research questions and the problem area is described. The last part of the chapter is mainly devoted to related literature.
2. *Theory*: The theory chapter includes relevant background theory for the rest of the thesis.
3. *Method*: In this chapter the methods for the preparatory studies, the development of the concept and testing are presented. The two, major methods used for concept development are included in a custom model. Each step in the model is firmly described. Additionally a method for ranking requirements, the MoSCoW prioritization method is mentioned.
4. *Observations and Experiments Prior to Concept Development*: The chapter contains observations and experiments done during the project. Firstly a morning meeting agenda, used in actual meeting activities in the oil and gas industry, is presented. Then information gathered during the requirement generation phase is presented, followed by evaluations of currently available meeting software and prototypes, according to these requirements. Lastly the chapter presents functions found in other software, possibly suited for the prototype developed in this project.
5. *Concept Development*: This chapter presents a review of how the concept was generated, and the choices made during the development iterations. The prototype was firstly developed following a horizontal strategy - designing lightweight slides of the future system. Then a vertical strategy was used for creating a part of the prototype in code. Lastly the findings made, during the vertical stage, resulted in several alterations in the overall design. Lastly the chapter contains information about how the prototype was tested.
6. *Concept Evaluation Results*: This chapter presents the results of the tests. Three test styles were used - a cognitive walk-through, heuristic evaluation, and a formal usability inspection.
7. *Discussion*: Here the results are discussed with regard to the research questions.
8. *Conclusion*: Finally the last chapter contains a conclusion, aiming for answering the projects research questions.

1.4 List of Abbreviations

- *ASD*: Agile Software Development.
- *CASE*: Computer-Aided Software Engineering.
- *CSS*: Cascading Style Sheets.
- *CTI*: Consumer Price Index.
- *DCS*: Distributed Collaboration Structures.
- *DTD*: Document Type Definition.
- *ESREL*: European Safety and Reliability Conference.
- *HSE*: Health, Safety and Environment.
- *HTML*: Hyper Text Markup Language.
- *ICT*: Information and Communications Technology.
- *IFE*: Institute for Energy Technology.
- *IPL*: Integrated Planning.
- *IO*: Integrated Operations.
- *IO-MAP*: The Integrated Operations Maintenance and Modification Planner.
- *JS*: JavaScript.
- *MCS*: Main Collaboration Session.
- *MoSCoW*: Must (o) Should Could (o) Would (prioritization).
- *MVC*: Model View Controller.
- *MVVM*: Model View ViewModel.
- *NCS*: Norwegian Continental Shelf.
- *NOG*: Norwegian Oil and Gas.
- *OLF*: The Norwegian Oil Industry Association.
- *PHP*: PHP: Hypertext Preprocessor.
- *PSA*: Petroleum Safety Authority Norway.
- *RAD*: Rapid Application Development.
- *RDM*: Requirement Definition Methodology.
- *RGM*: Requirement Generation Model.
- *XML*: Extensible Markup Language.
- *XSD*: XML Schema Definition.

1.5 Related Literature

The goals of the master thesis is to study if a new software tool could better support decision making in morning meetings than already developed software and prototypes. Such a tool should fit into the industry's setting, promoting the Integrated Operations concept.

However, it has been hard to find studies of software made specifically for the chosen meeting type. On the other hand, several studies have been made of software usage in the particular industry.

This part of the thesis firstly presents studies found of how morning meetings are conducted, and how they are related to the IO context. This hopefully elucidates some of the software needs in such an IO meeting, as requested in the second requirement question (Section 1.2.1).

Secondly the section presents a study which aims for setting collaboration sessions in oil and gas organizations into a broader perspective. The term distributed collaboration structures widens the meeting concept, and includes phases both before, during and after the actual sessions. This study relates to both the research questions, enlightening the scope of the meeting concept with regard to the current IO context.

Further the section examines studies of how decision support already works in the industry, and how decision support systems affect the quality of virtual collaboration. Such studies may be related to the first of the two research questions in Section 1.2.1.

1.5.1 Morning Meetings in an IO context

Ose and Steiro [26] emphasize that morning meetings are important in modern oil and gas organizations. Such meetings are held every day between individual oil rigs and the onshore office. All personnel supporting each rig are present in the meetings and communicate with their offshore colleagues either by phone or by using videoconferencing tools.

In a document [9], published by the Australian part of the oil and gas organization ConocoPhillips, important topics discussed in morning meetings are mentioned. Although this organization has another name for the meeting type - "pre-start meetings". In the document it is emphasized that planned work tasks offshore, permits for conducting jobs and inter alia HSE matters are commonly discussed.

IPL and Morning Meetings

Since planned tasks and jobs are discussed, morning meetings may be seen as the last stance of a long and comprehensive range of planning activities leading down to the final execution of tasks. The planning process in IO organizations is called integrated planning (IPL).

Ramstad et al. [28] have studied IPL and describes the process as more transparent than traditional planning. According to the researchers it should be possible for different actors to share planned activities during the whole process in IPL. They mention the following bullet points as essential:

- Interplay between planning levels.

- Interplay between organizations, groups, units and professions involved in planning and the execution of planned work.
- Inter-dependencies that have significant consequences for operational performance.
- A feedback loop for continuous improvement of the IPL-process.

Ramstad et al. point out that the transparency of IPL should work both horizontally and vertically in an organization. By vertically it means that planned work should be visible for actors responsible for both long-term, mid-term and short-term planning. On the other hand, the current organization of the oil and gas industry, formulating plans in the same time-span, consist of a long range of geographically spread domains. By integrating IPL horizontally, plans with the same kind of time-span should be more transparent across domains.

The article authors mentions ICT tools as one of the main parts of the IPL implementation. They have identified the following important aspects related to such technology:

- Availability of real-time information. This aspect is seen as especially important for short-term planning, as in morning meetings.
- Aggregation of data, information processing and data sharing.
- Visualization of dependencies in the planning process, and consequences of altering plans.
- Collaboration surfaces. Such tools could make communication and collaboration across organizational, professional and geographical borders possible.

However, the study performed by Ramstad et al. indicate that there may exist several challenges related to today's implementation of ICT tools in the oil and gas industry. The researchers studied three companies and found problems caused by an extensive use of different planning tools. These tools rarely supported automatic data sharing. Spokespersons from two of the companies also admitted that the lack of harmonizing tools threatened information integrity and led to unnecessary use of resources.

Further all companies stated that they were in need of ICT tools which could make real-time information accessible and available for all involved parties. The requested solution should also contain a shared collaborative surface.

HSE and Morning Meetings

As mentioned in the initial part of Section 1.5.1, health, safety and environmental (HSE) matters are discussed in morning meetings. Activities affecting safety on board installations are therefore relevant topics.

Traditionally the oil and gas business has been viewed as a high-risk environment, prone to accidents and precursors. An apposite example on the NCS is the major accident at the Alexander L. Kielland rig, earlier mentioned in Section 1.1.2.

But since the implementation of IO it is questionable if the industry still should be defined as a high-risk business. Vinnem [39] has studied offshore accidents occurring within two decades - a period stretching from 1987 to 2009. He claims that, although accidents still happen, the frequency seems to be decreasing.

In Vinnem's article "Evaluation of offshore emergency preparedness in view of rare accidents" the number of fatalities per 100 million work hours are compared with respect to the years 1987, 1997 and 2009. For mobile units the frequency of accidents shrunk from more than 50 in the period around 1987 to less than ten in 1997. Similarly the frequency further decreased from 1997 to 2009 - where less than five fatal accidents were registered. The trend is almost the same for production installations, but in a smaller scale and with less difference between the years 1987 and 1997. In the article Vinnem points out that the huge decrease in fatal accidents from 1987 to 1997 may be a result of the work done in the aftermath of the Alexander L. Kielland incident.

Okstad et al. [24] explains that the Petroleum Safety Authority Norway (PSA) today is supervising emergency preparedness, safety and the working environment in the modern offshore business. PSA establishes investigations of major incidents in the North Sea, and is putting pressure on companies to record, examine and investigate accidents through management regulations. Their work may contribute to the decreasing rate of accidents.

But there are exceptions.

One, prime example is the Deepwater Horizon accident where an oil rig sank in the gulf of Mexico as late as in 2010. 11 people were killed in an explosion and a following fire on board the installation. The oil spill caused by the incident has been considered the biggest environmental disaster in US history.

Another more relevant incident, possibly connected to morning meetings, happened on the 16th of November 2012 in the same part of the world - the gulf of Mexico. Then multiple explosions, followed by a fire occurred on the WD 32E offshore platform. Three workers died in the incident, and others were seriously injured. The direct cause; welding work on an oil tank igniting hydrocarbon vapor ¹⁰.

An investigation [23] performed by the Bureau of Safety and Environmental Enforcement (BSEE) found that the operator, authorized to approve the welding work, was not present in the morning meeting prior to the incident. He was occupied in another meeting conducted at the same time. The task was therefore approved by a C operator - a role which normally does not possess knowledge and experience to evaluate dangerous activities.

If the incident could have been avoided by restricting the C operator from approving the fatal task is questionable. But BSEE's report states that a possible contributing factor to the accident was the work permission given in the morning meeting.

These examples, both occurring after the accidents studied by Vinnem, may prove that safety still is an important matter that has to be taken seriously, also in short-term activities such as morning meetings.

1.5.2 The Structure and Scope of Collaboration Sessions

From the outside oil and gas organizations could be viewed as overall unities. Such view is called an organizational perspective. But organizations could also be viewed from the inside and seen as a collection of individuals. This focus is called the individual perspective.

¹⁰<http://www.offshoreenergytoday.com/bsee-black-elm-contractors-to-blame-for-wd-32-e-platform-explosion/>

Skjerve et al. [34] have studied collaboration activities in modern oil and gas organizations, implementing the IO concept - and have introduced a new perspective, in addition to the ones mentioned above. They have named it the distributed collaboration structures (DCS) perspective. This perspective describes the collaboration activities taking place inside the organizations, involving individuals. It may therefore be viewed as a linkage between the overall, organizational and the individual perspective.

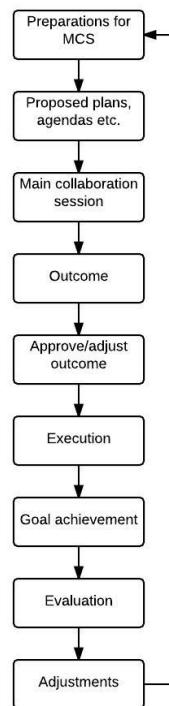


Figure 1.1: A redrawn version of the DCS model described by Skjerve et al. [34]

A model describing how a DCS may be built is presented in Figure 1.1. This is a redrawn version of Skjerve et al.'s model, which could seem fairly complicated. For the sake of simplicity the model has been redrawn for this project. Several of the stages were merged into three, main parts. Figure 1.2 contains a simpler version of the model described in Figure 1.1.

The simplified model contains three stages, involving individuals. The preparation stage includes work to be done before a main collaboration session (MCS) could start. For an interactive morning meeting such work may include tasks as planning, information gathering and the design of a meeting agenda.

The MCS is the main event in the structure, and frequently involves discussing proposed plans. Morning meeting sessions could be viewed as such an activity. However, given the nature of the modern oil and gas organizations, the meetings may involve geographically dispersed individuals and teams, with highly different backgrounds, collaborating in the same sessions. A model illustrating a MCS may be viewed in Figure 1.3. In the presented model several

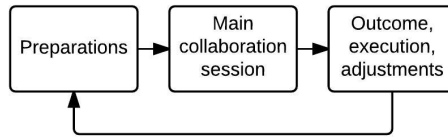


Figure 1.2: A simplified model of distributed collaboration structures, described by Skjerve et al. [34]



Figure 1.3: A simplified model made of the main collaboration sessions, described by Skjerve et al. [34]

individuals, located on different places, are involved.

The last stage contains the outcome of the MCS. This stage could include execution of plans discussed, for possibly accomplish some sort of goal achievement. Lastly activities for adjusting both the DCS as a whole and the structure of the main collaboration sessions could be performed, based on newly acquired knowledge of the process.

After conducting the last stage, the process starts from the beginning again in an iterative manner.

Skjerve et al. have not made a perspective specifically illustrating how morning meeting structures are performed in today's oil and gas organizations. However, the model may be seen as important for understanding how collaboration activities, including such sessions, are performed.

Chapter 2

Theory

2.1 Decision Support Tools in IO organizations

As mentioned in the initial chapter, the thesis' main research question asks if software tools can support decision meeting in morning meetings, and how. According to Kaarstad and Rindahl (2012) decision support implies supporting people making decisions. Decision systems are components of decision support, which also consists of decision theories. Decision support systems are software which help their users solving problems and to accomplish complex decision making.

2.1.1 A Study of an Email Platform and Other Decision Tools in the Oil and Gas Industry

One software platform used for decision-support in morning meetings was a platform, described by Sarshar and Rindahl [31]. This was a simple platform based solely on emails and the SAP software environment. The platform was used both before, during and after meetings in the following way:

- *Before the meeting:* The management prepared a handover and a status report interacting with an existing email template containing predefined headers and information points. This document was sent to the operation manager before the meeting started, and made it possible for this leader to be familiar with the information given.
- *During the meeting:* During the meeting the same email was used as a shared surface and was then updated with new information generated i the meeting.
- *After the meeting:* When a given meeting finished, the email acted as a status report. The document was at that point sent to all of the meeting's participants and stakeholders.

Sarshar and Rindahl concluded that this approach was the best solution they had seen during their five years of studying IO collaboration technologies in the business. But the platform was dependent on experienced and skilled leadership.

However, further in their study the researcher found several examples of issues connected to the use of more advanced state-of-the-art decision tools, although these tools were much more complicated than the simpler email platform. Some of them were, according to Sarshar and Rindahl, both prone to errors and had a low degree of user friendliness.

One described example was a collaboration system implementing a "shut down"-button in its interface. This button ended however not the current collaboration session, but only shut down the monitors used. It was observed that personnel misinterpreted the use of this interface element.

Other example involved 3D technology producing dual images, that could be viewed as blurry and unclear images if the correct adjustments were not initiated by the personnel. Meetings were also often canceled due to common, technical failures. Lastly it was also observed installed cameras in video conferencing rooms not capable of showing the faces of the participants involved in a virtual conversation.

One crucial advice, supported by the observations conducted, was that "new technology should enable a desired work practice, and not the other way around". The poorly functional software and equipment could, according to Sarshar and Rindahl, have been implemented due to vendors unable of seeing the users' actual needs and habits. Another cause could be linked to the ICT and procurement departments. These groups had often a low degree of contact with the operative groups, leading to difficulties with defining system requirements. System orders were therefore often over- or undersized.

This study may emphasize that decision support systems in activities, such as morning meetings, do not have to be the most expensive and state-of-the-art solutions. Simpler, but effective tools, may be more user friendly. It also pointed out that requirements for implementing software should be formed in closer collaboration with the operating groups.

2.2 Systems Designed for Meetings in the Oil and Gas industry

There are several software systems which could be used during morning meetings in the oil and gas industry. In this section current software have been described. These systems are later analyzed further to find solutions to implement in a new meeting system.

2.2.1 Epsis Teambox

Epsis Teambox¹ is a collaboration solution, designed especially for activities such as meetings. The Epsis Teambox is delivered on a single workstation, which manages the entire system. A new version of the solution has however been created since it was tested in early Spring 2015.

The system makes the user able of managing screen setups with several windows presenting video conferencing, meeting information, documents, etc. The screen layout may also be edited, and saved as templates.

Epsis Teambox is designed for collaboration in small, medium or large organizations, as well as enterprises. Outside meetings, the system makes it possible for the users to collaborate and share information through the platform.

According to a document by Epsis [10], describing the solution, one of the major advantages of Epsis is that it is not only a software, but also a hardware solution. Both software and hardware are delivered in a single tower, or in a rack.

A part of the software is, according to the same document, referred to as an engine. this engine is capable of tailoring the display environment, to mange external sources, executing meetings, creating layouts, share screen content, and call for collaborative meetings.

2.2.2 ARKit

ARKit [12] is a software system developed by the company Ark Platforms. The tool is built around a map interface. The map is zoom-able and may display information according to the selected magnification. The oil companies vendors, work sites, personnel, etc are visible in the map. Item-specific information is shown when these objects are clicked.

Further the software displays activities as projects or jobs performed in a company as geographical information on the map.

Additionally the interface has an activity stream - a column containing project updates, added documents, and other news.

2.2.3 Protosphere

Protosphere [14] is a 3D tool built for meeting activity. The software lets the user navigate a 3D environment by taking control of a personalized, human avatar.

The virtual environment may be edited by the user and generally looks like a huge meeting space with monitors hanging on the walls. These monitors may

¹<http://www.epsis.no/epsis-teambox/>

be used for presenting meeting agendas, document cooperation, displaying real-time information, etc. The avatars may also interact with gestures and voice over IP communication.

Implementing a set of extensions enables further functionality. With Polycom [13] the monitors in the Protosphere environment may present video conferencing. Lync [27] presents the user with a contact list, to start collaboration sessions. Other extensions are also available.

Protosphere may be taken into use from both ordinary computers and mobile devices [15].

2.2.4 Email platform

The origin of the email platform has not been made public. But the concept is similar to the one described by Sarshar and Rindahl [31] in the article "Integrated Operation Collaboration Technologies - Remaining Challenges and Opportunities". Here the authors mention an email platform used as both a status report, an agenda and as a minutes document before, during and after meetings.

The aforementioned email surface was not designed as a groupware solution, nor was it particularly sophisticated. However, the solution was found to be one of the most successful by Sarshar and Rindahl. They emphasize that the simplicity of the system was one of its major advantages.

The work flow was as follows: The onshore manager firstly received a status and handover report from the offshore part of the organization as in email before a meeting. This email was then used as a meeting platform during the meeting, and updated with new information. After the meeting the email was forwarded to participants and stakeholders with the newly added changes.

Sarshar and Rindahl point out that high performance may be achieved with simple tools as the email surface. They also state that technically sophisticated tools were observed to perform less efficient than this simple one.

2.2.5 IO-MAP

IO-MAP is an abbreviation for "the Integrated Operations Maintenance and Modification Planner". As the name of the tool indicates, IO-MAP was developed to support users planning offshore modifications and maintenance. The prototype is, as mentioned by Skjerve et al. [35], part of a project called "Future Collaboration Environments" at the IO CENTER. The tool's testbed has however been developed at IFE.

Skjerve et al. [35] emphasize that the developers focused on visualization of risks when creating the system, and found ways to promote risk of decision-making in a map surface displaying the current working environment. Risk understanding was tried to be increased by highlighting hazards, prohibits, connectors and work tasks on the map.

The tool provide the users a map area of an installation showing planned jobs, requiring work permits. Here new tasks also may be added by planners. The visualization of the installation may be zoomed, panned, and the users may also decide to navigate through different decks [36].

Elements as hazards, prohibits and connectors may be added to the map area. Hazards represents safety issues related to specific tasks. Prohibits are

mandatory initiatives needed, like wearing safety helmets in certain areas of the installation. Connectors are represented as lines between tasks, and visualize hazards associated with connected tasks [29].

In addition to jobs the map visualization may show areas where there may be certain hazards - as the risk of falling objects (implemented in the second version of the tool), zones with danger of explosions, etc [7].

In addition to the map the tool also show the user information about e.g. weather conditions, a calendar to select interval for showing jobs, workload in specific time intervals and possibilities for accepting or rejecting work permits

The process of planning modification and maintenance tasks offshore/onshore in a petroleum organization may be a complex business. However, the IO CENTER² has, according to [29, 32, 35] using tools as IO-MAP may be a mean to overcome cultural and skill related differences. The tool itself is meant to be a groupware available for planners across a restricted network. Taylor et al. [38] also argues that the tool may be useful for management purposes in the oil and gas industry.

Skjerve et al. [35] inform that the first version of IO-MAP went through testing with several participants from the onshore oil industry. The outcome of the tests indicated that lack of local knowledge of offshore installations among onshore personnel could to a degree be compensated by a tool as IO-MAP. It could therefore make onshore planners more familiar with offshore installations, since they could get more knowledge of such systems' characteristics. In addition the test indicated, according to Taylor et al. [38], that the software's user interface could promote the personnel's ability to visualize safety hazards.

2.2.6 Wisio

Olsen et al. [25] have created a prototype of a collaboration surface displaying activities on an offshore installation. The project was partly inspired by IO-MAP.

The authors stressed the importance of usability and design when developing the software. The final product is a prototype of a shared platform where users may register activities and extract information from them by clicking an interactive map. The activities are also displayed in a timeline, making it possible for the users to compare activities in time perspective.

Additionally the design includes weather forecasts, transportation and inter alia help information. The overall user interface is beyond this much like IO-MAP.

²<http://www.iocenter.no/>

2.3 System Development Methodology

2.3.1 Traditional methods

Traditional software development methodologies have well-defined documentation and plans. This subsection mentions several of these, and what type of development method to select with respect to a specific project.

The code-and-fix method

This is one of the earliest software methodologies. It is a simple method with only two stages. In the first stage code is written, then in the next the problems with the previously written code are fixed. This simple methodology lacks however recognition of requirements, planning and other design stages [21].

The stagewise method

The stagewise-method was designed in the 1950s. Unlike the code-and-fix-method, the stage-wise method has several stages. These are creation of an operational plan, coding specifications, coding parameter testing, assembly testing, shakedown and finally system evaluation. The method has however not included possibilities for improving previously finished stages, although new experience may demand such changes [21].

The waterfall method

This method has, like the stagewise-method, several stages. It has also implemented prototyping as a separate stage. Unlike the stagewise-method, the waterfall method has implemented feedback-loops for changing previously finished stages. A previous stage may then be changed in the same fashion salmon is using a salmon staircase - changes has to reset the project to an earlier stage [21].

The waterfall method has traditionally been used for large-scale projects, like development of compilers and operative systems [21].

Evolutionary software development

In this method the operational experience of users is included in the software development. It is the users of a future application who specifies the functional characteristics of it.

The transform method

In the transform method all later modifications to code are only made in the specification, and not in the design, testing and implementation stages. It therefore must include options for transforming specification to code. This method has proved to be effective for reducing development time and cost [21].

The spiral method

The spiral method is first and foremost a further development of the waterfall method. Unlike the water method's sequential nature, this method implements iterativity. The life cycle of a project is formed like a spiral, and small parts

of a planned project are iteratively created for each turn of the spiral including different stages [21].

Prototyping

Burns and Dennis [8] presents a traditional set of development methods in the paper "Selecting the Appropriate Application Development Methodology" written in 1985. Unlike Misra they also mention prototyping as a software development method. Prototyping is compared with other, sequential methods, which are summarized as life cycle methods.

Prototyping is more loosely defined. The method requests collection of information about what the users don't like with an existing system, or missing features. The developer then creates and presents the user with a prototype - instead of comprehensive design specifications. From this prototype the user may clarify system requirements. These requirements may later be included in a future prototype. The iterative process ends when the user is satisfied with the created software [8].

The advantages of the method may be lower development cost, and high user satisfaction. The negative sides may be creation of less effective systems than with sequential methods, since prototyping lacks analysis and design stages [8].

Mixed methodology

Burns and Dennis [8] also propose a mixed methodology, including parts from system life cycle models and prototyping. Such method may propose sequential development of subsystems. But the complete system is first designed coarsely. Then each of the subsystems are refined and shown for the user. If the software part is approved, the subsystem is designed and developed sequentially and in detail [8].

But this is only one sort of mixed methodology. Several other forms exist. Common to all such methodologies is that they consist of both a life cycle and a prototype part [8].

How to decide upon which method to incorporate

Burns and Dennis [8] state that the level of project complexity and project uncertainty should be used to determine the selection of development methodology. They have created a matrix for this purpose. The matrix has been reproduced and is shown in Figure 2.1.

According to Burns and Dennis [8], a system life cycle method should be chosen if a project is complex and the uncertainty is low, the mixed methodology should be chosen if the complexity is high and the uncertainty is high. At last the prototyping methodology should be preferable if the complexity is low, and the uncertainty is either low or high.

The following bullet points may be used to determine uncertainty [8]:

- The degree of structure. If it should be a fixed system, not subject to change, the uncertainty is low. If for example the inputs and outputs are clearly defined, the uncertainty is also lower than if the opposite is the case.

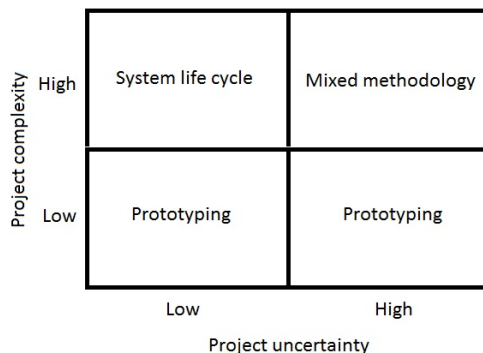


Figure 2.1: This illustration shows how a software development methodology may be selected, on the basis of a project's complexity and uncertainty [8].

- User task comprehension. The degree of the users' understanding of the future systems' tasks, and how to perform them.
- Developer task proficiency. If the developer will be able to understand the system's requirements exactly, may have consequences for the uncertainty. The developer's familiarity with the tools needed in the development process and the users' environment is also included in this point.

And this bullet list contains points affecting complexity [8]:

- Project size. Usually measured in man-hours, may give an indication of size. Projects which have long development time are often more complex than others.
- Number of users. It is stated that a system with many users is in general more complex than a system with few.
- Volume of new information. A system may be more complex if the volume of new information generated is high.
- Complexity of new information production. If the processes used for creating new information are complex, the system also gets more complex.

Criticism of traditional methods

Critics have been promoted against the traditional software development methods. The critics argue that such methodologies are not flexible enough, difficult to learn, labor intensive, slowing down the development process, poorly defines, and so on. The more recently developed agile methods is however a counterpart of the traditional ones [21].

2.3.2 Rapid Application Development

There is no universal definition of the Rapid Application Development (RAD) methodology. But RAD may however be characterized in two ways; as a methodology with a set of phases and as a class of tools making it possible to develop software in a speedy fashion [1].

RAD is normally implemented - with the following possible phases - requirement planning, user design, construction and cut over. As agile methods, described in Section 2.3.3, RAD also require customer involvement. But the methodology is also associated with the use of prototyping and incremental development [1].

RAD may be described as a methodology deploying tools, guidelines and techniques for a short period of time for developing a part of a software system. Such a part are defined as a "time box", a module developed in close cooperation with the customers. Software is delivered in pieces, and the most important parts are delivered first [1].

RAD teams consist of small groups of highly skilled individuals. These members cooperate to finish the software product before an end date, and the product delivery cycle is more concerned about deadlines than traditional application development. But the development team has to keep a form of memory of the structure of the process while working inside a time box [1].

The Australian company Mediasphere [20] has created a document briefly describing the RAD development process, and it's stages. A reproduced model of a figure presented in the company's document is shown in Figure 2.2.

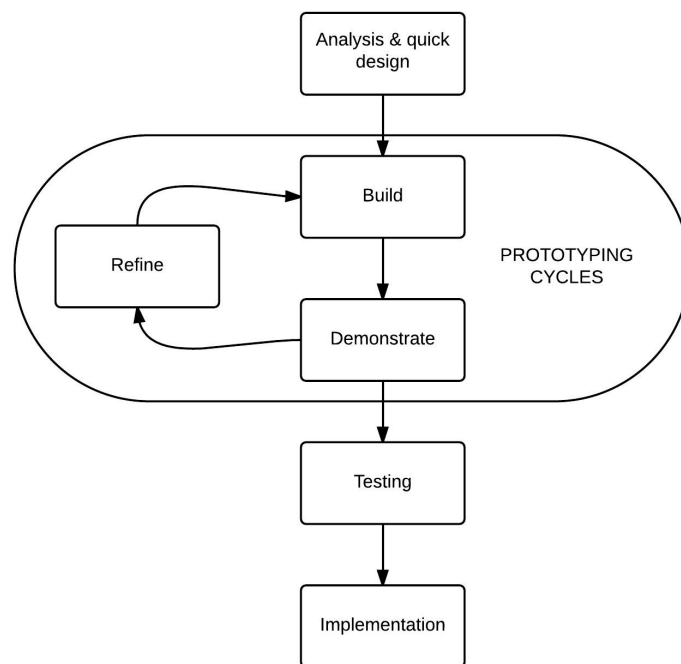


Figure 2.2: An illustration of the RAD development process, as described by Mediasphere [20].

In the presented model RAD is divided into six stages; analysis and quick design, build, demonstrate, refine, testing and implementation. The build, demonstrate and refine stages are all part of an iterative loop called prototyping cycles. The other stages are sequential in nature. Each of the stages are briefly

explained in the text below [20].

Analysis and quick design

In this stage one of the main goals is to perform a software analysis gather requirements for the upcoming development. Meetings between the key customers and the planning team should also be conducted to establish communication between the involved personnel [20].

The quick design part aims for presenting the detailed requirements from the analysis phase - developing a required data model. The concrete results should be screen flows and layouts of essential parts of the system. These may be used in the process to find core requirements for the entire, future system. The use of CASE-tools are common in this stage [20].

Ganesh Krishnamurthy ³ describes CASE-tools (CASE: Computer-Aided Software Engineering) as technologies used to provide an automated assistance in the software development process. Special RAD tools are freely available for for example the Visual Studio development environment. Such a tool is for example "RAD Studio Code Generation Toolkit" ⁴, which is especially made for supporting software development following the RAD development life cycle.

Prototype cycles - build, demonstrate & Refine

During the prototyping cycles the three stages build, demonstrate and refine are repeated iteratively. The software is gradually developed, tested and the requirements are progressively formed. Functionality is added during each iteration, and afterwards new functionality is scheduled for development [20].

Several documents are maintained and updated during the cycles. For all iterations the core and secondary requirements are designed or redesigned. The data model, test scripts and project plan are also updated. For each cycle an iteration plan is created, and handed to the customer [20].

After the final iteration the development team updates the software documentation, performs user acceptance tests and prepare the steps necessary for the upcoming implementation process [20].

Testing

In this stage the finished application is tested. The tests are performed by the technical team, and also the software users [20].

Implementation

Here the software is converted to a live system. The development team is responsible for the transfer and migration of the system [20].

2.3.3 Agile methods

Software developers have in recent years adopted to new development methodology. Agile software development (ASD) is a popular set of methods, which aim

³<http://www.umsl.edu/~sauterv/analysis/F08papers/View.html>

⁴<http://radstudio.codeplex.com/releases/view/66805>

for simplifying development and making it more adopted to later requirement changes [21].

Such methodologies relies on frequent contact between developers and customers - preferably face-to-face communication. The techniques implement iterative software deliveries of finished system parts - instead of complete software systems [21].

According to the manifesto made for agile software development, the following bullet list illustrates the methodologies' main values [21]:

- Individuals and interactions should be prioritized over processes and tools.
- One goal of a project should be to create working software, and give less priority to comprehensive documentation.
- Customer collaboration should be sought instead of contract negotiation.
- The development process should respond to change, instead of just following the plan in such situations.

Although the techniques have been praised by parts of the development community, critics to the methods have also been raised. According to Misra [21] the critics have mostly been promoted by large private, governmental and bureaucratic organizations. The critics argue that ASD is over-hyped, reduces quality due to the lack of rigor, has limited support for distributed environments and has inter alia limited support for development involving large teams.

2.4 Requirement Methodologies

2.4.1 Requirement Definition Methodology

Arthur and Gröner [3] present a model for requirement generation - the Requirement Generation Model (RGM). The method divides the activity into sub-phases, which may be repeated iteratively until a final set of system requirements are defined. According to studies performed by the authors this procedure improves the quality of produced software.

The method described in the paper "An operational model for structuring the requirements generation process" begins with a indoctrination phase. Then three sub-phases are repeated in a loop in a phase called the requirement capturing phase until the requirements have been defined. They may then be implemented into the requirement analysis. These three sub-phases are; the preparation, the elicitation and the evaluation [3].

The Indoctrination Phase

One of this phase's goals is to introduce the customer (involved in the process) to how RGM works. It also aims for providing the requirement engineer with information about the domain and customer needs. The participants' responsibilities and tasks should be described at this stage [3].

The Requirement Capturing Phase

As earlier mentioned, this phase contains three sub-phases. It is an iterative process repeating the phases below in the order they are mentioned in this section. The process continues until all system requirements have been set [3].

Preparation In this sub-phase there should be a meeting where the roles should include the customer, the requirement engineer and possibly also the user of the future system. If this meeting takes place after finishing a requirement iteration, issues revealed in earlier discussions should be reviewed. Otherwise the meeting takes place after completing the indoctrination phase [3].

The purpose of this phase is to review any open issues discovered in earlier phases. Then work has to be done to arrange the next meeting between the requirement engineer and all the other stakeholders. In the preparation phase it could be decided that such a meeting should be a declarative meeting - about communication of needs. Other forms of meetings may be explorative (aiming for discussing and finding new issues) or for example elicitive (focusing on identifying individual requirements). Common to all such meetings is that the requirement engineer is taking lead role, but is actively consulting the customers [3].

Elicitation This focuses on capturing of system requirements. In this phase the customers convey the information needed, while the requirement engineer captures the needs [3].

Evaluation In this phase it should be determined if the objectives of the previous elicitation meeting were met. The identified requirements should also

be examined, as independent artifacts and in relation to the previously identified requirements. Lastly it should be determined if a new iteration is needed [3].

2.4.2 MoSCoW Prioritization of Requirements

The MoSCoW method is a technique for prioritizing system requirements. According to Project Smart's Duncan Haughey ⁵ MoSCoW is used for ranking a selection of defined requirements - whether they should be implemented or not, in what order functionality should be developed and for defining crucial system parts.

MoSCoW is an abbreviation for the phrases "must have", "should have", "could have" and "would like to have". When reviewing a list of system requirements these phrases are used for defining the need for implementing each of them.

According to Haughey the requirements defined as "must have" parts of a system are crucial for the system's success. If they are not present, a developed system may be viewed as a failure.

"Should have" requirements are not as important as the type mentioned above, and a project's success is therefore not depending on the implementation of them. However, as many of these requirements as possible should be added to a new system.

"Could have" requirements are nice to have, but they don't have to be added - only if there are available resources. It is however important that implementation of such requirements is not negatively affecting requirements having a higher priority.

"Would like to have" (or "won't") requirements is the fourth and last category of the MoSCoW prioritization method. These requirements would not be added to the version of the system that is to be developed. Instead they may be implemented later.

The DSDM CONSORTIUM⁶ points out that using MoSCoW as prioritization technique has benefits. According to DSDM this prioritization technique makes it easy for the stakeholders to get a grasp of how implementation of system functionality will affect the end-result. Ranking requirements as "high", "medium" or "low", on the other hand, would not specify the need of implementation.

But there are also critics with an opposite opinion about usage of the technique. Kukreja et al. [17] claim that the method is not truly value-centric. Instead MoSCoW is seen as a technique assuming that the stakeholders instinctively know the correct value of each requirement priority, something that may seem quite utopian. Further they state that MoSCoW may not work well when system changes are promoted. It could then be hard to incorporate the priority value of the newly desired functionality with the current release.

⁵<https://www.projectsmart.co.uk/moscow-method.php>

⁶<http://www.dsdm.org/content/10-moscow-prioritisation>

2.5 Prototyping Techniques

Beaudouin-Lafon and Mackay [5] describe how prototypes of future software could be made. According to the authors such representations of interactive systems could be developed both offline and online during the design process.

2.5.1 Offline and Online Prototypes

Offline prototypes, also called paper prototypes, do not have to run on computers when reviewed. These representations are often easy and quick to develop. They are commonly designed early in the development process, and are usually thrown away after serving their purpose.

Online prototypes, on the other hand, require a computer for presentation. Such representations may be designed in digital interface builders, or for example as programs written in code. The cost of developing these prototypes are commonly higher than for the above category. They are therefore better suited for being implemented later in the process, when a design strategy has been defined.

However, Beaudouin-Lafon and Mackay do not recommend selecting only one of the two, overall techniques when designing a new system. They argue that the prototyping process should start with developing simple, offline system representations.

According to the authors there are several benefits of producing paper prototypes. Firstly these representations do not have to constrain the thinking of the designers - limiting creativity. By creating such prototypes the number of ideas discussed could be wider than by spending much more time on creating advanced representations.

Secondly Beaudouin-Lafon and Mackay state that creating design offline, as mentioned above, is inexpensive and may entail more rapid prototyping iteration. Different solutions may therefore be tried, instead of binding the developers to first, but possibly not best, working solutions.

Lastly another benefit of producing paper prototypes is that they easily may involve other parties than programmers in the process of producing them. They may therefore lead to increased collaboration in the design process.

Online prototypes are however better suited for evaluating more particular design ideas, where offline prototyping may prove to be insufficient. Complex, dynamic visualizations and advanced interactions may require online prototyping.

2.5.2 Prototyping Strategies

Further Beaudouin-Lafon and Mackay present four strategies for developing prototypes. These may be used for creating:

1. *Horizontal prototypes.*
2. *Vertical prototypes.*
3. *Task-oriented prototypes.*
4. *Scenario-based prototypes.*

Horizontal Prototypes

Horizontal prototypes include the whole system in the designing process. They may therefore present the reviewers with a overall perspective of how the future system will work.

Such prototypes may begin with rapid, offline techniques, and gradually expand to working design. They may therefore be built with interface builders, without making use of underlying functionality. However, since simulations of the future application may require some sort of simulation of the underlying functionality and data, the prototyping process tends to be evolutionary - gradually expanding the prototype into a working system. In that process the prototyping may need the whole development team to produce them.

Vertical Prototypes

Vertical prototyping strategies may be made when the designers should test if working functionality is implementable. In that process layout algorithms and performance could be tested, in addition to interaction techniques. Such prototypes have therefore in general a high precision, and are suited for validating specific ideas.

Task-oriented Prototypes

Designers may choose this approach when they start off with a task analysis, identifying user needs. Such analyzes may require functionality, which is developed in task-oriented prototypes. These prototypes are generally organized as tools sets, allowing each set test specific tasks. When following this strategy the system is therefore systematically created by implementing functionality to accomplish tasks.

Scenario-based Prototypes

These prototypes are much like task-oriented prototypes, but do not limit the prototyping process to tasks. Instead the representations should be developed for more realistic scenarios, used in real-world.

2.6 Testing

Shneiderman and Plaisant [33] have described a selection of testing techniques. Two, overall techniques are describe in detail - usability testing and the heuristic expert reviews approach.

According to the researchers the conduction of testing may take place both in the beginning, the middle and the late stage of the development process. The following aspects regarding a project are mentioned by Shneiderman and Plaisant as important before settling with a specific technique:

- *Stage*: The current state of the development, if it is early, middle or late in the process.
- *Novelty*: If the project is unique or new.
- *Number of users*: How many users the future project will have.
- *Criticality*: How critical the functionality of the interface will be.
- *Costs*: Resources available for conducting testing.
- *Time*: The time available for testing.
- *Experience*: The experience level of the design and evaluation team.

2.6.1 Usability Testing

Shneiderman and Plaisant have described usability testing as a method involving the future users, or a representative selection of users, in tests. The experiments may take place in controlled environments as usability labs to find flaws in developed software.

Such experiments may involve as few as three participants. These participants should possess the future users' knowledge of computing, experience with the task domain and represent the users' education level and background.

2.6.2 Experts Reviews

Experts reviews, on the other hand, do not involve the future users of a system in the testing phase. Shneiderman and Plaisant emphasize that such tests may be conducted in both a formal and an informal manner.

Informal experiments may be carried out by presenting an interface of a system, or a part of it, for colleagues or users and then ask for their opinion. This approach may produce some useful feedback. More formal experiments may involve experts, and the conduction of tests resulting in a test report. According to Shneiderman and Plaisant, the latter approach has proved to be the most effective.

Experts reviews may be conducted during any stage of the development process. However, involving domain experts in the experiments may be crucial for producing better results. It may also be useful to choose both reviewers earlier involved in the development process and fresh reviewers in such tests.

Shneiderman and Plaisant suggest several techniques for conducting experts reviews. The following bullet list mentions some of them:

- *Heuristic evaluation*: This technique asks the experts to review an interface and to compare implemented elements with a list of heuristics, as for example the Eight Golden Rules of interface design. It is then important

that the experts are familiar with the heuristics selected, and that they are able to apply them.

- *Guidelines review*: This approach requests the experts to review the interface with a guidelines document.
- *Cognitive walk-through*: The experts are asked to perform a walk-through of selected parts of the interface, to carry out tasks.
- *Formal usability inspection*: The interface's weaknesses and strengths are discussed in a formal courtroom-styled meeting.

Experiences with the Use of Usability Inspection Methods

Holligsted and Novick [11] have studied the use of usability inspection methods as heuristic evaluation, cognitive walk-through and formal usability inspections - all mentioned earlier in the bullet list in Section 2.6.2.

According to the authors both the cognitive walk-through and the formal usability testing techniques were more popular tools among usability professionals earlier than they are today. Cognitive walk-throughs, while still in use, are implemented less frequently in today's projects than earlier. The formal usability testing technique was popular in the mid 90's, but is now less used.

Heuristic evaluation, on the other hand, is still in frequent use. Half of ten intranets winning a competition in 2005 reported that they had used heuristic evaluation while testing their software.

According to Holligsted and Novick cognitive walk-through and Novick are methods easy to deploy in a project's development stage. These techniques are both cost-effective and may be conducted rapidly.

Additionally the authors claim that such testing techniques should be combined to achieve better results while experimenting, than with the use of solely an individual technique. Further Holligsted and Novick advise the use of several participant groups - both domain experts and end-users.

2.7 Visualization

This master thesis is aiming for creating a tool which may present relevant information to participants in morning meetings in the oil and gas industry. The tool have to have some kind of visualization of data to make this goal possible.

Visual Representation

The purpose of a good visual representation is to guide the users and help them to understand a visualization. The designer may choose among several viewing formats [37]:

- *Color*. Color is commonly used to visualize large data sets. It is less useful for smaller data sets, or data sets which contains a small range of differentiation. People may identify many gradients and shades of color, which makes color ideal for representing big-picture trends.
- *Size*. People tend to judge quickly between sizes, but this representation is widely used, and perhaps overused.
- *Location*. This technique may be used to attach data to a map or an other element corresponding to a real or virtual place. This kind of representation may be a good solution if the viewers are familiar with the visualized location.
- *Networks*. Networks may be used to for example show social connections between people. Networks may contain data points and the links between them. However, this technique should be used with caution, since it may be hard to understand a messy network.
- *Time*. Time animations may be used as a technique to represent data changing over time.
- *Multiple methods*. The representation techniques mentioned in the list of bullet points above may be combined in some visualizations. An example on a visualization technique which uses both the size and the time dimension is a stacked time series. This representation of data may be a chart containing elements with different sizes - changing over an animated time.

2.7.1 Map Visualizations

Steele and Iliinsky [37] has created a digitally available map for the New subway system. In the book "Beautiful Visualization" they present the solution and argue for choices done in the process of making the digital map. The purpose of the map is not to describe the subway system in unnecessary details, but to show the viewers a simplified and understandable image of the New York subway system. Several advice are presented. Some are mentioned below:

- *Include essentials*. According to Steele and Iliinsky only the essentials of complex geography must be drawn. Extra, unnecessary details may only cause visual noise for the viewer.
- *Avoid disorientation when a map contains several layers*. Iliinsky and Steele state that it is important to avoid disorientation when creating a

map. In the subway system map both the places on the surface and underground had to be drawn, simplified - but not too simple. The audience had to recognize the environment in both layers [37].

- *Create a recognizable environment.* Steele and Iliinsky state that it is important to create a map which mirrors the viewers relationship to the environment. The subway map was divided into several neighborhoods, since the people of New York (according to the authors) look upon their city as a collection of neighborhoods. nm jk bvb
- *Create possibilities for zooming* The authors implemented possibilities for zooming in their digital map, which made it possible to get more detailed sub-maps.
- *Change map if context is change.* Iliinsky and Steele presents a map which changes from day- to night-time. Sine fewer routes are active at night, the map mirrors the changed context.
- *Create situational maps.* Iliinsky and Steele point out that maps don't have to be the same in all situations. According to the authors maps should be slightly different in for example on the walls in train cars, in digital form or on big posters at underground train stations. The maps should be optimized for their specific environment.

2.8 User Interface Design

Shneiderman and Plaisant [33] state that gathered evidence and theories applied for designing user interfaces may be organized into three, main categories:

- Guidelines.
- Principles.
- Theories and models.

According to Shneiderman and Plaisant [33] theories and models belong to the highest and most overall level of these three. Principles are more specific, and may be used to analyze and compare design alternatives. Guidelines are the most specific, and are often platform specific.

In this thesis design principles are used. The section below concerns design principles, especially the the Eight Golden Rules of interface design, but also principles for implementing interaction styles.

2.8.1 The Eight Golden Rules of Interface Design

Shneiderman and Plaisant [33] claim that the Eight Golden Rules of interface design may be seen as a good starting point for mobile, desktop and web development. They also point out that this set of underlying rules could be refined and extended with regard to the specific solution. Below follow each of the eight rules, with accompanying explanations.

1. **Strive for consistency.** This involves consistent sequences of actions, colors, textual information, etc.
2. **Cater to universal usability.** The user interface should be adapted to different users such as experts and novices, different age,ranges, users with disabilities and different level of technological knowledge.
3. **Offer informative feedback.** For every user action there should be some sort of system feedback. But frequent or minor actions should however result in modest sorts of feedback.
4. **Design dialogs to yield closure.** Sequences of actions should all have a beginning, a middle and an end.
5. **Prevent errors.** The developers should strive for reducing errors. The user interface should therefore be designed for error reduction by for example graying out not appropriate items in menus, etc.
If errors occur the user interface should notify the user about the error. The user interface should remain consistent if an error occurs, or give the user instructions on how to regain consistency.
6. **Permit easy reversal of action.** All actions should easily be reversible, supporting user exploration.
7. **Support internal locus of control.** The user interface should be designed for experts. When operating a UI such users usually don't want surprises, they like to stick to familiar patterns, don't like tedious data-entries and demand systems capable of rapidly produce desired results.
8. **Reduce short-term memory load.** Solutions where the users have to remember chunks of information from one screen to another should be avoided.

2.8.2 Interaction Styles - Advantages and Disadvantages

Shneiderman and Plaisants [33] mention principles for implementing interaction styles in a user interface, and have listed both advantages and disadvantages connected to each of them.

Direct Manipulation

Advantages: Presents concepts visually, allows easy learning and retention, allows errors to be avoided, encourages exploration and lastly leads to higher, subjective satisfaction.

Disadvantages: May be hard to develop and may require graphics display and pointing devices.

Menu Selection

Advantages: Shortens learning, reduces keystrokes, structures decision-making, permits use of dialog management tools and allows easy support of error-handling.

Disadvantages: Presents danger of many menus, may slow down frequent users, consumes screen space and requires rapid display rate.

Form Fill-in

Advantages: Simplifies data-entry, requires modes training, gives assistance and permits use of form management tools.

Disadvantages: Consumes screen-space.

Command Language

Advantages: Flexible, appeals to frequent expert users, supports user initiative and may support macro creation.

Disadvantages: Poor error handling and training and memorization needed.

Natural Language

Advantages: Not necessary to learn syntax.

Disadvantages: Unpredictable, may require many keystrokes, may hide context and may require clarification dialog.

Chapter 3

Method

3.1 Methodology Selection

3.1.1 Development Methodology

Before a new system could be proposed, and possibly designed, based on the current platform, methodologies for the development process had to be selected.

There are several methodologies existing, each with specific weaknesses and strengths. But the major challenge when selecting a methodology was the limited possibility to communicate with the project's "customers" - the petroleum organization offshore and onshore participating in morning meetings. This project is instead taking place at IFE, and the student is communicating with personnel there, rather than directly with the personnel in the oil and gas industry.

At first an agile methodology for the development of the new software was considered. But as mentioned in Section 2.3.3, agile methods involve users to a high degree in the development process, and two of four points in the manifesto for agile software methodologies stress that individuals and interactions should be prioritized before processes and tools, and that customer collaboration should be sought. It therefore seemed impossible to select an agile methodology for investigating and producing the new software.

Another option could be to still implement an agile and modern methodology, but redefine who the users are. Until now the users have been defined as personnel participating in morning meetings in the oil and gas industry. But since this is a research project, without a commission from the industry, it is likely that the results may only be used for a research purpose. The users may therefore be redefined to the personnel working at IFE, who may be interested in the future solution. But the target audience may still be the personnel in the oil and gas sector, since IFE is a research facility partly working with projects for this type of industry. Selection of an agile method may still require a lot of user collaboration, but with reachable users at IFE instead of users from the oil and gas sector.

A downside to this approach is the nature of agile development. According to the manifesto the goal of such methodologies should be to create working software, rather than creating comprehensive documentation. Since this project is a research into the selected topic, paying little attention to the documentation,

could be undesirable.

A traditional method could therefore be the best suited for this project. Dennis and Burns' matrix, presented in Figure 2.1, was used to decide if a traditional system life cycle method, as for example the waterfall method, should be chosen. But making this selection demanded a further investigation of the project's nature - the degree of complexity and uncertainty.

Firstly it was hard to predict if the current project was to be complex, or not. According to Dennis and Burns the complexity could partly be determined by evaluating the man-hours needed. On one side this project could be considered small and not complex, since the development needed for fulfilling the thesis would be nearly as much as for a large project. However, the student was to develop a prototype of a part of a bigger system. The man hours needed for completing the whole system could justify a complex definition.

Further the number of users could be fairly high, to make it possible for the users to decide an arbitrary number of participants. This could increase system complexity.

The complexity and the amount of generated information was on the other hand difficult to foresee, since there were no system requirements made before the project started.

It was easier to define the project's degree of uncertainty. There were no defined inputs and outputs. And it was also impossible to determine if the users understood a system not yet planned or requested, and if the developer would be familiar with not yet selected software tools. This project is all in all not an assignment commissioned by the oil and gas sector.

To summarize, a high degree of uncertainty was associated with the future system. But if it would be complex, or not, was harder to define at this point. According to Dennis and Burns, either a prototype methodology or a mixed methodology should be chosen for the development process on this basis. And since prototyping lacks analysis and design stages, the methodology could be less suited for the task than a mixed methodology - this is a research project where such stages likely would be important.

The mixed methodology proposed by Burns and Dennis could be suited for this project, but a drawback with this technique is the lack of implementations by the development community. It seemed to be a proposal promoted in the 1980s, which was not used as much as other methods.

Finally the student settled with creating the software by using the RAD methodology. The methodology has several similarities with Burns and Dennis' method. Like Burns and Dennis' methodology RAD has an initial stage where the system is designed coarsely. Then the following stages are overall similar, since RAD also implements iterative development of software. Additionally RAD also is an established methodology with available toolkits supporting developers who would like to implement this way of work.

As stated in Section 2.3.2, there is no universal definition of how RAD development should take place. The methodology was therefore customized for this student project.

The users were firstly defined as to be the personnel at IFE, rather than offshore/onshore personnel in the oil and gas industry. As mentioned earlier in this section, the lack of possibilities for frequent communication with offshore/onshore personnel is the background for this choice.

In the initial design and analysis stage system requirements and prototypes

were developed. This process was done in cooperation with the contracting authority at IFE. Since this is a relatively small scale student project, it was not possible to establish persistent communication with several of IFE's research personnel. Instead the contracting authority primarily represented the users.

In the later prototype cycles, modules were developed iteratively in cooperation with the contracting authority. An iteration plan was developed for each iteration and delivered by email. Deadlines for the deliveries were introduced, in accordance with the RAD methodology.

When the student finished the project, and it was ready for testing, this was done in collaboration with IFE's research personnel, as later described in Section 3.5. The tests also aimed for investigating how well the system could fit into the oil and gas sector, and if it could be a good replacement for current meeting platforms and suggested prototypes.

3.1.2 Requirement Definition Methodology

Although the RAD methodology includes a phase containing shaping of requirements, the RAD templates found don't specify the process of defining requirements specifically. Since the requirements promoted for the new system was essential for how the software would be designed, an own methodology was selected for creating requirements - the Requirement Definition Methodology (RDM). This process was determined to be completed before the initiation of the project's RAD methodology - and the task of developing a requirement analysis.

It was decided that the requirement definition would be accomplished almost as described in the methodology. The student was to take the role of the requirement engineer, while researchers at IFE described the needs of a future meeting software. Since the personnel at IFE has extensive knowledge of the oil and gas sector, they were defined as expert customers - shaping such a system's needs.

In addition to requirements promoted by the researchers, requirements were also made by studying a major oil and gas company's internal guidelines for establishing collaboration and support rooms. Requirements gathered from the guidelines are presented in Appendix B.2. These requirements were in an early stage merged with the list of requirements based on the needs promoted by IFE's researchers, and later changed. Proposals for ranking them were therefore only promoted by the student himself at their time of writing.

It was not practically feasible to arrange several meetings for both preparing and eliciting requirements. Instead these meetings were merged together.

3.1.3 Complete Project Methodology

The project methodology may be seen as a combination of a customized requirement definition and the RAD methodology. Figure 3.1 illustrates the model described in Section 3.1.1 and 3.1.2.

The first package illustrated is the use of the requirement definition methodology, containing an initial indoctrination phase, where the methodology is to be described. This is followed by an iterative process for producing requirements.

After finishing the requirement definition the result was implemented into the requirement analysis document. This was the first encounter with the RAD

methodology. When the requirement analysis document had been produced, the first paper prototypes were developed. The next stage involved the iterative prototyping cycles, aiming for gradually producing software. These stages will be described more specific later in this chapter.

According to the described model the software produced should lastly be tested and implemented. But since this is a student project, the last implementation phase was never initiated. Below follows more specific descriptions of the conduction of the RAD methodology.

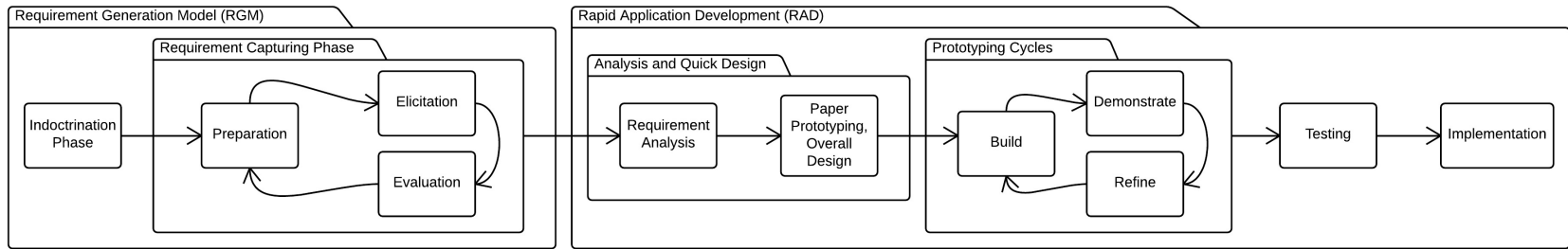


Figure 3.1: An illustration of the combination of the customized requirement definition and the RAD methodology used in this project. [20].

3.2 The initial phase - the analysis

In the analysis phase the current system and the system to be developed was analyzed. A requirement analysis document was written. The document is partly made according to a template used for RAD analysis ¹. The analysis is closely related to theory about usability, visualization, integrated operations, etc.

However, the template used didn't specify prioritization of the requirements defined. An analysis was conducted for ranking the requirements, whether functionality should be implemented or not in the future system. The MoSCoW method was chosen for this analysis.

3.2.1 MoSCoW Ranking of Requirements

Two researchers at IFE carried out the prioritization of requirements defined in an earlier project phase. This work was done individually by each researcher, resulting in two, independent analyzes. By considering the priorities given a final list of "must have" requirements was formed. The technique used was fairly similar to the MoSCoW technique, but with custom phrases.

The reason why not one, but two reviewers analyzed the requirements was to hopefully decrease the possible downsides promoted by critics as Kukreja et al. [17] in Section 2.4.2. By letting more than one researchers conduct their analyzes independently, the risk of creating priorities based on instinctive and wrong judgments would probably decrease. The requirements ranked as very important were therefore not added to that distinct category if not both researchers had ranked the same requirement as a crucial functionality.

3.2.2 Evaluation of Current Systems

Before ending the thesis' requirement analysis phase, currently developed prototypes and systems' functionality was evaluated and compared with the list of requirements. This work was conducted by studying available system documentation found online.

When this study was conducted, it also became clear that several functions, implemented in the evaluated systems, also could be implemented in the future application. A table containing solutions considered desirable in the future meeting system has been added to this part of the project. It is also mentioned in the table if the functions found have been implemented in the prototype concept.

¹<http://www.bruegge.informatik.tu-muenchen.de/twiki/bin/view/OOSE/RequirementsAnalysisDocumentTemplate>

3.3 Quick design

After conducting the requirement analysis, prototyping was conducted. However, it was decided to implement both offline and online prototyping techniques, as described by Beaudouin-Lafon and Mackay in Section 2.5. The authors argue for creating offline prototypes in an early stage, for later continuing with online prototypes, for example developed in code.

The prototyping in the quick design part therefore proceeded by producing simple, paper prototypes of the future system. These were designed as horizontal prototypes, covering as much of the system's functionality as possible.

This in accordance to Beaudouin-Lafon and Mackay's recommendations, since simple and rapid to develop representations should, according to the authors, take place in a project's initial stage. The later, online and more functional prototype was therefore developed in the prototype cycles, described in Section 3.4.

The initial prototypes were however made in close cooperation with the contracting authority at IFE. The prototype was changed several times during the stage, to make the result as good as possible.

Simultaneously while new design was developed and specified, work with the data model was conducted, to prepare the prototype for actual development.

Both this work and the design of prototypes are in compliance with the selected project development model.

3.4 Prototyping Cycles

3.4.1 Prototype Fundamentals

After developing a paper prototype the process continued by migrating the prototype to a digital tool where simple, not-interactive design was implemented. Since this also was easy to conduct, without any need of coding, the process of creating a horizontal system prototype continued. The tool selected for the digital prototyping was the free JustInMind² prototyping toolkit, resembling a CASE tool, but without options for exporting projects for further development.

While producing this digital, but simple prototype, changes were made in the project's path. Particularly one choice, inspired by the existing software Epsis Teambox³ (described in Section 2.2.1), affected the overall system design. This choice consisted of abandoning the idea of developing internal functionality to display data. Instead the system was decided to be sort of an overlay to other systems, making it possible to conveniently interact with them during a meeting.

After several iterations in the prototyping the tool, a deep dive producing a functional prototype was conducted, before continuing with the overall design. An online prototype technique was chosen, but only some parts of the system was included in the process. The strategy for producing the representation went therefore temporarily from a horizontal to a vertical prototyping strategy.

The reason why the strategy was changed at this stage was the resources available for producing the system. A horizontal strategy may often lead to gradually expanding the representation into a full, working system, making use of the entire design and development team. Such resources were not available in this project. Instead a vertical strategy could prove if certain parts of the design could be implemented in the future system. The work revolved around making the main user interface, only of the system's meeting module, to work. It was also developed more complex functionality as changing a map view of an oil installation and capabilities for drawing and writing on digital whiteboards. These part of the horizontal design were therefore tested, before the overall design process could continue.

This part of the prototyping was conducted by creating a web site containing JS and PHP scripts, XML, CSS and HTML. This approach was chosen to make the project available online to the supervisor and other stakeholders at IFE. Frameworks as ExtJS⁴ could have been selected to make it easier to develop following recognized standards as MVC⁵ or MVVM⁶, and to make the website project fully object oriented. But since frameworks as ExtJS may have a steep learning curve, these suggestions were abandoned. Instead the project only made use of the JQuery⁷ and Bootstrap⁸ libraries (Bootstrap was abandoned in the second version of the functional prototype).

One of the drawbacks of abandoning ExtJS, and later Bootstrap, implied possibly making it harder to achieve cross-browser compatibility. The project

²<http://www.justinmind.com/>

³<http://www.epsis.no/product/>

⁴<https://www.sencha.com/products/extjs/#overview>

⁵<https://msdn.microsoft.com/en-us/library/ff649643.aspx>

⁶<https://msdn.microsoft.com/en-us/library/hh848246.aspx>

⁷<https://jquery.com/>

⁸<http://getbootstrap.com/>

has however only been tested in the Google Chrome⁹ browser, where it works as intended.

Implementation of a server-side database was omitted to make the project more transferable, if the later users don't possess the required database system. XML was instead chosen for storing data server-side. Such documents are created by PHP-scripts. The XML is also accompanied by XSD-schemes to make it easy for later users to know which attributes and elements they are allowed to remove or add, as well as to understand the structure of each XML document.

3.4.2 Iterative Development

According to the RAD methodology, described in Section 2.3.2, the prototyping cycles are closed in a loop producing new software for each iteration. This consists of three, embedded stages - build, demonstrate and refine. In the first stage software is developed, before demonstration is conducted in the second stage. If the development process should continue after the demonstration, the loop continues with the refinement stage before a new iteration is initiated. Otherwise the loop breaks after the second stage.

This method fitted well with the current prototyping, since demonstrations were conducted in collaboration with the project's supervisor. Any changes suggested during supervisions were later refined in the third stage before new functionality was implemented in the next iteration.

During the prototyping cycles, producing the functional prototype, modules of the user interface were produced. All suggested functionality was however not implemented. As mentioned above, a test bed consisting of a window containing a menu with a toolkit, an interact-able map surface and a whiteboard was developed - solely for testing purposes.

Several findings were made, during the vertical prototyping, which resulted in a need to redraw the overall, horizontal prototype. The last iterations were therefore conducted with a horizontal strategy - aiming for the implementation of missing features.

⁹<https://www.google.com/chrome/browser/desktop/index.html>

3.5 Testing

Shneiderman and Plaisant describe the use of both usability testing and experts reviews in Section 2.6. In this project the usability testing was excluded since it has not been possible to find relevant participants for such experiments. The end-users of the system are thought to be onshore and offshore personnel working in an IO oil and gas organization. Without possibilities for contacting this user group, it was decided early on to focus more on experts in the current domain.

The available expertise could however be considered highly adapted for the tests following the experts reviews methodology. These are researchers at IFE who are daily involved in projects linked to the domain, and who have conducted several studies of usability of software systems in the oil and gas sector. Shneiderman and Plaisant's requests for involving expertise with knowledge of the domain and the selected heuristics.

However, the use of tests during the system development differs slightly from the model presented in Figure 3.1. Although a major test was conducted after finishing a sufficient amount of development cycles, small and informal tests have been conducted both during the development cycles, and the earlier quick design stage. The project's supervisor is currently employed at IFE, and has extensive knowledge about both system development and oil and gas organizations. Experts reviews have therefore been conducted frequently during the design and development stages. But since these tests have been of a more informal nature, they were conducted by asking the supervisor about his opinion about implementations. As Shneiderman and Plaisant point out, such informal experts reviews may provide useful feedback, although more formal tests produce even better results.

After ending the prototyping cycles a major test was therefore conducted in collaboration with two researchers at IFE. As Shneiderman and Plaisant recommend, one of the participants had less experience with the project work conducted than the other.

According to Holligsted and Novick, it could be beneficial to conduct tests with a combined use of usability inspection methods, as described in sub Section 2.6.2. The major test was therefore including both the use of heuristic evaluation, guidelines review, cognitive walk-throughs and a more formal usability inspection. But since cognitive walk-throughs and formal usability inspections seem to be less frequent in use by usability professionals today, than earlier, these parts of the test were prioritized less than the heuristic evaluation.

The test started with presenting the developed concept for the participants who were asked to perform a cognitive walk-through of central parts. Then a part containing both the heuristic evaluation and the evaluation of guidelines was conducted. This part consisted of reviewing the current solution with respect to the Eight Golden Rules of Interface Design and the project's requirements. Lastly a formal usability inspection was conducted by letting the experts suggest functionality to change, exclude or implement in the future.

Chapter 4

Observations and Experiments Prior to Concept Development

This chapter firstly contains observations done during the project and descriptions of executed experiments before the software development was conducted. Lastly suggested functionality to implement in a future application, based on the study of other systems, has been added.

The chapter does not contain information about the prototype development process, although the activity could be considered as experiments. Since this was both a lengthy and a major process, an own chapter has been dedicated to prototyping. Chapter 5 firmly describes both the paper, digital and functional prototypes developed in iterations.

The current chapter's sections are organized as follows:

- *The Studied Email Platform:* This section contains gathered information about an email platform earlier used for morning meetings at an operating company in the oil and gas industry. A specific meeting document has been studied, and each part of the document is firmly described.
- *Defining Requirements for Future Software Supporting Morning Meetings:* The section contains information about the requirement definition process, including a table displaying groups of the requirements. Requirements found crucial in the new system, by use of the MoSCoW method, are also described.
- *Evaluation of Existing Software's Functions:* Here an evaluation of existing systems is presented. The evaluation was conducted by comparing these systems with the defined requirements.

4.1 The Studied Email Platform

4.1.1 Findings from Studying an Email Platform Document

This thesis is investigating if information flow in offshore/onshore morning meetings may be handled differently than today. An email sent internally in an oil and gas organization, containing meeting information, may contribute to the design of the meeting software. This email has more concretely been used as an agenda, and a shared platform during such a meeting back in 2008.

The organization of morning meetings in other companies have not been considered, although an agenda used in a second company was swiftly examined. This agenda was quite similar to the studied one.

Since the studied morning meeting document is confidential, it has not been possible to present the actual content in this thesis. But permissions have been granted for reproducing some of the presented information - in an anonymous manner.

The document is a static email containing information written directly into it and screen dumps obtained from different types of software. The document is divided into eight sections. These are:

1. HSE (Health Safety Environment).
2. Production.
3. New notifications and work orders.
4. A review of the operational plan, including logistics.
5. Drilling.
6. New actions from a handover meeting, earlier the same day.
7. Focus points before a well meeting, later the same day.
8. Status of ongoing actions.

The next sub sections will briefly describe the content in each of these eight parts.

HSE

This section begins with a general description of new events affecting HSE. This information is written directly into the email.

Then a screen dump of a software presenting current cases with deviations is shown. These cases are mostly physical deviations, as cracks and leaks in objects or structures, but also notifications about upcoming events having impact on HSE, as a mentioned hygiene inspection.

After the screen dump a new section with written information follows. This includes a list of the different platform teams' focus on HSE in upcoming work. Then a list describing lift activities is presented. The written part ends with information about the operation, and mentions interruptions too.

Production

This section presents a screen dump from an EXCEL document containing an overview of the oil production at the offshore installation in table form.

Notifications and Work Orders

This section has an own walk-through in SAP¹ (this is described at the beginning of the part). Then a screen dump of notifications and work orders are presented in the email. The screen dump is of a SAP table. The columns describe inter alia notification IDs, functional location and descriptions.

Review of Operational Plan

This contains an excerpt of a SAP table as a screen dump (a text above the table describes when such screen dumps are normally taken - on Mondays or Fridays). The table contains a listing of work to be done. The listed work in the table are mainly modifications and repairs.

The table is then followed by an activity description. This is also on table form and contains information about activities to be done, priorities, responsibilities and start and finish time.

The tables are followed by a weather forecast, which is also a static screen dump. This illustration contains a long term plot of the weather at the installation. It informs about announced swell periods, wind-wave periods, swell height, wave height, wind speed and swell as well as wind and wind-wave direction.

Lastly the section contains information about helicopter and boat traffic to the installation. A screen dump from a software used to show such traffic is included in the document. This is also on table form informing the meeting participants about the helicopter's flight number, the time of flight, call sign, type and the number of vacant seats to and from the platform.

Since there is no reported boat traffic, only a message of this is written directly into the document.

Drilling

This section contains both a screen dump from a software used for drilling, and information written into the document. The written information is about the status of the current drilling operation, and new events. It also contains a part about planned work.

The illustration is a colorful overview of a drilling pipe and its location in the ground below sea level.

New Actions from Handover

The part contains only written information. An explanation is given at the beginning, which says that the section also includes information about coordination meetings the same day. It is also specified that critical operations are written with red color, and clarified operations in green.

The section further contains information written in blue. This information is about some tools that will be transported with the helicopter flight later that day, and the time of a meeting.

¹<http://go.sap.com/index.html>

Focus Points Before Well Meeting

Nothing is mentioned here, besides information about a meeting between different parts of the organization.

Status of Ongoing Actions

Like the section above, this only contains information written directly into the document. A lot of internal abbreviations are used. As the meeting part described in Section 4.1.1, text in this section is also colored red or green based on priority and clarification.

The section describes the status of the current operations and who are responsible for completing them.

4.2 Defining Requirements for Future Software Supporting Morning Meetings

This section contains an analysis of requirements defined in the project. The goal of this analysis was to define a number of key requirements for comparison with the systems they should be compared with in Section 4.3. These efforts are described in Section 4.2.1. In addition the section contains a review of the conducted MoSCoW method for ranking requirements.

4.2.1 Defining Key Functionality

The requirements build upon needs forwarded by researchers at IFE in a workshop early in the project. In addition some requirements were formulated on the basis of a study of guidelines for establishing collaboration and support rooms.

In Appendix B requirements for a future system are defined. Some of the requirements describe specific functions in detail. But it is irrelevant if current software have identical functionality to the defined requirements, if only insignificant elements are compared.

To cope with this possible pitfall the requirements were put into main categories. The current software and prototypes' implementation of these categories are described in Section 4.3.1.

The below Table 4.1 contains the main categories and their accompanying requirements.

Id	Title	Requirements
1	Two-way communication with users operating external mobile devices during collaboration sessions.	0002, 1000, 1001, 1002, 1003, 1004, 1005, 1006, 1007, 1010, 1011, 9001, 9002, 9003, 9004, 9007, 9008, 9009, 9010
2	Sharing of view with mobile devices during collaboration sessions.	1008, 1009, 9005, 9006
3	Possibilities for registering information as for example keywords during meetings.	1100, 1101
4	Must take the existing meeting roles into account by being designed for both meeting participants, meeting leaders, etc.	1200, 1201
5	Multiple modes for varying system flow based on current state in the meeting process. Examples include an editor, a meeting mode and a report mode.	1300,
6	Possibilities for editing and prepare a future meeting's setup.	1301, 1302, 130, 1304, 1305, 1306, 1307, 1308, 1309
7	Implementing options for presenting real-time data.	1310
8	Implementing possibilities for presenting fixed data.	1311

9	Implementing an own mode active during meetings, especially designed for such a stage in the planning process.	1312, 1313
10	Implementing an own mode active after meetings, able of producing minutes or other kinds of reports.	1314, 1315
11	Implementing modes for different types of devices to make the interface user friendly, regardless of device/monitor used.	1316, 1317, 1318, 1320
12	Collaboration participants should see the same platform view regardless of physical location or monitors in use.	1319
13	Sharing of cursor.	1400, 1401, 1402, 1403, 1404, 1405
14	Agents for decision-support	1500, 1501, 1502, 1503, 1504, 1505
15	Page hierarchy with maximum of two levels.	1600, 1601, 1604, 1605
16	Register and time meeting attendance	1602, 1603
17	Functionality for open other systems.	1606
18	Present clear navigation information during interaction with system, as well as when other systems are active inside the software's context.	1607, 1608, 1801
19	Having an overview of personnel competence, and possibly also use the competence as decision-support	1700, 1701, 1702
20	Easy to use system stripped of unnecessary functionality.	1800, 1802, 1803, 1804, 1809, 1813
21	Understandable and intuitive automation.	1805
22	Clear system feedback, including graphical feedback.	1806, 1807, 1808, 1810
23	Measures for focus as warm and cold colors for inter alia setting priority.	1811, 1812
24	Implementing functionality capable of giving readability feedback based on the current adjustments.	1814, 1815
25	Readability test	1816
26	Implementing possibilities for readability adjustments.	1817, 1818
27	Pre-selection of options when possible, to make it easier for users to follow ordinary system flow.	1819
28	Support for display reduction.	1820, 1821
29	Different complexity levels enabling both expert and novice use.	1900
30	Notifications about important news.	2200
31	Local display settings, as brightness, color and contrast	2300, 2301, 2302
32	A system capable of taking available hardware in current meeting rooms into account.	2400
33	A system capable of sharing the required information in morning meetings.	2500

34	Whiteboard functionality.	2600
----	---------------------------	------

Table 4.1: Main requirement categories

4.2.2 Importance of Functionality in the Future System

When reviewing requirements two IFE's researchers went through a list of such needs. The researchers are anonymously presented as "reviewer 1" and "reviewer 2". During the review the researchers gave feedback on requirement formulation, and rated each of them based on importance. Although they didn't made use of the phrases "must have", "should have", "could have" and "would like to have", as in the MoSCoW method, they used quite similar phrases: "Important", "should implement" and "nice to have". The following sections present their opinion about the requirements for the future meeting system:

Communication with mobile devices

Communication with mobile devices was rated as important in a future system. However, according to the document, the scientists had different opinion about how such a solution should be designed in details.

Reviewer 1 emphasized the importance of being able to notify third-part mobile device users, and receive messages from them, and rated these functions as mandatory. But the reviewer gave lower priority to functionality making it possible for meeting attendees to communicate with third-party mobile users, by rating this as a "nice-to-have" need.

Reviewer 2, on the other hand, rated all the mentioned functions as something that should be implemented in a future system, but not as mandatory system-parts.

Implementation of Agents

Reviewer 1 and reviewer 2 slightly disagreed on how important it is to integrate agents in the future system. Reviewer 2 meant that agents, including what-if-agents, should be implemented in the system, but defined it not as mandatory. Reviewer 2 meant that such functionality was nice to have, but that other parts were much more important.

However, both scientists thought that if agents should be implemented, it is important to ensure that the expert system is user friendly. They emphasized that such a system must have to be intuitive, and that it is important to easily add and understand information. Lastly the system must explain to the user how reliable the information output is.

Overview of Competence

None of the scientists ranked this functionality as mandatory. Reviewer 1 thought it sounded ambitious to make the system able to control if required competence is present in meetings, when tasks are discussed. Reviewer 2 stated that if such functionality is to be implemented, the system must be able to store data about competence requirements in information points that should be

discussed during meetings. If so reviewer 2 meant that this functionality should be implemented, but not as a mandatory system part.

Readability Feedback and Tests

Both scientists pointed out that text must be readable for participants when using the future system. They also thought that manual readability adjustments must be implemented, to in compliance with rules regarding universally usable.

The readability test, however, was thought to be a good idea, but was not rated mandatory. However, both reviewer 1 and reviewer 2 stated that the test should be implemented in the future system.

Automatic readability adjustments, based on distance to the monitors in the meeting rooms, was not rated as important. Reviewer 1 thought that such functionality was too ambitious. Reviewer 2 noted that this functionality possibly could be in conflict with other, automatic adjustments, if they were to be implemented.

4.2.3 Requirements the Reviewers Defined as Important

Although the reviewers had different opinions about some requirements, both defined several requirements as important for the future system. To summarize the experts unanimously gave top-priority to the following functionality (complete descriptions are presented in Table 4.1):

- *Id 1*: Functionality for contacting users operating on mobile devices.
- *Id 3*: Possibilities for registering information during meetings.
- *Id 9*: Functionality for displaying the same platform to all users.
- *Id 10*: Functionality for producing reports.
- *Id 16*: Functionality for registering meeting attendance.
- *Id 20*: An easy to use system.
- *Id 21*: Understandable automation.
- *Id 22*: Clear feedback.
- *Id 26*: Possibilities for setting readability adjustments manually.
- *Id 32*: The system must take available hardware into account.
- *Id 33*: The system must be capable of sharing the required meeting information.

4.3 Evaluation of Existing Software's Functions

In Section 4.2 requirements for a future meeting system were gathered. But to find the actual needs for developing such software, existing systems and prototypes were decided to be evaluated. This work is presented in Section 4.3.1.

The same section contains a table presenting a list of key functions, and columns describing implementations of these functions in existing systems. The table also includes a review of the systems' functions, since some of the discovered functionality was also found to be useful in this project's prototype. A list of these elements were therefore added.

4.3.1 Current Implementations of Key Functions

In this section current software systems' fulfillment of the list of defined requirement categories presented in Table 4.1 has been evaluated. However, the available information about the software systems has been varying, making it hard to tell if some functions are implemented or not. The evaluation should therefore not be viewed as a complete evaluation of systems. The data could instead be considered as indications of areas supported or not by currently available software.

The findings are presented in Table 4.2. Below follows a brief explanation to the table's organization:

- *Ids*: The data used in the leftmost table column is IDs representing requirement groups evaluated. These are not described in this table. Descriptions are however presented in Table 4.1.
- *Systems/prototypes*: Systems and prototypes are mentioned as headers above each column, except the first. Information in each of the cells in these columns explains if a requirement is implemented or not.
- *"?"*: Question marks are used in cells to tell the viewer that either documentation was found to be unclear, or if implementation of the current requirement may be discussed.
- *"-"*: The dash symbol is used when a requirement is found evaluated as not relevant.

Below follows descriptions about the sources used for gathering information about the evaluated systems and prototypes:

- *ARKit*: Information was gathered from Ark Platform's homepage ², and by ordering a pdf document containing functionality (the pdf was ordered from the same site).
- *Epsis Teambox*: Epsis AS's homepage ³ was used for information gathering by watching video tutorials and reading published descriptions of functionality. Particularly the user manual [4] was a helpful document in the evaluation.
- *IO-MAP*: A report [35] about the system has been used for the evaluation.

²<http://www.arkplatforms.com/>

³<http://www.epsis.no/>

- *ProtoSphere*: Information was gathered from ProtonMedia's homepage ⁴ by investigating published documents.
- *Wisio*: Wisio [25] was evaluated by reading the report published at Østfold University College's sites.

4.3.2 Functionality Based on Other Systems' Solutions

In addition to the proposed functionality, each of the analyzed systems had implemented functions not included in the requirements list. This section contains a review of the current systems' functions that could act as a source of inspiration to the future meeting system.

Teambox

Epsis Teambox [4] has implemented modes adapted to the work process. These are globally available from the main window, and may be accessed by clicking on described symbols. By making modes globally accessible, like in Epsis' solution, the modes of the future software system could be easy to access.

Further the modes are not linked especially to morning meetings. They are instead customized to a more general use. The following modes are accessible; "Workflows", "Environment", "Conference", "Collaboration", "Publisher" and "Sources". By creating a more general set of modes, the future system could be used by a broader user base.

It should also be noted that Epsis' "Workflows" and "Environment" modes could be useful for organizing future collaboration sessions. The "Workflows" mode is used for adding information, such as documents, to different steps of for example a collaboration session. The "Environment" mode could later be used for storing changes to the layout of this session.

Another functionality that Epsis Teambox has implemented, which could be transferred to a future application, is the ability to present various information in windows. By doing so the interaction with other systems is left to the user to decide.

ProtoSphere

It could be argued that ProtoSphere ⁵ is not fitted for ordinary collaboration sessions. The application lets individual users control avatars in virtual meeting environments. How this approach could be integrated in a meeting setting, is not further discussed.

However, the solution has some major advantages. The participants in virtual meetings may for example easily change between meetings they are participating in. Managers and other key personnel may therefore be able to easily switch between several meetings.

Another functionality that could be implemented in the future application is the use of a contact list, presenting available personnel. This list could be used for communicating with not-participating personnel on for example mobile devices. It could also be used for registering attendance.

⁴<http://www.protonmedia.com/>

⁵<http://www.protonmedia.com/>

Id	Protosphere	Mail surface	ARKit	Epsis	IO-MAP	Wisio
1	Yes	No	No	?	No	No
2	No	No	No	No	No	No
3	Yes	Yes	Yes	Yes	Yes	Yes
4	Yes	?	Yes	Yes	Yes	Yes
5	Yes	No	No	Yes	No	No
6	Yes	Yes	No	Yes	No	No
7	Yes	No	Yes	Yes	Yes	Yes
8	Yes	Yes	Yes	Yes	Yes	Yes
9	Yes	No	No	Yes	No	No
10	Yes	No	No	Yes	No	No
11	?	No	?	?	?	No
12	?	?	?	Yes	?	?
13	Yes	No	No	?	No	No
14	No	No	No	No	No	No
15	Yes	Yes	Yes	Yes	Yes	Yes
16	Yes	No	No	No	No	No
17	?	No	No	Yes	No	No
18	Yes	-	Yes	Yes	Yes	Yes
19	No	No	No	No	No	No
20	Yes	Yes	Yes	Yes	Yes	Yes
21	?	No	?	?	?	?
22	Yes	?	Yes	Yes	Yes	Yes
23	-	Yes	Yes	-	Yes	Yes
24	No	No	No	No	No	No
25	No	No	No	No	No	No
26	Yes	No	?	Yes	No	No
27	?	-	?	?	-	-
28	No	No	?	Yes	?	No
29	Yes	No	Yes	Yes	Yes	Yes
30	No	No	Yes	Yes	Yes	Yes
31	?	No	No	No	No	No
32	No	Yes	Yes	Yes	Yes	?
33	?	Yes	?	?	No	No
34	Yes	No	No	No	?	No

Table 4.2: Implementation of requirement categories in evaluated systems.

IO-MAP

Something that has not been discussed much to this point is how the future application should look like. IO-MAP's [35] solution should be considered. Here tasks on a platform may be visible in a map environment, consisting of graphical representations of platform decks. This makes the user able of visually get a grasp of the jobs to be done, conflicts between tasks, etc. by just looking at the map. This could be a greater challenge if the users had to look at table presenting task data instead.

The future application could therefore partly or completely adapt a map environment to present meeting information.

Another property of IO-MAP is that some information is always visible, like meteorological data. The use of global information should also be further discussed in the future application.

Wisio

Wisio has the same advantages as IO-MAP, since this is a map solution that has been inspired by the previously mentioned one. However, there are some differences.

In addition to the information displayed on the map representations, Wisio also presents the user with a time line where each of the tasks, as well as transportation information, is shown. This functionality could be implemented in the future application when presenting information on upcoming tasks and means of transports. However, it should also be tested if this information could fit into a suitable window, and that no important information is hidden from the users.

Wisio also displays rolling text with important notifications. This is globally available information. The future application could have this functionality integrated, or a notification solution more like the one presented in Section 4.3.2.

ARKit

As WISIO and IO-MAP - ARKit also presents the user with map data. The map is however a real image, with information not only about tasks, but also available personnel, etc. It is possible for the user to move the location of the map and to zoom in or out. The users may also edit the map image by adding graphical information in different colors when for example illustrating an oil field.

ARKit's map solution could be integrated into the future application. However, this zoom-able map could be more suitable for a global view of an onshore petroleum business than offshore businesses. If both vendors and onshore and offshore operations should be visible in such a map, a huge map may be implemented. This is a major contrast to the map needed for presenting information about an oil platform. Further oil platforms has several decks, making this solution less attractive.

A more suitable functionality to implement is ARKit's notification system. The application is capable of displaying newly added documents and comments in a notification list, which may also be sorted or limited. Such functionality may be used in the meeting system to access documents of importance and to display important messages during meetings.

Summary of Functionality Discovered in Current Solutions

Table 4.3 summarizes all the functionality discovered in systems and prototypes which could act as sources of inspiration for the future system.

ID	Description
1	Epsis Teambox have several global modes linked to a meeting's work process.
2	Epsis Teambox displays external software in windows customized by the user for each of the meeting's steps.
3	Epsis Teambox implements a large cursor, if the users need such object for increased readability.
4	Protosphere implements avatars for each of the meeting participants.
5	Protosphere may display a meeting's participants and available personnel in a contact list.
6	IO-MAP displays tasks in a map environment of the current off-shore installation.
7	IO-MAP has integrated information which is always visible for the users.
8	WISIO has a time line presenting tasks on the current installation.
9	WISIO presents important information as rolling text.
10	ARKit displays a zoom-able map of the current organization's locations (zoom functionality is also integrated in IO-MAP and WISIO).
11	ARKit has implemented a list of notifications, which may display important information and documents. The visible elements in the list may be sorted or limited.

Table 4.3: This table presents functionality found in evaluated prototypes and systems, which could be implemented into the future meeting system.

The features discovered in this part of the project could have been added to the list of requirements for a future meeting system. However, not all of the solutions were considered favorable. But some were, and these are described in the upcoming prototyping cycles.

Regardless the number of suggestions, at this point in the project the preparatory work was considered thoroughly enough for commencing the initiation of the prototyping cycles. These are described in the upcoming Chapter 5.

Chapter 5

Concept Development

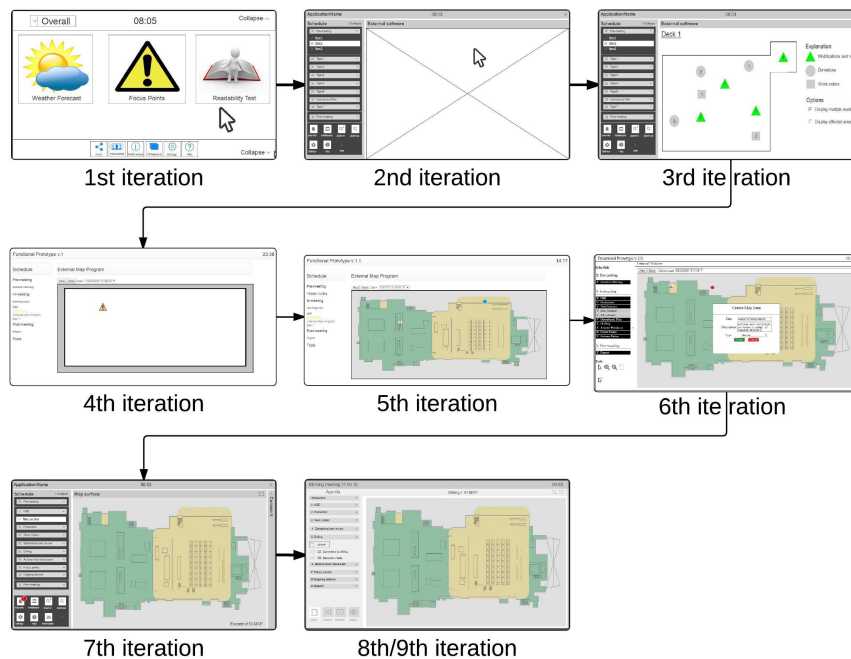


Figure 5.1: The concept was changed in nine iterations. The gradual evolution of the design is illustrated in this figure. The design in the fourth, fifth and sixth iterations were developed in code.

This chapter contains the complete process of developing the project's prototype. It includes both choices made during the quick design phase and the later prototyping cycles.

During the prototyping the system requirements defined were tried to be fulfilled. If some proved to be hard to implement, this is described. Additionally the evaluation of other systems resulted in producing a list of desired functionality that could be implemented in the future system. These functions are

presented in Section 4.3.2. If such functionality is implemented in the prototype, this is pointed out.

Theory about visualization, mentioned in Section 2.7, and principles for user interface design (Section 2.8) have been integrated into the prototypes. Therefore both principles and theory regarding user interface design has been an important contribution to the development process. Such choices are mentioned in the parts describing the prototype development.

The actual development took place in several iterations. As mentioned in Section 3.4.2, this process was conducted as described in the overall RAD methodology. Demonstrations of developed material were regularly presented to the supervisor, before possible refinement was conducted, and a new cycle began. The results of each of these iterations are visible in Figure 5.1.

However, the concept development was conducted according to two prototyping strategies. The initial, horizontal strategy creating a global design, and the later vertical strategy implementing a subset of the future system's functions and design. Since the outcomes were produced on the basis of highly different strategies, both results were implemented in the final test, described in Section 6.1.

5.1 Quick Design - Paper Prototyping

Before creating the initial paper prototype overall functionality and screen types to be implemented were listed. Here the functionality considered global was separated from internal functions.

The following functionality was considered to be globally reachable:

- A list of participants and possibilities for changing the list.
- A clock to clearly display time for participants.
- Notifications.
- Settings.
- Help.
- A whiteboard.

The reason why this list of functionality was decided to be globally reachable was that the users could have a need for taking benefits of the functionality at any time. The list of participants should be changed if some was to leave a meeting. This could happen at any point during a meeting. The clock could help the participants keep track of time. Notifications could possibly appear at any time, also during a meeting. Settings should be changed if needed. A help view should be accessible at any time, if help was required. A whiteboard functionality should also be accessible at any to make the meeting leader capable to sketch explanations, etc., related to the presented information.

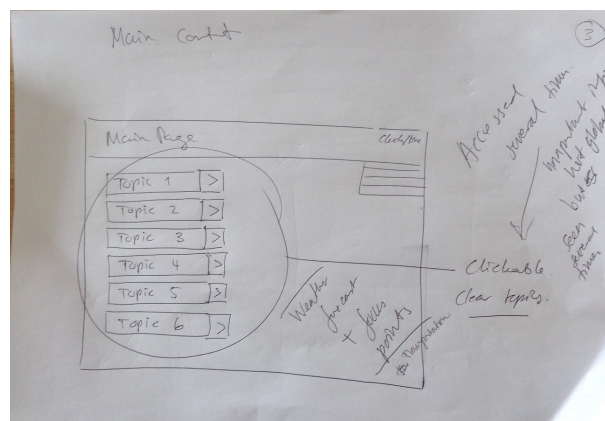


Figure 5.2: A screen made for the initial paper prototype.

The paper prototype was made by drawing screens to present the system flow during a morning meeting. An example of such a screen is shown in Figure 5.2.

When designing all sites of the paper prototype the list of topics presented in the email platform, described in Section 4.1.1, was used for determining the content of each site. A problem with this approach was that it was hard to fit all the required information into single sites for each topic. The idea of creating sites for topics was therefore abandoned in favor of splitting the topics into user-defined sites. This solution was implemented in the first digital prototype, presented in Section 5.2.

5.2 Prototyping Cycles - Horizontal Prototyping

As mentioned in Chapter 5 and Chapter 3, the concept development was ultimately divided into three stages - based on their underlying prototyping strategy. Each contained several iterations. The first followed a horizontal approach, designing overall prototyping functionality. This stage is described in this section. The second, following a vertical strategy, is described in Section 5.3. Then a new state, refining the horizontal prototype, was added. This last stage is described in Section 5.4.

Further in this section the prototyping iterations are mentioned as sub sections. Choices made and argumentation for those selections are described for each of the iterations.

5.2.1 Iteration I

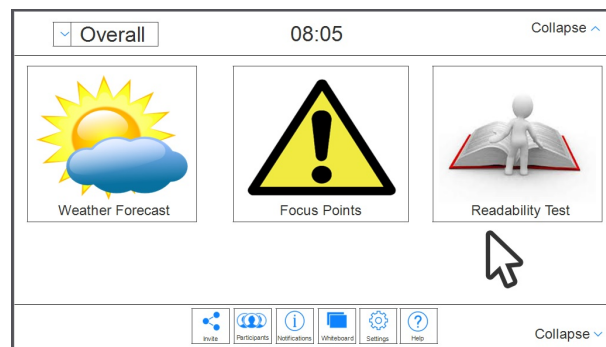


Figure 5.3: The initial digital prototype's main screen.

The first prototype was partly inspired by Epsis Teambox, a software briefly described in Section 2.2.1. The major similarities were the use of links to external software and a huge mouse cursor - for increased visibility in a collaboration session. The main page of the initial prototype is shown in Figure 5.3.

The prototype was thought to support an arbitrary number of documents and internal/external software in each of the sites. The requirements mentioned may however specify more than the selected groups in Table 4.1. A complete set of requirements is however presented in Appendix B.1.

Fulfilling requirements

The requirements set to the project were important. Several of the main categories, earlier listed in Table 4.1, were thought of in this process.

As Figure 5.3 shows, the prototype's page hierarchy was fairly shallow with a drop down menu to navigate between topics. Each topic contains large, square buttons for reaching individual sites, documents, etc. The requirement set for restricting the page hierarchy to two levels could therefore be accomplished.

Further the prototype could implement readability tests by adding these as sites to topics pages, also shown in Figure 5.3.

The requirements state that the future system should be stripped of unnecessary functionality, and be able of sharing the required information in morning

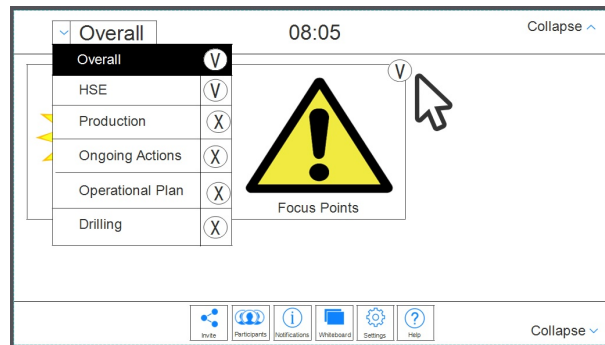


Figure 5.4: Selection of topics in the initial prototype.

meetings. As Figure 5.4 displays, accessing the drop-down menu could present users with topics used in today's morning meetings. These would be displayed as accessible pages, possibly fulfilling the goal of presenting required information. Further the top bar was thought to be stripped of unnecessary functionality, making inexperienced users capable of navigating between sites by only accessing the drop-down menu and the subsequent sites below.

The bottom bar contained several choices. These choices were thought to be global, and accessible from all parts of the application. One of these options could present the users with a list of participants in the current session. Figure 5.5 shows the participants, their current collaboration group (offshore, onshore, or another group), how they are joining, etc.

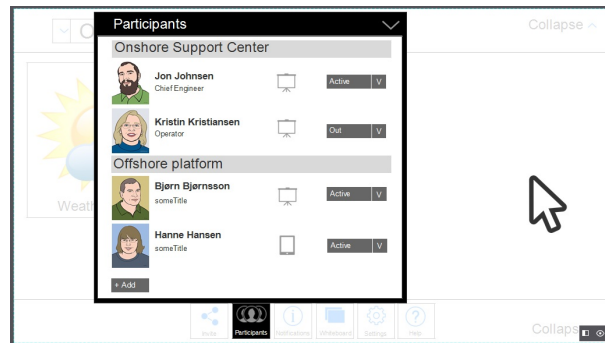


Figure 5.5: A pop-p window containing participants in a collaboration session.

The main purpose of the participants window was to make the application's users capable of registering and time meeting attendance - which are also system requirements. This was thought to be done by selecting the drop-down menu to the right of each user. The meeting leader could then be capable of deselecting participants if someone left the group meeting, or to set participants as active users.

Although communication with mobile devices was not prioritized in the design of this initial prototype, the participants window was thought to be capable of displaying information of how personnel were communicating with each other. The symbol to the right of the participants' title and name was thought to be

visualizing a tablet if the current person was operating such device during the collaboration, or similar for other kinds of devices.

The requirement for having an overview of the personnel's competence, and possibly use the competence for decision-support, was only partially met. It was thought that the titles below the participants' names in the participant window could serve the purpose of displaying competence to participating personnel.

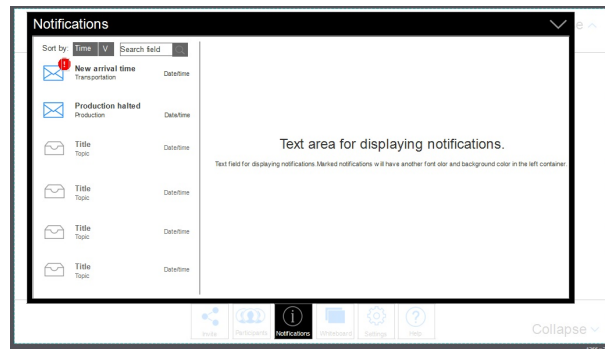


Figure 5.6: A pop-up window containing notifications in a collaboration session.

Another selectable option in the bottom bar was the notification option. This choice could make the users capable of reading notifications, as shown in Figure 5.6. An exclamation mark was also thought to be displayed on the bottom bar if new notifications had arrived, whether the bar was collapsed or not. These notifications features could help fulfilling the requirement of displaying important news - as notifications.



Figure 5.7: A pop-p window containing a whiteboard.

Lastly the prototype attempted to fulfill the requirements connected to the whiteboard functionality. A whiteboard should be globally accessible from all parts of the application, as shown in Figure 5.7.

The whiteboard could let users either draw or write directly onto the surface. Several whiteboards could also be added by pressing the plus-symbol. The whiteboards were then thought to be added as tabs. However, none could be deleted during a session. This choice was made for preventing any information from being lost.

Design principles used when creating the digital prototype

The interaction styles thought of when designing the initial prototype was mostly one-dimensional menu selection, although direct manipulation was considered for the editor. But since no editor was designed at this stage, the direct manipulation alternative was shelved.

As for the golden rules of interface design several were considered.

For example the first rule, *strive for consistency*, was thought of when designing the up- and down-symbols for expanding or minimizing actions. A cross-symbol was for example considered, but not implemented, for closing the global pop-up windows. A down-symbol was used instead, perhaps making it easier for users to understand that changes could be saved after minimizing the window. The up- and down-buttons were also commonly used other places in the application's design, and could therefore be considered as well-known symbols.

Further the second golden rule was thought of when designing the interface. Text was intentionally large and the mouse pointer's size was also increased to cater for universal usability. Adjustments to these settings could be done in a globally accessible setup window. A help site was also designed for being globally accessible, if users had problems during their interaction with the system.

Review of the initial prototype

The design of the first prototype was discussed during a guidance meeting with the contracting authority. Several remarks were made to the initial design. Additionally the contracting authority drew possible design solutions adding to or changing the proposed design.

One remark to the design was that it could be undesirable to gather all topics into one drop-down menu. If the topics to discuss instead were visible at all times, the participants could better prepare for upcoming topics.

Although the design of the window displaying participants appealed to the contracting authority, some remarks were made about the size of such pop-up windows in general. If the window's content didn't have to be visible for other personnel than the meeting's leader, the window itself should possibly not cover the whole screen. Instead the size could be decreased, for still displaying underlying and important information.

The mouse cursor, and possibilities for adjusting size, appealed to the contracting authority.

Further the contracting authority proposed an own tool menu for selecting options - as adding whiteboards to specific software elements or other types of manipulation.

The contracting authority proposed an own map surface to be made for supporting some kinds of morning meeting topics. This surface was later abandoned in favor of using external software for fulfilling this purpose. The application's possibilities for displaying external software could therefore be the software's only focus, instead of creating such internal modules. But requirements for other software to be used by the system could still be clearly defined.

5.2.2 Iteration II

Before the final version of the second prototype was made, three alternative solutions were designed. These were all further developments of the prototype described in Section 5.2.1, and tried to solve the contracting authority's remark about the visibility of the meeting's schedule.

However, all solutions were fairly different from the initial one. They were all made as overlays managing different forms of external software. A column containing the morning meeting schedule, with clickable options and tools, was the main part of all prototype alternatives. But the alternatives had different solutions regarding if the schedule column should be fixed, collapsible or divisible.

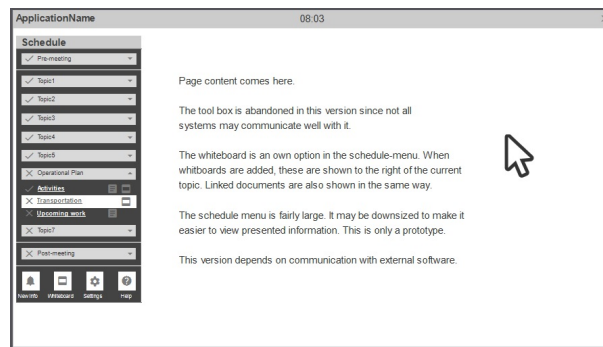


Figure 5.8: The second prototype - alternative 1.

The alternatives were considered in a second guidance meeting. A final proposition to the second prototype was then made on the basis of the three alternatives and meeting propositions, and is presented in Section 5.2.2.

Figure 5.8 displays the first alternative - where the main column was fixed. The idea behind the design was to always make the schedule visible on the same part of the screen, independent of meetings. This could perhaps make the users familiar with the schedule and the interaction routine.

One immediate disadvantage with this design could be that a fixed column decreases the software surface, making the available screen space left for other programs smaller. This could again contribute to poorer readability, since other programs could be designed for being viewed on the whole surface.

The second prototype alternative, presented in Figure 5.9, had menus that could be moved around the screen, and locked into the original, fixed position - if necessary.

The second alternative permitted the use of the whole screen, and all, underlying information could be visible by moving the menus around. However, the external software's use of the surface could be disturbed if it was covered by an overlaying menu.

The third alternative was then made to solve both disadvantages of the earlier presented solutions. This design is visible in Figure 5.10. It had two menus presenting the tools and the schedule, which could individually be collapsed if the users needed more space.

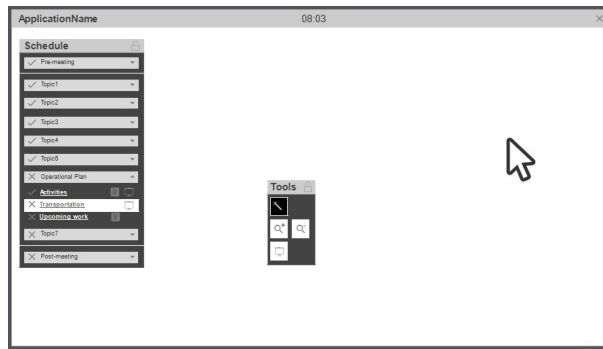


Figure 5.9: The second prototype - alternative 2.

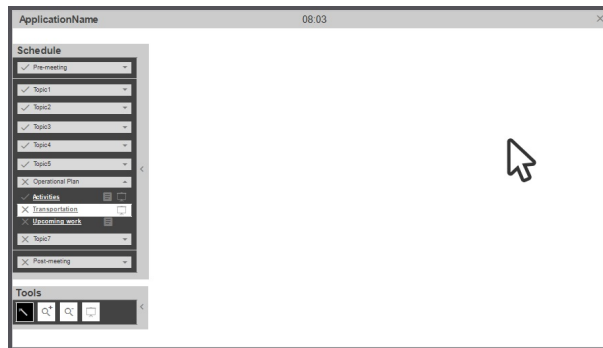


Figure 5.10: The second prototype - alternative 3.

The second prototype version

The contracting authority's immediate reaction to the alternatives was that the menu should not be divided into two parts. Further the idea of collapsing menus could be interesting, especially when presenting information on smaller devices with limited screen space. The contracting authority also mentioned that it could be advisable to present the user with the current topic on screen if the menu was collapsed, to prevent confusion.

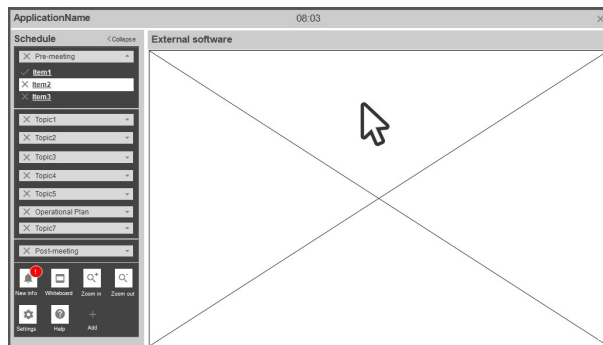


Figure 5.11: The second prototype.

On the basis of the feedback a new version was made. In this design, presented in Figure 5.11, the menu was a single column - containing both tools and the schedule. The surface space available for presenting external software was positioned to the right of this menu. However, if the user needed to present the external software on a larger part of the screen, the menu could be collapsed, leaving more room for the external software. If so, the current topic could be visible on an expandable tag in the top bar. A view of this event is visible in Figure 5.12.

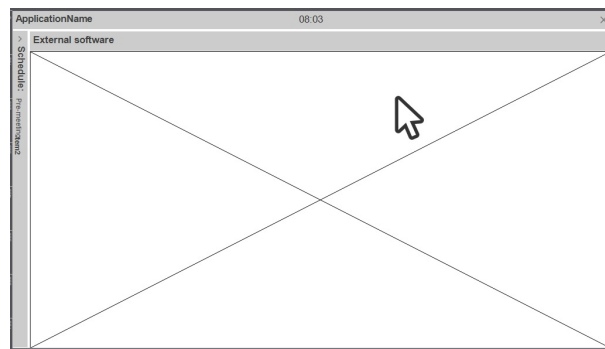


Figure 5.12: The second prototype - collapsed.

However, it has not been considered whether expanding or decreasing of window size could affect the external software's view. A possible disadvantage of the approach is only increasing the views width, while the height remained the same for the collapsed version.

Fulfilling requirements

By creating a surface for viewing external software, the system could become less capable of controlling readability requirements, since most, displayed text was to be presented by other software. However, by implementing options for zooming, the size of the text could be increased to fulfill the users' needs. Figure 5.13 presents these options.

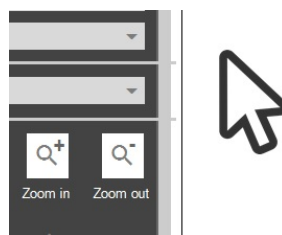


Figure 5.13: Magnifying glasses to compensate for lack of readability control.

As for prototype 1, the second, digital prototype displayed solutions for how notifications and whiteboards could be implemented. These functions could be activated by tapping/clicking buttons in the bottom of the menu, displayed in Figure 5.11.

As for prototype 1, the system could also alert users of new notifications by displaying an exclamation mark above the specific button, as displayed in Figure 5.14.

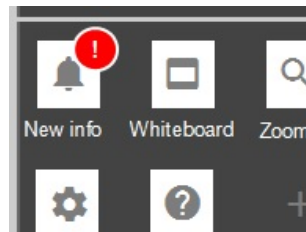


Figure 5.14: Notifications alert symbol.

information about notifications, whiteboards, settings, help and participants were considered internal pop-up windows that could be visible by clicking/tapping options in the bottom of the internal menu. Figure 5.15 presents an example of implementation of pop-up windows - the settings window.

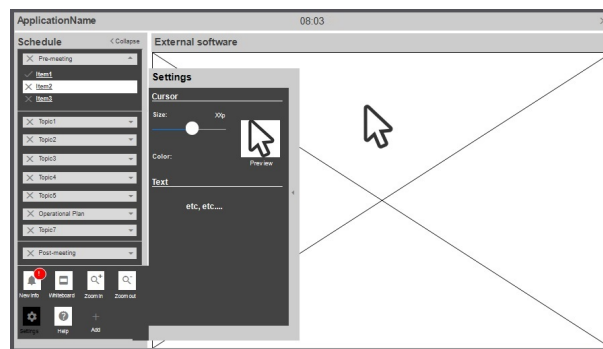


Figure 5.15: A pop-up window displaying system settings.

It was both discussed and considered if the participants option should not be part of the bottom menu, and that the option instead should be visible for all participants before a meeting started. However, since one of the requirements for the system's states is that the software should be intuitively made, it was decided that all internal and global functionality could be placed in the bottom of the menu - perhaps preventing confusion.

The pop-up windows were further thought to be fairly similar to the ones described for prototype 1, and the requirements fulfilled could therefore be quite similar.

However, the use of colors were more thought out than for the initial prototype. Neutral black, white and grayish were chosen for the system interface to avoid taking too much attention. This could perhaps move the users focus to the presented information, or important notifications, as displayed in Figure 5.14. These choices could fulfill requirements set for using colors for manipulating how users interpret the interface.

Further the new prototype had integrated an own schedule for displaying information. Figure 5.16 displays this system part. The schedule was now a

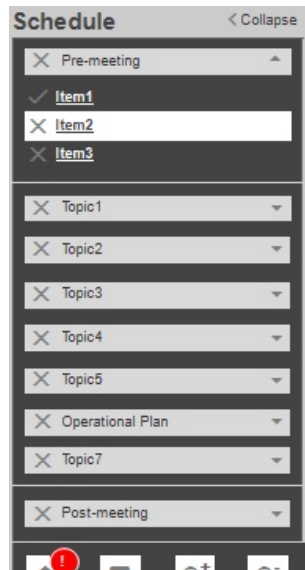


Figure 5.16: A menu organized as a schedule used during a morning meeting.

more important component than in prototype 1, where it was a mostly hidden menu for navigating between internal and external documents, software, and other information sources. Now the menu also served other purposes. Firstly it was a visible overview of the current meeting’s schedule. But it was also a module-based component, since it contained pre-meeting, post-meeting and meeting options.

With respect to the requirement analysis the menu now could satisfy the requirements for implementing a module-based solution, as well as creating a system based on the current organization of meetings offshore.

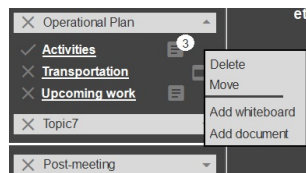


Figure 5.17: Adding documents/whiteboards to specific item in menu.

In the new menu, documents and whiteboards could be added to each step in the schedule, by right-tapping or clicking an item in the menu, or by clicking the whiteboard functionality in the bottom bar. Figure 5.17 presents symbols used for displaying attached documents and whiteboards, as well as how these objects may be added by right-clicking. These functions could fulfill requirements for the whiteboard functionality and requirements for information-sharing in meetings, since the documents could contain important information not accessible in external software. If more than one document was to be presented, when accessing a menu item, the number of documents available was thought to be presented above the symbol, as shown above the topmost document item in

Figure 5.17.

Some of these documents could also be generated internally in the solution and linked to the specific step in the schedule. By implementing an internal text editor it could be easier to save information generated during the meeting, and to add choices made in a report document generated in a later stage.

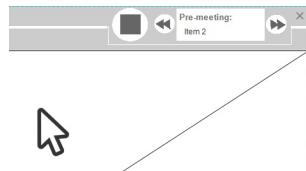


Figure 5.18: Functionality for "playing" through a meeting's topics.

Lastly, it was also considered to implement options for "playing" a meeting. But since the functionality introduced a whole new concept of accessing meeting information, it was decided to discuss this feature with the supervisor before introducing it in a prototype design.

The media-player in Figure 5.18 presents this functionality. It is thought that by playing a meeting, the user could easily navigate between each step in the schedule's menu by clicking the forward- or back-buttons. The menu was also thought to be accessible when the system was in this meeting-state.

Design principles used when creating the digital prototype

As for the first prototype the interactions styles were mostly limited to menu selection.

The Eight Golden Rules of interface design were also tried fulfilled mostly in the same fashion as for the initial prototype design. Adding participants as an option to the buttons in the bottom bar was an attempt to cater for consistency.

Universal usability, on the other hand, may have been reduced by supporting external software tools, in a larger degree.

However, the fourth rule could be strengthened by adding functionality as the proposed media player for controlling a meeting's flow. The fourth rule elaborates that actions should have a beginning, middle and an end. Adding the media player functionality could make it easier for users to separate the editor phase from the meeting phase.

Adding the schedule menu, on the other hand, could reduce short-time memory load, since participants did not have to remember previous or upcoming meeting phases. These were visible in the menu. This could strengthen the fulfillment of the eighth rule - "reduce short-time memory load".

Review

In the supervision of the second iteration the contracting authority emphasized the need for specifying guidelines for external software. This software was to be displayed inside the prototype's view. Since external software, until this point, had been thought of as sort of black boxes, displaying information in various ways, there could be needs for specification. The contracting authority recommended performing an analysis of an actual morning meeting's agenda,

and to suggest how such information may be displayed with the use of the future system.

Additionally the contracting authority advised not to implement the suggested media player for navigation between meeting topics. It was argued that clicks on a menu could be easier to follow by a meeting's participants than the suggested solution.

5.2.3 Iteration III

In the review of the second iteration the contracting authority emphasized the need for creating suggestions for external software. This iteration dealt with producing such information. However, it could be discussed if this work should be considered a part of the prototyping. But since these visualization propositions were made during the prototyping phase, and indirectly effected the system's design, the work was included in the iteration.

During this iteration the different meeting topics were analyzed, based on the topics presented in the mail platform (Section 4.1.1). Such topics could be visible in the system's menu. Function suggestions for each topic have also been made.

The studied document has been used for conducting morning meetings in the oil and gas industry. It has not been proposed a changes to this list of topics, since we did not possess knowledge to do so, and this would also be outside this thesis' scope. Instead new visualization techniques were proposed to present the information described in the email platform.

All information in the studied document is static. It is either written in the document or presented as screen shots from external software. Much of the information is also shown in table form without the use of extensive visualization techniques.

Map Visualization

"HSE", "Notifications and Work Orders" and "Review of Operational Plan" are all topics that visualize data in table form in the studied document. Common to this data is that it may contain information about position, and may therefore be placed on a map.

Steele and Illiinsky (ref. 2.7) claim that map visualization may be a good technique for presenting data when viewers are familiar with the visualized location. In morning meetings detailed information about installation repairs, production, etc. is shared. It is therefore reasonable to believe that the participants have detailed knowledge about the current platform. Map visualization is therefore proposed to visualize data at least in the three, mentioned topics.

According to Steele and Illiinsky several aspects are important when designing map visualizations. Essentials should be included, the environment should be recognizable, disorientation should be avoided, possibilities for zooming should be included, and lastly maps could be changed with respect to the current situation.

However, these guidelines are somewhat general, and may be open for discussion. For example restricting information to essentials could entail visualizing only topic-specific data in the map for each of the three topics. But by doing so,

information about overlapping work in some areas could get lost. It can therefore be argued that tasks linked to different topics may be visible in the map at the same time, possibly creating situational awareness among participants in the meeting. Color, as a visualization technique, may be used to focus on the currently discussed tasks.

By implementing more than one kind of information in a map for each of the topics may also be in accordance to The Eight Golden Rules of interface design, since this design choice may reduce the short-time memory load. With several items included in the map the participants do not have to remember tasks from one map to the next, if they wonder if tasks are overlapping in the same area. A visualization proposition has been developed to display such map design. It is presented in Figure 5.19.

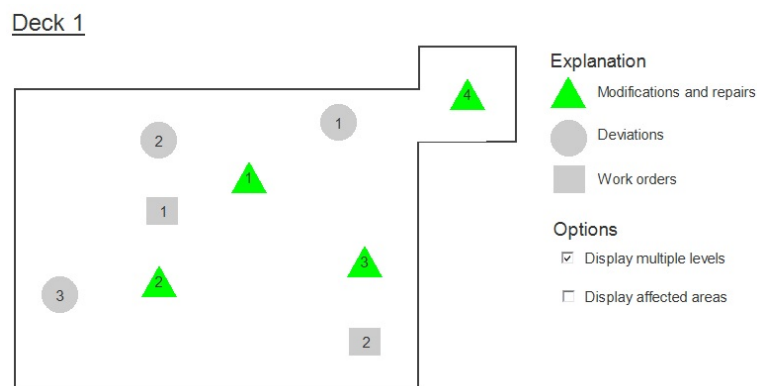


Figure 5.19: A simple map displaying an oil platform, with tasks connected to several topics. Notice that the tasks that are not directly relevant are gray.

Further the software presenting map visualizations should be recognizable for the morning meeting participants, although different decks of a platform is visible. This is in accordance to Steele and Illiinsky's suggestions, but have not been implemented in the displayed figure. One solution to overcome this possible confusing pit-fall is to implement elements that the users may relate to, in the same fashion as neighborhoods for American subway passengers.

Current software solutions implementing the proposed software demands may be both IO-MAP, presented in Section 4.3.2 and WISIO, mentioned in Section 4.3.2. Both solutions implement possibilities for visualizing tasks in maps, they present simplified visualizations of offshore installations, and have possibilities for both zooming and jumping between different platform decks.

Written information

Most topics, described in the mail solutions, have written information connected to them. This information may be written both prior to or during a meeting.

The second prototype, presented in Section 5.2.2, implements both possibilities for connecting documents and whiteboards to items in topics. The whiteboard is an internal functionality, while the documents are external. However, if external documents should be presented during a meeting, there could be set some requirements to the editor used for displaying them.

Common to the written information in the email is that some of the text has been modified to highlight information. An editor should therefore have functionality to change text emphasis. It may also be argued that the internal whiteboards also should implement functionality for altering text - to emphasize information.

Other Visualizations

In the mail platform there are some other kinds of visualizations that should be mentioned.

The "Drilling" topic is for example a screen dump of an external software with a colorful animation of a drilling sequence. This software visualization could be continued in the new application, perhaps as a live drilling demonstration, if needed.

Further the topic "Review of Operational Plan" has a static weather forecast. External software could implement a real-time weather forecast.

If this weather information should be accessible in more visualizations than a single weather forecast, may also be discussed. The mail platform is for example containing information about lifting tasks to be done on the platform. The weather may affect such operations. It is therefore arguable that weather forecasts should be presented together with described tasks. This is also in accordance to The Eight Rules of interface design, since the users do not have to remember information about weather when discussing tasks to be done on the offshore installation.

External software presenting tables, with no position data, may be continued in the new application. However, some suggestions for visualization techniques for visualizing comparable information have been proposed.

In for example the production-topic, a table presents an overview of current production levels. External software which have possibilities for easily change visualization of such data may be favored. Steele and Illiinsky argues that for example color may be used to visualize differences in huge data sets, while size, as a visualization technique, may help morning meeting participants to easily compare small sets of data.



Figure 5.20: Comparison of visualization techniques.

The example in Figure 5.20 builds on a visualization presented by Steele and Illiinsky, and presents how table data may be changed to a size visualization, for increasing the viewers understanding. The data in the table is a fictive version of data presented in the mail platform, which is a small data set. Such concrete visualizations may perhaps make the users capable of easily compare numbers currently presented in tables.

Such visualizations may also be used for illustrating other kinds of information, as for example the amount of oil torched. Large deviations for such data are currently highlighted in the mail platform. By automatically creating visualizations the highlighting process may be abandoned, and deviations may be easier to discover.

Summary of Proposed Function Suggestions for the Meeting Topics

Table 5.1 presents suggestions for function implemented in external software used for each of the morning meeting’s topics, included in the menu in the prototype system. The full titles of the topics are presented in Section 4.1.1.

Table 5.1: Function Suggestions for External Software

Topic	Functions
HSE	Text editor, map solution
Production	Table (size visualization)
Notifications & Work Orders	Map solution
Operational Plan Review	Map solution, text editor, (weather visualization)
Drilling	Own software
Actions from Handover	Text Editor
Focus Points	Text editor
Action Status	Text editor

This table presents function suggestions for external software used in combination with the future system proposed in this thesis.

5.3 Prototyping Cycles - Vertical Prototyping

After conducting the design of the meeting system, by following a horizontal strategy, it was decided to narrow the approach. The development of functionality, presented in this section, took much longer time than design of the user interface with drawing tools and paper prototyping. This part was on the other hand developed as a traditional online prototype - with code. Therefore only a handful functions were gradually implemented in a proof on concept prototype for the active part in meeting sessions.

5.3.1 Iteration IV

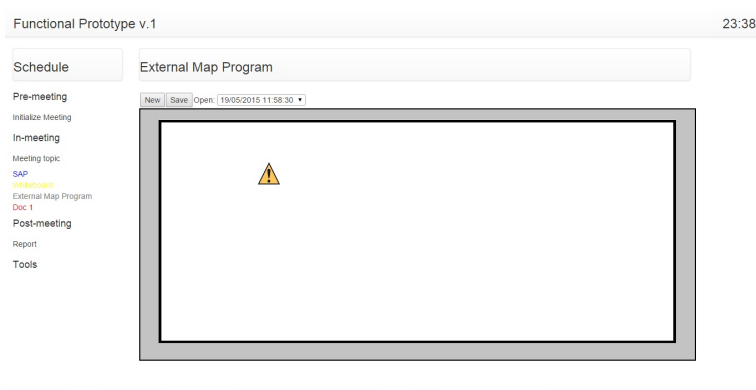


Figure 5.21: The initial prototype version's user interface.

The initial prototype was thought to implement only two, major functions:

1. *An interactable schedule menu*: As in the prototype design, described in Section 5.2.2. The schedule should represent a meeting's content and make the user capable of navigating between topics and related documents and software.
2. *Map visualization*: Since the software could be used for displaying a map view of installations, an interactable map functionality was proposed to be implemented.

Additionally it was decided that the prototype could implement minor functionality, as a clock displaying time globally, a help menu, and some settings, if there was time to spare.

A screen shot of the first online prototype is presented in Figure 5.21.

The menu in the left part of the view let the users cycle through a meeting. External software and information could be presented in the right part of the view. Figure 5.22 presents a view of this initial and simple implementation of the menu.

This version of the prototype differed from the previously presented ones. It was simpler and contained less functionality than for example the design shown in Figure 5.15. The reason for these differences is that it was quicker to design features than to actually implement them in code. A lot of features earlier presented were therefore not included in this initial, functional prototype.

A Menu Inspired by Working Solutions

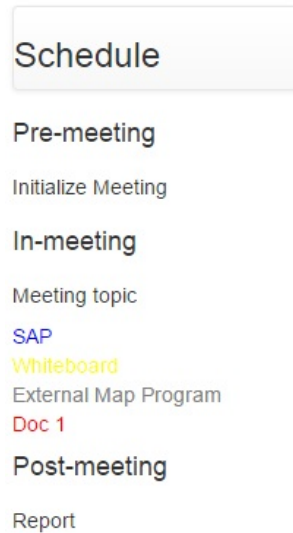


Figure 5.22: The interactable menu - version I.

The setup for the menu itself was retrieved from the server as an XML document. It was thought that this document could have been set in an editor on a previous stage. The below excerpt of a DTD¹ scheme shows the layout of the XML, used for structuring the menu:

```
<!ELEMENT menu (schedule, tools)>
<!ELEMENT schedule (meeting_stage*)>
<!ELEMENT meeting_stage (topic*)>
<!ELEMENT topic (item*)>
<!ELEMENT item (heading?)>
<!ELEMENT heading (#PCDATA)>

<!ATTLIST item type (participants | external |
whiteboard | txt | map) #REQUIRED>
<!ATTLIST item url CDATA #IMPLIED>
<!ATTLIST topic n CDATA #IMPLIED>
<!ATTLIST meeting_stage n CDATA #IMPLIED>
```

The menu was built as a tree structure. At an overall level the structure of the XML resembled the meeting structure mentioned in Section 2.1.1, implementing the "before", "during" and "after" meeting parts. In the XML such elements were added below a singleton schedule element, and each had to be of the type "meeting_element", as described in the DTD above. But to make the schedule more flexible, and to support other kinds of meeting structures, an arbitrary number of meeting stages could be added below the schedule. In the currently displayed menu these elements' values were "Pre-meeting", "In-meeting" and "Post-meeting".

¹http://www.w3schools.com/xml/xml_dtd_intro.asp

On a lower level, the menu was inspired by a previously used morning meeting agenda, described in Section 4.1. This agenda contained eight topics with underlying sub topics presenting information. The menu imitated this structure by letting the users add a custom number of meeting topics for each of the meeting stages. But in the initial prototype, only one topic had been added to the displayed menu for the "In-meeting" stage. This was called "meeting topic", but could have been named both "HSE", "Production", "Drilling", etc. - as in the meeting agenda.

The item elements in the DTD resembled each of the actual agenda's subtopics. For example the "review of the operational plan" topic in the agenda displays information gathered from SAP, some descriptions, a weather forecast and transportation. All such sub topics were called items in the DTD, and are shown in the menu as colored text. In the currently displayed example "SAP", "Whiteboard", "External Map Program" and "Doc 1" are such items. Clicking on one of them could display the underlying information in the window to the right of the menu.

Map Visualization Inspired by Prototype Software

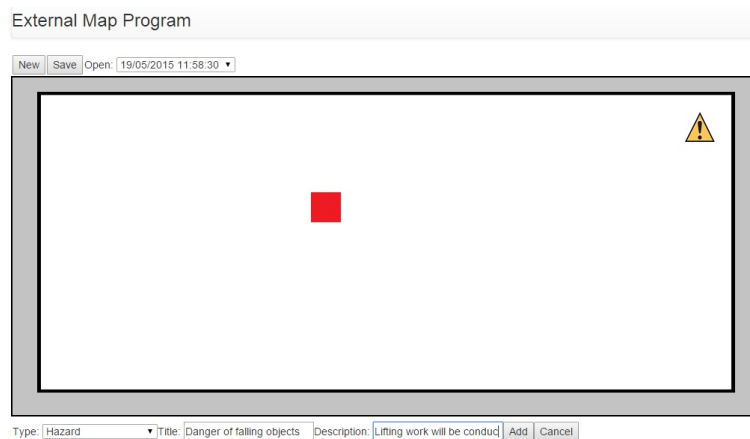


Figure 5.23: The first version of the functional map visualization.

The window to the right of the menu, displayed in Figure 5.21, was meant to present information to the users during meetings. Currently a map visualization was displayed, as shown in Figure 5.23. Other kinds of software were in this prototype only implemented as screen shots of external programs.

The map functionality was highly inspired by existing prototypes with a similar purpose - to display hazards and tasks on a map surface. Especially the prototypes IO-MAP and Wisio (described in Section 2.2.5 and 2.2.6, respectively) were sources of inspiration for the developed prototype's initial and simple map surface.

As for the menu, the map surface also made use of XML to display information, and to store information about maps created by the users. The following tags are excerpts of the DTD-file describing these XMLs' structures (the complete content of the file has been added to Appendix F.1):

```

<!ELEMENT map (item*)>
<!ELEMENT item (id, title, description, start,
finish, modified, xcord, ycord)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT title (#PCDATA)>
<!ELEMENT description (#PCDATA)>
<!ELEMENT start (#PCDATA)>
<!ELEMENT finish (#PCDATA)>
<!ELEMENT modified (#PCDATA)>
<!ELEMENT xcord (#PCDATA)>
<!ELEMENT ycord (#PCDATA)>

<!ATTLIST map src CDATA #REQUIRED>
<!ATTLIST map created CDATA #REQUIRED>
...

```

Similar to IO-MAP and Wisio the interface displayed tasks and hazards on a map surface. Although the mentioned prototype systems implement more functionality, like weather forecasts and upcoming transportation possibilities, the presented interface only displayed a simplified map surface. The reason why this approach was selected was a recognized need for displaying external map data. Table 5.1 in Section 5.2.3 summarizes that such visualization could be used for presenting information about HSE, notifications, work orders and review of the operational plan in morning meetings. Each of these topics were earlier defined with respect to the evaluated meeting agenda described in Section 4.1.1.

The structure of the XML, linked to the map surface, was however simple. Each XML had a root element - a map, with a file path to the background surface image, stored as an attribute. The map node could contain several items. These items were the individual objects to be displayed on the map surface. In this prototype version the id, title, description and the objects' x- and y-coordinates were used.

Fulfilling Requirements

Since this prototype was the first to be developed in the project following a more vertical strategy, all requirements were not met. The requirements were set to a complete system, while this prototype only implemented small parts of the future application. However, some requirements were taken into account during the development.

With regard to the list of important system requirements, presented in Section 4.2.3, three needs were partly or completely fulfilled. These are as follows:

Requirement group 3: The requirements state that it should be possible to register information during a meeting. This was partly fulfilled by letting the users access and manipulate information in the displayed map surface. The initial prototype had implemented functionality to create map items, to store the complete map, including these items, and to retrieve the information with an Ajax proxy to a server side PHP script - reading generated XMLs.

Requirement group 20: According to the requirement the future system should be easy to use and understand. By creating a menu based on an agenda for morning meetings, it could be possible that users working in the oil and gas industry, could already be known to the menu's structure. If some companies execute meetings differently, or implement other agendas, the underlying XML may be changed - possibly in a future editor.

Requirement group 33 : The last of the implemented list of important requirements state that the system must be capable of displaying required meeting specific information. By creating the menu based on an agenda, the topics mirrored actual meeting specific information. In the initial prototype this information was shown on a map surface, although the data was fictive and not gathered from actual sources. In addition to the important requirements, the prototype was also made to fulfill some of the other groups mentioned in Table 4.1.

Requirement group 4: The prototype took partly existing meeting roles into account. The displayed information resembled an actual meeting agenda, and could be changed with respect to the users' needs.

Requirement group 5: Although the prototype was made for meeting sessions, the underlying XML structure could be changed in an editor, and information could be retrieved in a later report state.

Requirement group 6: The use of XML could also be handy when editing a future meeting's setup, although the setup could only be changed directly in the XML document, at this point.

Requirement group 7/8: The future application should display both fixed and real-time data. These requirements could be fulfilled by displaying external software, if the process of doing so permitted it.

Requirement group 9: The system should have an own mode active during meetings. This was currently implemented as the meeting's schedule.

Requirement group 15: The page hierarchy of the prototype had only two levels, although the menu had several levels of topics.

Requirement group 17: The requirements connected to displaying other system's data was imitated by displaying the map surface in the window where external systems could be presented.

Requirement group 18: Clear navigational information was visible in the menu when interactions were made. The bar above the external window also informed the user about the current navigation.

Requirement group 20: Unnecessary functionality was considered not implemented. The main parts of the initial prototype were simple - only the map surface and the schedule were implemented.

Design Principles Followed

Few design principles were fulfilled in the initial prototype. The user interface was simple and contained test information, as the use of colors for element types in the schedule menu.

Review

The initial prototype was reviewed in collaboration with the contracting authority after completing the first cycle. The contracting authority thought that the map surface had room for several refinements. Adding items to the map currently showed some input fields, buttons and a combo box below the surface. Alert pop ups also informed the user about changes made and illegal actions. Such information could instead be presented inside the 'external' window, providing the user the experience of accessing an external application.

The contracting authority also emphasized that the background image in the map surface should be changed to present the user with an overview of an offshore installation.

In addition the menu was considered too simple, and bewildering. The initial schedule menu had no indents, making it hard for the user to separate topics from items. The contracting authority also emphasized that the menu should link to more information, making the user capable of viewing images of external software, while interacting with the menu.

A last implementation possibility, according to the contracting authority, was to add opportunities for displaying which topics that had been viewed. Thus the users should be capable of separating between already displayed information, and information that was yet to be viewed.

5.3.2 Iteration V

The fifth prototype cycle was much less comprehensive than the fourth. It was decided early in this cycle that an upcoming, and extensive version should be made without the use of external design tools - as bootstrap. Instead the design was to be made from scratch, preventing additional functionality as dynamical shrinking of panels, etc, included in the design tool. Thus the student was thought to be provided with more control of the design process. The fifth prototype was therefore not considered a new version, but as a product containing minor changes to the first version.

Changes made

The biggest changes made in the second cycle was to implement a view of the map surface, and to include some extra images of other, thought to be external, systems. These changes are presented in Figure 5.24 and Figure 5.25.

The image of the installation, used when presenting information in the map, was retrieved from a screen shot of the working IO-MAP prototype.

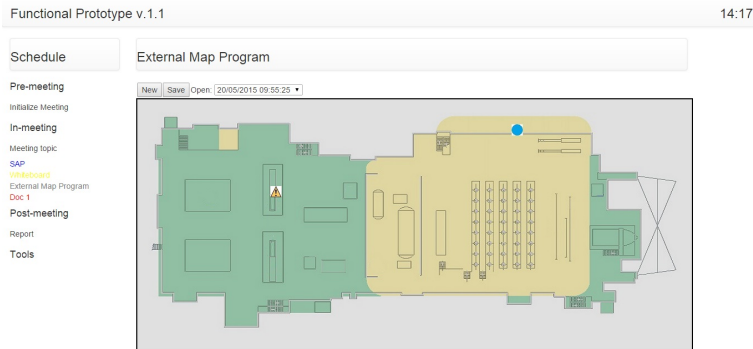


Figure 5.24: Adding an installation background to the map surface.

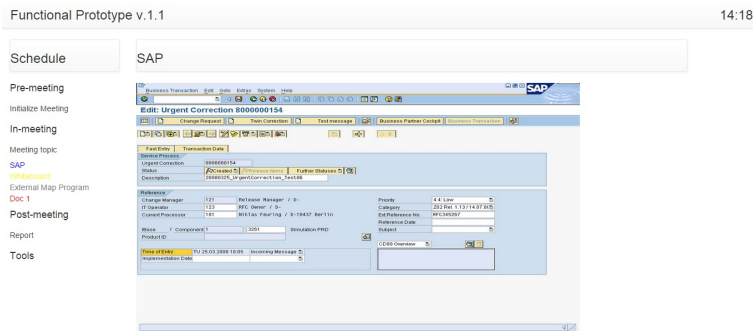


Figure 5.25: Adding images of other software systems.

The second figure, displaying the external software SAP, was also a screen shot of the working system.

Requirements, design principles and review of the changes

The minor changes made had little effect, if any, on the prototype's fulfillment of requirements and design principles beyond what is mentioned for the initial, vertical product in Section 5.3.1. The contracting authority, reviewing the changes, was therefore notified about what was done. Most of the sought needs were to be fulfilled in the sixth cycle, mentioned in Section 5.3.3.

5.3.3 Iteration VI

In the sixth iteration the vertical prototype was redesigned. The menu was made collapsible, to add possibilities for displaying a larger view of inter alia external software. Possibilities for performing simple draw and write operations on whiteboards were added. The external map surface was also improved, making it possible to add new map elements by clicking the surface and writing information in input fields in a div - designed as part of the map software. Additionally the map got functions for displaying information about each of the elements by accessing the tools menu, and selecting a help cursor icon.

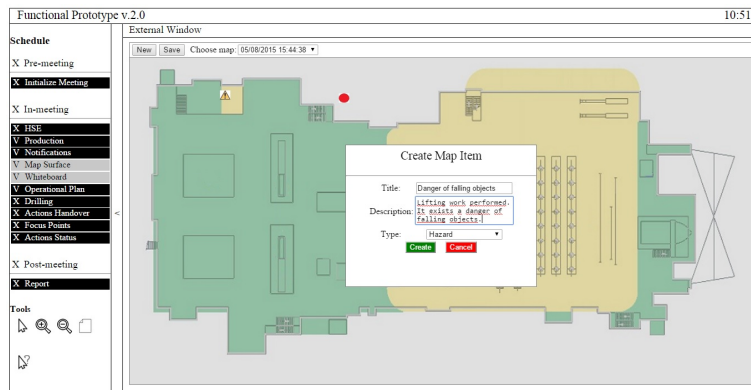


Figure 5.26: Interaction with a map surface in the last, vertical prototype.

Further the xml for the tools menu was implemented into the prototype. The tools were made mutable or final, separating them into categories of tools displayed for specific sites, or available globally. A small part of the DTD file (available in Appendix F.1), with this functionality described on a lower level, is presented below.

```

...
<!ELEMENT tools (tool*)>
<!ELEMENT tool (heading?, img?, func, id?)>
...
<!ATTLIST tool type (final | mutable) #REQUIRED>
...

```

In the script the attribute "type" is introduced in the DTD. This is a required attribute, making the tools either mutable or final.

Lastly, the information shown in the prototype was closely linked to the discoveries made in Section 5.2.3 - the third design iteration. Both the map view, the textual parts, tables and size and weather visualizations were added.

Fulfilling Requirements

By implementing the described changes it was thought that the prototype could fulfill more of the defined requirements. The newly added whiteboard functionality could satisfy requirement 34 in Table 4.1, defining the need for such a system part.

Further, adding different types of tools, both global and more specific, could increase the fulfilling of both requirement 20 and 21. Since only the needed tools could be presented, the automation of the system would perhaps be more understandable, since tools not applicable for an external system would be hidden. This could also strip the system for showing an unnecessary amount of different types of tools at all times.

But a major discovery made, during this vertical prototyping stage, was that the current design had problems fulfilling important functions. It could be hard to implement possibilities for alteration of the menu, if the system had to be operated from the menu itself. It was further noted that it could be hard to

share the required information in morning meetings. The current prototype design, both on this narrowed, vertical level and a higher, horizontal level, did not contain possibilities for simply assign decisions made and comments during a meeting.

5.3.4 Review

The current prototype was reviewed by the contracting authority. It was decided to end the vertical development - implementing functionality in code. Instead it was decided that the horizontal prototyping could continue, to implement missing, important functionality. With this, a seventh iteration cycle could commence.

5.4 Prototyping Cycles - Adjustments to the Horizontal Prototype

5.4.1 Iteration VII

The work with the prototype following the vertical strategy spanned over several weeks. During this immersion into more specific functionality, new ideas came up and was added to the horizontal part of the prototype, although they were not included in the vertical, timeconsuming prototype. Since both prototypes were to be tested, the horizontal prototype was changed after the vertical prototype was finished.

All these ideas were reviewed with the contracting authority before finishing the cycle, prior to the initiation of the first testing stage, which also forced a last design iteration.

The Comments Window

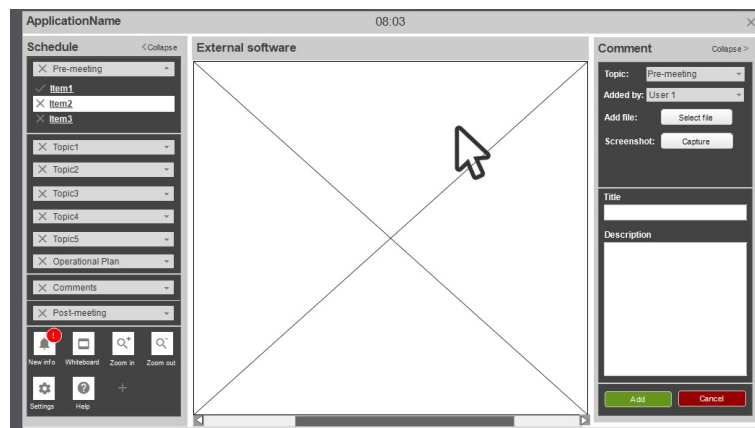


Figure 5.27: The overall prototype was changed during vertical work with specific system parts. A collapsible comments window was firstly added.

One of the perhaps most important functions added to the horizontal design was a collapsible comment-field, reachable during the meeting. This comment field had been added to the horizontal prototype, and is presented in the right-most part of Figure 5.28.

In the earlier developed prototype text documents were to be added with external software and stored. This process could however be timeconsuming, and was meant for altering earlier written documents. However, if new information came up during a meeting, and this information could not link directly to a written text document, problems could occur.

In the email solution, described in Section 4.1, the meeting participants were able to write information and comments directly into a displayed document during meetings. Such changes could however not be possible if the participants instead were working with an external application or a map surface during meetings.

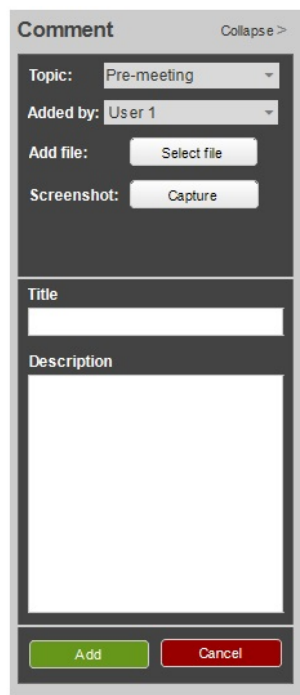


Figure 5.28: A new comment option was added to the prototype developed vertically.

Therefore the horizontal design was supplemented with a collapsible comments field as shown in Figure 5.28. This comments field was meant to be globally accessible during meetings, and could be expanded or collapsed based on the users' needs. If the users wanted to write comments, based on the displayed information, they now could add this to the comments window while still viewing a part of the underlying information.

Requirements Thought Fulfilled by Implementing Comments Adding functionality for writing comments, and possibly attaching documents and screenshots to them, could strengthen fulfillment of requirements.

Some requirements defined as highly important, presented in the bullet list in Section 4.2.3, were assumed to be strengthened by adding such a window. The background for the alterations, based on the requirements, are presented in the bullet list below.

- *Id 3*: The possibilities for registering information during meetings was thought to be increased. By implementing the comments window the users could have the ability to easily access a menu and enter information.
- *Id 10*: After a meeting it was thought that the leaders could have to produce reports, and share them. By implementing an internal comments tool, it could be easier to fetch produced data when a meeting had finished, without establishing communication with external tools.
- *Id 20*: It could be easier to use the system, since the users would have easy access to a window made for writing meeting specific information.

Meeting Handler

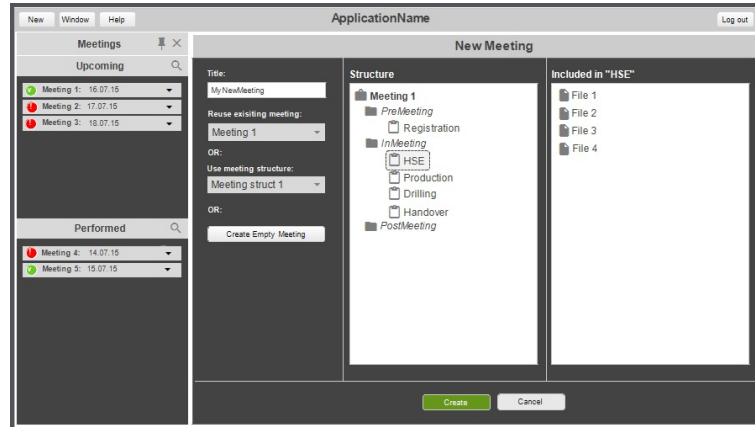


Figure 5.29: This is a slide illustrating how the outer part of the system could be made, for handling several meetings.

While creating the vertical prototype problems were discovered with the current design. It seemed difficult to alter the meeting's setup from within an already existing meeting menu. The overall handling of different meetings, reuse of meeting elements and importing new ones also seemed difficult to commit with the current design. Lastly, creating reports seemed hard in the vertical prototype, if they had to be sent after the meeting had finished.

This paved the way for the implementation of a design handling all meetings, with possibilities for editing, creating, starting and reporting such sessions.

A part of the design, displaying how meetings could be created, is presented in Figure 5.29. The other slides are available in the later concept evaluation - added to Appendix D.

This part of the design was thought to implement a similar XML structure, as the one developed during the work with the functional prototypes in earlier iterations. The vertical prototype had therefore already shown that implementation of a menu, built on XML, could work in practice.

The new design took however the implementation of the XML a step further, and based the layout on more functionality than the ones designed into the functional prototypes. It contained, among others, functionality for creating both meetings and meeting structures. Structures were thought to be the meeting skeletons, without the information needed to be displayed in meetings. Meetings, on the other hand, could contain files, links and references to external software to be displayed.

Additionally the design had implemented possibilities for inviting personnel to morning meetings. The lists of personnel were to be built on an XML-like structure, with team elements containing children elements describing each of the actual persons.

Created meetings were listed in a menu containing two, sub-menus displaying both upcoming and conducted meetings. The upcoming meetings, in the first list, could be accessed for altering meeting setup, as adding files and external software to the structures, inviting personnel and commence sessions.

The meetings in the list with performed sessions could be accessed for sending reports to participants and other stakeholders.

A fundamental idea behind the design of the meeting handler was to reduce the time the users had to spend creating and altering meetings before the meetings could be commenced. The design had therefore implemented a lot of possibilities for reusing earlier information, as preset lists of participants, meeting structures and meeting content.

Requirements Fulfilled Implementing a part of the system handling all the meetings could strengthen the fulfillment of several requirements. One of them, considered highly important, is that the system should be capable of sharing the required meeting information. Meetings could however be viewed as a part of a bigger whole, the distributed collaboration structure, described by Skjerve et al. in Section 1.5.2.

According to Skjerve et al. the process of conducting a meeting, or a main collaboration session, is closely linked to stages involving both the preparations, and the outcome, execution and adjustments made in the aftermath.

The new system design could perhaps make the future system more connected to the model by implementing functions for performing each of the three stages. Preparations could be conducted by inviting personnel, attaching meeting content and editing the meeting's setup. The main collaboration session, on the other hand, could be performed with the earlier designed main part of the future system. Lastly the activities needed after the meeting could be supported by making it possible to send a meeting report and adjusting the structure of future meetings.

Review

The design was not reviewed with a minor expert review, as for the earlier iterations. At the current time the design was considered finished, and was therefore reviewed with a more comprehensive test - the cognitive walk-through described in Section 6.2. However, the test's results revealed numerous weaknesses, forcing another design iteration, described below.

5.4.2 Iteration VIII

In this iteration several changes were conducted in accordance to feedback given during the cognitive walk-through. Both the system part designed for handling meetings, and the part active in sessions, were redesigned. The new slides have been added to Appendix C.

In the part of the system handling performed and upcoming meetings, a new window was added for starting sessions, displaying meeting specific information. The functions used for adding files, links and references to external software were also changed. The list containing the meeting skeleton now includes the content, such as the specific files. Lastly roles were added to the invitations setup, making it possible to contact the active roles before the meeting, instead of only specific personnel, who may rotate with other personnel.

The biggest changes were however not done with meeting handler part of the design, but with the in-meeting part, visible in Figure 5.30. The tool box was replaced with a row of new buttons, making it possible to add both actions,

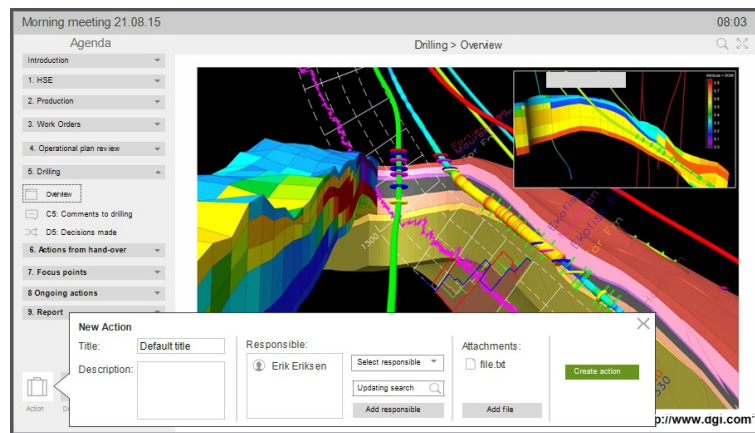


Figure 5.30: This illustration presents the design generated for the meeting application during the eighth prototyping iteration.

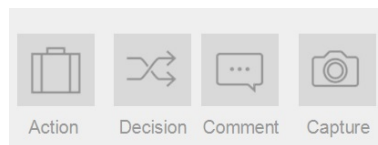


Figure 5.31: An excerpt of the design displaying the new buttons replacing the previous tools menu. In Figure 5.30 this menu is mostly hidden below the "New action"-window.

decisions and comments during a meeting, plus capturing screen shots of the displayed content. This menu is presented in Figure 5.31.

Clicking on either the comments, decisions or actions buttons could further lead to displaying a custom form window, accepting user input. An example is presented in Figure 5.30. If created, the information was thought to be available as either a comment, decision or action item inside the schedule menu.

The schedule menu was also altered further. In previous versions cross- and tick-symbols were used to inform if topics had been viewed, or not. In this version a new approach, inspired by email readers, was implemented. The menu items were now displayed as bold text if they had not been viewed yet. Otherwise they were displayed with regular text formatting.

Another functionality added was a more advanced report window, accessible in the end of a meeting. The report displayed all information gathered during the meeting, as comments, decisions and actions added, and for example screen dumps captured. This information was however not thought to be fixed, and could be edited if needed, before the report could be sent.

Several tools as whiteboards, notifications, help and settings were omitted from the prototype. The testers argued that interacting with such functionality could be too time-consuming. The prototype was therefore simplified, for saving time.

The visual presentation of the design was lastly transformed. In the visualization of the outer part of the system, the use of the color red was switched with

yellow, for not triggering perceptions of alerts. More comprehensive changes were undertaken with the inner meeting-specific part of the design. Here the layout, previously using a lot of borders, was replaced with a cleaner, coherent design, as displayed in Figure 5.30. Space-demanding borders, surrounding menu items and the window displaying external information, were all removed. Instead the grayish background color itself, separating the external information from the rest of the design, was used as a window border.

Review and Guidelines Followed

The eighth iteration of the design was tested with a heuristic evaluation, and a usability inspection - aiming to uncover if the future system could fit into today's oil and gas organizations. The heuristic evaluation included a questionnaire to test in what degree the requirement groups were implemented into the current design. A second questionnaire aimed for examining if interface design guidelines could be considered satisfied. These tests are presented in Section 6.1.

Since the tests reviewed the interface design, the separate evaluation of the requirements has been excluded from this sub section.

5.4.3 Iteration IX - Final Changes

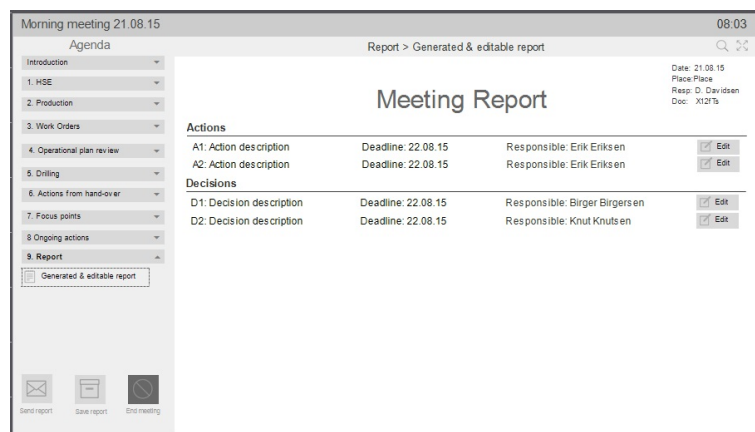


Figure 5.32: A slide presenting the report with action and decisions shown without the need of scrolling.

After conducting the heuristic evaluation and the usability inspection, the testers still felt that changes could improve the system. The feedback revolved around changes to the part active during meetings, and minor changes to the outer subsystem, managing meetings.

According to the testers the yellow warning triangles, symbolizing missing input, should be changed, since no real hazards could be connected to them. Execution dates were requested for actions and decisions in the in-meeting subsystem. Here the report was requested changed, to display all relevant information on a single site, not demanding scrolling.

Although these changes have not been reviewed, the testers suggestions have been designed as a couple of prototype slides. Figure 5.32 displays a proposal

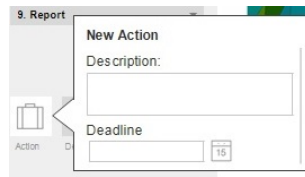


Figure 5.33: An excerpt of a design slide presenting possibilities for adding a deadline to actions and decisions.

for the design of the report slide. Figure 5.33 presents how a deadline could be added to actions and decisions. And lastly Figure 5.34 presents an excerpt of a slide representing the outer part of the application, with circles added for telling the users that reports had not yet been sent.

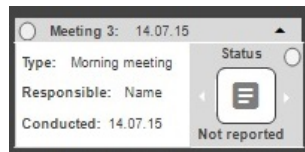


Figure 5.34: An excerpt of a slide presenting information about reports, without the yellow warning sign used in the previous iteration.

5.5 Requirements and Features Thought Implemented when Developing the Horizontal and Vertical Prototypes

The requirements set to the system, and inspiration from current systems and prototype solutions, form the basis of the future system's functions. During the work with the prototypes several functions were implemented, and some were omitted along the way. Table 5.2 in this section sums up all requirements and other features that was tried implemented in each prototype version. The 34 requirement groups, as described in Table 4.1, are presented. Further the table presents which of the features, discovered in other systems, that were tried implemented into the prototypes. These features are listed in Table 4.3. In Table 5.2 all IDs are prefixed to separate requirements from discovered features. Requirements are prefixed with the letter "R", while other features are prefixed with "F".

Requirements or features which are designed, but not tested, are marked as implemented, but are surrounded by parenthesis. The reason for this choice is that the design contains a lot of complicated functionality. It is not possible to test all by prototyping vertically in a project of this size.

ID	v1	v2	v3	v4	v5	v6	v7	v8	v9
R1	No	No	No	No	No	No	No	No	No
R2	No	No	No	No	No	No	No	No	No
R3	No	(Yes)	(Yes)	No	No	No	(Yes)	(Yes)	(Yes)
R4	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R5	No	No	No	No	No	No	Yes	Yes	Yes
R6	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
R7	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)
R8	(Yes)	(Yes)	(Yes)	Yes	Yes	Yes	(Yes)	(Yes)	(Yes)
R9	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R10	No	No	No	No	No	No	Yes	Yes	Yes
R11	No	No	No	No	No	No	No	No	No
R12	No	No	No	No	No	No	No	No	No
R13	No	No	No	No	No	No	No	No	No
R14	No	No	No	No	No	No	No	No	No
R15	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R16	(Yes)	No	No	No	No	(Yes)	(Yes)	(Yes)	(Yes)
R17	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)
R18	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R19	(Yes)	No	No	No	No	(Yes)	(Yes)	No	No
R20	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
R21	No	No	No	No	No	No	Yes	Yes	Yes
R22	No	No	No	No	No	Yes	Yes	Yes	Yes
R23	Yes	Yes	Yes	No	No	No	No	Yes	Yes
R24	(Yes)	No	No	No	No	No	No	No	No
R25	(Yes)	No	No	No	No	No	No	No	No
R26	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)
R27	No	No	No	No	No	No	Yes	Yes	Yes
R28	No	No	No	No	No	No	No	No	No
R29	Yes	No	No	No	No	No	Yes	Yes	Yes
R30	Yes	Yes	Yes	No	No	No	Yes	No	No
R31	No	No	No	No	No	No	No	No	No
R32	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
R33	No	No	No	No	No	No	Yes	Yes	Yes
R34	(Yes)	(Yes)	(Yes)	(Yes)	(Yes)	Yes	(Yes)	No	No
F1	No	No	No	No	No	No	Yes	Yes	Yes
F2	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
F3	(Yes)	(Yes)	(Yes)	No	No	No	(Yes)	No	No
F4	No	No	No	No	No	No	No	No	No
F5	Yes	No	No	No	No	Yes	Yes	No	No
F6	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes
F7	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
F8	No	No	No	No	No	No	No	No	No
F9	No	No	No	No	No	No	No	No	No
F10	No	(Yes)	(Yes)	Yes	Yes	Yes	(Yes)	(Yes)	(Yes)
F11	Yes	Yes	Yes	No	No	No	Yes	No	No

Table 5.2: A table presenting functionality thought implemented in the prototypes of the future system.

Chapter 6

Concept Evaluation Results

6.1 Testing of the Developed Concept

During the development of the prototype there were conducted several minor tests in collaboration with the contracting authority. These were minor experts reviews, and are not described in this chapter. Instead they are shortly described for each iteration made for developing the concept.

The tests described in this section were on the other hand more extensive, and of a more formal nature. They were activities where two test individuals at IFE, with depth knowledge about software development in the oil and gas industry, were involved. These individuals were asked to perform a cognitive walkthrough of the prototype design, fill out a questionnaire linked to the project's requirements and design guidelines, and finally perform a usability inspection.

Initially the tests were meant to be performed in one, single session. However, the results of the first test activity, the cognitive walk-through, forced another prototype iteration. The tests were therefore performed in two separate parts. And so, further improvements to the prototyping design were developed before the design lastly was tested with a heuristic evaluation and the usability inspection.

The tests were otherwise performed as described in the methodology in Section 3.5. All tested slides are available in this document. The eight design iteration, conducted after the cognitive walk-through and the smaller, ninth iteration are however described in Section 5.4.2 and 5.4.3.

The cognitive walk-through was performed as mentioned in the test document, presented in Appendix D. The subjects were handed a document describing several walk-throughs and were asked to use the design to navigate within the user interfaces. They were told that the application was designed in two layers - an outer part of the system, letting personnel create, start, design and report meetings, and an inner part containing the design available in the meeting sessions.

The heuristic evaluation was meant to follow the cognitive walk-through in the same testing session. But since several implications with the design were discovered in the first test session, both the heuristic evaluation and the usability inspection were postponed.

However, when the heuristic evaluation finally was performed, the same experts were asked to fill out questionnaires to evaluate the system based on the initial requirements and interface design guidelines.

Lastly the usability inspection was performed to investigate how well the currently developed prototype could fit into the morning meeting context in the oil and gas industry.

The results of the tests are presented in in this chapter. The complete test document is available in Appendix D.

6.2 Cognitive Walk-through

During the cognitive walk-through several remarks were made, not only to design elements, but also the logical structure of the design concept. The prototype parts discussed was the outer system, managing meetings and reports, and the inner system active during meetings. Implications discovered during the review of the outer system are listed in "pre-meeting" and "post-meeting tasks". Feedback regarding the actual meeting system design is listed in "in-meeting tasks".

Mainly the implications discovered are mentioned, since these disturbed the cognitive walk-through, leading to functional feedback. Later tests presents more subtle reviews.

6.2.1 Pre-meeting tasks

Some of the feedback given affected minor elements in the design. But no matter if they where small, their impact could have been negative in a future system.

The first regarded the use of color. In the prototype the color red was shown to tell the users that reports and meeting invitations not yet had been sent. According to one of the reviewers red is considered an alarm color, and should not be used for such purposes.

Further the button designed for accessing a meeting's setup contained a sprocket symbol. Both reviewers meant that such a symbol should not be connected with harmless adjustments to a meeting's setup. They stated that sprockets should instead be used when performing deeper and more advanced system changes.

Two, structural implications were discovered - connected to a meeting's setup and to meeting invitations.

Firstly one of the reviewers noted that files, links and references to external software were not visible in the structure of the meeting, when adding content to an upcoming session. If the users could not see the relationship between the structure and the content, they could easily get confused.

Further a reviewer noticed that it could be unnecessary to invite specific personnel to meetings. By instead inviting roles, personnel in rotating shifts could be invited. On the other hand, when inviting specific personnel, one of the reviewers meant that the system should take more into account invitation of guests. It was therefore argued that the invitation structure should be changed.

The reviewers also noticed, when discussing the organization of files, etc., that the content to be displayed should be better explained. The current prototype informed the users about each of the files' sizes, creation dates and, among others, types. Instead this information was requested to be more closely linked to the users needs. By displaying read or write permissions the users could for example know in advance if they were able to alter the content to be displayed. Last edit date and a preview of the content were also desired information, partly available in the current design.

Lastly the testers found the pre-meeting prototype to provide the users with too little feedback when a meeting was to be started. One of the reviewers requested an own window displaying a summary of meeting related information, connected to the upcoming session.

6.2.2 In-meeting tasks

After conducting the walk-through, describing the pre-meeting functionality, the testers were presented a set of slides with functionality available in the actual meetings. The first implication was noticed early on. No welcome message had been included at the beginning of the meeting slides. Such text was requested.

When being displayed the different versions of participant registration, the reviewers found the current design to be cumbersome. Instead they requested a simple checkpoint list on the welcome site, enabling easy and fast registration.

They also noticed that the tools menu, available below the schedule, was providing the users with perhaps unnecessary and time-consuming functionality. They therefore argued for omitting the functions, and move specific tools to the top bar above the window displaying the meeting content.

One reviewer added that available functionality in morning meetings should be timesaving to reduce costs.

The testers argued that the previously mentioned tools menu could be replaced with a button group. These buttons could be used during the session for attaching actions, comments and decision descriptions to the content displayed in a meeting. One button could also be used for adding screen dumps as clarifications of decisions made. When creating such information one of the reviewers claimed that it should be attached to the schedule menu, and be available in a later report, displayed at the end of the meeting. Especially actions and decisions were considered as highly valuable meeting information, requiring more focus.

Both the reviewers agreed that the schedule should be presented more as an agenda, and be more striking than the current solution. One reviewer claimed that other functionality, as the use of tools, could steal too much attention from the schedule.

The schedule's display of files, links and external software were also considered too space consuming. Such information was requested to instead be displayed in a single line in the schedule, making space for more important information - such as attached actions, decisions and comments.

Further the schedule's use of symbols, for informing users of topics viewed, was considered misleading. Especially the "X"-symbol could be mixed with a symbol for closing something.

The whiteboard functionality, meant for replacing physical meeting whiteboards, was considered unimportant. Although many physical whiteboards are available in today's collaboration rooms, they are, according to the testers, rarely used.

Further the help menu and settings were considered unnecessary. According to one of the reviewers, the meeting participants didn't have time to access such features during a meeting.

It was thought that the report should only be available for viewing at the end of a morning meeting. But the testers stated that reports usually are sent at the end of the session. More functionality was therefore requested.

The testers also claimed that a meeting report should have possibilities for listing all the actions, decisions and comments made, plus screen dumps taken, during a meeting. Such information should have editing possibilities. The report could therefore be changed before it was sent. Then, in the next morning meeting, the actions and decisions made in the previous meeting should finally

be listed, according to the testers.

One remark was also given to the visual design of the future system. A reviewer thought that the interface design should be simpler, without unnecessary elements as borders or colored frames, stealing attention.

6.2.3 Post-meeting tasks

After conducting the review of the meeting design, the outer part of the system, handling the reports was evaluated. Although most reports should be send in the morning meetings, the testers could see the advantages of being able to send reports after the sessions. Such incidents could occur if the meeting leaders were to check something before information could be written in an unfinished report. But the current design only implemented thought functionality for viewing the reports. The testers requested functionality for also changing report content.

6.3 Heuristic Evaluation

The heuristic evaluation was divided into two parts. The first aimed for evaluating the prototype against the requirements made in the beginning of the project. In the other part the reviewers were asked to evaluate the prototype based on the Eight Golden Rules of Interface Design. Although the main focus for this prototype has been to adapt it to modern oil and gas industry, the student has also tried to design with established guidelines in mind.

To complete their task the reviewers were to rank the fulfillment of requirements and guidelines with regard to the developed concept. This scale was limited to five, possible selections. Additionally the reviewers were encouraged to append comments if they were unsure if requirements had been implemented, or the meaning of them.

The two reviewers participating in the test were the same individuals who evaluated the system by performing the cognitive walk-through.

6.3.1 Evaluation of the Concept Based on the Requirements

The testers ranked the system based on the 34 requirement groups, presented in Section 4.2. Table 6.1 shows their answers per requirement group.

ID	Requirement group	Reviewer 1	Reviewer 2
1	Two-way communication with users operating external mobile devices during collaboration sessions.	1, Design allows it, not demonstrated.	1, not relevant.
2	Sharing of view with mobile devices during collaboration sessions.	1, Design allows it, not demonstrated.	1, not relevant.
3	Possibilities for registering information as for example keywords during meetings.	5	4
4	Must take the existing meeting roles into account by being designed for both meeting participants, meeting leaders, etc.	4	4
5	Multiple modes for varying system flow based on current state in the meeting process. Examples include an editor, a meeting mode and a report mode.	4	4
6	Possibilities for editing and prepare a future meeting's setup.	5	4
7	Implementing options for presenting real-time data.	1, Design allows it, not demonstrated.	1, hard to rate.
8	Implementing possibilities for presenting fixed data.	1, Design allows it, not demonstrated.	1, hard to rate.

9	Implementing an own mode active during meetings, especially designed for such a stage in the planning process.	3		?
10	Implementing an own mode active after meetings, able of producing minutes or other kinds of reports.	4		?
11	Implementing modes for different types of devices to make the interface user friendly, regardless of device/monitor used.	1, not demonstrated.		1
12	Collaboration participants should see the same platform view regardless of physical location or monitors in use.	1, not demonstrated.		1
13	Sharing of cursor.	1, not demonstrated.		1
14	Agents for decision-support.	1, not demonstrated.		1
15	Page hierarchy with maximum of two levels.	3		3
16	Register and time meeting attendance.	4		3
17	Functionality for open other systems.	5		1, theoretically.
18	Present clear navigation information during interaction with system, as well as when other systems are active inside the software's context.	4		1
19	Having an overview of personnel competence, and possibly also use the competence as decision-support.	2		1
20	Easy to use system stripped of unnecessary functionality.	3, getting there - but a bit to go.		3
21	Understandable and intuitive automation.	4		
22	Clear system feedback, including graphical feedback.	4		3
23	Measures for focus as warm and cold colors for inter alia setting priority.	3, hazard sign.		1
24	Implementing functionality capable of giving readability feedback based on the current adjustments.	1, not demonstrated.		1
25	Readability test.	1, not demonstrated.		1

26	Implementing possibilities for readability adjustments.	3	1
27	Pre-selection of options when possible, to make it easier for users to follow ordinary system flow.	4	2,3
28	Support for display reduction.	3, unclear if it is in place.	1/2
29	Different complexity levels enabling both expert and novice use.	3	3
30	Notifications about important news.	4	
31	Local display settings, as brightness, color and contrast.	3, unclear if it is in place.	1
32	A system capable of taking available hardware in current meeting rooms into account.	3, unclear if it is in place.	1
33	A system capable of sharing the required information in morning meetings.	4	3
34	Whiteboard functionality.	3, not so important for morning meetings.	2/3

Table 6.1: A table presenting the results of the heuristic evaluation. The reviewers individual answers are shown in the columns to the right. The fulfillment of requirements are ranked on a scale from 1-5, where 1 means not fulfilled, and 5 stands for fulfilled. Additional comments are written in the same table cells.

While the table presents the review of the system, not all requirements were implemented, or possible to implement, in the prototype. Some functionality, as whiteboards, help, settings and notifications, were omitted after the review produced in the cognitive walk-through. At the same time the system's implementation of the overview of personnel's competence, was omitted based on the review. Figure 6.1 presents review scores in a diagram displaying requirements thought to be present in the design. The x-axis shows the requirements by id and the y-axis shows the score. The two types of bars in the diagram presents the reviewers ratings.

6.3.2 An Evaluation Based on the Eight Golden Rules of Interface Design

The system design was further rated in accordance to the Eight Golden Rules of Interface Design, as presented in Section 2.8.1. Table 6.2 shows the reviewers scores, given in a similar fashion as for Table 6.1.

ID	Golden Rule	Reviewer 1	Reviewer 2
1	Strive for consistency.	4	3
2	Cater to universal usability.	?, unclear	1

3	Offer informative feedback.	4	3
4	Design dialogs to yield closure.	4	3
5	Prevent errors.	?, unclear	1
6	Permit easy reversal of action.	?, unclear	1
7	Support internal locus of control.	4	4
8	Reduce short-term memory load.	4	2/3/4, external tools are not controlled.

Table 6.2: A table presenting the results of the heuristic evaluation in accordance to fulfillment of the Eight Golden Rules of Interface Design.

6.3.3 Comparison of the Reviewers Ratings

The heuristic evaluation of the design seemed to result in hugely diverging scores. It seemed that reviewer 1 tended to give the design more points for fulfillment of requirement groups and design rules, than reviewer 2. Therefore a quick comparison of the sum of the points given by the reviewers was conducted.

The pie chart in Figure 6.2 presents each reviewers percentage of the sum of all points given in the test. As the diagram displays, more than 60 percent of the summed points were given by reviewer 1. A further look into the composition of points given was therefore conducted for each reviewer.

The diagrams displayed in Figure 6.3 and 6.4 show the composition of requirement groups and guidelines scores given for each of the reviewers. The score designated as 0 in the diagrams were not rated guidelines or requirements in the questionnaire. The first reviewer didn't rate 7.1 percent, while reviewer 2 didn't rate 9.5 percent, which are fairly equal numbers.

However, there the similarities stop, according to the charts. While reviewer 1 rated 7.1 percent of the requirements or guidelines as 5 (fulfilled), and 35.7 percent as 4 (partly fulfilled), reviewer 2 had a different opinion. This reviewer didn't rate any of the requirement or guidelines as fulfilled. 11.9 percent were however rated as partly fulfilled.

Further the second reviewer gave the score of 1 (not fulfilled) to 47.6 percent of all requirements or guidelines, while reviewer 1 gave 23.8 percent the same score.

Scores presented as decimal numbers are however average scores given when more than one square were checked in the questionnaire.

As the presented pie charts show, the reviewers had different opinion about the concept. A final chart was made to find the average score for each of the test's parts; all requirements to the future system, the requirements thought to be implemented in the current solution, all of the Eight Golden Rules of Interface Design and the golden rules which were checked in the questionnaire by the reviewers. The findings are presented in the bar chart in Figure 6.5.

6.3.4 Usability Inspection

The testers performed a usability inspection, to discuss if the future system could be fitting in the oil and gas industry. According to the testers the current solution have several advantages.

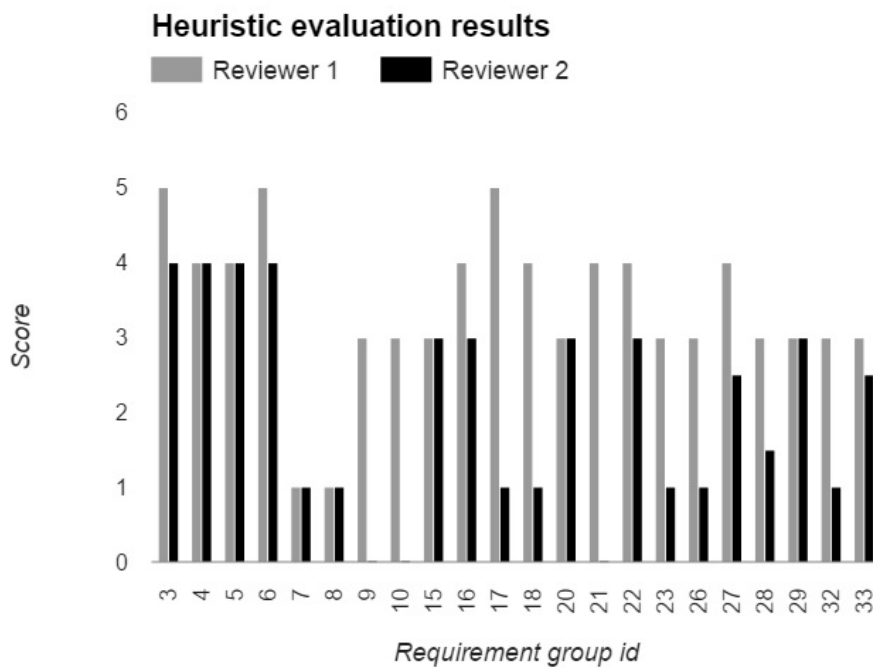


Figure 6.1: A bar diagram presenting how the reviewers scored the requirements thought to be implemented in the design.

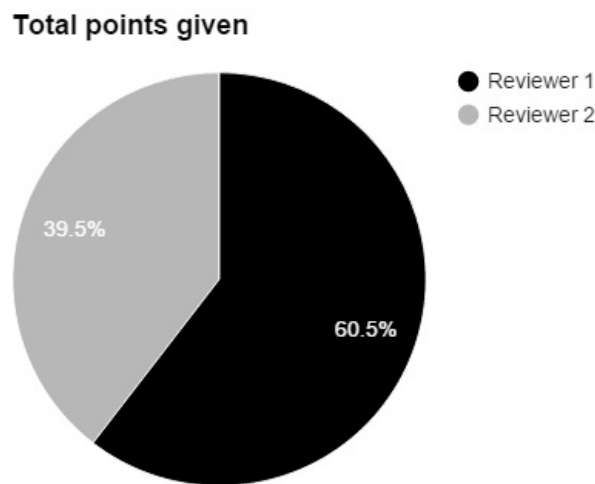


Figure 6.2: The percentage of points given by each of the reviewers, with regard to the total number of points given in the heuristic evaluation.

It was stated that well functioning work surfaces are necessary for successful implementation of the IO concept. The prototype was considered a step in the right direction.

It was also stated that the developed prototype tries to support a work

Percentage of scores given: Reviewer 1

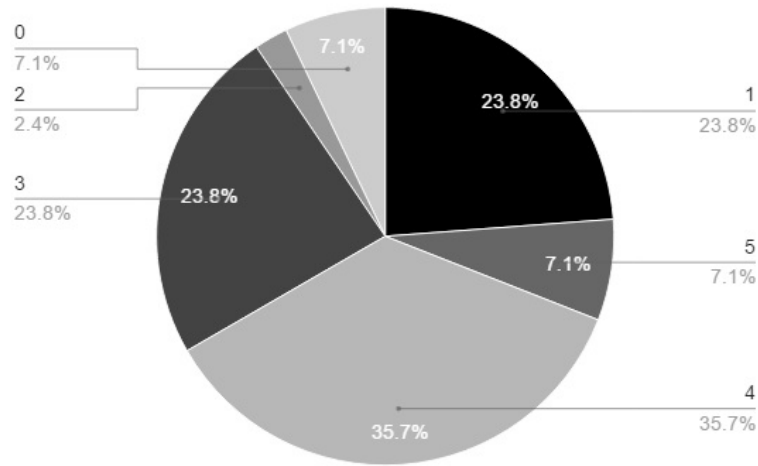


Figure 6.3: Composition of points given by reviewer 1.

Percentage of scores given: Reviewer 2

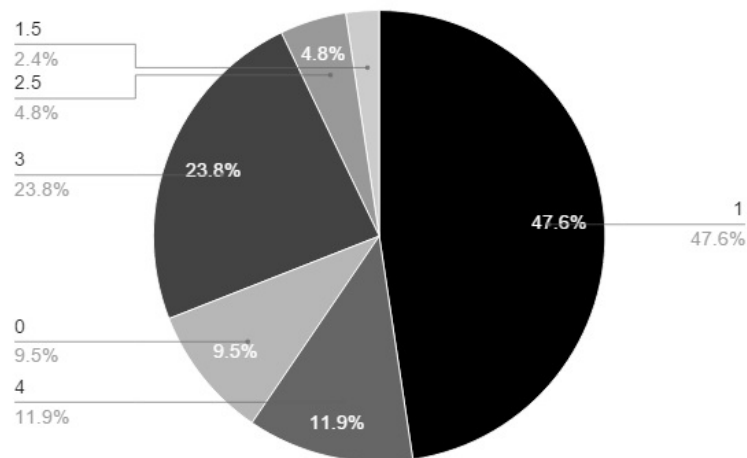


Figure 6.4: Composition of points given by reviewer 2.

practice for meetings. This is unique for such systems. The tool should therefore be adapted well to today's collaboration meetings.

Further one of the testers said that the tool now stores and presents actions made during meetings. By implementing such functionality the chances of an agreed job succeeding increases, alongside with an increase of situational awareness.

However, there are challenges connected to the current prototype. It was pointed out that existing tools are shown in meetings, and those, external tools may still be hard to use.

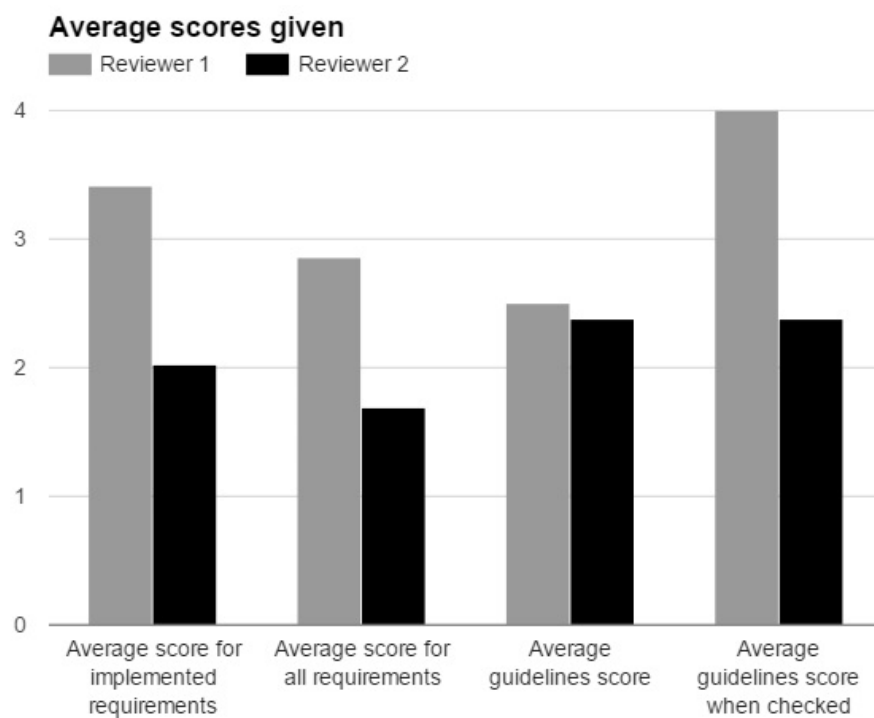


Figure 6.5: Average scores given in the different parts of the heuristics.

Chapter 7

Discussion

In this project several experiments have been conducted to try to answer the research questions, and to find whether software tools may be used for supporting decision making in morning meetings, and how. Additionally such a software solution should be adapted to the current Integrated Operations concept.

To find answers to these questions a set of requirements for a future morning meeting system was formed in collaboration with experts on the field. Current solutions were tested on the basis of these requirements. A new concept was partly designed as a proposal for a new solution, built on many of the requirements. Additionally the concept was based on information given in a meeting schedule, previously used in morning meetings. Finally the developed solution was tested on the basis of the requirements, guidelines and how well it could fit into the today's oil and gas industry.

In this chapter these experiments and results are discussed with regard to the research questions.

7.1 How May Software Promote Overall Decision Support in the Oil and Gas Industry?

Decision support implies, according to Kaarstad and Rindahl [16], supporting people making decisions. Software tools is a part of the decision making concept, aiding the users to solve problems and accomplish decision making.

However decision support tools may not implement the most advanced and state-of-the-art functionality to work well in practice. On the contrary, researchers as Sarshar and Rindahl [31] have studied collaboration technologies and found problems often connected to such solutions. According to their research modern and complex tools, implementing advanced functionality, tended to be prone to errors and to have a low degree of user-friendliness.

The same researchers observed the use of a simple solution, and found it to be the one working best in practice. This was a platform based on email and the SAP environment¹. The meeting leaders used the solution for conducting work both prior to, during and after collaboration meetings. The platform presented the users with fixed screen shots and email text.

Further the researchers gave the following advice to avoid implications when designing new decision support tools: "New technology should enable a desired work practice, and not the other way around."

How software could promote decision making in the oil and gas industry may therefore be linked to how well these tools are designed for supporting the current work practices.

However, collaboration meetings contain a range of activities - constantly reviewed and changed. The work practice may therefore be rather complex. According to Skjerve et al. [34], the actual meetings, called main collaboration sessions, are only parts in a bigger picture. Processes involving individuals and resources take place both before and after meetings. Preparations are made, the meetings outcome is analyzed and activities, determined in the meeting, may be executed and registered in the aftermath. The whole meeting process could also be changed and tuned for each iteration.

A software supporting such a work practice may therefore not only be made for actual meetings. It should also support the phases prior to and after the meetings.

The email solution, described by Sarshar and Rindahl, had embedded features for supporting such a process. The meeting leaders could send the email document to the participants before a meeting, use the same document as support in the actual meeting, and send it as a report when the meeting had finished. Such an approach may therefore be seen as supporting some of the work practices connected to meetings.

The continuous implementation of the Integrated Operations concept may however make solutions, as the email platform, obsolete over time. According to Ramstad et al. [28] the planning processes in current IO organizations, requests technology able to share real-time information. This is seen as especially important in short-term planning.

The software supporting decision making in meetings should therefore be able to display real-time information - and not only fixed screen dumps as in the email platform.

¹<http://go.sap.com/index.html?source=contentsynd-mrr-sap.co>

Further goals of the IO concept is, among others, reducing operational costs and increasing safety levels, according to the Kongsberg Group². A decision support tool should for that reasons mirror these goals by for example effectively serve the users needs, avoiding time-consuming and costly delays.

Functionality decreasing risks of accidents could also be included. A recent event, exemplifying the dangers connected to the oil and gas industry, is the explosions and fire on board the C32E platform in the Gulf of Mexico in 2012, where three people were killed. The reason for the accident was welding work igniting hydrocarbon vapor. Decisions made in the morning meeting, prior to the incident, could however have contributed to the accident. According to BSEE's report the operator authorized to approve the fatal welding work, was not present in the morning meeting. Instead another operator, without enough authority, made the decision of conducting the work.

²<http://www.kongsberg.com/nb-no/>

7.2 Defining Requirements for Decision Support in Morning Meetings

Following Sarshar and Rindahl's [31] advice for designing a future collaboration surface, presented in Section 2.1.1, more knowledge had to be obtained about the work-practices in morning meetings.

At IFE in Halden some researchers are currently studying how software tools may be designed for the oil and gas industry. They have conducted studies of personnel collaborating virtually, and observed their use of the available software tools. At the beginning of the project it was envisaged that personnel, working in collaborative environments in the oil and gas sector, could be observed or interviewed for studying the work process in morning meetings. This approach was however not possible to conduct, and two experts were instead interviewed for mapping the ideal features of a software supporting such meetings. Additionally a major oil company's guidelines for establishing collaboration rooms, and a morning meeting schedule, containing actual information, was handed out to the student.

If the student would have obtained more or better data about the work process and decision-making by observing morning meetings is unsure. It is possible that such an approach could have lead to better understanding of the work practices in the industry. But on the other hand, the interviewed experts had already obtained extensive knowledge about how morning meetings are conducted.

Therefore the experts were asked to suggest improvements and needs to a meeting system, designed as a collaborative platform for future morning meetings. The needs were not to be fundamented on already existing software. Instead the researchers were asked to promote all features they could think of being important in a future application. All the needs were then collected and written as individual requirements. This was done with the use of the Requirement Definition Methodology (RDM), presented in Section 3.1.2. The student took the role of the requirement engineer, and interviewed IFE's experts.

Additionally, guidelines for establishing collaboration rooms, written as an internal document in an oil company, was studied for adding requirements to the list. The guidelines were evaluated for associating the needs to today's standards for establishing collaboration rooms. Such requirements included the software not to play any sound effects that could disturb ongoing conversations, or for example to support different screen resolutions on the basis of the standard definitions.

A major pitfall with this approach was that, although the requirements were captured from polite sources and guidelines, the definitions were written by the student himself. Misunderstandings could lead to badly written requirements, ultimately undermining the process of defining needs for software supporting decisions in morning meetings. Therefore the list of requirements was reviewed by the experts at IFE. In this process some requirements were added, some were changed, and other removed. A final list containing about 100 requirements was ultimately produced.

At this point several requirements to a software supporting morning meetings had been formed. However, this list didn't specify if any of the requirements were more important than others, or in what order they could be implemented in a

new system. If the list was taken literally, when designing a new software, much resources could be used to implement unnecessary, or in worst case, unwanted features.

To minimize these risks a method, much like the MoSCoW prioritization technique, was used for rating each requirement to determine their importance. This approach led to the creation of a prioritized list of requirements, based on each of the reviewing experts' ratings.

The MoSCoW technique has been applauded for its simplicity - making it easy for stakeholders to get a grasp of how system functionality could affect the end-result. However, there are also critics. Kukreja et al. [17] argue that the method makes it hard to change prioritization if new functionality was to be added to the list. Additionally Kukreja et al. claim that it is not certain that the stakeholders are able to instinctively know the value of each requirement priority.

The advantages of choosing the technique were considered more valuable than the disadvantages. And since the experts were to define a set of ideal needs for systems, and not a specific or existing system, the list should not have to be changed. Therefore the disadvantages connected to future changes, could be minimized.

The other disadvantage, that the reviewers instinctively should know the prioritization of requirements, were not handled. It was thought that their knowledge about software development and the domain was enough for prioritizing requirements satisfactory with regard to the users' needs. It was however hard to overrule the prioritization, since the student did not possess the same knowledge about work practices in morning meetings.

However, a third problem occurred while conducting the MoSCoW prioritization technique. The reviewers ratings seemed to vary, making it hard to determine whether requirements should be rated as important or not. These variations could also indicate that the reviewers didn't instinctively know correct requirement prioritization, as mentioned in Kukreja's critics.

Further, the requirement list was fairly long, and contained detailed information. Some requirements were also almost similar to others. Most of them were therefore grouped together, forming a new list of 34 requirement groups. It was thought that this measure could simplify the evaluation of current prototypes and software. However, the groups were more generally written, without specific and highly detailed information.

Some requirement groups' features were rated as important to implement by the experts at IFE. A prerequisite for finally ranking these priorities as important was that both reviewers had agreed upon the importance of the current group. This approach was chosen to minimize the possibilities for producing a list of badly prioritized requirements validated as important.

11 requirement groups were finally defined as important. The list is presented in Section 4.2.3. Common to many of these requirements were that they seemed closely linked to the current work practice in morning meetings. They included possibilities for registering information, displaying the same platform to all users, producing reports, registering attendance, sharing the required meeting information, and taking the available hardware into account. Formed in this fashion the requirements would be designed to support a current work practice, hence following Sarshar and Rindahl's advice for designing collaboration tools.

Further an important requirement revolved around contacting users operat-

ing mobile devices. This could improve communication and collaboration with people not present in a meeting. The requirement could therefore be in accordance to important aspects of ICT tools in IPL, promoted by Ramstad et al. They claim that collaboration surfaces should make cooperation possible across organizational, professional and geographical borders. By extending the limits of a meeting to people not present in collaboration rooms, the collaboration could also be improved.

Lastly the set of important requirements defined several demands which could be in accordance to more general guidelines of interface design. These included an easy to use system, understandable automation, clear feedback and possibilities for adjusting readability options manually.

7.3 Testing Current Solutions

The main research question asks if software tools could better support decision-making in morning meetings, and how. One way to find if there were potentials for improvement in today's solutions, was to evaluate current software and prototypes based on the defined list of requirements. Such an evaluation was conducted, and the results of this activity is presented in Table 4.2 in Section 4.3.1.

The selected systems were however not tested, but evaluated based on online documentation and reports. It was considered to post questionnaires to all selected companies, and an online test was made for this purpose. A compressed pdf version of the test has been added to Appendix E.

The second approach, handing out questionnaires to companies, could have produced better results than by evaluating the systems with regard to perhaps incomplete documentation online. But this would have been a timely process, and the results obtained from the companies could also be incorrect by letting the companies decide if requirements were fulfilled or not. If one or more of the companies for example wanted to set their products in favorable light, this could undermine the evaluation.

Instead the evaluation was conducted by the student himself for all of the selected companies. This approach reduced the time spent on the evaluation, and provided perhaps more neutral results, than with the alternative approach, since all the evaluations were performed by the same person. But a disadvantage with this approach was the possibilities for not identifying all software features. More features could have been discovered if the student was able to study depth documentation, or to test the systems.

The systems and prototypes evaluated in the test was:

1. An email platform, previously used in morning meetings.
2. The prototype system IO-MAP, developed at IFE.
3. Wisio, a continuation of IO-MAP, developed at Østfold University College, in collaboration with IFE.
4. Protosphere, a meeting application where participants may control avatars in online meetings.
5. ARKit, a tool built around a map interface.
6. Epsis Teambox, a collaboration solution using windows to display video-conferencing and meeting information.

The systems were selected on the basis of their highly different nature, with one, major exception. Wisio and IO-MAP are designed relatively similar, and were both included in the test. This choice was made since the current project takes place at Østfold University college, in collaboration with IFE. IO-MAP was developed at IFE, while Wisio were a former project at Østfold University college.

The results of the evaluation is presented in Table 4.2 in Section 4.3.1. As the table shows, none of the prototypes or systems fulfilled all requirements defined earlier in the project. But it is important to point out that only implemented functionality has been evaluated. Software, as for example ARKit, could have fulfilled more requirements if an email solution was added to the current one, by sharing such a document on the system's activity stream. If so, ARKit would have scored better.

Another important remark is that some of the systems are designed for off-shore meeting activity, others are not. Epsis Teambox, Wisio, the email solution and IO-MAP are examples on systems which are designed especially for the off-shore oil and gas industry. ARKit and Protosphere are not. It may therefore seem unfair to compare these systems with the others, and score them. However, the solutions promoted by ARKit and Protosphere were considered both creative and innovative, and the systems were therefore added to the software list.

Further it is also important to notice that some functionality may be hard to implement in systems not designed for very specific tasks. Agents for decision support may for example not be fitted for systems such as Epsis Teambox, ARKit and ProtoSphere, since these systems may support a range of industries with a lot of different data and sources involved. Designing agents for such a huge span of businesses may be an overwhelming task.

Lastly the readability requirements may lead to overwhelming changes to some of the systems' design philosophies. Epsis Teambox is for example a system publishing all kinds of information in virtual screens, and that functionality may be seen as one of the system's main advantages. Readability tests, etc. may be more fitted for systems presenting integrated text.

With all this in mind, it appears that some areas in Table 4.2 are just partly supported, or not supported at all, by any of the analyzed systems. This applies especially to functions associated with communication with mobile devices, implementation of agents, integration of an overview of competence, and lastly readability feedback and test.

Further the fulfillment of the requirement groups defined as the most important, varied among the systems and prototypes. Epsis Teambox and Protosphere stood however out as the systems fulfilling the most of these. But both systems were found to implement limited functionality when communicating with users operating external devices. Epsis Teambox neither implemented embedded functionality to time meeting attendance. This was found to be at place in Protosphere with the use of an extension. But Protosphere was perhaps less fitted when it came to taking the available hardware in meeting rooms into account. Protosphere have implemented avatars for conducting meetings by individual participants. But morning meetings in the oil and gas sector are not organized in that way.

7.4 A Concept Made for Promoting Decision Support in Morning Meetings

On the basis of the evaluation of current systems and prototypes, it was found that neither of the systems fulfilled all the requirements defined. It was therefore considered naturally to try to implement most of these features in a concept illustrating a future meeting system.

The concept was developed in several iterations with a RAD-like method. New functionality was added by the student from the beginning of each iteration. Later the created functionality was demonstrated for the contracting authority. Lastly refinements were made in cooperation with the contracting authority before a new iteration could commence. This last activity could also be viewed as a minor expert review.

The actual prototypes, created in the iterations, followed on the other hand both horizontal and vertical strategies. If this choice was beneficial may be discussed. Since when deciding to create a vertical prototype, much time was spent coding features, that may have been easy to implement with a simpler technique - saving time, and allowing more iterations.

On the other hand, the use of vertical prototyping lead to discovering several weaknesses in the former system. The severity of the weaknesses triggered new, overall design iterations.

If these weaknesses also would have been detected, by only developing overall, light-weight prototypes, is not certain. The product of the vertical prototyping itself, on the other hand, may therefore be seen as an important part of the development design process, and a mean to test features to find new to implement in the overall design.

How the prototyping was commenced is described in Chapter 5. In addition to the requirements defined, some functionality discovered in other systems were also implemented as parts of the prototyping design, when they could contribute to the overall goal - to strengthen meeting decision support.

7.4.1 Using Features Discovered in Other Software

Seven of the eleven alternatives, presented in the list in Section 4.3.2, were at some point implemented in the project.

From Epsis Teambox the global modes (linked to the work process), the presentation of external software in windows, and the large cursor were all sources of inspiration during the prototyping development.

The prototyping design implemented in stage seven an outer subsystem containing the pre-meeting and post-meeting activities. This software part was inspired by the work processes in a DCS structure, as described by Skjerve et al. [34]. But it was also inspired by Epsis Teambox, and the solution's use of global modes.

Further the current prototype design implemented capabilities for displaying meeting information in a single window inside the application. Here external systems, documents and web sites could be presented in a future system. By implementing a window, capable of displaying live information, it was thought that the system could better support decision making. Further IO is, accord-

ing to NOG³, in constant development. By locking the system to integrated functionality it could perhaps be less suited to the changing IO-environment.

Epsis Teambox was however the main source of inspiration for including the window in the prototype. Epsis have implemented a design capable of displaying numerous windows. But in this prototype functionality for displaying several windows was not implemented, to avoid confusion among the participants in the meeting.

Further Epsis Teambox' implementation of a large cursor was also used in early versions of the current prototype. The background for including the large cursor was to make it easy for meeting participants to view the object on the screen. This feature was however omitted after conducting the cognitive-walk-through - narrowing the system design.

A single feature was also obtained from Protosphere when developing the design. Protosphere has possibilities for displaying a contact list displaying personnel's contact information, and a portrait images of them. A similar feature was implemented in the first prototype iteration, and later removed in after the cognitive walk-through.

The reason for implementing the contact list was to inform users about personalia and each of the participants' work titles. This was thought to strengthening the fulfillment of requirement group 19 in Table 4.1, emphasizing the need for displaying competence and possibly use this as decision support. It was thought that displaying work titles could indirectly inform users about the current participants' authority to make decisions during meetings.

One feature was implemented from IO-MAP. The map environment, as the one in IO-MAP, was included in the design, and was also created in code in the vertical prototypes. Although the final prototype depend on showing such feature as an external software, the importance of presenting information on a map was highlighted. This was especially pointed out during the third prototyping iteration, described in Section 5.2.3.

Two features, discovered in ARKit was lastly included - zooming and notifications, although notifications also were described in the requirements.

Zooming was implemented as a tool for the window displaying external software, if the systems had support for it. Since external software do not necessarily implement features for readability adjustments, the zooming tools were thought to strengthen fulfillment of requirement 26 - describing needs for such adjustments.

Further possibilities for displaying notifications were implemented in all versions of the prototype until version eight.

7.4.2 Major Changes to the Design

During the development of the prototypes several changes were made to the design. Some of them were however considerably large - completely changing the prototypes. Especially two events involved such changes - the transition from prototype one to prototype two, and the transition between prototype six and seven.

All changes were however committed to strengthen decision support provided by the future software.

³<https://www.norskoljeoggass.no/en/Publica/Guidelines/>

Transition from Prototype 1 to 2

The initial prototype design, presented in Section 5.2.1, consists of several sites, accessed by changing selection in a combobox. The sites have large buttons, illustrating the topics to be discussed in a meeting. If HSE was selected in the combobox, such topics could include deviations, focus points and new events affecting HSE. These examples are also mentioned in the actual email document studied in Section 4.1.

Additionally the initial design have implemented several features thought to be used globally - as for example notifications, help and settings.

But when undergoing the transition from the initial to the second prototype design, all functionality previously available in all parts of the design, were placed together into a single menu, consisting of tools and a schedule imitating the meeting's agenda. A large window to the right of the menu was allocated to displaying meeting specific information.

The reason why a new menu was made, was to make this object visible to the meeting participants at all times. It was considered utterly important that the participants knew what information they were viewing, and what information that remained to be shown during the meeting. The new menu could at some degree support that. This was also in accordance to requirement 18 (present clear navigation) and 33 (sharing required information in morning meetings) in Table 4.1.

Additionally the new menu could be easier to understand by the participants than the previous, since it was closer linked to schedules used in current morning meetings, hence supporting current work practices.

The window to the right of the menu was positioned there to make the users known to the organization of the design. By always displaying meeting specific information at the same place, the interface could perhaps be more intuitive and easy to use - supporting requirements 20 and 22 in Table 4.1.

Transition from Prototype 6 to 7

The second, major change to the design was carried out when creating the seventh prototype. After conducting the functional prototyping it became clear that it could be hard to create new meeting setups from within an already existing meeting menu. This could again affect the design's ability to support collaboration in an IO-environment, by making it hard for users to perform tasks before a meeting starts. The model presented by Skjerve et al. [34], emphasizes the importance of several stages both before and after a meeting.

The design was therefore expanded to offer support for more stages, according to the DCS perspective. The seventh prototype therefore included possibilities to create new meetings, alter existing ones, alter the information to be presented, and for example to send reports when meetings had been conducted. This was done by implementing a new, outer part of the system - acting as an editor and a meeting organizer.

Another advantage of this new design was perhaps the possibilities to support the IPL process. According to Ramstad et al. [28], it is essential that there exists an interplay between planning levels in IO environments. Such interplay should involve both organizations, groups, units and professions. The new design in iteration seven offered possibilities to invite personnel from within the

organization and to publish meeting transcripts to both meeting participants and others, hopefully in accordance with IPL and the IO concept.

During this transition the in-meeting part of the application was also changed. A new window, where users could add written information, files and screen dumps, was added. It was thought that the feature could increase decision support by fulfilling the third requirement in Table 4.1 - supporting possibilities for registering information during meetings. This could also be in accordance with requests promoted by Ramstad et al.'s [28]. The authors emphasize the need for transparency in IO planning, and the need for sharing. By storing comments, and other written information, discussions in meetings could be more visible and transparent for third party personnel.

7.4.3 Removing Functionality

The prototype was tested after completing the seventh iteration. A cognitive walk-through, involving two reviewers, was conducted. Several remarks were made, and some of them regarded functionality implemented on the basis of the earlier defined requirements. This led to a new iteration for changing the prototype design. During this activity several features were abandoned, although they fulfilled described requirements. This section tries to explain those choices.

During the cognitive walk-through the tool box, implemented at first in the second prototype, was considered unnecessary. The features available inside it involved both functionality for creating whiteboards, displaying notifications, altering settings, requesting help and viewing a list of the participants. All these functions were requested omitted to make the future software quicker to use, preventing costly and time-consuming detours involving the tools.

The eight prototype iteration therefore didn't include the tools, other than the capabilities for zooming - located differently in the design.

But if the act of removing the tools could be in compliance with increasing support for decision-making is questionable. Some of the tools, as the ones used for displaying notifications and whiteboards, were defined as requirements for the future system.

The reviewers, on the other hand, pointed out that participants didn't have time to access such functions during a meeting. This is described further in Section 6.2.2. Such an argument could also be in compliance with the goals of the IO-concept, as described in Section 7.1. Then it is discussed how systems could promote overall decision support in oil and gas organizations. It was suggested that systems should strive for the avoidance of time-consuming and costly delays.

But if the meeting participants didn't understand how a function worked, or something unexpected occurred preventing them from finishing their job, a help menu could be used for solving their problems. If so, the tool could perhaps reduce the time spent on figuring out an undocumented feature.

Removing settings may also defy the second of the Eight Golden Rules of Interface Design; "cater for universal usability", as described by Shneiderman and Plaisant [33], and may dismiss possibilities for adjusting the interface to universal usage.

The arguments for removing such tools are that the tools, if available, could have been accessed too much - causing an unnecessary amount of time spent on

such features in meetings. And time is money.

As mentioned in Section 1.1.2, the annual cost inflation in the offshore industry have been much higher than the CPI in Norway for several years. At the time of writing the value of European crude oil has sunk and is less than 50% of the value in the summer of 2014 - when this project was about to start. The recent cost development may therefore have had an impact on the business and also the reviewers of the project. It is possible that such functionality could have been more welcome if the economic situation was different.

But this hypothesis may not be accurate. Another argument for removing the tools is that a meeting software should be simple and highly intuitive (as described in the requirements) - perhaps reducing needs for such tools as the help menu.

Further it may be argued that tools could divert the participants' focus. If so, their attention could be directed too much against the tools, ultimately reducing decision support.

Nevertheless, most of the tools were removed in the eight prototype. But if the choice was correct in regard to support for decision-making is questionable.

7.4.4 Features not Tested

Several features designed in the prototypes have not been tested. It is therefore hard to determine if they would work in practice, if the features cost a lot to implement, or if it could be a timely process adding them to the system.

Necessary features not tested included for example the window displaying external software. How manipulation of software presented in the window would work has not been taken into account. Such manipulations could include zooming the window's content, or for example conducting writing operations.

But since current solutions as Protosphere⁴ and Epsis Teambbox⁵ already has implemented quite similar functionality, it was taken for granted that it may be implemented in a future software.

This is also the reason why the prototype developed vertically, in the iterations 4-6, didn't include testing of displaying other systems. Instead the vertical prototype implemented functionality for accessing a crucial part in the design - the menu displaying the meeting's schedule.

The menu has been developed in code with zooming functions, whiteboards, a working map interface and, among others, possibilities for expanding or collapsing the content displayed in the external window.

It may, on the other hand, be argued that the vertical prototype should not have been concentrated towards developing this menu. The menu could be considered easy to develop, and therefore not necessary to test.

However, if there had been time to perform more tests it could have been interesting to figure out if the window displaying meeting specific information could work in practice. It could also be interesting to try to communicate with personnel operating other devices - and to find if these devices were capable of displaying the meeting platform.

⁴<http://www.protonmedia.com/>

⁵<http://www.epsis.no/product/>

7.5 Testing of the Developed Concept

The concept was evaluated in three individual tests - a cognitive walk-through, a heuristic evaluation and a usability inspection.

Other techniques could have been chosen. But tests involving users were excluded because of lacking access to this group. The background for the selection of test methods is described in Section 3.5.

7.5.1 Cognitive Walk-Through

The cognitive walk-through, presented in Section 6.2, resulted in an additional development iteration, which has been thoroughly described in Section 5.4.2.

As earlier mentioned, the feedback given during the test resulted in several design changes. It was considered beneficial to implement the proposed changes into the project before conducting the later tests. The reviewers had extensive knowledge about software development in oil and gas organizations, and were therefore considered capable of adjusting the design to the work practices in morning meetings, hence improving the future system's decision support.

If the researchers assumptions about the design were correct, may be argued. But since the student possessed little knowledge of the practical execution of morning meetings, it was hard to overrule the reviewers' feedback. The proposed amendments were therefore implemented into a new design.

The student limited knowledge of the practical execution of morning meetings, beyond available theory, was also expressed in how the system was designed for delivering reports. The student had thought about implementing a reporting feature outside meetings, making it possible for meeting leaders to send the reports after meetings had ended. But the reviewers noted that reports usually were sent during morning meetings, and rarely at a later stage - something the student did not know. The test therefore helped trimming the design to the practical work situation in morning meetings, perhaps improving the quality of decision support.

The practical execution of the cognitive walk-through may also be discussed. To make the reviewers familiar with the design, while still testing the system, they were given several tasks to accomplish. These tasks are described in Appendix D and included creating new meetings, editing meeting setup, starting meetings, navigating in meeting content, and reporting meetings. The tasks were however numerous to test as many parts of the system design as possible.

But it may be argued that the cognitive walk-through could limit the users' review of the system only to the tasks given, and that not mentioned parts could be forgotten.

The feedback given may to a certain extent disprove this. As described in Section 6.2, the reviewers noticed that symbols used in the interface sent the wrong signals. Reports not sent, or meetings without invitations, could be misinterpreted as hazards or alerts. Additionally the reviewers noticed that the actual design could be simplified by removing borders or limiting the use of contrasting colors. None of these reviews were part of the tasks given in the cognitive walk-through. It may therefore be argued that the test found weaknesses in the design beyond the tasks given, and that the solutions to the weaknesses could support decision-making, as for example by removing symbols that could be misinterpreted and taking too much attention.

7.5.2 Heuristic Evaluation

As mentioned in Section 3.5, the heuristic evaluation was considered an important part of the concept test. It was divided into two questionnaires - one including all the requirements defined for the future system, the other including the Eight Golden Rules of Interface Design, as presented by Shneiderman and Plaisant [33]. The reviewers were then asked to rate the fulfillment of each requirement or guideline based on their impression of the design.

It may be argued that the scope of this approach was unnecessary wide. Some of the requirements had not been designed, as for example communication with mobile devices or implementation of agents to improve decision support. But it was considered favorable to list all guidelines and requirements to accentuate the design's potential for improvements and to find successful implementations.

Varying Test Feedback

The reviewers feedback varied however a lot, as presented in the test results, described in Section 6.3.3. By comparing the reviewers' ratings of requirement fulfillment, the results showed that reviewer 1 had given more than 60 percent of the total points for rating the system. Reviewer 1 therefore tended to hand out higher scores, and to approve more of the system's design features, than the second reviewer.

A closer look into the ratings, presented in Figure 6.3 and 6.4, showed that reviewer 1 had rated more than 40 percent of the requirements and guidelines as fulfilled, or partly fulfilled. The second reviewer had rated only about 12 percent of the heuristics as partly fulfilled, and none as fulfilled.

Statistics, presented as a bar chart in Figure 6.5, further showed that reviewer 1 had given an average score for all the requirements as a little less than 3, which stands for "either fulfilled or not fulfilled". The second reviewer had at average given the requirements the score of about 1.5, which is between "partly not fulfilled" and "not fulfilled".

When it came to the guidelines, the second reviewer had omitted answering several guidelines. All the guidelines that had been rated, were rated as 4 - "partly fulfilled". The same guidelines were rated approximately as 3, "either fulfilled or not fulfilled", by the second reviewer.

These results may indicate that the first reviewer thought the system could be better suited for morning meetings than the second reviewer. Reviewer 1 had generally given the system higher scores than reviewer 2, therefore approving more of the implemented features.

On the other hand, it may also be possible that the reviewers' differing ratings could be linked to their opinions on how the prototype could work in practice. The designed prototype is only containing slides presenting future features. The features themselves are not part of a working system, except the minor vertical prototype implementation in code. If the first reviewer rated the system on the basis of a future, working implementation, it could explain the differences, if the second reviewer did not.

The above assumption may be further strengthened by looking closer into how the reviewers rated the different requirements. Reviewer 1 rated for example the design's intuitive and understandable automation as partly fulfilled, while

the second reviewer didn't answer anything according this need. This may indicate that reviewer 1 rated the system based on how it could work in practice in the future, while the second only rated the current design slides.

Nevertheless, the differences between the test results varied a lot, something that could indicate a need for involving more people as reviewers in the heuristic evaluation. If so, the results could be compared, and hopefully be better fitted for rating the current design. But when the test was conducted, only two reviewers had time to perform it. Later tests, involving more people, could have been favorable, but such tests have not been conducted in this project.

The results of the heuristic evaluation may therefore be viewed as non-conclusive. However, they may also indicate that the current design has weakness and possibly a large potential for improvement.

Fulfillment of Most Important Requirements

In Section 4.2.3 several requirements defined as especially important in a system supporting morning meetings are presented. They were not highlighted in the heuristic evaluation handed out to the reviewers, to avoid special treatment of these requirements. The reviewers' ratings of all requirements are presented in Table 6.1. The requirements considered important were ranked in the following way:

- *Requirement 1*: According to both reviewer 1 and reviewer 2, functionality for communicating with users operating mobile devices is not existing. This may also conform to the implementation of the design - where such features are not present.
- *Requirement 3*: Further the possibilities for registering information was rated as fulfilled by the first reviewer, and partly fulfilled by the second. This may indicate that the implementation of this feature is mostly satisfactory in the current design.
- *Requirement 9*: Implementation of a meeting-mode was only rated as 3, either fulfilled or not fulfilled, by the first reviewer. The second one was unsure if the requirement was fulfilled, or possibly the formulation of the requirement. It may therefore be hard to consider if the requirement is satisfied in the system, or the degree of satisfaction.
- *Requirement 10*: Reviewer 2 was also unsure on either wording or the fulfillment of requirement 10, while reviewer 1 rated the implementation of an after-meeting mode as partly fulfilled. As for requirement 9, this may indicate that there are uncertainties connected with the current implementation.
- *Requirement 16*: The ability to register and time meeting attendance was rated as partly fulfilled by the first reviewer, and either fulfilled or not fulfilled by the second. Although the system has integrated possibilities for registering participants, the functionality may be improved to satisfy the users' needs.
- *Requirement 20*: Both reviewers rated the requirement, demanding the creation of a system stripped of unnecessary functionality, as either fulfilled or not fulfilled - indicating needs for improvement.
- *Requirement 21*: Reviewer 2 didn't answer if the system has understandable or intuitive automation. Reviewer 1 meant that this requirement

was partly fulfilled. Nevertheless, the inconclusiveness of the result may indicate that more work is needed to fulfill the requirement.

- *Requirement 22*: Reviewer 1 thought that the design partly fulfilled the need for presenting the viewers with clear system feedback. Reviewer 2 rated this need as partly fulfilled. More work may therefore be needed for satisfying the requirement.
- *Requirement 26*: Since software, that could be displayed in the future application, may not have possibilities for setting readability adjustments, zooming tools were added to the design. However, reviewer 1 rated the current implementation of readability adjustments as either fulfilled or not fulfilled, while reviewer 2 rated it as not fulfilled.
- *Requirement 32*: It may be considered unclear if the current system may take the available hardware into account. The first reviewer rated this requirement as either fulfilled or not fulfilled, while the second thought that this need was not satisfied at all. More work may therefore be needed for adjusting the design to the current hardware available.
- *Requirement 33*: The design's capabilities for sharing the required information in morning meetings was however rated as partly fulfilled, by the first reviewer, and either fulfilled or not, by the second. This indicates that there may be a need for implementing more features supporting morning meetings.

According to the MoSCoW method, described in Section 2.4.2, all the must-requirements have to be fulfilled if a project may be viewed as a success. With respect to this criteria the current design may be seen as a failure, by comparing the fulfillment of the most important requirements with the results of the heuristic evaluation.

In the prototype's defense it may be possible that it is impossible for a meeting software to fully implement all defined requirements. Such a solution could be a Utopian one. As earlier mentioned in Section 7.2 the experts did not build their needs on existing solutions when defining demands for a future meeting system. Combining the requirements in practice may therefore be hard. As described in Section 7.3 software as Epsis Teambox implements possibilities for displaying external programs, but lacks, according to the evaluation, readability support. However, it may be extremely hard to implement functionality for adjusting readability when displaying information not embedded in a system.

It may also be argued that the evaluation of the prototype gave mediocre results, and that there exists software that fulfill more requirements than the proposed concept, with regard to the earlier evaluation of prototypes and systems. This may be the case, although it could be difficult to compare results of evaluations carried out in different ways, and with different reviewers. The evaluations of the systems were conducted to determine if they had implemented some sort of functionality similar to the requirement definitions. Such an evaluation was also done by the student himself with all the prototype versions, and is displayed in Table 5.2.

It is possible that if the same reviewers conducted heuristic evaluations with the software and prototypes, in the same fashion as for the prototype developed in this project, the evaluation's results could have been highly different.

Nevertheless, the design proposed in this project is not ready to be released, and was never meant to be. It is a concept made for illustrating how a system

may be made to fit into a morning meeting context. How suitable the idea of a future system is for morning meetings may not be clearly enough explained by only evaluating the results of the heuristic evaluation. These results only rank the current design, but not how a fully developed and enhanced version of the design would work in its context. The formal usability inspection, meant for evaluating the concept on a higher level, may answer just that.

Formal Usability Inspection

The reviewers were asked to evaluate if the future system could be adapted to morning meetings. According to the feedback the prototype system may be viewed as a step in the right direction for supporting decision-making. The reviewers found several advantages, which are mentioned in Section 6.3.4. The idea was considered unique, since it was adapted to a specific work practice in meetings. Further it was considered useful that the prototype supports and displays actions and decisions. A disadvantage could on the other hand be linked to displaying other software, that may perhaps be hard to use.

However, a countermeasure to the the system's disadvantage, regarding the lack of control of external systems, have earlier been described in the third prototyping iteration in Section 5.2.3. Although full control may not be possible to achieve, proposals for visualization techniques were mentioned in respect to a meeting's different topics. Table 5.1 presents an overview of techniques to be used with the current design. the below proposals were considered important, replacing the perhaps widespread use of tables, as in the studied email document described in Section 4.1.

Firstly it could be favorable to include software visualizing maps for sharing information, perhaps by displaying platform decks. According to Steele and Illiinsky [37] map visualizations may be a good technique while displaying data to users familiar with a location. It has therefore been proposed to include such software.

In the third prototyping iteration it was also noticed that table data could be converted to other kinds of visualizations. One proposed technique, described in Section 5.2.3, presented size visualizations of data formerly displayed in tables. The example was inspired by a visualization by Steele and Illiinsky [37], and could make it easier for users to compare and separate data just by looking at their visual representations.

7.5.3 The Design's Ability to Provide Decision Support with Respect to the Test Results

The concept produced in this project has been tested in several ways. The cognitive walk-through indicated strongly that a new design iteration was needed. This was conducted to adjust the design to the reviewers' feedback. The test was therefore useful for, among others, trim the design to the current work practices in the industry - perhaps strengthening support for decision making.

The heuristic evaluation, on the other hand, provided varying results. Although the test may be considered inconclusive, the results indicated needs for improving the current design.

The results of the heuristic evaluation were however not satisfactory in respect to fulfillment of requirements and the MoSCoW perspective. None of the

reviewers rated all important requirements as fulfilled. But the concept is not meant to be released in the current state.

The formal usability inspection, may on the other hand, indicate that the concept idea was both unique and adapted to the current IO context in the oil and gas industry.

Chapter 8

Conclusion

This project has been conducted in several phases to answer if there may be better ways to support decisions in morning meetings than what is available today. The project also tries to answer how software, supporting such meetings, may promote important elements of the IO concept.

To answer these questions a list of requirements were made in cooperation with experts in the field at IFE in Halden. These requirements were to mirror ideal system goals for improving decision support.

Features available in current solutions and prototypes were compared with the requirements to find if any satisfied the defined needs. Several weaknesses were found in this process, paving the way for a new concept adapted to the requirement list, and the current IO context.

However the tests of the developed prototype indicated that this system was not fulfilling all needs, and had several weakness, although the outcome of the heuristic evaluation may be seen as inconclusive.

The current design is therefore not the ideal solution for improving decision support in morning meetings. But future systems may take lessons from some aspects of the concept development. The creation of a prototype, based on work practices in morning meetings, was considered both unique and a well-suited approach for developing a system adapted to today's collaborative meetings.

Further the system was considered a step in the right direction in the current IO context.

Additionally the list of requirements indicate that there is a need for improving decision support with regard to the current use of systems in morning meetings. Features requested by experts in the business include possibilities for displaying a static view of a work surface to users in meetings and personnel operating mobile devices. Such functionality was not found in any of the evaluated systems. By studying the requirement list, provided in this project, both future and current solutions may improve decision support in collaborative meetings.

References

- [1] R. Agarwal, J. Prasad, M. Tanniru, and J. Lynch. Risks of rapid application development. 2000.
- [2] E. Albrechtsen. Integrated operations concepts and their impact on major accident prevention. In E. Albrechtsen and D. Besnard, editors, *Oil and Gas, Technology and Humans: Assessing the Human Factors of Technological Change*, pages 11–27. Ashgate Publishing Limited, 2013.
- [3] J. D. Arthur and M. K. Gröner. An operational model for structuring the requirements generation process, 2004.
- [4] Epsis AS. Epsis Teambox User Manual, 2014.
- [5] M. Beaudouin-Lafon and W.E. Mackay. Prototyping Tools and Techniques. 2002.
- [6] H.C. Bjørnland. Oljeprisfallet – Økonomiske og politiske utfordringer. 2015.
- [7] A.O. Braseth and S. Sarshar. Improving oil & gas installation safety through visualization of risk factors. 2012.
- [8] R. N. Burns and A. R. Dennis. Selecting the appropriate application development methodology. 1985.
- [9] ConocoPhillips Australia Pty Ltd. ConocoPhillips Australasia Business Unit, Contractor HSE Management Process – Attachment A1, Health, Safety and Environmental Requirements for Contracts. http://www.conocophillips.com.au/Documents/SMID_022_West_HSE_Req.pdf.
- [10] Epsis, Norway. Epsis Teambox: The beauty of collaboration - done right. 2015.
- [11] T. Hollingsed and D.G Novick. Usability Inspection Methods after 15 Years of Research and Practice. 2007.
- [12] Ark Platforms Inc. The ARKit Feature List 4.3. <http://www.arkplatforms.com/arkit/feature-list>.
- [13] Polycom Inc. Protosphere Virtual Immersive Environments for Learning and Collaboration. <http://www.protonmedia.com/resourcepage/protonmedia-partner-playbook-4962-0113-enus.pdf>, 2013.

- [14] ProtonMedia Inc. Protosphere - a Highly Engaging 3D Virtual Workspace. http://www.protonmedia.com/resourcePage/ProtoSphere_Product_Datasheet.pdf, 2013.
- [15] ProtonMedia Inc. Protosphere for ipad. <http://www.protonmedia.com/resourcePage/ProtoSphere-for-iPad-Data-Sheet.pdf>, 2013.
- [16] M. Kaarstad and G. Rindahl. Shared collaboration surfaces to support adequate team decision processes in an integrated operations setting. In C. Berenguer, A. Grall, and C.G. Soares, editors, *Advances in Safety, Reliability and Risk Management*, pages 241–242. CRC Press, London, 2012.
- [17] N. Kukreja, B. Boehm, S. Payyavula, and S. Padmanabhuni. Selecting an Appropriate Framework for Value-Based Requirements Prioritization. 2012.
- [18] A. Lågbu, S. Sarshar, and G. Rindahl. Improving ICT Tools as Support for Morning Meetings in the Oil and Gas Industry. 2015.
- [19] McKinsey & Company. Meeting the challenge of increasing North Sea costs. 2014.
- [20] Mediasphere. Rapid application development methodology. <http://www.mediasphere.com.au/files/file/MediasphereRapidApplicationDevelopmentMethodology.pdf>.
- [21] S. Mira. Agile software development practices: evolution, principles, and criticisms. 2011.
- [22] Norsk Olje og gass. Norway’s petroleum history.
- [23] U.S. Department of the Interior Bureau of Safety and Environmental Enforcement. Investigation of November 16, 2012, Explosion, Fire and Fatalities at West Delta Block 32 Platform E, 2013.
- [24] E. Okstad, E. Jersin, and R. Kviseth Tinmannsvik. Accident investigation in the norwegian petroleum industry - common features and future challenges. Technical report, SINTEF Technology and Society, 2011.
- [25] C. S. Olsen, O. G. Nedrebø, P. J. Berg, J. M. Røsok, and M. Eskerud. Web based Information Surface for a Petroleum Installation. <http://www.it-stud.hiof.no/h13d28/bachelorReport.pdf>, 2013.
- [26] G.O. Ose and T.J. Steiro. Introducing IO in a Drilling Company: Towards a Resilient Organization and Informed Decision-making? In T. Rosendahl and V. Hepsø, editors, *Integrated Operations in the Oil and Gas Industry: Sustainability and Capability Development*. Business Science Reference, Hershey, 2013.
- [27] ProtonMedia. Microsoft Lync - Learning in New Ways, Anytime, Anywhere. http://www.protonmedia.com/resourcePage/ProtonMedia_ProtoSphere_Learning_on_Lync.pdf.

- [28] L. S. Ramstad, K. Halvorsen, and E. A. Holte. *Integrated Operations in the Oil and Gas Industry*, chapter Implementing Integrated Planning: Organizational Enablers and Capabilities, pages 171–188. Business Science Reference, Hershey, 2013.
- [29] S. Sarshar, B.A. Gran, S. Haugen, and A.B. Skjerve. Visualisation of risk for hydrocarbon leakages in the planning of maintenance and modification activities on offshore petroleum installations. 2012.
- [30] S. Sarshar, S. Haugen, and A.B. Skjerve. Factors in offshore planning that affect the risk for major accidents. *Journal of Loss Prevention in the Process Industries*, 2014.
- [31] S. Sarshar and G. Rindahl. Integrated operation collaboration technologies - remaining challenges and opportunities, 2014.
- [32] S. Sarshar, A.B. Skjerve, G. Rindahl, T. Sand, and B. Hermansen. Quality aspects in planning of maintenance and modification on offshore oil and gas installations. 2011.
- [33] B. Shneiderman and C. Plaisant. *Designing the User Interface*. Pearson Higher Education, Boston, 2010.
- [34] A. B. Skjerve, E. Nystad, G. Rindahl, and S. Sarshar. Assessing the Quality of collaboration in an Integrated Operations Organization. 2013.
- [35] A. B. Skjerve, S. Sarshar, G. Rindahl, A. O. Braseth, H. O. Randem, O. Fallmyr, T. Sand, and C. K. Tveiten. The Integrated Operations Maintenance and Modification Planner (IO-MAP) - the first usability evaluation - study and first findings. 2011.
- [36] A.B. Skjerve, S. Sarshar, G. Rindahl, A.O. Braseth, H.O. Randem, O. Fallmyr, T. Sand, and C.K. Tveiten. The integrated operations maintenance and modification planner (io-map). 2011.
- [37] J. Steele and N. Iliinsky. *Beautiful Visualization - Looking at Data Through the Eyes of Experts*. O'Reilly Media Inc., 2010.
- [38] C. Taylor, S. Sarshar, and S. Larsen. How io leaders can use technology to enhance risk perception and communication. 2014.
- [39] J.E. Vinnem. Evaluation of offshore emergency preparedness in view of rare accidents. Technical report, University of Stavanger, 2010.
- [40] J.E. Vinnem, J.A. Hestad, J.T. Kvaløy, and J.E. Skogdalen. Analysis of root causes of major hazard precursors (hydrocarbon leaks) in the norwegian offshore petroleum industry. Technical report, University of Stavanger and Safetec Nordic AS.
- [41] A.M. Wahl, H. Sleire, T. Brurok, and B.E. Asbjørnslett. *Agility and Resilience in Offshore Operations*, 2008.

Appendix A

Research Article Written During the Project Work

Improving ICT Tools as Support for Morning Meetings in the Oil and Gas Industry

A. Lågbu

Østfold University College, Halden, Norway.

S. Sarshar

Østfold University College and Institute for Energy Technology, Halden, Norway.

G. Rindahl

Institute for Energy Technology, Halden, Norway.

ABSTRACT: The petroleum industry in Norway operates their offshore installations with an offshore organization and an onshore support organization. One of the most important collaboration meetings between these is the morning meeting. The software tools used in such collaboration meetings have an important role in focusing and communicating important information to decision makers. Decisions made in morning meetings may impact both production and safety of the installation. This paper reports on an evaluation of existing and research software tools applicable for morning meetings and how well they support the meeting. Based on this study and workshops with experts in collaboration surfaces and environments, requirements for an improved tool were established and a first concept of a new software framework based are developed and presented.

1 INTRODUCTION

The petroleum industry in Norway operates their offshore installations with an offshore organization and an onshore support organization. While the production and maintenance work is executed offshore, more of the administrative work and preparation are managed by the onshore organization. One of the most important collaboration meetings between these is the morning meeting.

The overall goal of the meetings is to make sure that the organization is planning proactively, and is capable of detecting future, possible conflicts on an early stage. A typical agenda include HSE (Health, Safety and Environment), production, new activities, status of ongoing and planned work, and new actions.

The technologies used to keep these meeting are videoconferencing for communication and different software tools which are displayed in a shared surface, e.g. SAP data on maintenance activities.

The software tools used in such collaboration meetings have an important role in focusing and communicating important information to decision makers. Decisions made in morning meetings may impact both production and safety of the installation. Sarshar et al. (2015) have studied investigation reports and iden-

tified hydrocarbon leakage incidents which can be traced to misunderstandings and lack of overview of simultaneous activities when making decisions in the morning meeting.

This motivates the following research question: *Can software tools better support decision making in meetings, and how?*

This paper reports first on a study of existing and research software tools applicable for morning meetings and how well they support the meeting. Based on this study and workshops with experts in collaboration surfaces and environments, requirements for an improved tool were established and a first concept of a new software framework are developed and presented. The currently developed prototype only illustrates a future system's concept for research purposes and is not a fully working system which can be used during operation.

The paper is structured as follows: Section 2 provides context and background. In section 3, the research methodology applied in this study is described. Section 4 presents the main results from the research, covering the study of existing tools, establishing new requirements and developing a new concept framework. Section 6 provides conclusions and further work.

2 CONTEXT AND BACKGROUND

2.1 *Integrated operations*

The oil and gas industry has in recent years started implementing a concept called integrated operations (IO)¹. In theory IO integrates people, organizations, work processes and ICT. However, this process depends on a global access to real time data and collaboration technology. This latter dependency makes it possible for all involved parties to communicate across disciplines, organizations and geographical borders².

According to the Kongsberg Group³ the introduction of IO has several benefits. An article on the company's websites states that implementation of IO leads to reduced operating costs, increased oil and gas recovery, accelerated production, longer life-spans, extended field-life and higher safety levels.

2.2 *Morning Meetings in IO*

Skjerve et al. (2013) have described how meetings are organized in an IO context. The actual meeting, between onshore and offshore personnel, is called a main collaboration session (MCS). However, this session is only a part of a bigger structure - named a distributed collaboration structure (DCS). This structure contains steps before, during and after a meeting. These are activities needed for completing the organizational goal. The steps are repeated after each iteration.

Figure 1 is a simplification of Skjerve et al's own illustration of a DCS. In the figure the initial step includes work to be done before a MCS. More specifically the step contains general meeting preparations and selection of proposed plans, agendas, etc which should be presented in the meeting (the MCS).

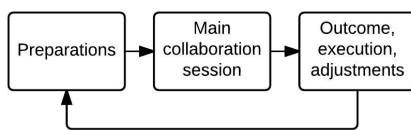


Figure 1: DCS steps before, during and after a MCS. The illustration is a simplification of a figure presented by Skjerve et al.

The collaboration session itself is an activity involving both offshore and onshore personnel, usually in the form of geographically dispersed teams. The personnel participating in such meetings may have a variety of backgrounds.

The final step after the MCS involves approval and adjustment of the meeting's outcome before execution of adopted tasks. This execution may lead to

¹<http://www.norskoljeoggass.no/>

²<http://www.iocenter.no/>

³<http://www.kongsberg.com/>

some sort of goal achievement, which again may be evaluated. Finally the execution of the MCS, or the DCS, may be adjusted on the basis of the evaluation.

Morning meeting sessions may be seen as examples on MCSs. These sessions are described by Ose and Steiro (2013) as meetings which are held every day. These kind of meetings involve onshore and offshore personnel at individual rigs. The onshore personnel supporting the rigs are attending the meetings, and they communicate by phone or video conferencing.

The agenda in such meetings may, according to Wahl et al. (2008), include notifications for the last 24 hours prior to the meeting, and clarification and distribution of actions from the onshore operation center to offshore personnel. The meetings could also concern changes involving planned work. Wahl et al mention a concrete example where a plan had to be updated during the meeting. For accomplishing this task several decision support tools had to be accessed and used. These tools may therefore be important when executing a morning meeting.

2.3 *Decision support tools in IO*

According to Kaarstad and Rindahl (2012) decision support implies supporting people making decisions. Decision systems are components of decision support, which also include decision theories. The systems are software which help their users to solve problems and to accomplish complex decision making.

Sarshar and Rindahl (2014) have observed the use of different types of decision systems in the oil and gas sector over a period of five years, ranging from 2008 to 2013. The purpose of their study was to inter alia find if the software implemented by the industry supported meeting agendas and overall organizational workflow. The researchers found several challenges related to the studied tools.

Although some of the systems were highly advanced, personnel had problems using them. One system had for example implemented a sophisticated 3D technology, rendering stereographic images. This functionality was turned off by some of the users, since they misinterpreted the advanced functionality as noise.

On the other hand a much simpler system was found to work extremely well. The researchers observed the meeting leaders' use of an email as a collaboration platform. Although the email only contained fixed data, in the form of written text and screen dumps of other systems, the meeting leaders managed to turn the simple surface into a highly useful collaboration tool.

Sarshar and Rindahl point out that lessons learned from the study included that functionality, which seems nice to have from the vendors' point of view, may be unnecessary and turn out cumbersome to manage for the users. Simple tools may also be just as

good collaboration tools as advanced and expensive solutions, if the team managing them possess sufficient collaboration skills and technology literacy.

2.4 *On the consequence of tools for IO meetings*

During a study of collaboration technologies (Sarshar & Rindahl 2014) in meetings between offshore and onshore personnel, several effects of implementing tools were observed. These are listed in this section.

In virtual collaboration rooms there usually are monitors to share and work on information, [the shared work surface], in addition to inter alia video conferencing equipment (Hjelle & Monteiro 2011). In the study performed by Sarshar and Rindahl it was found that the main surface in a successful IO meeting is not the video surface, but the shared work surface. Ideally, this tool should not be meeting specific, but reflect the work surfaces mainly used by the participants during their workday. Still, any good work surface is not necessarily fostering collaboration in video meetings. Below follows several effects of utilizing work surfaces from (Sarshar & Rindahl 2014).

Positive effects observed:

1. *Identifying and clarifying misunderstandings:* In an observed meeting series, issues kept resurfacing after they had apparently been resolved. The team then started noting down actions, decisions and key information points on the shared surface in the meeting. Often such items were corrected by the participants. Statements as "No, this concerns the A27, not the A22" or "No, I am not taking responsibility for fixing this, I just promised to talk to a coworker about it" were noted. By verifying a shared understanding then and there, recurring discussions were avoided, and actions could be taken faster and by the right people.
2. *Getting information without cluttering up the meeting:* In some cases, people attended morning meetings to stay informed, and spent much time waiting for the information points they really needed. By logging key information then and there, all necessary information was available for use in action to those attending and those not attending at the same time. Only people who should offer input or counsel on issues discussed attended the meeting. The people only receiving information could save time and room in the video picture and get their information as soon as the meeting ended all the same. [It would, however, require high quality agendas and minutes to facilitate people in identifying whether they should attend or not].
3. *Management showing the way:* In an observed case a manager used a tool in a daily morning meeting. On purpose, he opened the tool there and then and navigated to find the right information. He logged new issues and information, and pointedly hit save when a decision was

made or an issue was agreed by the participants to be correctly described. After a week it was observed that several colleagues start using the same technology in and outside meetings, logging the same type of information.

Negative effects observed:

4. *Surface not suited for video use:* A frequently used type of collaboration surface is spread sheets. These are highly useful tools, and often contain much of the information relevant in meetings. However, only the most skilled virtual collaborators (probably 1 out of 50) are able to use such a surface in a way that makes it easy for collocated and remote participants to partake in the shared information. Another favorite is PowerPoint. An excellent tool if you know the truth and want to share it. But it is not a good tool when what you need is to have your understanding challenged and updated based on input from subject matter experts available in virtual 15-30 minute morning meetings.

In addition to the described effects of collaborative work surfaces, it was observed that time may be saved if:

- Background information is available in digestible form to all parties prior to a meeting - without data mining or long time preparation.
- Facts shared in a meeting are clearly presented both vocally and visually.
- Misunderstandings are discovered and remediated while experts are still present.
- Decisions are both made and understood at once.
- Only contributors need to spend time in meetings, and recipients get their information as quickly as if they were in the meeting.

There were finally observed several issues due to shortcomings in collaboration or information sharing:

- Delay of the restart of a temporarily shut down well can cost millions a day. Start-up may have environmental consequences if not done correctly. Misunderstandings around risk of startup often causes several days delay.
- Planned jobs that cannot be performed after all is also a cost and sometimes risk driver. Misunderstandings can include e.g.:
 - Not seeing the whole picture in terms of simultaneous activities,
 - Double booking of resources (staff, equipment, beds, lifeboat capacity)
 - Wrong parts sent out or sent out at the wrong time

3 METHOD

Requirements for the future system have been gathered by interviewing two researchers at Institute for Energy Technology (IFE) in Halden, Norway. They are currently studying how software tools may be used to support planning processes in the IO-environment. A major oil company's guidelines for establishing collaboration rooms has also been studied to identify system needs.

The Requirement Generation Model (RGM), described by Arthur and Gröner (2004), was selected as methodology for the requirement definition process. RGM is an iterative model where the main stage, *the Requirement Capturing Phase*, is repeated until a generated set of system requirements is satisfactory. This phase contains three sub-phases. In these phases meetings between customers and requirement engineers are prepared, information is conveyed from the customers to the engineers and gathered information is transformed to requirements, and evaluated. When these phases have been completed it is decided if a new iteration is needed.

In this project the experts have acted as the customers in the description of RGM, and the first author took the role of a requirement engineer. But since the experts possessed extensive knowledge about the specific field and system development, they have also acted as advisers through the requirement generation process.

In this respect the experts initialized a MoSCoW prioritization⁴ of the generated requirements. This technique is commonly used for evaluating requirements to find if a future system *must*, *should*, *could* or *won't* have the needs for implementing proposed functionality. If the first category of essential "must"-requirements is not fulfilled, the product development may be viewed as a failure.

3.1 Evaluation of current systems

In parallel with the process of defining requirements, currently available software and prototypes developed for research purposes have been evaluated based on the acquired requirements. Common for all software were that they were designed (or partly designed) for conducting meeting activity. The software tools were selected on the basis of their ability to present a broad range of meeting topics.

A total of six existing tools and prototypes were selected for the evaluation. The studied systems were Epsis Teambox, Protosphere, ARKit, IO-MAP, Wisio and an email platform successfully used for collaboration support in morning meetings.

The whole evaluation was conducted by studying documentation and other kinds of information online. None of the systems were actually tested. There may

therefore be some degree of uncertainty connected to the results.

Short descriptions of the systems:

- *Protosphere* is, according to ProtonMedia (2013), a system making use of avatars during meetings. Participants may control their own avatar when interacting with other participants. Information may be shared with other users. The software is commercially available.
- Sarshar and Rindahl (2014) describe the *email platform* as a collaboration surface built upon the email and SAP technologies. The solution was earlier in use in an oil company.
- *ARKit* is described by Ark Platforms Inc. as a tool developed for several kinds of industries, including the oil and gas industry, and is commercially available. It has not been specialized for the offshore industry. The software presents information on maps and makes it possible for users to communicate and share information.
- *Epsis Teambox*⁵ offers access to other kinds of software during meetings, and to share information outside meetings. It is commercially available for several industries, including the oil and gas industry.
- *IO-MAP* is, according to Rindahl et al. (2013), not commercially available, but instead a prototype of a future system. It has been designed as a collaboration surface presenting meeting specific information on maps. The software has been specialized for the oil and gas industry.
- *Wisio* is, as described by Olsen et al. (2013), designed in the same fashion as IO-MAP, and is quite similar to the above system.

4 RESULTS

4.1 System requirements

Several requirements were defined for the future morning meeting software, but no individual requirements are presented in this article. Instead they have been merged into groups to make it easier for the reader to keep track of them.

The initial set of requirements were developed after conversations with experts at IFE. Additionally a small study was performed of guidelines for design of collaboration rooms. During the requirement generation process this selection was changed in consultation with IFE. The groups are based on the final selection containing about 100 individual requirements, and are displayed in Table 1.

Requirements have been prioritized with the MoSCoW method. But since individual requirements have been merged into groups, and the experts had varying rating of several of them, specific results are

⁴<http://daily-scrum.com/features/moscow-prioritization>

⁵<http://www.epsis.no/epsis-teambox/>

not listed in this paper. However, to summarize the experts unanimously gave top-priority to the following functionality (complete descriptions are presented in Table 1):

- Functionality for contacting users operating on mobile devices (id: 1).
- Possibilities for registering information during meetings (id: 3).
- Functionality for displaying the same platform to all users (id: 9).
- Functionality for producing reports (id: 10).
- Functionality for registering meeting attendance (id: 16).
- An easy to use system (id: 20).
- Understandable automation (id: 21).
- Clear feedback (id: 22).
- Possibilities for setting readability adjustments manually (id: 26).
- The system must take available hardware into account (id: 32).
- The system must be capable of sharing the required meeting information (id: 33).

4.2 Evaluation of current systems and prototypes

Current solutions and prototypes were evaluated by comparing their functionality with the list of requirements generated in the project. The result is presented in Table 1. All requirement groups, described in Table 1, have IDs matching the IDs of the requirements in Table 2. Further Table 2 contains words and symbols in each cell. If "yes" or "no" is written the requirement is either implemented or not by the analyzed system, respectively. The question mark is used when implementation may be discussed, or if the documentation was found to be unclear. The dash symbol is used when a requirement is not relevant for the current system. This could be the case when requirements are not satisfying underlying demands, making them irrelevant for the current solution. This is for example the case for the mail platform and requirement 27: "Pre-selection of options when possible, to make it easier for users to follow ordinary system flow" (Table 2). This requirement is considered not relevant since the surface is not implementing any functionality for selecting or deselecting options in the first place.

In the evaluation two of the systems fulfilled most of the requirements. Both Protosphere and Epsis Teambox implements more requirements than any of the other software tools. However, it is not certain that the system with the highest number of fulfilled requirements is the software best adapted to morning meeting sessions in particular.

For example Protosphere fulfill at least 18 of the 34 requirement groups, but is not designed for being used in collaboration sessions with several participants watching the same monitors. Participants using Protosphere are mainly avatars in the program, which

possibly makes the software better suited for personnel operating individual work stations.

Some of the system requirements could also be conflicting. This could be the case for inter alia Epsis Teambox. The software fulfill requirements for displaying external systems, but is not found to implement a function to give readability feedback. If Epsis was to display such feedback for the users, it could have been to analyze other tools' use of fonts, text sizes, etc. Such readability requirements may therefore not go along with requirements for displaying external software, since the requirements then should be delegated to the external systems instead of the system displaying them.

Further some requirement groups were not found to be implemented by any of the systems:

- None of the systems were found to be capable of sharing the same view of the work surface, as the one displayed in collaboration rooms, with participants operating mobile devices.
- None had implemented agents, or were keeping track of personnel's competence, to use as decision support when operating the work surface.
- Functions for detecting current readability adjustments or readability tests, to find satisfactory adjustments before sessions.

4.3 Concept proposition

To create a new system capable of fulfilling the requirements mentioned in table 1, a digital prototype was designed. The prototype illustrates a future design, but is at this point not a working prototype.

The current platform has therefore not been designed for being accessed by users operating mobile devices and group collaboration participants. This design is only a simple prototype, not taking security, platform independence, network connections, or other, possible necessities into account.

Figure 2 shows a screen shot of the current prototype concept, displaying an external window containing the prototype software IO-MAP. In addition to the requirements the prototype is being developed on the basis of generally accepted design guidelines. Existing systems have also affected the design. Especially Epsis Teambox⁶, a software solution which partly uses windows to display external systems. The prototype solution do not display several windows, but instead a single, large frame where external software systems may be displayed.

Further the email platform has influenced the design. This system contains of a software surface based on a morning meeting's context, producing a status report, an agenda and a minutes document. The future solution is also thought to be designed with the meeting's context in mind. Requirements have been made for the system to implement modes mirroring

⁶<http://www.epsis.no/epsis-teambox/>

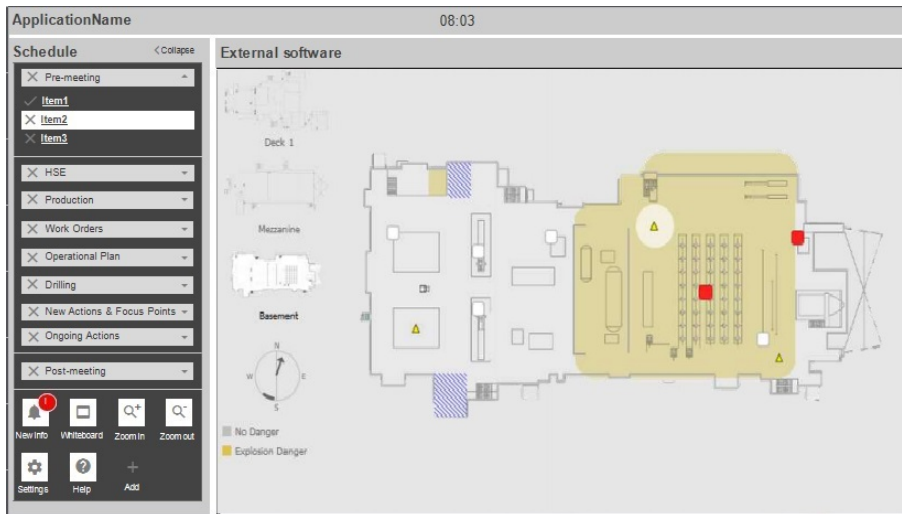


Figure 2: An image of the concept being developed at the time of writing, displaying an excerpt of the external software IO-MAP as presented by Braseth and Sarshar (2012).

the work process before, during and after meetings, and the meeting's agenda.

However, the schedule presented in the email is static headers in a fixed document. The menu presented in the left bar of Figure 2 is based on this agenda, and is a dynamic and accessible menu. It notifies the users if all topics are read or not, and could be manipulated during several steps of the distributed collaboration session - both before, during and after a meeting.

Additionally the menu has a bottom sub-menu with tools to zoom in and out on the content to increase readability, create virtual whiteboards, read notifications among other functions.

4.4 Use of external software in the framework

The framework is meant to make use of existing software to provide the necessary information into the meeting. However, rather than solely be based on e.g. SAP to display maintenance work, solutions like IO-MAP and WISIO are recommended when it is important to establish a picture of simultaneous activities, their locations and local area hazards. The main advantage of such systems is that they position activities on map surfaces instead of presenting the data in a perhaps less intuitive table-form.

5 DISCUSSION

Except the email surface, none of the evaluated systems have been particularly designed for morning meetings in the oil and gas industry. By creating a system in close relation to this context, the software may be well adapted to morning meetings. The system could therefore be easier to understand for new

users familiar with the specific meeting activity - possibly resulting in an increased ability to make decisions.

By designing a future system integrating with other software, and displaying live data, the system could also provide the users with an overall view of the current situation. If so the system could possibly decrease the amount of misunderstandings and the needs for temporarily shut downs, hereby decreasing costs. But this is not unique for the proposed concept. Already available software, as for example Epsis Teambox and Protosphere, may present the users with windows into external tools.

Another advantage of displaying external software is in avoiding the system from being meeting specific. Although it is thought that a future system should be developed for other kinds of meetings as well, the current prototype has only been designed with morning meetings in mind. By connecting the system to external software, possibly well known by the meeting participants, the level of meeting specificity should decrease.

Automatic readability adjustments are difficult to implement when a framework is based on external software which one can not control parts of. As a compromise in the prototype, zoom tools have been added to the menu to make the user capable of zooming on certain parts of the displayed software's view. But how this should work in practice has not been tested thoroughly.

What makes the prototype unique at this point is its close relation to the morning meeting's context, while in the idea still being able of presenting the users with live data from external software.

A future prototype should however add functionality to the existing design, and make the system implement other kinds of functions.

The studied systems were not found to fulfill requirements set for sharing view with mobile devices, implementing agents for decision support, keeping track of the personnel's competence and to perform readability tests.

According to the requirements the future system should implement sharing of the view of the surface with mobile users. Such functionality could promote information, communication and data sharing. As mentioned by Wahl et al. (2008) it is possible that work tasks have to be changed during morning meetings. Implementing possibilities for involving third-party personnel, and presenting them with information about upcoming tasks, could ultimately strengthen the participants capability of making decisions.

None of the systems were nor found to be capable of fulfilling all requirements with top-priority, as proposed by IFE's experts in section 4.1. These critical requirements may be considered mandatory in a future meeting application.

6 CONCLUSION AND FURTHER WORK

In this paper, software systems for morning meetings have been studied and requirements for a new concept framework have been proposed. Parts of these have also been implemented in a prototype.

The main findings are that the currently available systems and prototypes lack several defined requirements, and more critically - requirements defined as mandatory for such a system by experts in the field. The prototype developed in this project should at best implement these, and focusing on fulfilling the most important requirements.+

The prototype is closely linked to morning meeting sessions in the oil and gas industry. It has implemented several stages of the distributed collaboration structure, including the meeting activity. This would probably make the system easy to comprehend for the users. Additionally the prototype has been designed for making use of external software tools.

Future work could include further development of the concept and usability studies with real users from the industry.

REFERENCES

- Ark Platforms Inc. The ARKit Feature List 4.3. <http://www.arkplatforms.com/arkit/feature-list>.
- Arthur, J. D. & M. K. Gröner (2004). An operational model for structuring the requirements generation process. pp. 1–3.
- Braseth, A. & S. Sarshar (2012). Improving Oil & Gas Installation Safety through Visualization of Risk Factors.
- Hjelle, T. & E. Monteiro (2011). Tactics for Producing Actionable Information. In H. Salmela and A. Sell (Eds.), *Nordic Contributions in IS Research*, pp. 72. Springer.
- Kaarstad, M. & G. Rindahl (2012). Shared collaboration surfaces to support adequate team decision processes in an integrated operations setting. In C. Berenguer, A. Grall, and C. Soares (Eds.), *Advances in Safety, Reliability and Risk Management*, pp. 241–242. London: CRC Press.
- Olsen, C. S., O. G. Nedrebø, P. J. Berg, J. M. Røsok, & M. Eskerud (2013). Web based Information Surface for a Petroleum Installation. pp. 2–5.
- Ose, G. & T. Steiro (2013). Introducing IO in a Drilling Company: Towards a Resilient Organization and Informed Decision-making? In T. Rosendahl and V. Heps (Eds.), *Integrated Operations in the Oil and Gas Industry: Sustainability and Capability Development*, pp. 382. Hershey: Business Science Reference.
- ProtonMedia Inc. (2013). Protosphere - a Highly Engaging 3D Virtual Workspace. pp. 1–2.
- Rindahl, G., A. B. Skjerve, S. Sarshar, & A. O. Braseth (2013). Promoting Safer Decisions in Future Collaboration Environments - Mapping of Information and Knowledge onto a Shared Surface to Improve Onshore Planner's Hazard Identification. In E. Albrechtsen and D. Besnard (Eds.), *Oil and Gas, Technology and Humans - Assessing the Human Factors of Technological Change*, pp. 172–173. Surrey: Ashgate Publishing Limited.
- Sarshar, S., S. Haugen, & A. Skjerve (2015). Factors in offshore planning that affect the risk for major accidents. Volume 33, pp. 188–199. *Journal of loss prevention in the process industries*.
- Sarshar, S. & G. Rindahl (2014). Integrated Operation Collaboration Technologies - Remaining Challenges and Opportunities. pp. 3–4. Institute for Energy Technology, Center for Integrated Operations in the Petroleum Industry: Society of Petroleum Engineers.
- Skjerve, A., E. Nystad, G. Rindahl, & S. Sarshar (2013). Assessing the Quality of Collaboration in an Integrated Operations Organization. pp. 1–2. Institute for Energy Technology, Center for Integrated Operations in the Petroleum Industry: European Safety and Reliability Conference, 2013.
- Wahl, A., H. Sleire, T. Brurok, & B. Asbjørnslett (2008). Agility and Resilience in Offshore Operations. pp. 6.

Table 1: System requirements defined after conversations with experts at IFE.

Id	Description
1	Two-way communication with users operating external mobile devices during collaboration sessions.
2	Sharing the same view with mobile devices during collaboration sessions.
3	Possibilities for registering information as for example keywords during meetings.
4	Must take the existing meeting roles into account and designed for both meeting participants, leaders, etc.
5	Multiple modes for varying system flow based on current state in the meeting process.
6	Possibilities for editing and prepare a future meeting's setup.
7	Implementing options for presenting real-time data.
8	Implementing possibilities for presenting fixed data.
9	Implementing an own mode active during meetings, especially designed for this activity.
10	Implementing an own mode active after meetings, able of producing minutes or other kinds of reports.
11	Implementing modes for different devices to make the interface user friendly, regardless of device/monitor.
12	Collaboration participants should see the same platform view regardless of physical location or monitors in use.
13	Sharing of cursor.
14	Agents for decision-support.
15	Page hierarchy with maximum of two levels.
16	Register and time meeting attendance.
17	Functionality for open other systems.
18	Present clear navigation information during interaction with system, and when external systems are displayed.
19	Having an overview of personnel competence, and possibly also use the competence as decision-support.
20	Easy to use system stripped of unnecessary functionality.
21	Understandable and intuitive automation.
22	Clear system feedback, including graphical feedback.
23	Measures for focus as warm and cold colors for inter alia setting priority.
24	Implementing functionality capable of giving readability feedback based on the current adjustments.
25	Readability test.
26	Implementing possibilities for readability adjustments.
27	Pre-selection of options when possible, to make it easier for users to follow ordinary system flow.
28	Support for display reduction.
29	Different complexity levels enabling both expert and novice use.
30	Notifications about important news.
31	Local display settings, as brightness, color and contrast.
32	A system capable of taking available hardware in current meeting rooms into account.
33	A system capable of sharing the required information in morning meetings.
34	Whiteboard functionality.

Table 2: Implementation of requirement categories in analyzed systems.

Requirement id	Protosphere	Mail surface	ARKit	Epsis	IO-MAP	Wisio
1	Yes	No	No	?	No	No
2	No	No	No	No	No	No
3	Yes	Yes	Yes	Yes	Yes	Yes
4	Yes	?	Yes	Yes	Yes	Yes
5	Yes	No	No	Yes	No	No
6	Yes	Yes	No	Yes	No	No
7	Yes	No	Yes	Yes	Yes	Yes
8	Yes	Yes	Yes	Yes	Yes	Yes
9	Yes	No	No	Yes	No	No
10	Yes	No	No	Yes	No	No
11	?	No	?	?	?	No
12	?	?	?	Yes	?	?
13	Yes	No	No	?	No	No
14	No	No	No	No	No	No
15	Yes	Yes	Yes	Yes	Yes	Yes
16	Yes	No	No	No	No	No
17	?	No	No	Yes	No	No
18	Yes	-	Yes	Yes	Yes	Yes
19	No	No	No	No	No	No
20	Yes	Yes	Yes	Yes	Yes	Yes
21	?	No	?	?	?	?
22	Yes	?	Yes	Yes	Yes	Yes
23	-	Yes	Yes	-	Yes	Yes
24	No	No	No	No	No	No
25	No	No	No	No	No	No
26	Yes	No	?	Yes	No	No
27	?	-	?	?	-	-
28	No	No	?	Yes	?	No
29	Yes	No	Yes	Yes	Yes	Yes
30	No	No	Yes	Yes	Yes	Yes
31	?	No	No	No	No	No
32	No	Yes	Yes	Yes	Yes	?
33	?	Yes	?	?	No	No
34	Yes	No	No	No	?	No

Appendix B

Requirements

B.1 Requirements Defined for the Project

Meeting System Requirements, version II

Amund Lågbu

February 4, 2015

1 About

This document contains requirements based on needs promoted at a workshop, and a study of guidelines defined for collaboration and support rooms.

Each of the requirements in this document have been numerated, given a title, a description, a purpose and finally possible connections to parent requirements.

This is the second version of the description of the requirements. The first version had a separate list of requirements based on their origins - if they were promoted in the workshop or based on guidelines. This version, on the other hand, contains a merged list of all requirements.

Secondly the first list of requirements have been analyzed by the project's contracting authority. Some have been removed, others have been added to the new list. Additionally wording has been changed to make the message of each requirement easier to understand, if ambiguities were detected during the first analysis.

A model displaying connections between requirements has been added at the end of the document.

2 Requirements

Requirement 0000

Title: Create two systems for morning meetings.

Description: Two systems should be produced for supporting morning meetings in the oil and gas industry. One system should be used by meeting participants in sessions in collaboration rooms. The other should be used by third-party personnel available for contact via their mobile devices.

Purpose: To support personnel in morning meetings, and to let morning meeting participants be able of contacting third-party personnel.

Dependencies: NULL.

Requirement 0001

Title: System for collaboration rooms.

Description: A software system should be designed for morning meetings held in collaboration rooms. Such meetings takes place in rooms with usually two monitors displaying video conferencing and information. The software system should be displayed on the monitor not used by the video conferencing.

Purpose: To create a new system to support participants in morning meetings.

Dependencies: 0000.

Requirement 0002

Title: Mobile application.

Description: An application should be produced to operate on mobile devices. This application should be able to communicate with personnel participating in traditional morning meeting sessions. The communication should occur between the mobile application and the software system used in morning meetings.

Purpose: To make third-party personnel able of participating in morning meetings via their mobile devices.

Dependencies: 0000

Requirement 1000

Title: Interface to contact personnel not participating in current session.

Description: An interface to contact personnel not participating in a morning meeting session, but who are available via a mobile application, should be created. This interface should be global and accessible from all parts of the software system.

The interface is not supposed to run on the VC application, but to be an option available in the software running on a secondary screen in the meeting session.

Purpose: To involve third-party key-personnel in the decision-making process.

Dependencies: 0001.

Requirement 1001

Title: Select available user of mobile application.

Description: The interface should have functionality to show available personnel running a mobile application. Information such as names, titles and portrait images should be presented in such a view.

Purpose: To make it possible to select personnel who may join a meeting session via their mobile devices.

Dependencies: 1000.

Requirement 1002

Title: Send messages to third-party personnel running a mobile application.

Description: The main meeting software should have implemented a function for sending text messages to personnel running a mobile application, outside the meeting's context.

Purpose: To make it possible to contact personnel with messages, if they are not available at the moment, or if a number of personnel should be contacted at the same time.

Dependencies: 1000.

Requirement 1003

Title: Receive messages from third-party personnel running a mobile application.

Description: The interface should have implemented possibilities for receiving messages from key-personnel. Notifications in the interface could be used for showing the meeting participants that a message has been intercepted.

Purpose: To make it possible for third-party personnel to communicate textually with the software system from their mobile devices, outside a session.

Dependencies: 1000.

Requirement 1004

Title: Establish voice communication with personnel running mobile applications.

Description: The interface should implement possibilities for establishing voice communication with personnel outside a session. Video communication is not considered, since the users outside the session could be holding their devices in their hands, while simultaneously viewing information on their devices. The quality of a video call could therefore be uncontrollable.

Purpose: To make it possible for meeting participants and personnel using a mobile application to communicate by voice in a meeting session.

Dependencies: 1000.

Requirement 1005

Title: Present image, title and name of personnel participating with a mobile-device.

Description: An image of the contacted personnel should preferably be shown in the software system's interface, and not on the VC screen, if contact is established. The name and title of the person should also be visible in the interface as long as the communication lasts.

Purpose: To minimize the notion of distance while the communication with the mobile device user is active, and to make it clear for all meeting participants who they are communicating with.

Showing this information on the VC screen is omitted since may be hard to find a place to put such information on the VC screen without disturbing an ongoing session.

Dependencies: 1000.

Requirement 1006

Title: Present timer displaying duration of communication with mobile application.

Description: The participants in a meeting session should be displayed a timer showing the duration of the communication with a mobile user.

Purpose: The application should strive for time-efficiency. Displaying a time counter, as for example a stop-watch, could make the participants eager to make a call time-efficient.

Dependencies: 1000 & 1313.

Requirement 1007

Title: Clear information about ongoing communication with mobile application user.

Description: Clear information with an ongoing session, involving a mobile device user, should be presented as long as the communication lasts. If a screen has been shared or not must also be included as such information.

Purpose: To prevent communication from mistakenly continue after a session with a mobile device user should have been ended.

Dependencies: 1000.

Requirement 1008

Title: Share surface view with mobile application.

Description: The meeting leader should be able to share the view of the software system with a mobile application user.

Purpose: To make it possible for a mobile user to view the information displayed on a screen in the meeting.

Dependencies: 1000.

Requirement 1009

Title: End sharing of interface during session.

Description: It must be possible for the meeting leader to end the sharing of the user interface during communication with a mobile application user. The sharing must also automatically end if the voice communication with the mobile application is ended.

Purpose: To make it possible for the meeting leader to end sharing of the software system's interface.

Dependencies: 1000.

Requirement 1010

Title: End communication with mobile application.

Description: The meeting leader should have possibilities for ending ongoing communication with a mobile application.

Purpose: To make the leader of a meeting session capable of ending communication with a third-party participant during a meeting session.

Dependencies: 1000.

Requirement 1011

Title: Enable voice communication with third-party personnel.

Description: While the communication with the third-party personnel is active, the participant on the mobile device should be included in the ongoing meeting session. He or she should be able to communicate in line with the others, when it comes to voice communication. All participants in a meeting should be able to speak with the user of the mobile device.

Purpose: To include third-party personnel in the active meeting session.

Dependencies: 1000.

Requirement 1100

Title: Register key-information during a meeting.

Description: Possibilities for registering actions, decisions and key-information points. Examples on such interaction tools may be text boxes for registering information directly into the platform during a meeting session.

Purpose: Easy registering of information in a meeting.

Dependencies: 0001.

Requirement 1101

Title: Restrict long text strings.

Description: Maximum limits should be implemented to prevent users from registering long text strings during meetings. A counter presenting characters left could be implemented for each text area.

Purpose: To force the participants to limit written information.

Dependencies: 1100.

Requirement 1200

Title: Multiple Roles.

Description: It must be taken into account that four different roles could be using the system; the meeting leader, participants, mobile application users and finally non-participating viewers.

Purpose: To support different types of user roles.

Dependencies: 0001.

Requirement 1201

Title: Different views outside meeting mode.

Description: Different views of information may be available for different types of users interacting with the application when meeting mode is not active.

Purpose: To avoid some users from having access to all information.

Dependencies: 1200.

Requirement 1300

Title: Modes.

Description: The system must have several modes.

Purpose: To make the different types of interactions with the system more structural - before after and during a meeting, and with different types of devices.

Dependencies: 0001.

Requirement 1301

Title: Editor mode.

Description: the system should have an editor. The editor should make it possible to change a meeting's templates by selecting what to include in later meeting agendas.

Purpose: To edit meetings' templates.

Dependencies: 1300.

Requirement 1302

Title: Drag and drop interactions to edit.

Description: Drag and drop interactions should mainly be used for editing the meetings' templates in the editor.

Purpose: To make editing as simple as possible.

Dependencies: 1301.

Requirement 1303

Title: Save and load editor meeting templates.

Description: It should be possible to save edited meeting templates and to load previously stored ones.

Purpose: To make it possible to create several meeting templates. These may be filled with information based on type of morning meeting in the later preparation phase.

Dependencies: 1301.

Requirement 1304

Title: Set default meeting template.

Description: One meeting template should at all times be set as the default template, available in the preparation mode. This template could be changed if needed.

Purpose: To set a meeting template as default, making it easy for the user to select a meeting type in preparation mode.

Dependencies: 1301.

Requirement 1305

Title: Preparation mode.

Description: An own mode should be implemented to handle tasks needed for preparing the meetings by the meeting leader (meeting owner).

Purpose: To make it possible to distinguish the preparation of meetings into an own system mode.

Dependencies: 1300.

Requirement 1306

Title: Rapid preparation.

Description: A meeting leader should be able to generate a morning meeting rapidly.

Purpose: To prevent unnecessary time spent during preparations.

Dependencies: 1305.

Requirement 1307

Title: Select template.

Description: It should be possible to select a meeting template produced in the editor. The default one should be selected automatically. But this may be changed to a different one, if needed.

Purpose: To easily create a meeting, based on produced meeting templates.

Dependencies: 1305.

Requirement 1308

Title: Add information to meeting template.

Description: Possibilities for adding/removing information to a template or previously stored meeting.

Purpose: To create a meeting of a template.

Dependencies: 1305.

Requirement 1309

Title: Save/load meeting.

Description: To save or load a meeting.

Purpose: To save a meeting for later use, or load a meeting for reuse/further changes.

Dependencies: 1305.

Requirement 1310

Title: Selection of real-time information display.
Description: Information boxes created in the editor could be set to contain automatically generated information, based on the current box' purpose, and the available system data.
Purpose: To make it possible to display real-time information.
Dependencies: 1305.

Requirement 1311

Title: Selection of manual information display.
Description: Boxes created may contain manually selected information.
Purpose: To avoid real-time display, if a fixed data display is preferable.
Dependencies: 1305.

Requirement 1312

Title: Meeting mode.
Description: An own system mode should be active during meetings.
Purpose: To distinguish the meeting activity into an own system mode.
Dependencies: 1300.

Requirement 1313

Title: Designed for fast walk-through.
Description: In oil and gas organizations morning meetings typically lasts for 15 minutes. The system must support a walk-through designed for the meeting type - avoiding unnecessary time-consuming actions.
Purpose: To design a mode adjusted to the meeting type's typical duration.
Dependencies: 1312.

Requirement 1314

Title: After-meeting mode.
Description: A mode for handling the activities needed after a meeting has been held.
Purpose: To distinguish the activities needed after a meeting into an own system mode.
Dependencies: 1300.

Requirement 1315

Title: Report.

Description: The after-meeting mode should be able to present the user with a report describing changes made and new actions and decisions. Examples may be rejections of notifications and movements of jobs.

Purpose: To summarize changes.

Dependencies: 1314.

Requirement 1316

Title: Monitor modes.

Description: The system should implement modes for different types of monitors/devices.

Purpose: To make the interface of the system user friendly regardless of used device/monitor.

Dependencies: 1300.

Requirement 1317

Title: Manual selection of monitor mode.

Description: It should be possible to manually select the desired monitor mode.

Purpose: To make it possible for the users to override the system's choice of mode.

Dependencies: 1316.

Requirement 1318

Title: Automatic selection of monitor mode.

Description: The selection of monitor mode should at default be done automatically. If a communication between different devices is taken place, the adjustments should be set based on the poorest device.

Purpose: To make it possible for all participants to view the same content, and make the view(s) optimized for the current device(s).

Dependencies: 1316.

Requirement 1319

Title: Same platform view.

Description: The selected monitor mode should make the users able of seeing the same platform view during a meeting.

Purpose: To prevent misunderstandings, etc. due to users not watching the same view of information.

Dependencies: 1316.

Requirement 1320

Title: Optimized for hd resolution.

Description: The system's interface should be optimized for a frame size of 1920 x 1024 pixels, which is the minimum required frame size proposed in the study of the oil company's guidelines.

Purpose: To ensure that the users' experience of the interface is as good as possible. (A zoom (pinch) functionality is proposed for the mobile application to allow full hd viewing in the main system. See requirement 9008 for more information about the mobile application's resolution).

Dependencies: 1316.

Requirement 1400

Title: Pointer control.

Description: The system should have possibilities implemented for sharing of pointer control.

Purpose: To in a greater degree involve participants in meetings, and to make implementation of a future technology easier.

Dependencies: 0001.

Requirement 1401

Title: Apply for pointer control.

Description: The participants should be able of getting control of a global mouse pointer.

Purpose: To make the participants a more active part of the meetings.

Dependencies: 1400.

Requirement 1402

Title: Receive pointer control.

Description: Participants may receive control of a global pointer.

Purpose: To make the participants a more active part of the meetings.

Dependencies: 1400.

Requirement 1403

Title: Retrieve pointer control.

Description: The meeting leader should be able to forcefully retrieve control of the global pointer.

Purpose: To make the leader capable of retrieving pointer control if it has to be retrieved with force (for example when errors occur, participants can't find out how to hand over the pointer, etc).

Dependencies: 1400.

Requirement 1404

Title: Hand over pointer control.

Description: Functionality should be implemented for the leader (and the participants) to hand over pointer control.

Purpose: To make the participants a more active part of meetings.

Dependencies: 1400.

Requirement 1405

Title: Present appliances for pointer control.

Description: The display should have notifications and a reachable list of appliances for pointer control.

Purpose: To make it easier for the meeting leader to hand over pointer control to participants.

Dependencies: 1400.

Requirement 1500

Title: Agents.

Description: It should be implemented automatic agents for decision support.

Purpose: Automatic agents may add a new level to the decision-making process, making it possible to test different options in real-time.

Dependencies: 0001.

Requirement 1501

Title: What-if agents.

Description: A concrete type of agent that should be implemented is the what-if agent. This type of agent may act as a virtual meeting participant.

Purpose: Automatic agents may add a new level to the decision-making process, making it possible to test different options in real-time.

Dependencies: 1500.

Requirement 1502

Title: User friendly expert system interface.

Description: The expert system interface should be intuitive, and easy to use.

Purpose: To make the interface easy to use and to stimulate for more use.

Dependencies: 1500.

Requirement 1503

Title: Reliability of expert system information.

Description: The information output from the agents should be as reliable as possible, and information about the degree of reliability should also be returned.

Purpose: To prevent misinterpretation of information.

Dependencies: 1502.

Requirement 1504

Title: Easy to add information into the expert system.

Description: Information input into the expert system should be easy to perform.

Purpose: Making the agents attractive to use and to prevent misunderstandings.

Dependencies: 1502.

Requirement 1505

Title: Easy to understand information.

Description: The output from the agents should be easy to understand.

Purpose: To prevent misinterpretation of information.

Dependencies: 1502.

Requirement 1600

Title: Page hierarchy in the system with a maximum of two levels.

Description: The system should be organized with a main page, and possibly underlying sub-pages. But too many levels may lead to confusion.

Purpose: To organize the system and make it intuitive to use. It should therefore not consist of too many hidden levels underneath a main page.

Dependencies: 0001.

Requirement 1601

Title: Main page.

Description: The system must have a main page displaying topics and possibly sub-pages.

Purpose: To make a page with global information. From here possible sub-pages may also be reachable.

Dependencies: 1600.

Requirement 1602

Title: Registration of participants.

Description: The main page should contain possibilities for easy registration of participants into a list with checkpoints.

Purpose: To register who attended meetings.

Dependencies: 1601.

Requirement 1603

Title: Time meeting attendance.

Description: The meeting attendance should be timed when selecting participants in the list described in requirement 1602.

Purpose: Registering who were participating, and duration and time of attendance.

Dependencies: 1602.

Requirement 1604

Title: More than one panel.

Description: The system should consist of more than one global panel. It should be easy to navigate between panels.

Purpose: To show information in different views, serving different purposes.

Dependencies: 1600.

Requirement 1605

Title: Sup-pages.

Description: Possibilities for implementing sub-pages must be considered. These kind of pages may contain each of the topics.

Purpose: To make simple overviews of topics, and to organize information.

Dependencies: 1600.

Requirement 1606

Title: Open other systems.

Description: It should be possible to open other systems, and to interact with them. If sub-pages are implemented, these should contain such functionality.

Purpose: To enable interaction with other systems.

Dependencies: 1600, 1605.

Requirement 1607

Title: Show navigation when interacting with third-party software.
Description: A navigation bar must be visible when interacting with other systems. This navigation bar must contain a back button, to return to the main system, and a header describing the currently used software.
Purpose: To prevent confusion due to misunderstandings of navigation.
Dependencies: 1606, 1801.

Requirement 1608

Title: Animation when interacting with other software.
Description: Animation should be implemented to give the users tangible illusions of opening other systems.
Purpose: To prevent user confusion.
Dependencies: 1606.

Requirement 1700

Title: Overview of competence.
Description: The system should have an overview of what competence is needed to perform different tasks.
Purpose: The system should support the participants in controlling if the required competence is available in a meeting.
Dependencies: 0001.

Requirement 1701

Title: Hot permit warnings.
Description: The system should be able to give warnings informing of required competence when a hot permit job is to be approved.
Purpose: To reduce the possibilities of hot permit jobs approved without the required competence available in a meeting.
Dependencies: 1700.

Requirement 1702

Title: Control competence.
Description: The system should have listings containing the main competence of meeting participants. If the required competence to perform a task is not at place in a meeting, the system should give a warning. These operations should be performed by an agent.
Purpose: To ensure that the required competence is at place in a meeting.
Dependencies: 1500, 1700.

Requirement 1800

Title: Easy to use system.

Description: The system should be easy to use and not more difficult to understand than Facebook.

Purpose: Increase usability and the users' confidence in the system.

Dependencies: 0001.

Requirement 1801

Title: Display navigation information.

Description: If the implementation contains several pages, the layout must clearly tell the user about the current navigation. Measures as back-buttons, page titles, etc. must be implemented in such cases.

Purpose: To inform the user about the current navigation.

Dependencies: 1800.

Requirement 1802

Title: Stripped of unnecessary functionality.

Description: Functionality, which is not necessary for the meeting participants to perform their job, should not be implemented. Complex functions should not be part of the ordinary system flow, but hidden in for example a drop-down menu.

Purpose: To make the system intuitive and simple.

Dependencies: 1800.

Requirement 1803

Title: Operate without required learning.

Description: The system must be usable, and users should be able to operate the basics without having read instructions.

Purpose: To ensure system usability, and to make new meeting leaders capable of operating the basic functions of the system on the fly, in case of for example illness.

Dependencies: 1800.

Requirement 1804

Title: A layout with minimal distractions.

Description: The system surface should have a clean look, avoiding unnecessary distractions.

Purpose: To avoid distractions due to a messy layout.

Dependencies: 1800.

Requirement 1805

Title: Understandable automation.
Description: Automation of functions should be implemented intuitively.
Purpose: Automation of some functions should not make the user think that other, manual functions are automated.
Dependencies: 1800.

Requirement 1806

Title: Clear feedback.
Description: Clear feedback should be given on interactions with the system's interface. The feedback should make the viewers aware of options made.
Purpose: To present the viewers with understandable feedback and make them aware of their interactions.
Dependencies: 1800.

Requirement 1807

Title: Graphical feedback.
Description: Graphical feedback should be given any time an important choice is to be made. Such feedback may be given through a blinking paperclip or a dialog box. However, it is important that such measures is used with caution, see requirement 1809.
Purpose: To avoid erroneous choices made.
Dependencies: 1806.

Requirement 1808

Title: Unambiguous naming.
Description: Avoid confusing and hard to understand naming.
Purpose: Prevent misunderstandings.
Dependencies: 1806.

Requirement 1809

Title: Avoid unnecessary disturbing elements.
Description: Unnecessary disturbing elements should be avoided.
Purpose: To avoid unnecessary graphical and other types of noise in the application.
Dependencies: 1806.

Requirement 1810

Title: Avoid sound.

Description: Sound should not be used as a feedback effect.

Purpose: In collaboration rooms there may be noise made by for example telephones, voices, fans, etc. Sound should therefore not be used as feedback, since it may add noise and possibly disturb an ongoing collaboration session.

Dependencies: 1809

Requirement 1811

Title: Measures for focus.

Description: Measures, such as warm and cold colors, should be used for focusing on information parts. This should be tested thoroughly in advance of implementation.

Purpose: To make it easy to highlight important information, or tell the user about navigation.

Dependencies: 1806.

Requirement 1812

Title: Specify priority.

Description: It should be implemented functionality for setting priorities in the tool before and during a meeting. This may include marking a text in a specific color.

Purpose: To highlight some information, and to disregard other.

Dependencies: 1806, 1811.

Requirement 1813

Title: Intuitive interface.

Description: Information should be found where it could be expected to be found.

Purpose: Make the system intuitive.

Dependencies: 1800, 1806.

Requirement 1814

Title: Graphical requirements for readability.

Description: Universal usability should be pursued when presenting text.

Purpose: Prevent text from not being read due to low degree of readability.

Dependencies: 1800.

Requirement 1815

Title: Readability feedback.

Description: The system should give feedback if the readability is found to be low. An agent may be implemented for controlling readability. If the display is showing text in a way that triggers the agent's algorithm, the users should be presented with a warning.

Purpose: To contribute to good readability.

Dependencies: 1500, 1814.

Requirement 1816

Title: Implementation of a readability test.

Description: The editor must support implementation of a potential readability check for meetings.

Purpose: To implement a check which may verify that all participants, including visitors, can read displayed information. Visitors may be seated randomly, not following room design plans, making it hard to automatically decide text sizes.

Dependencies: 1301, 1814.

Requirement 1817

Title: Automatic readability adjustments.

Description: Automatic choices for readability should be implemented. The system should access information about the meeting rooms, and display readable text for potential viewers in the farthest seats. The system should take the room with the poorest readability into account, when deciding upon displayed text sizes. The text sizes should be found with an algorithm calculating displayed text size with screen size and distance to viewers.

Purpose: To automatically make sure that viewers may read the displayed text.

Dependencies: 1814.

Requirement 1818

Title: Manual readability adjustments.

Description: The system must support possibilities for manual adjustment of text sizes.

Purpose: To adjust text to support the individual viewer's needs.

Dependencies: 1814.

Requirement 1819

Title: Automatic option selection.

Description: When possible options should be selected automatically in advance of the user's final choice. Such selections may apply to radio buttons, check boxes, etc.

Purpose: To make inexperienced users able of using the system without much training.

Dependencies: 1800.

Requirement 1820

Title: Positioning of important information in the system's interface.

Description: The important information should not be positioned in the view in a way that makes it hard to view by the meeting's participants. According to the guidelines of the studied oil company, all important information should be placed within a 70 degree lateral viewing angle, 40 degrees below and 20 degrees above the line of vision in relation to the front of the video transfer, information and visualization surfaces.

Purpose: To prevent that information gets difficult to be seen, because of the meeting room's design.

Dependencies: 1800.

Requirement 1821

Title: Support for display reduction.

Description: The system must implement possibilities for decreasing either width or especially height of the displayed visualizations - and still strive for working optimally.

Purpose: Height from for example floor to ceiling may vary in different meeting rooms. This may affect the participants capability to view the screens. In some occasions the displayed interface may have to be cropped to ensure that the participants may view the information. In such cases the system should still strive for working optimally.

Dependencies: 1820.

Requirement 1900

Title: Levels of complexity and details.

Description: The system should have different levels of complexity, enabling both expert and beginner use. As an example a technical meeting leader may show more detailed information than a beginner, if demanded.

Purpose: To adopt the system to the user's needs.

Dependencies: 0001, 1800.

Requirement 2000

Title: Ensure that communication works.

Description: It should be tested that the system is able to communicate properly with mobile devices and other systems.

Purpose: To prevent loss of functionality due to communication problems.

Dependencies: 0001.

Requirement 2100

Title: Operational before deadline.

Description: The system should be implemented and ready to use before the specified deadline, if it should be implemented in a company in the future.

Purpose: If a system is not operational in time, personnel tend to avoid using it.

Dependencies: 0001.

Requirement 2200

Title: Notifications about news.

Description: If important information has been generated after the meeting leader has prepared a meeting, and possibly also during a meeting, the system should notify about the new information - if relevant. A subscription arrangement may be set in preparation mode to ensure that only relevant notifications are shown.

Purpose: To ensure that important information is displayed.

Dependencies: 0001.

Requirement 2300

Title: Display settings.

Description: The system should implement settings for colors, brightness, white balance, etc.

Purpose: To make customization of the display available, if needed.

Dependencies: 0001.

Requirement 2301

Title: Pre-designed display settings.

Description: Users should be able of adjusting display settings by selecting pre-designed display options. Such options may be adjusted for meeting rooms with natural light from windows or for example rooms illuminated by light bulbs, fluorescent lamps, etc.

Purpose: To make the users able of easily adjust display settings to the meeting room's lighting conditions.

Dependencies: 2300.

Requirement 2302

Title: Advanced, manual display settings.

Description: There should be integrated advanced options for changing display settings. This option should also implement functionality for resetting the adjustments to default.

Purpose: How a meeting room is illuminated may affect the information displayed on screen. Possibilities for changing color templates, contrast, etc. may be implemented to ensure that participants may easily view the information on screen.

Dependencies: 2300.

Requirement 2400

Title: The system must take available hardware into account.

Description: Collaboration rooms contain audiovisual collaboration equipment. The participants may view one, common set of dual viewing surfaces. The system must take this context into account, by for example not make the system dependent on more screens than the one not used for video conferencing.

Purpose: To avoid creating a system not usable with today's groupware implementations in the oil and gas industry.

Dependencies: 0001.

Requirement 2500

Title: Requirements for information sharing in morning meetings.

Description: The following information has to be shared in morning meetings to make participants able to solve tasks and decisions: Operation status, action log, risk management and documents such as SAP-documents. The meeting systems must at a minimum implement functionality to fulfill these goals.

Purpose: To ensure that the required information is shared in morning meetings.

Dependencies: 0001.

Requirement 2600

Title: Whiteboard functionality.

Description: In some meeting rooms, whiteboards are used for displaying information. The system may implement a whiteboard functionality which has possibilities to store the drawn screens.

Purpose: To create a substitute functionality for whiteboards used in some meeting rooms. This functionality may also store the written information, and ensure that this is not lost after meetings have finished.

Dependencies: 0001.

Requirement 2601

Title: Easy to access whiteboard.

Description: The whiteboard should be a globally accessible function. Animation could be used to open and close the whiteboard from anywhere in the application.

Purpose: To make whiteboard as accessible as actual whiteboards in meeting rooms. A global, virtual whiteboard may then hopefully be preferred over the actual one.

Dependencies: 2600.

Requirement 2602

Title: Whiteboard tabs, or similar functions.

Description: Whiteboard tabs could be implemented to make the user able of creating new whiteboards and navigate between existing ones. All whiteboards should be accessible within the tool.

Another alternative is to make the whiteboard increasingly larger when the user maneuvers inside it. A mini map may be used to tell the user where information already has been added.

Purpose: To make sure that the users always have available whiteboard space.

Dependencies: 2600.

Requirement 9000

Title: Receive messages from meetings.

Description: The mobile application should be able to receive textual messages from meeting sessions, and to be notified of received messages.

Purpose: To make it possible to communicate with participants in meetings without having to make voice calls.

Dependencies: 0002.

Requirement 9001

Title: Participate in meeting sessions.

Description: Mobile application users should be able to join meeting sessions as guests. When they are participating they may communicate with voice to all reachable participants in the meeting.

Purpose: To make the application users able of communicating with participants in meeting sessions by voice.

Dependencies: 0002.

Requirement 9002

Title: Respond to meeting requests with messages.
Description: The mobile users must be able to respond to meeting requests textually.
Purpose: Answering meeting requests.
Dependencies: 0002.

Requirement 9003

Title: End meeting participation.
Description: The mobile application users should be able to end participation with a meeting, if voice communication has been enabled.
Purpose: To make the mobile users able to end their involvement in a meeting.
Dependencies: 0002.

Requirement 9004

Title: Mute ongoing voice communication.
Description: If the mobile user is in a noisy environment, functionality should be implemented to mute his/her mobile microphone.
Purpose: To avoid unnecessary noise disturbances for all meeting participants, although a hands free set may be preferable means for communication.
Dependencies: 0002.

Requirement 9005

Title: Receive shared version of screen.
Description: The mobile app should implement functionality for receiving a shared version of a meeting's displayed view (of the software, not the video conferencing display).
Purpose: To make the user see what the other meeting participants see.
Dependencies: 0002.

Requirement 9006

Title: End sharing session.
Description: The user must be able to end the sharing session, and hereby closing the view of the meeting's display.
Purpose: To make the user able to end a sharing session.
Dependencies: 0002.

Requirement 9007

Title: Follow established guidelines.

Description: The mobile application must follow the design guidelines established for the current platform.

Purpose: To create an application as recommended for the current platform, hopefully making it more intuitive to understand for the mobile platform users.

Dependencies: 0002.

Requirement 9008

Title: Zoom shared view.

Description: The mobile application users must be able to zoom the received shared display used in the meeting session. Pinching movements on the device is one solution, must this must be platform independent.

Purpose: To make it possible to zoom in on different parts of the screen used in the meeting sessions, since the monitors in the morning meetings may support higher resolution than the mobile devices do.

Dependencies: 9007.

Requirement 9009

Title: Sign in and sign out

Description: The user must enter credentials when using the application.

Purpose: To prevent other users from participating in communication with meeting sessions.

Dependencies: 0002.

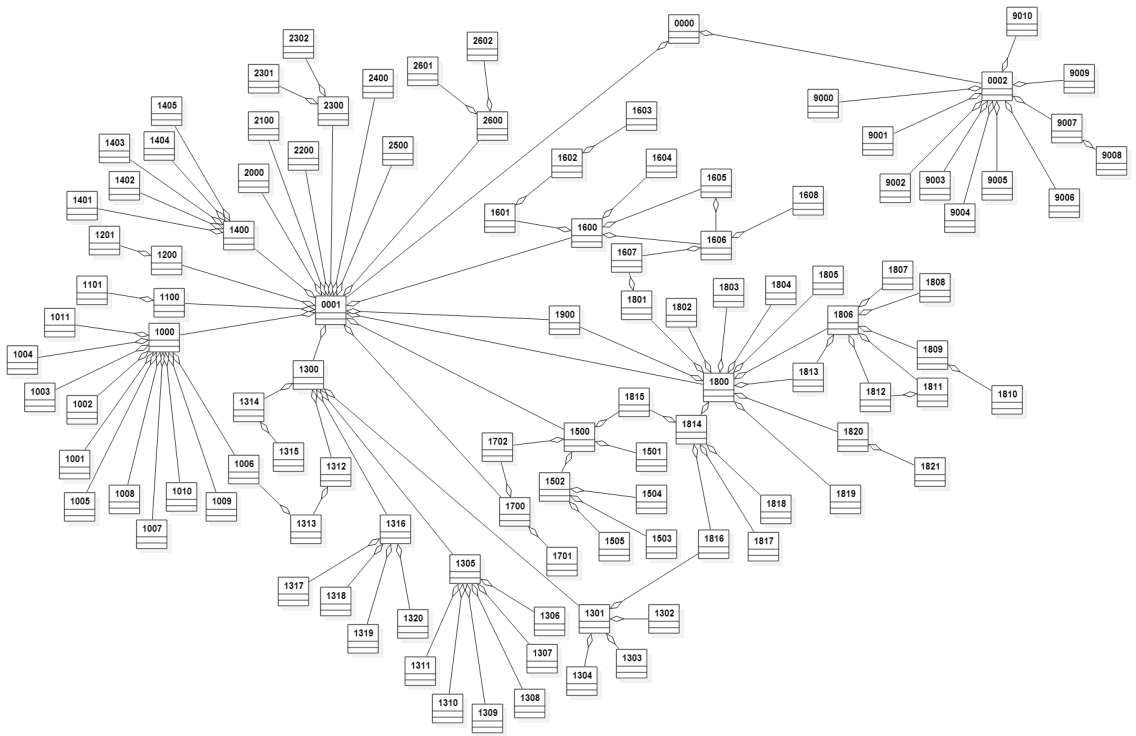
Requirement 9010

Title: Incoming meeting alarm.

Description: The users must be notified of upcoming meetings, when the meeting software tries to communicate with the mobile application.

Purpose: To make the users aware of incoming meeting calls.

Dependencies: 0002.



B.2 Requirements Based on a Company's Guidelines

Requirements Based on Guidelines Defined by Oil Company

Amund Lågbu

January 16, 2015

1 About

This paper contains requirements to the meeting system. These have been obtained from an oil company's internal guidelines for establishing support and collaboration rooms. All requirements will later be included in a joint document. However, the requirements promoted in this document lack references due to confidentiality.

The id of the requirements start on 4000, which is in compliance with requirements promoted earlier in another document. This will make it easy to create a new document containing all requirements when the ones in this document have been approved.

After discussions with the project's supervisor rows containing information about implementation necessity have been added to each requirement, as mentioned in the other document. The first row is called "student's proposal", the second "experts' proposal". The purpose of these rows are to map the usefulness of the implementation of each requirement.

Four ratings have been defined for the purpose; "Mandatory", "should implement", "nice to have" and "unclear". "Mandatory" requirements must be implemented in the new system. Those that should be implemented are not mandatory, but should be taken into account, and are to be rated as "should implement". "Nice to have" requirements are not needed in the new system, but are nice to have. "Unclear" requirements are difficult to rate because of unclear information given, etc.

Lastly a "comments" field has not been added. Experts may instead write comments in the margin to the left or the right of each requirement.

2 Requirements

Requirement 4000

Title: The system must take the available hardware in the meeting room into account.

Description: Collaboration rooms contain audiovisual collaboration equipment. The participants may view one, common set of dual viewing surfaces. The system must take this context into account.

Purpose: To avoid creating a system not usable with today's groupware implementations in the oil and gas industry.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
X			

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>

Requirement 4100

Title: Requirements for information sharing in morning meetings.

Description: The following information has to be shared in morning meetings to make the participants able of solving tasks and make decisions: Operation status, action log, risk management and documents such as SAP documents.

Purpose: To ensure that the required information is shared in morning meetings.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X (user-decided)		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>

Requirement 4101

Title: Collaboration purpose of morning meetings.
Description: The system must fulfill the purpose of morning meetings. In morning meetings all participants have to be updated on the status of operations, risks and ongoing actions. They must also be capable of establishing new actions.
Purpose: To fulfill the goals of morning meetings.
Dependencies: 4100.
Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
X (user-decided)			

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4200

Title: System designed for fast walk-through.
Description: In oil and gas organizations morning meetings typically last for about 15 minutes. The system must support a walk-through designed for the meeting type - avoiding unnecessary time-consuming actions.
Purpose: To design a system adjusted to the meeting type's typical duration.
Dependencies: NULL.
Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
X			

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4300

Title: Readability according to the Human-System Interface Design Review Guidelines(NUREG 0700, rev. 2).
Description: The Human-System Interface Design Review Guidelines should be followed to ensure that the system provides readability for all viewers in a meeting.
Purpose: To display readable information.
Dependencies: NULL.
Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
X			

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4301

Title: Automatic readability adjustments.

Description: Automatic choices should be made according to the Human-System Interface Design Review Guidelines(NUREG 0700, rev. 2). The system should access information about the meeting rooms, and display readable text for potential viewers in the farthest seats. The system should take the room with the poorest potential viewers into account.

Purpose: Automatic choices to ensure that viewers may read displayed information.

Dependencies: 4300.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4302

Title: Implementation of readability test.

Description: The editor must support implementation of a potential readability check for meetings.

Purpose: To implement a check which may verify that all participants, including visitors, can read displayed information. Visitors may be seated randomly, not following room design plans, making it hard to automatically decide on text sizes.

Dependencies: 4300.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4303

Title: Manual readability adjustments.

Description: The system must support possibilities for manual adjustment of text sizes.

Purpose: To adjust text to support the individual viewers' needs.

Dependencies: 4300.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4400

Title: Positioning of important information in the system's interface.

Description: The important information should not be positioned in the view in a way that makes it hard to view by the meeting's participants. According to the guidelines of the studied oil company, all important information should be placed within a 70 degree lateral viewing angle, 40 degrees below and 20 degrees above the line of vision in relation to the front of the video transfer, information and visualization surfaces.

Purpose:

To prevent that information gets difficult to be seen, because of the meeting room's design.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4401

Title: Support for display reduction.

Description: The system must implement possibilities for decreasing either width or especially height of the displayed visualizations - and still strive for working optimally.

Purpose: Height from for example floor to ceiling may vary in different meeting rooms. This may affect the participants capability to view the screens. In some occasions the displayed interface may have to be cropped to ensure that the participants may view the information. In such cases the system should still strive for working optimally.

Dependencies: 4400.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4500

Title: Optimized for 1920 x 1024 resolution.

Description: The system's interface should be optimized for a frame size of 1920 x 1024 pixels, which is the minimum required frame size proposed in the studied oil company's guidelines.

Purpose: To ensure that the users' experience of the interface is as good as possible.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
		X	

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4600

Title: Whiteboard functionality

Description: In some meeting rooms, whiteboards are used for displaying information. The system may implement a whiteboard functionality which has possibilities to store the drawn screens.

Purpose: To create a substitute functionality for whiteboards used in some meeting rooms. This functionality may also store the written information, and ensure that this is not lost after meetings have finished.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
		X	

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>

Requirement 4700

Title: Feedback on interactions.

Description: Clear feedback should be given on interactions with the system's interface, to ensure that the viewers are aware of options made.

Purpose: To make the viewers aware of system interactions.

Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
X			

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>

Requirement 4701

Title: Prevent sound feedback.

Description: Sound should not be used as an effect to display feedback.

Purpose: In collaboration rooms there may be noise made by for example telephones, voices, fans, etc. Sound should not be used as feedback for interactions, since it may add noise and possibly disturb an ongoing collaboration session.

Dependencies: 4700.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>

Requirement 4800

Title: Options to change contrast/color of display.
Description: There should be integrated options for quickly changing the colors or contrast of the display.
Purpose: How a meeting room is illuminated may affect the information displayed on screen. Possibilities for changing color templates, contrast, etc. may be implemented to ensure that the participants may easily view information on screen.
Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Requirement 4900

Title: Usability of the system.
Description: The system must be usable, and users should be able to operate the basics without having read instructions.
Purpose: To ensure system usability, and to make new meeting leaders capable of operating the basic functions of the system on the fly, in case of for example illness, etc.
Dependencies: NULL.

Student's proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
	X		

Experts' proposal:

<i>Mandatory</i>	<i>Should implement</i>	<i>Nice to have</i>	<i>Unclear</i>
------------------	-------------------------	---------------------	----------------

Appendix C

Design

Design - Iteration VIII

Amund Lågbu

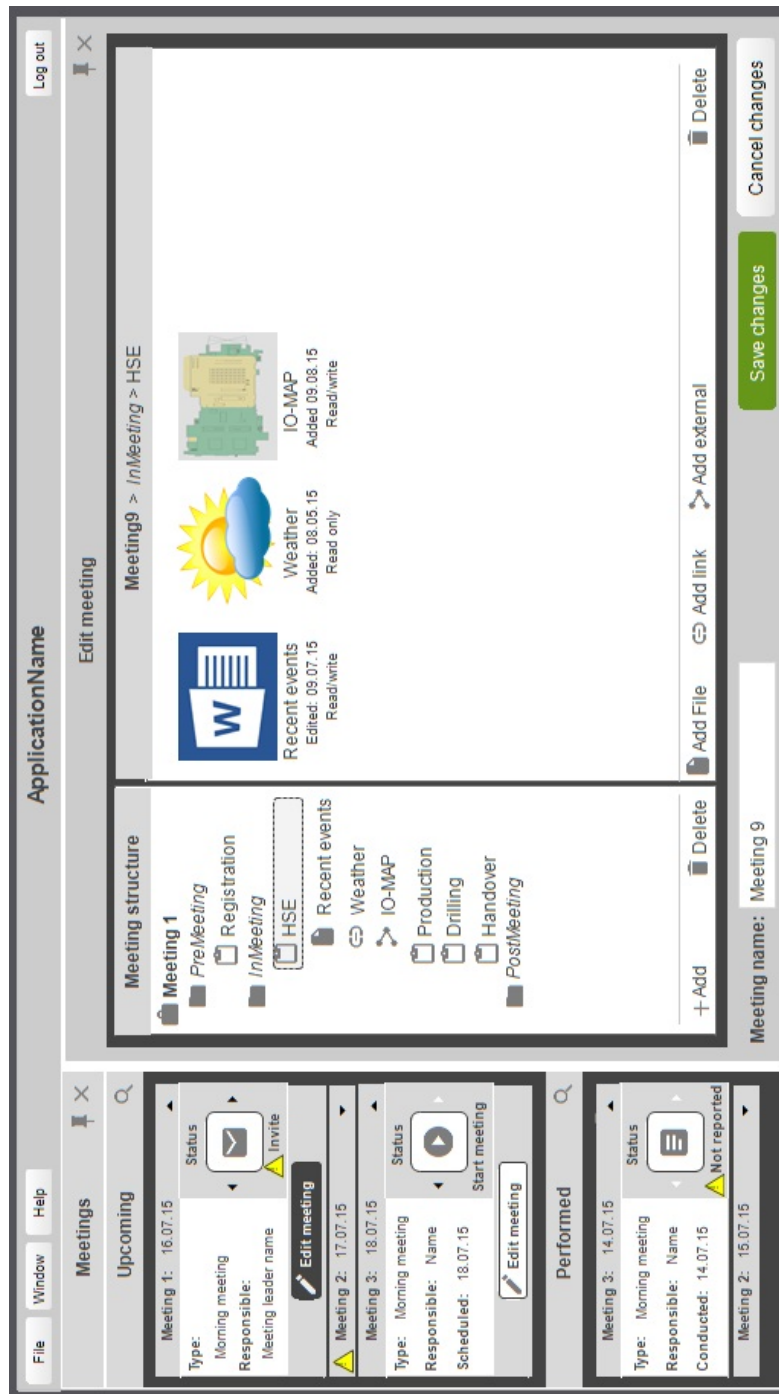
This document contains six slides of the future meeting system drawn during the eighth design iteration. Two of them are redrawn versions of the slides made in iteration VII presenting the meeting's sub system for handling several meetings. These are presented first. The last, four slides describe changes made to the part of the system active in the actual morning meeting sessions. In addition to overall design changes, these slides have also been made to support visualization guidelines.

The other slides, presented in the concept test, have not been changed. Instead they are thought to implement the same, overall changes as the ones presented in this document. It was not considered necessary to change all slides, if the current one could describe the overall and desired design.

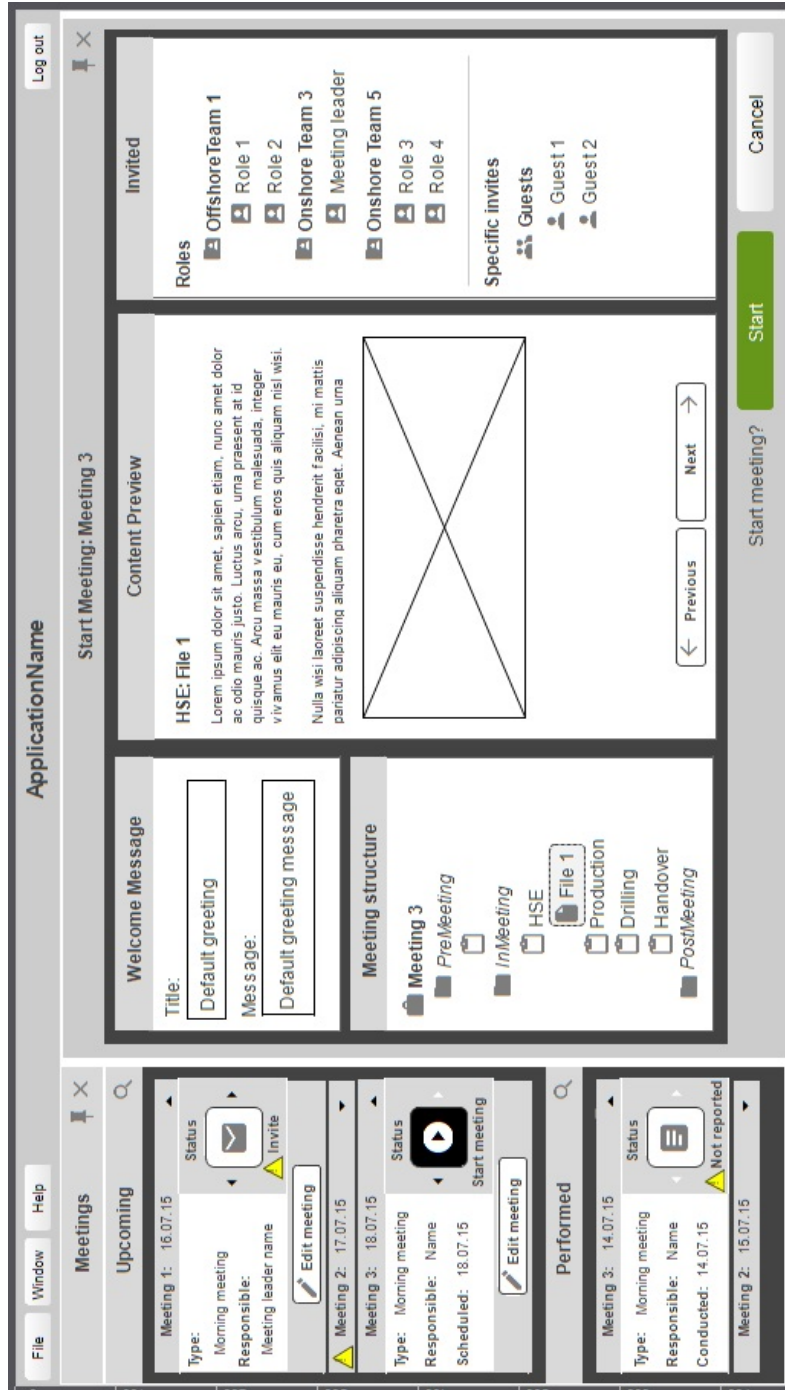
Below follows short descriptions of each of the slides:

- **1:** A design presenting how files, links and external software could be added and listed.
- **2:** This slide shows a new window visible when the user has clicked the start-button. Meeting related information is presented in this window.
- **3:** The slide presents a welcome message, in addition to possibilities for registering the attendances and important focus points.
- **4:** Here an activity is added while an external software is presented.
- **5:** A report is presented with actions, decisions, comments and screen captures, which may be edited.
- **6:** The slide shows how a meeting may be sent to both roles and individual personnel.

1 Meeting Handler: Adding Files to a Meeting



2 Meeting Handler: Initiating a Morning Meeting



3 Meeting Session: Welcome Screen

Morning meeting 21.08.15
08:03

Introduction > Welcome
🔍

Welcome to morning meeting at XXX

Meeting leader: David Davidsen

Agenda

- Introduction
- 📄 Welcome
- 📄 Actions, 20.08.15
- 1. HSE
- 2. Production
- 3. Work Orders
- 4. Operational plan review
- 5. Drilling
- 6. Actions from hand-over
- 7. Focus points
- 8 Ongoing actions
- 9. Report

Participants

Offs hore team 1	Onshore team 1	Onshore team 2	Guests
<input checked="" type="checkbox"/> Anders Andersen	<input checked="" type="checkbox"/> David Davidsen	<input checked="" type="checkbox"/> Gunn Gundersen	<input checked="" type="checkbox"/> Jorunn Johnsen
<input checked="" type="checkbox"/> Bjern Berg	<input checked="" type="checkbox"/> Erik Eriksen	<input checked="" type="checkbox"/> Hanne Hansen	<input checked="" type="checkbox"/> Kim Kristiansen
<input checked="" type="checkbox"/> Cato Chrostoffersen	<input checked="" type="checkbox"/> Fred Fredriksen	<input checked="" type="checkbox"/> Ida Iversen	

Add another

Today's focus points

Lorem ipsum dolor sit amet, sapient etiam,
nunc amet dolor ac odio mauris justo. Luctus
arcu, urna praesent at id quis que ac. Arcu

📄 Action

🗨️ Decision

💬 Comment

📷 Capture

4 Meeting Session: Adding Activity During a Meeting

Morning meeting 21.08.15 08:03

Drilling > Overview

Agenda

- Introduction
- 1. HSE
- 2. Production
- 3. Work Orders
- 4. Operational plan review
- 5. Drilling
- Overview
- C5: Comments to drilling
- D5: Decisions made
- 6. Actions from hand-over
- 7. Focus points
- 8 Ongoing actions
- 9. Report

New Action

Title: Default title

Description:

Responsible: Erik Eriksen

Attachments: file.txt

Updating search

Add responsible

Add file

Create action

www.dgi.com

5 Meeting Session: Displaying the Report

Morning meeting 21.08.15

08:03

Agenda

Report > Generated & editable report

- Introduction ▾
- 1. HSE ▾
- 2. Production ▾
- 3. Work Orders ▾
- 4. Operational plan review ▾
- 5. Drilling ▾
- 6. Actions from hand-over ▾
- 7. Focus points ▾
- 8. Ongoing actions ▾
- 9. Report ▾

Generated & editable report

Date: 21.08.15
Place: Place
Resp: D. Davidlsen
Doc: X12715

Meeting Report

1. HSE

Actions

Title: A1: Title of the action

Description:
Lorem ipsum dolor sit amet, sapien etiam, nunc amet dolor ac odio mauris justo. Luctus arcu, urna praes

Responsible: Erik Eriksen

Attachments: file.doc

Edit action

Title: A2: Title of the action

Description:
Lorem ipsum dolor m ipsum dolor sit amet, sapien etiam, nunc amet dolor ac odio mauris justo. Luctus arcu, urna praes

Responsible: Erik Eriksen

Attachments: file.doc

Edit action

Send report

Save report

End meeting

6 Meeting Session: Sending the Report

Morning meeting 21.08.15
08:03

Report > Generated & editable report
🔍

Agenda

- Introduction ▾
- 1. HSE ▾
- 2. Production ▾
- 3. Work Orders ▾
- 4. Operational plan review ▾
- 5. Drilling ▾
- 6. Actions from hand-over ▾
- 7. Focus points ▾
- 8. Ongoing actions ▾
- 9. Report ▾
- Generated & editable report

Meeting Report

1. HSE

Actions

Title: A1: Title of the action

Description:
Lorem ipsum dolor sit amet, sapien etiam, nunc amet dolor ac odio mauris justo. Luctus arcu, urna praes

Attachments: file.doc

Responsible: Erik Eriksen

Edit action

Title: A2: Title of the action

Attachments: file.doc

Responsible: Erik Eriksen

Edit action

Date: 21.08.15
Place: Place
Resp: D. Davidsen
Doc: X12F1s

Select report receivers

Select roles:

Preset list ▾

Add single

Select specific personnel:

Preset list ▾

Add single

Role 1

Role 2

A.guest

Send report

Appendix D

Test of Developed Concept

ØSTFOLD UNIVERSITY COLLEGE
FACULTY OF COMPUTER SCIENCES
HALDEN, NORWAY
MASTER IN APPLIED COMPUTER SCIENCE

CONCEPT TEST

By Amund Lågbu

September, 2015
Test of Developed Concept



1 Background

This document describes a test of a developed concept during the work with a master thesis in applied computer science. The concept has been developed as a mean for answering the thesis' research questions, which are:

1. *Can software tools better support decisions in morning meetings in oil and gas organizations, and how?*
2. *How may software, supporting morning meetings, promote important elements of the Integrated Operations concept?*

The developed concept is a non-interactive design of a future system. It has been developed as a shared software for interactive morning meetings between personnel located both onshore and offshore. The development and the study of support for other kinds of meetings was omitted with regard to the magnitude of such a task.

Further the software is a supplementary platform, and has not been designed to replace existing video conferencing. It is to be shown on a second monitor, alongside ongoing video streams.

The concept is founded on several system requirements formed in collaboration with personnel at IFE, a study of guidelines for establishing collaboration rooms and an evaluation of current systems and prototypes. In addition, more general design principles, as "the Eight Golden Rules of Interface Design", have been used during this process.

Major sources of inspiration were both the existing system Epsis Teambox, and a mail platform previously used in morning meetings in an oil company. Other, important inspiration sources were the prototype system IO-MAP (and WISIO), ArkIT and Protosphere.

The prototyping process followed a custom method. This method included initial stages for establishing requirements for a morning meeting system and an evaluation of how well current systems and prototypes met these requirements. After these stages the concept was developed in iterative cycles, resulting in producing both a horizontal and a functional prototype. However, the the functional prototype is not to be evaluated, since it does not differ much from the overall design. It was meant as a mean for testing design selections. Further design of the overall prototype followed after the functional prototyping.

2 Conduction of the Test

The following test will be performed in three stages. Firstly the produced prototype will be examined during cognitive walkthroughs of central parts. Secondly a heuristic evaluation of the design will be conducted in respect to guidelines and some system requirements. Then a usability inspection is to be performed, with regard to the usefulness of the system in current IO oil and gas organizations. This part of the test will not be performed individually, but with a custom method - a plenum session.

The backdrop for selection of test methods is as follows:

1. **Cognitive Walkthrough:** The cognitive walkthrough is performed first to make the participants aware of system functionality and the design, while still testing the design. The test aims for finding flaws in the design, and if the current implementations work well.
2. **Heuristic Evaluation:** By performing the cognitive walkthrough first, the participants should perhaps be familiar with the interface, making them better prepared for rating the system in this part.
3. **”Usability Inspection”:** The last part of the test is conducted not only test the current implementation based on defined limits as specific guidelines and requirements. It aims for analyzing if the current implementation is fitted for the IO oil and gas industry.

Consent Form

I have been informed about the nature of this study, and the test that I am about to participate in. I understand that participation in the test is fully voluntary, and that I am free to withdraw from the research at any time, or to refuse to take part in the study.

I have been informed that the gathered material will be handled anonymously, and that personal information (as my name or age), will not be published. However, I am also been notified that information regarding the participants as a group may be published. This information will include the time and place of the test, and an overall group description of the participants' background and work area.

If I have any enquiries regarding the research I may contact student at Østfold Univeristy College, Amund Lågbu, by email or phone. Contact information is as follows:

- Email: amundlag@hiof.no
- Phone: 922 23 903

By signing below I allow the gathered information during the test to be published as quotations, to be studied and compared, and reproduced in statistics.

Signed

Date and location

Name (please print)

3 Cognitive Walkthrough

This test will include different tasks to be performed by thought interaction with the system interface, which includes two, major parts: The main surface where meetings are created, updated and sessions are started. The second part is the actual meeting software, used during collaboration sessions. Each activity has a described starting point, a specific site. Navigation between sites will be simulated by navigating between prototype images.

Some functionality has not been designed, but is thought to be included. Examples include dialogues when a user wants to cancel or overwrite. Other examples are the help menu and the menu for rearranging the windows in the outer part of the application design. Additionally keyboard shortcuts are thought to be implemented in the outer part of the application, but not during meetings. Error handling should preferably be done automatically, but understandable messages should be shown for the user if error occurs. All errors should be logged.

However, the following tasks should be performed in this part of the test:

3.1 Pre-meeting tasks

1. Create a new meeting. Navigate to 3.4.
2. Invite personnel to the new meeting ("Meeting 3"). Navigate to 3.4.
3. Create a new meeting structure for future morning meetings, name it "morning meeting structure". Navigate to 3.4.
4. Include a new file in "Meeting 1". Navigate to 3.4.
5. Start "Meeting 1". Navigate to 3.4.

3.2 In-meeting tasks

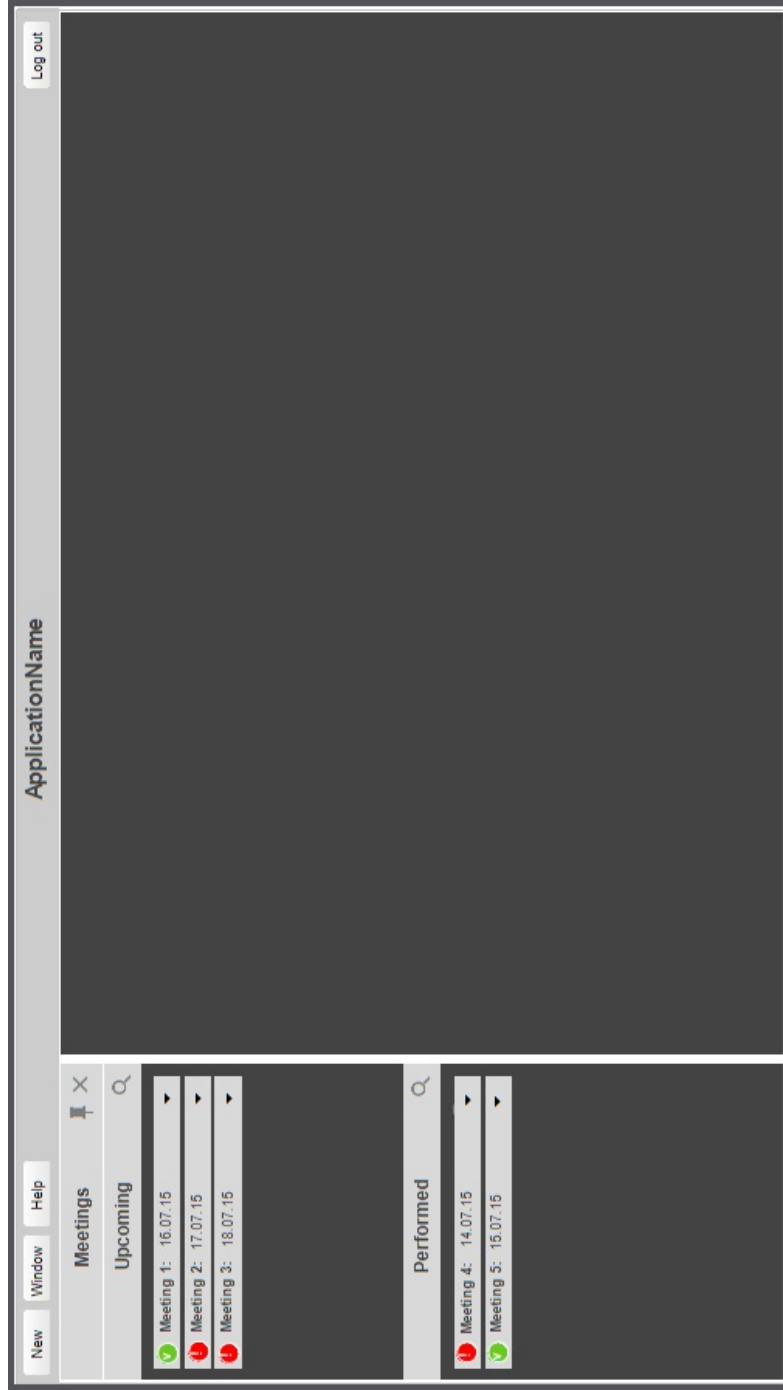
1. Perform the pre-meeting tasks. Navigate to 3.14.
2. Now, select the participants from the global tools menu. Navigate to 3.14.
3. If there are pending notifications, select the notification tool. Navigate to 3.14.
4. Access the settings and increase the size of the mouse pointer, to make it easier for the users to view selections made than with the default cursor. Navigate to 3.14.

5. Watch a map surface of the installation presented as HSE information. Navigate to 3.14.
6. Now, add a whiteboard describing the map. Navigate to 3.3.
7. Now add a comment to the selected map. Navigate to 3.3.
8. Then consider the alternative comment field. Navigate to 3.24.
9. Now select the production overview from the menu. Navigate to 3.14.
10. Then select the external drilling software by accessing the drilling topic. Navigate to 3.14.

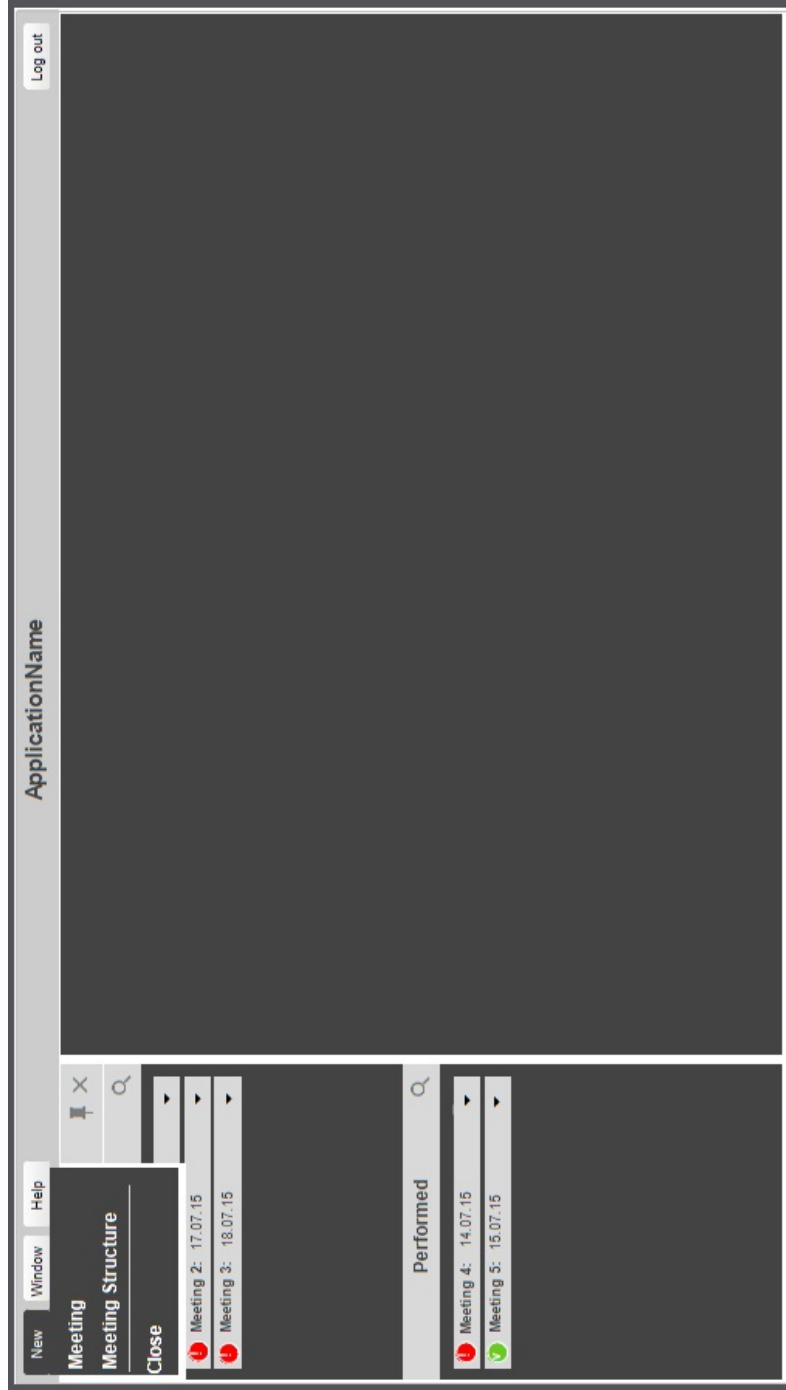
3.3 Post-meeting tasks

1. Create and send a report for the meeting that has not yet been reported. Navigate to 3.4

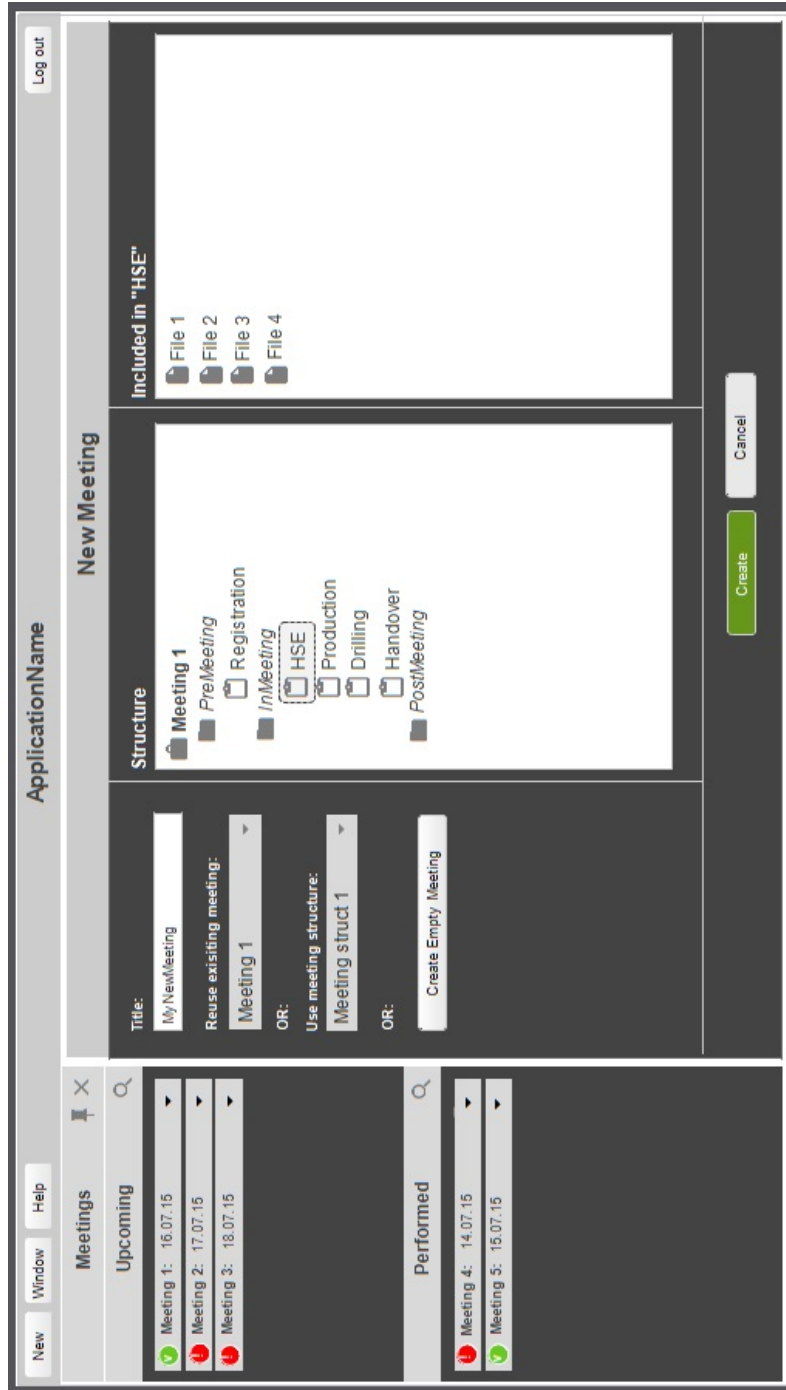
3.4 START



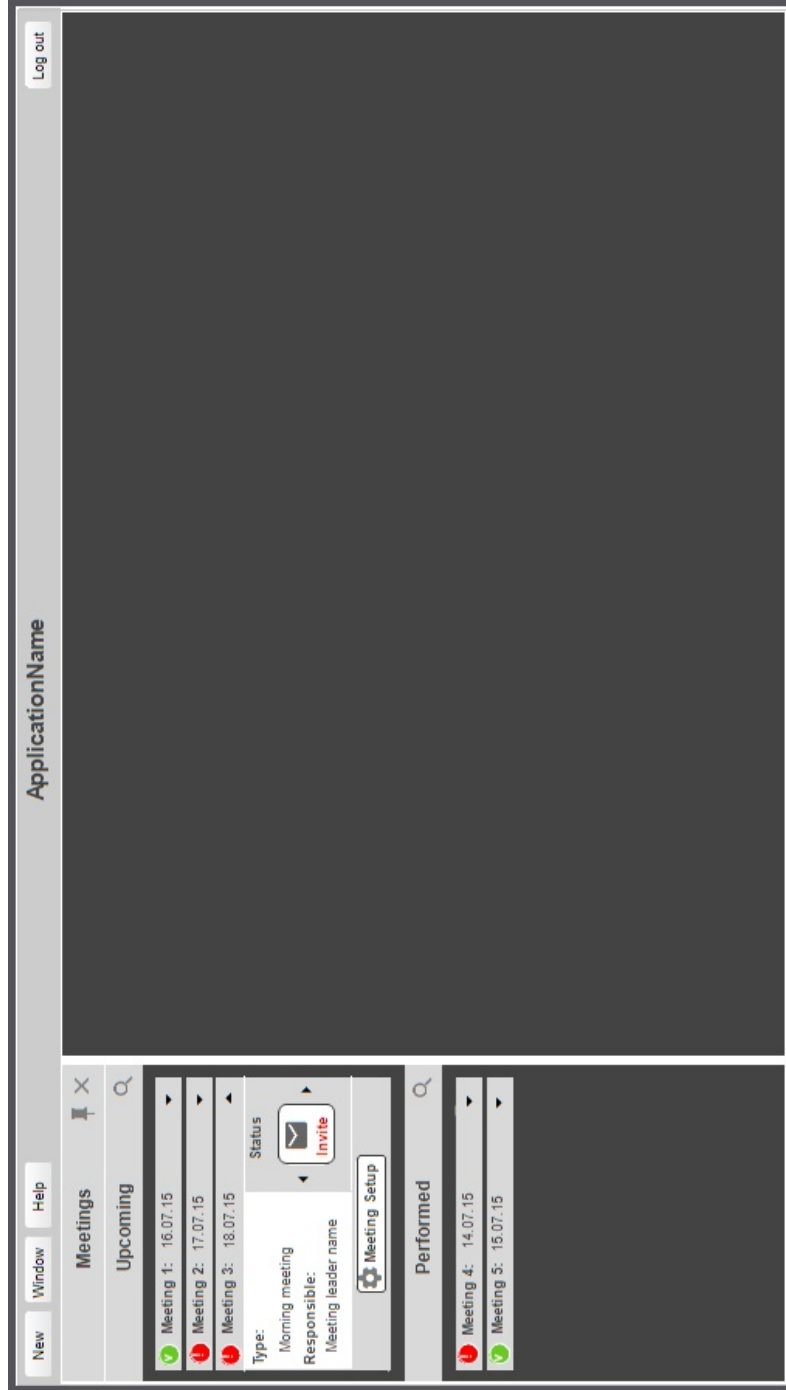
3.5 NEW



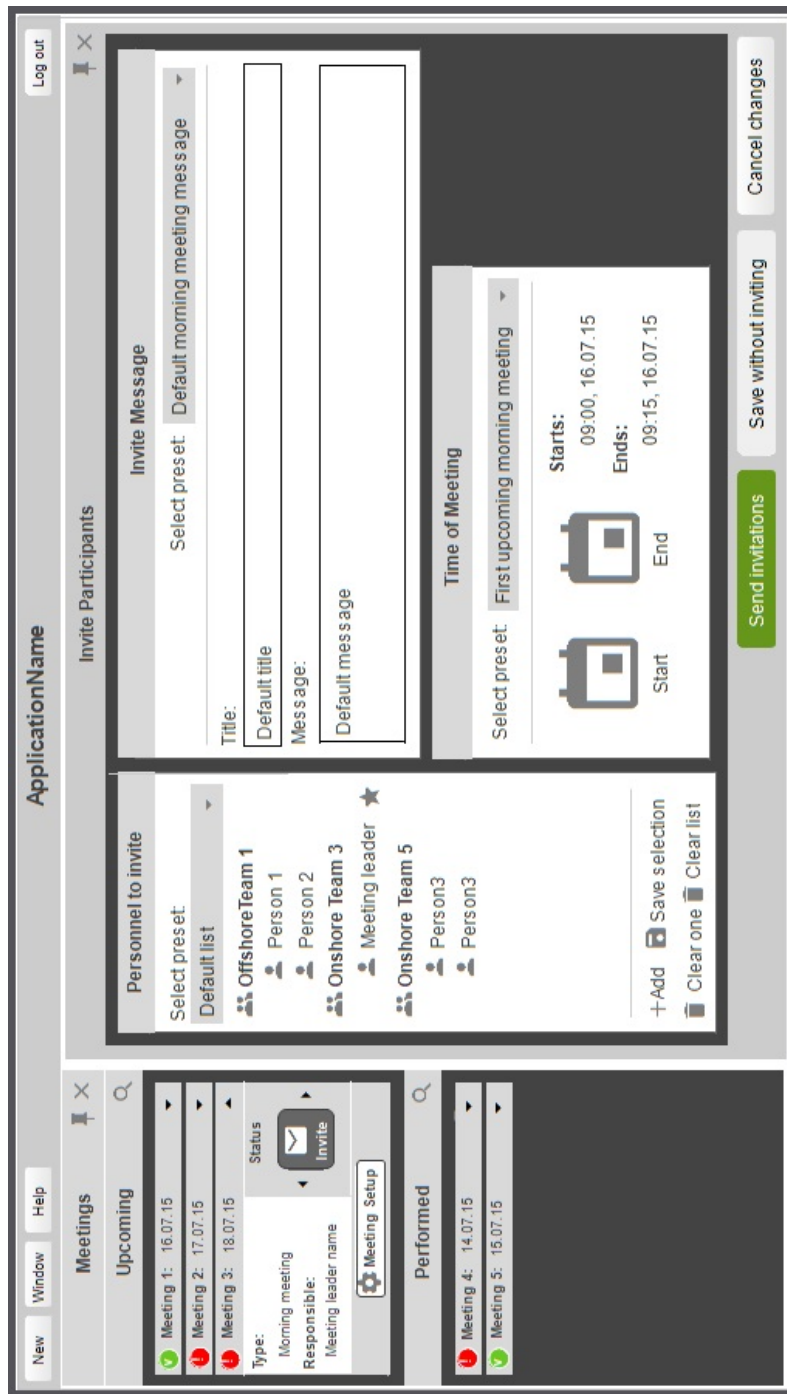
3.6 NEW MEETING



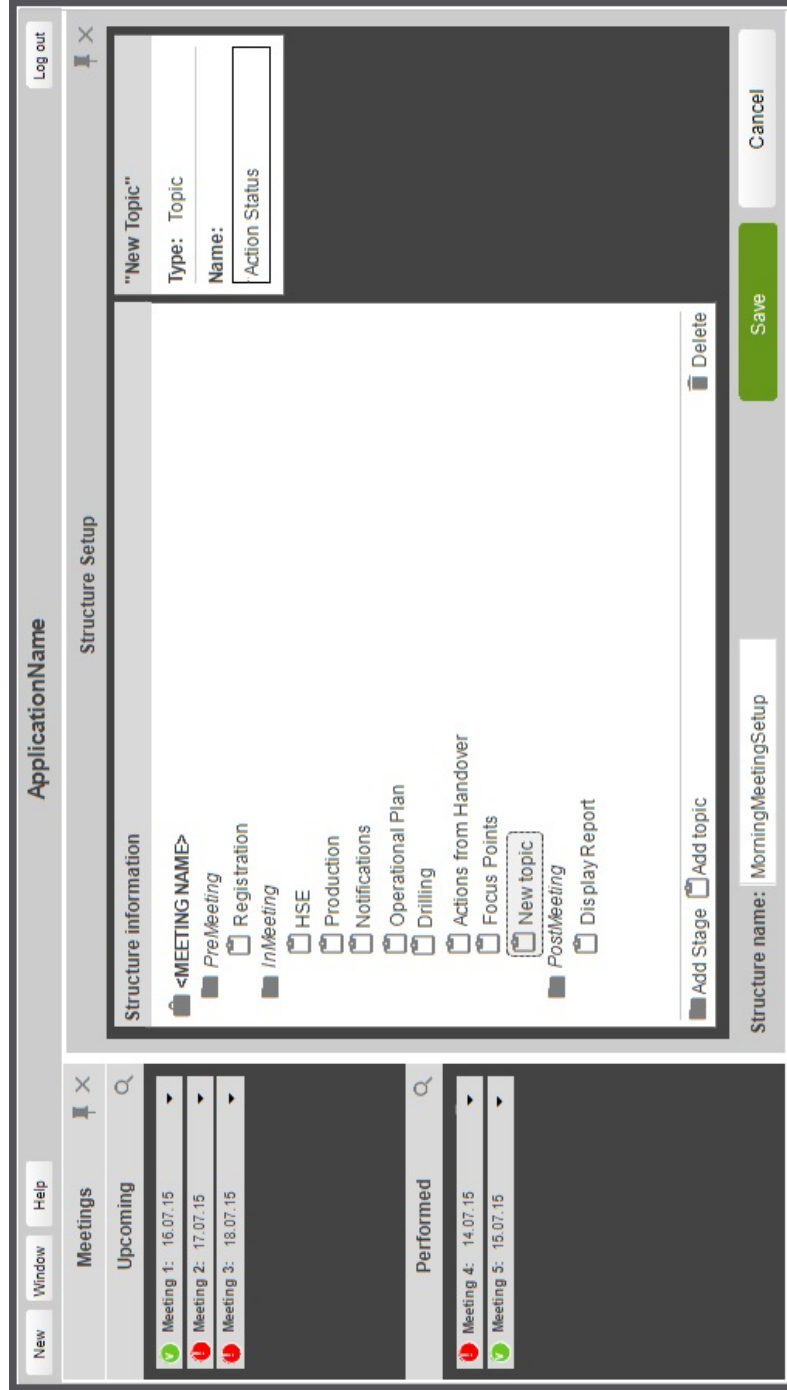
3.7 INVITATION SELECTION



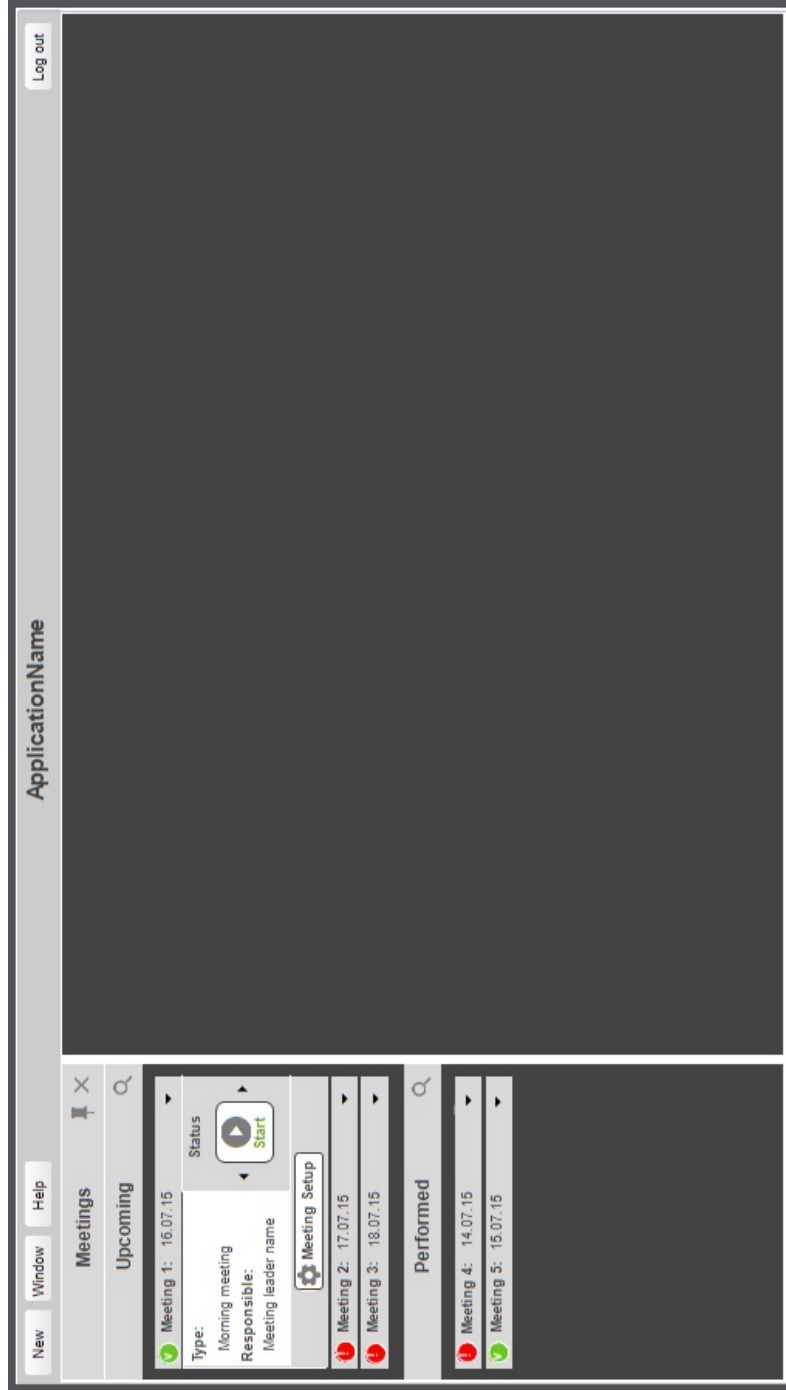
3.8 INVITATION



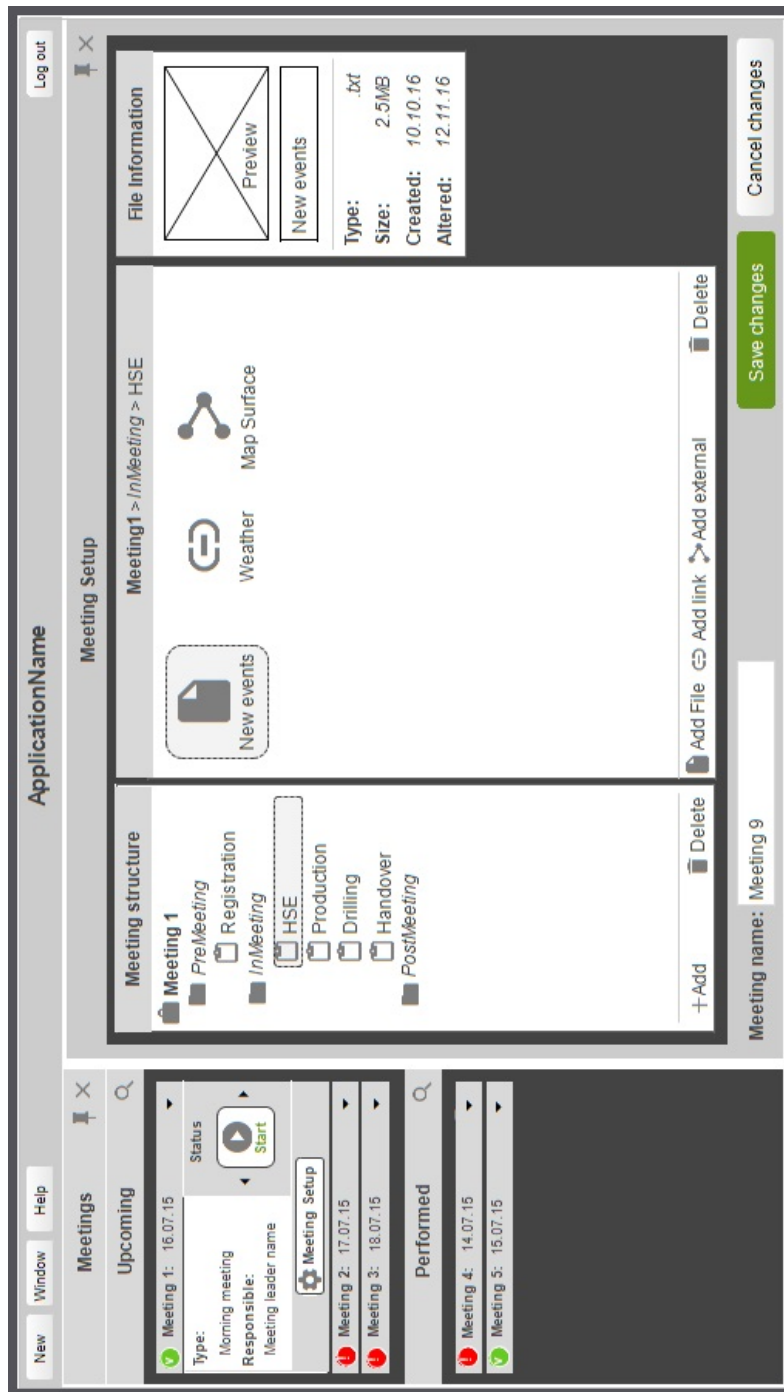
3.9 STRUCTURE



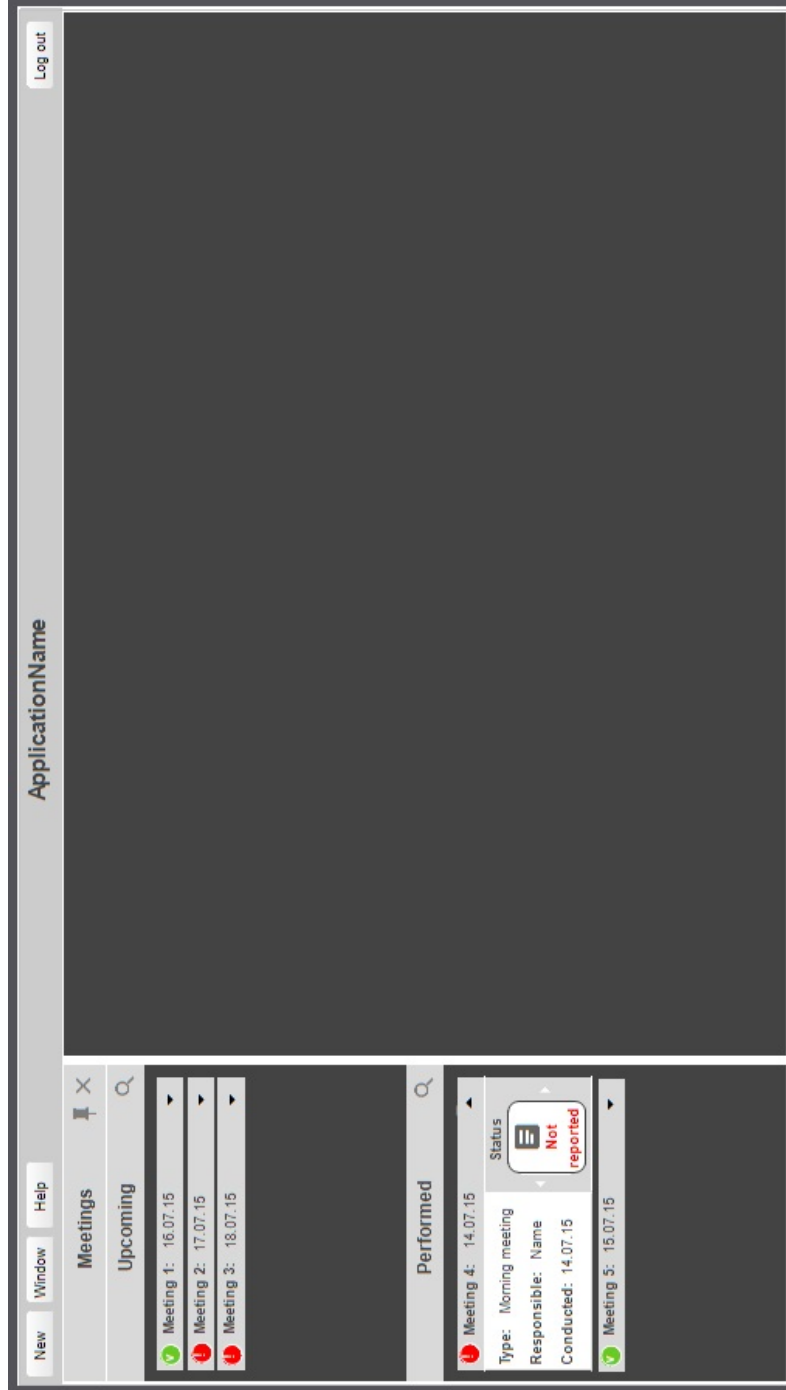
3.10 START SELECTION



3.1.1 START SETUP



3.1.2 REPORT SELECT



3.13 REPORT

The screenshot displays a web application interface for generating a report. The interface is organized as follows:

- Top Navigation:** Includes buttons for 'New', 'Window', 'Help', and 'Log out'.
- Left Sidebar:**
 - Meetings:** A list of three meetings: 'Meeting 1: 16.07.15' (green status), 'Meeting 2: 17.07.15' (red status), and 'Meeting 3: 18.07.15' (red status).
 - Upcoming:** A section for upcoming meetings, currently empty.
 - Performed:** A section for completed meetings, showing 'Meeting 4: 14.07.15' (red status) and 'Meeting 5: 15.07.15' (green status). A dropdown menu for 'Meeting 4' shows details: Type: Morning meeting, Responsible: Name, Conducted: 14.07.15, and Status: Not reported.
- Main Content Area:**
 - Message:** Contains a 'Title' field with 'Default title' and a 'Message' field with 'Default message'.
 - Preview:** Shows a preview of the report content, including sections for 'InMeeting' (with sub-section 'HSE'), a large empty box with a diagonal 'X', and 'Production'. The text in these sections is placeholder Lorem Ipsum.
 - Recipients:** A list of recipients to be included in the report. It starts with 'Select preset: Default list' and includes: 'Offshore Team 1', 'Person 1', 'Person 2', 'Onshore Team 3', 'Meeting leader', 'Onshore Team 5', 'Person3', and 'Person3'.
- Bottom Right:** Three buttons: 'Send report' (green), 'Save without reporting', and 'Cancel changes'.

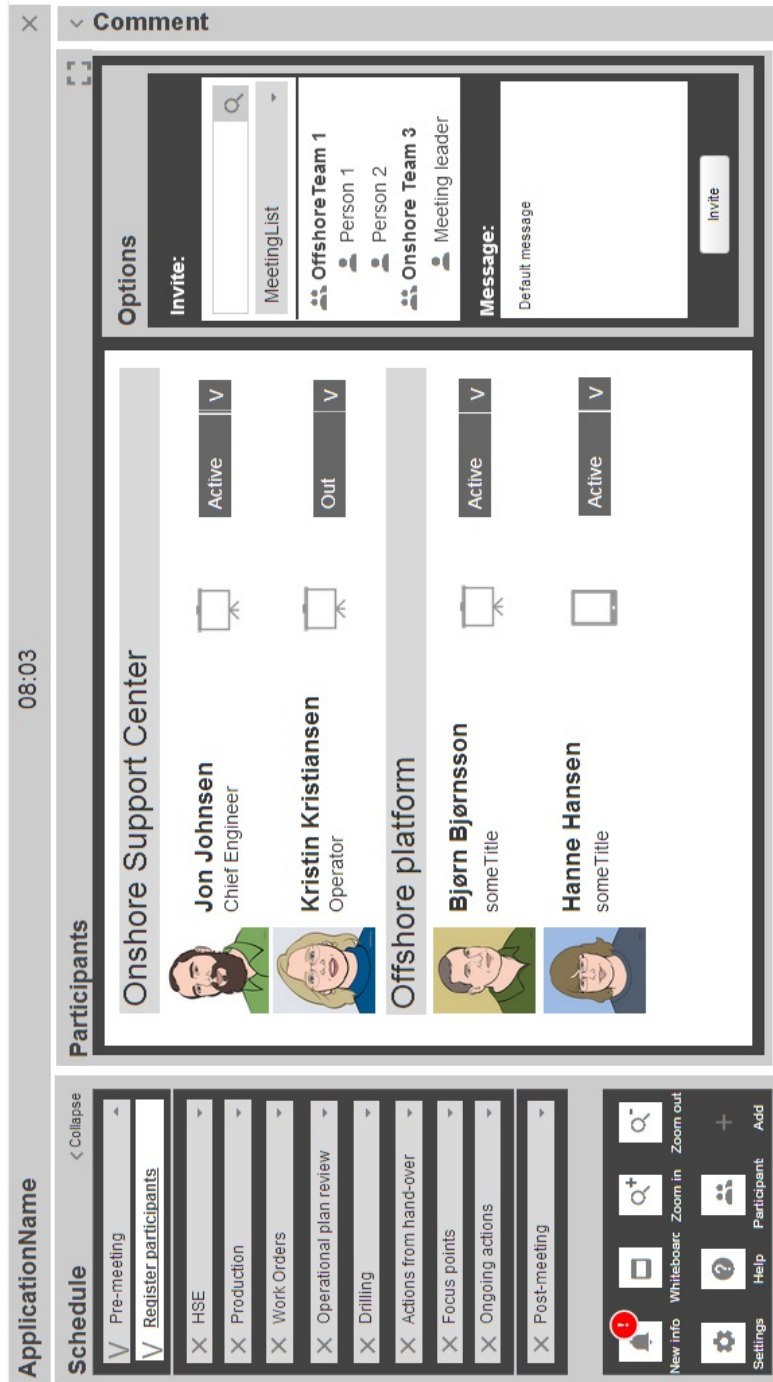
3.1.4 MEETING START



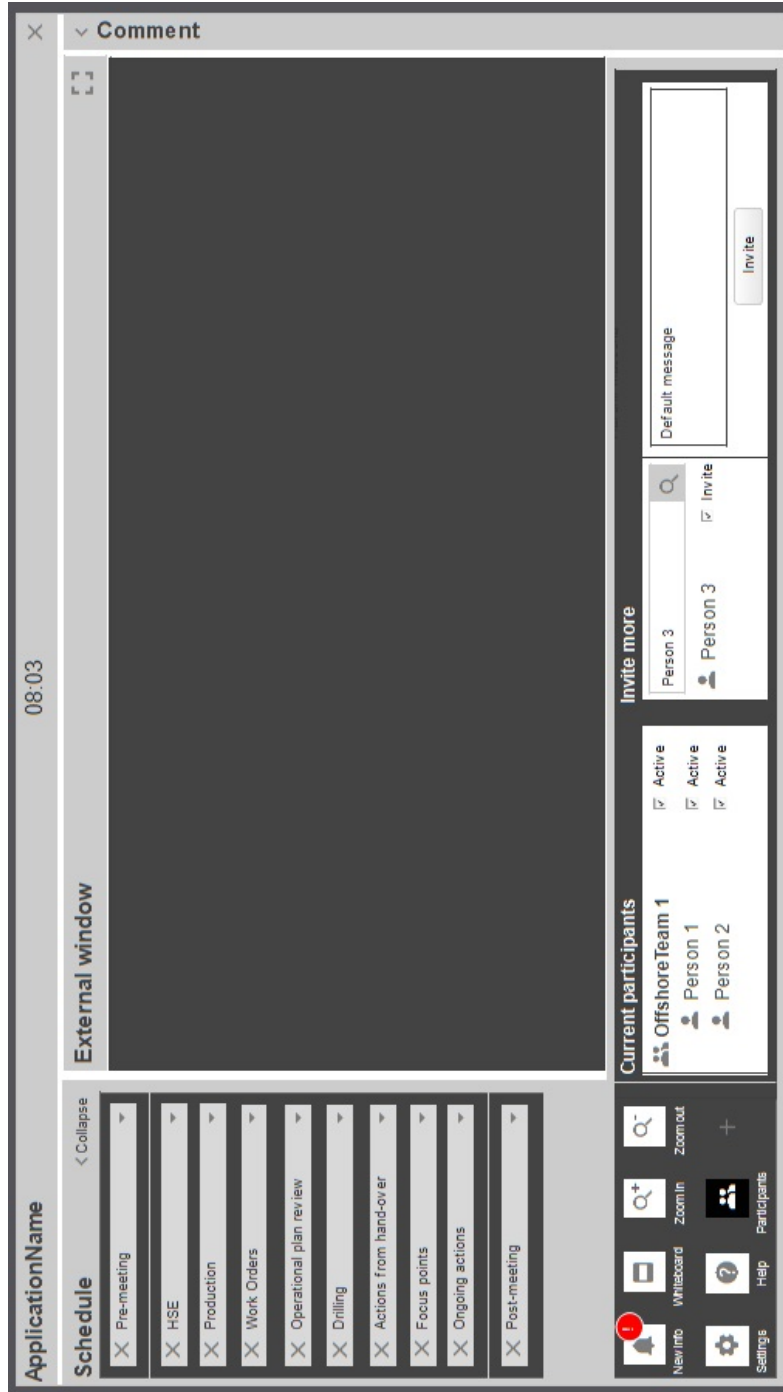
3.15 PRE-MEETING SELECTION



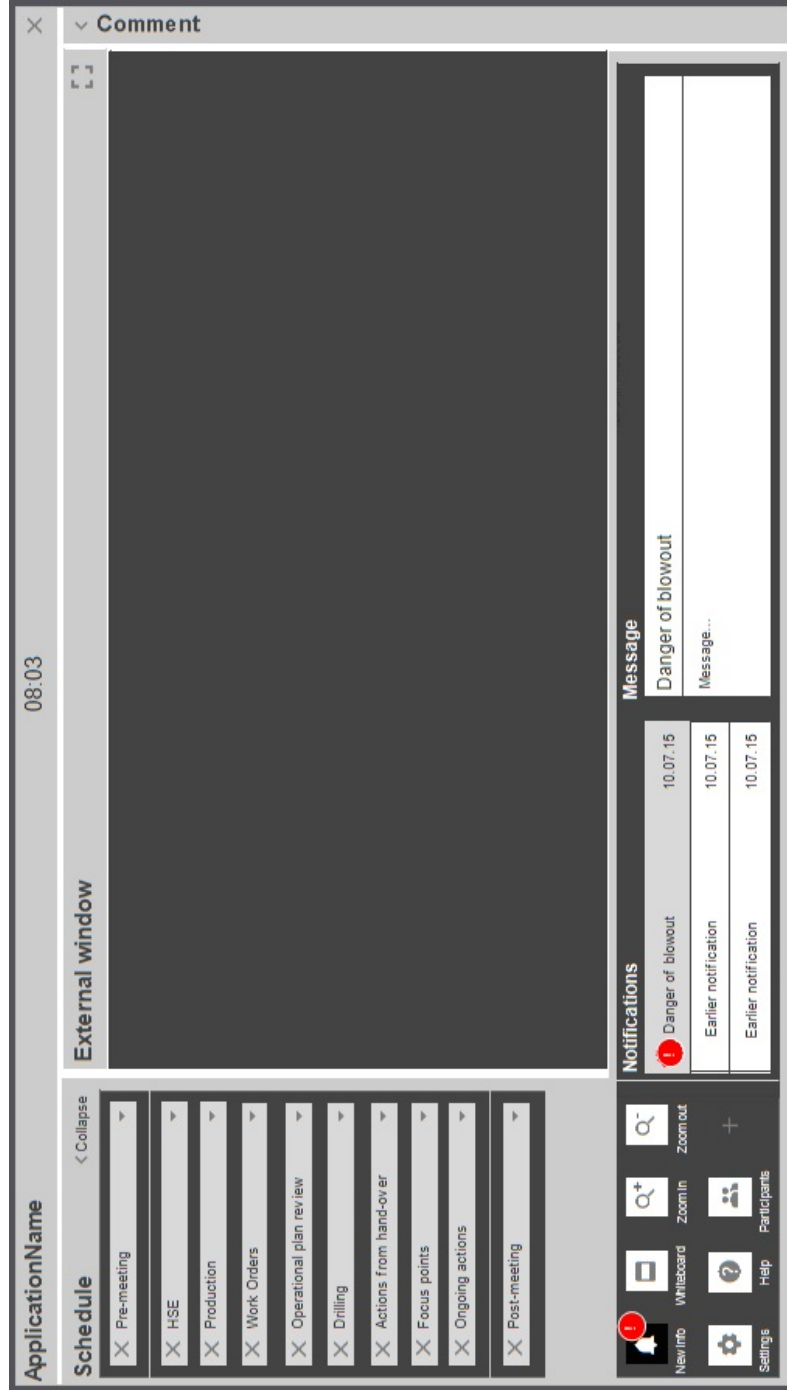
3.16 PRE-MEETING PARTICIPANTS



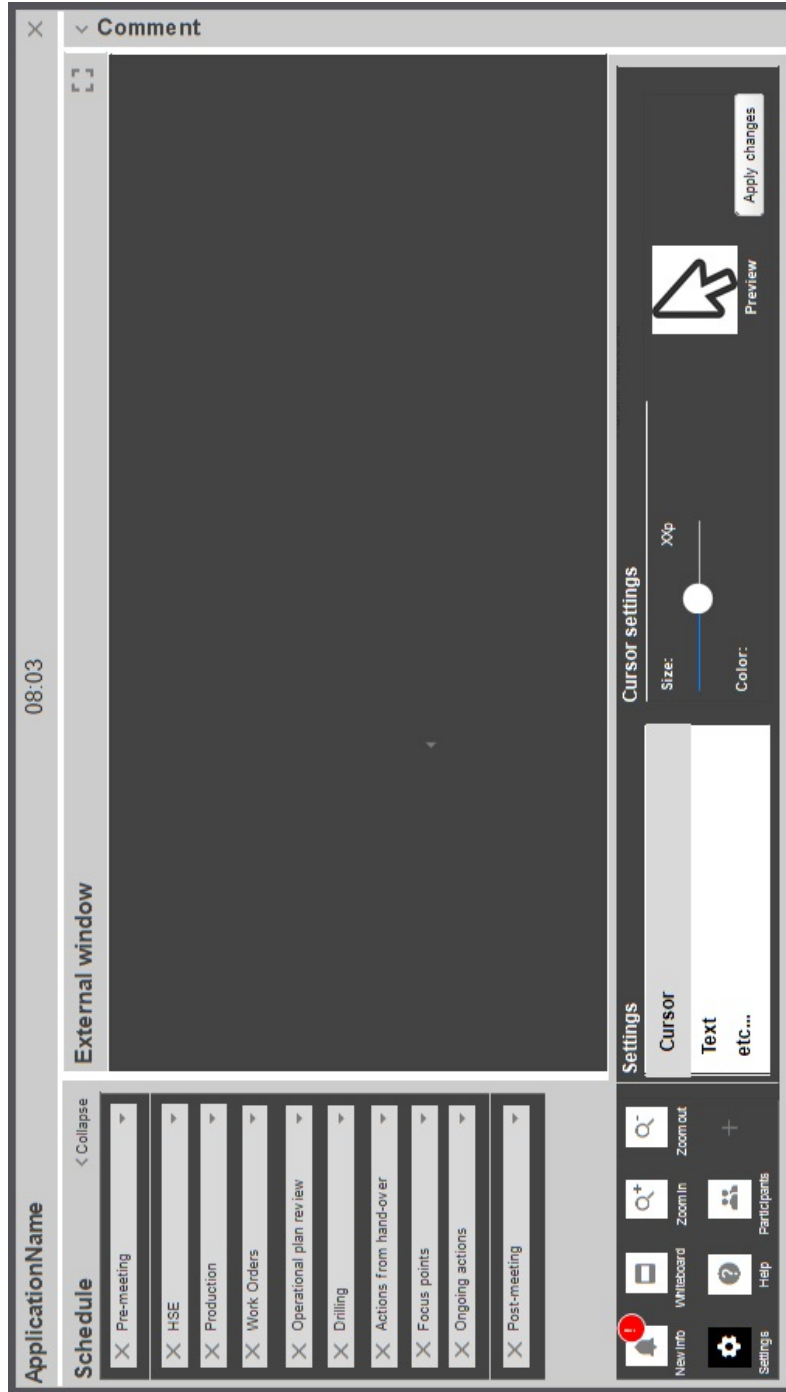
3.17 TOOLS PARTICIPANTS



3.18 TOOLS NOTIFICATIONS



3.19 TOOLS SETTINGS



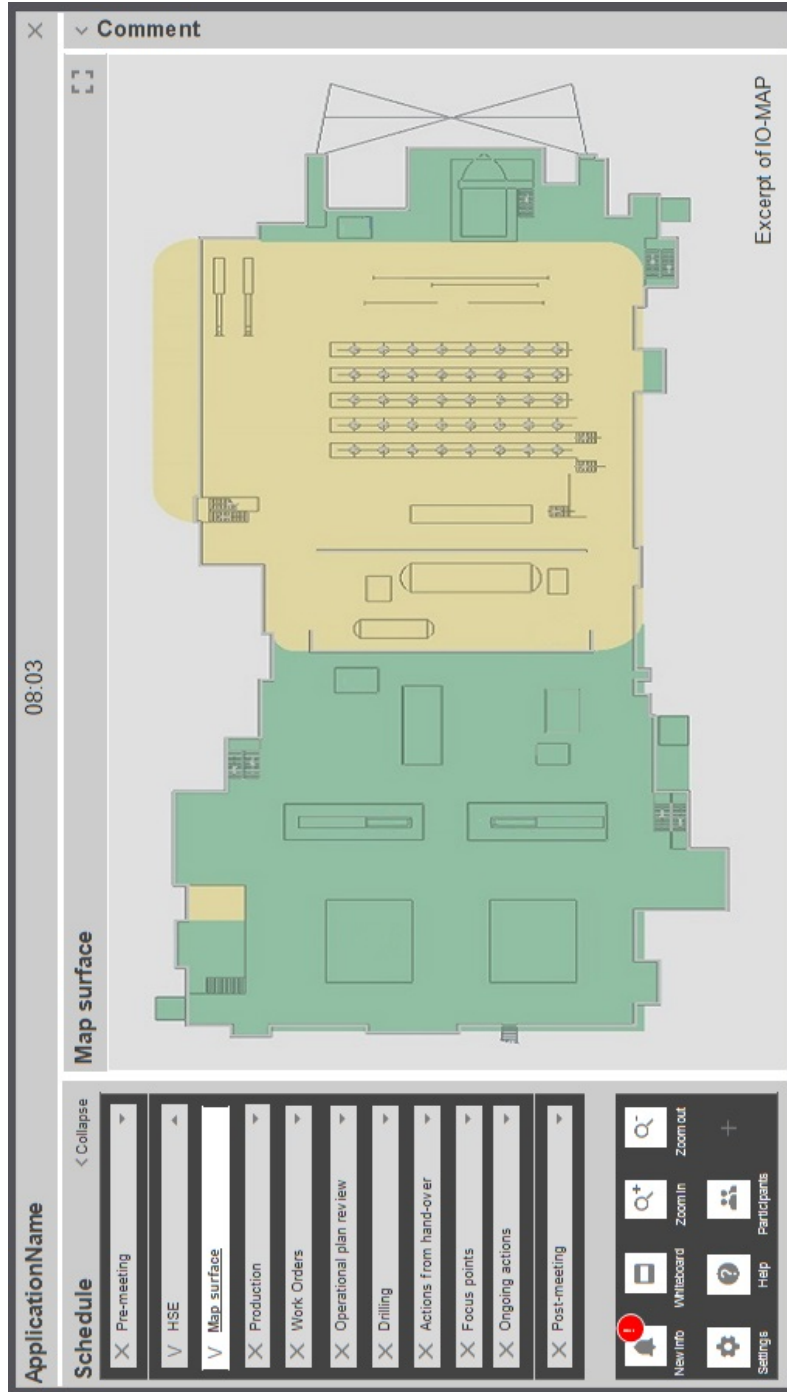
3.20 MAP MENU SELECTION



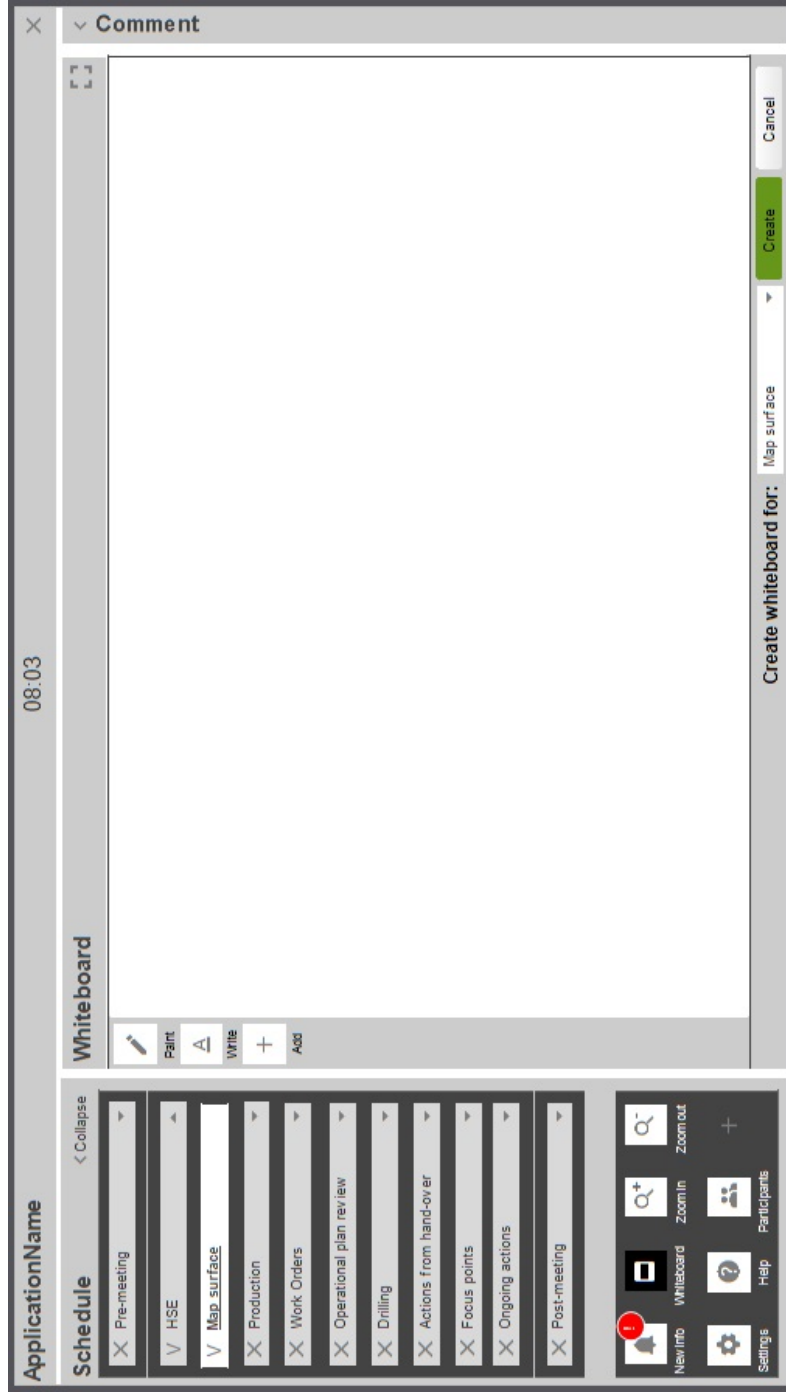
22

217

3.21 MAP SURFACE



3.22 WHITEBOARD



24

219

3.23 COMMENT

The screenshot displays a software interface for adding a comment to a map surface. The interface is divided into several sections:

- ApplicationName:** Located at the top left, it shows the current application name and a close button (X).
- Schedule:** A sidebar on the left containing a list of items with expand/collapse icons (X for collapsed, V for expanded). The items are: Pre-meeting, HSE, Map surface (expanded), Production, Work Orders, Operational plan review, Drilling, Actions from hand-over, Focus points, Ongoing actions, and Post-meeting.
- Map surface:** The central area showing a floor plan with a green area and a yellow area. A timestamp '08:03' is displayed above the map.
- Comment:** A form on the right for adding a comment. It includes:
 - Topic:** A dropdown menu with 'HSE: Map Surface' selected.
 - Added by:** A dropdown menu with 'User 1' selected.
 - Add file:** A button labeled 'Select file'.
 - Screenshot:** A button labeled 'Capture'.
 - Title:** A text input field.
 - Description:** A larger text area for the comment content.
 - Buttons:** 'Add' (green) and 'Cancel' (white) buttons at the bottom.

3.24 COMMENT - ALTERNATIVE

The screenshot displays a software interface with a central 'Map surface' showing a floor plan. The interface includes a top navigation bar with 'ApplicationName' and '08:03'. A left sidebar contains a 'Schedule' section with a list of items: Pre-meeting, HSE, Map surface (selected), Production, Work Orders, Operational plan review, Drilling, Actions from hand-over, Focus points, Ongoing actions, and Post-meeting. A bottom navigation bar features icons for New info, Whiteboard, Zoom in, Zoom out, Settings, Help, Participants, and Comment. A 'Comment' dialog box is open in the foreground, containing fields for 'Title' and 'Description', dropdown menus for 'Topic' (HSE: Map Surface) and 'Added by' (User 1), and buttons for 'Attach file', 'Capture screen', 'Add', and 'Cancel'.

26

221

3.25 PRODUCTION

ApplicationName
08:03
Comment

Schedule < Collapse

- Pre-meeting
- HSE
- Production
- Document
- Production
- Work Orders
- Operational plan review
- Drilling
- Actions from hand-over
- Focus points
- Ongoing actions
- Post-meeting

Produced Oil

Date	Produced oil
01.01.01	10000
02.01.01	21345
03.01.01	12345

01.01

10000 barrels

02.01

21345 barrels

03.01

12345 barrels

External window

New info

Whiteboard

Zoom in

Zoom out

Settings

Help

Participants

3.26 DRILLING

The screenshot displays a software interface for drilling operations. At the top, the application name is "Drilling" and the time is "08:03". A "Comment" section is visible at the top right. The main area shows a 3D geological model with a drilling path. The path is color-coded, with a legend on the left showing values from 0.0 to 0.9. The model includes labels for "Eroforien" and "Eroforien" and numerical values like "1500", "2000", "1590", "1000", "530", and "40". A toolbar at the bottom left contains a "Schedule" section with items: Pre-meeting, HSE, Production, Work Orders, Operational plan review, Drilling, Document, External drilling software, Actions from hand-over, Focus points, Ongoing actions, and Post-meeting. A bottom right toolbar includes icons for New info, Whiteboard, Zoom in, Zoom out, Settings, Help, and Participants.

ApplicationName: Drilling
08:03
Comment
Drilling
Schedule
Pre-meeting
HSE
Production
Work Orders
Operational plan review
Drilling
Document
External drilling software
Actions from hand-over
Focus points
Ongoing actions
Post-meeting
New info
Whiteboard
Zoom in
Zoom out
Settings
Help
Participants

1500
2000
1590
1000
530
40
Eroforien
Eroforien
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0

//Screenshot from "http://www.dgi.com"

4 Heuristic Evaluation

In this part of the test several questions should be answered in respect to the application design, presented in the cognitive walkthrough. The design is to be evaluated based on guidelines for application design and the requirements set for the current system. The questionnaire is divided into topics, and each topic leaves room for further comments.

The symbol used in the tables below are meant for evaluating if the current design fulfills or not fulfills guidelines or requirements. The symbols have the following meaning:

- - -: Not fulfilled.
- -: Mostly not fulfilled
- - / +: Neither fulfilled or not fulfilled.
- +: Mostly fulfilled.
- + + Fulfilled

If some guidelines or requirements are considered not relevant, or if other uncertainties may be connected to the current demand, please leave the box unchecked and write a short comment. Comments may also be written to specify why requirements or guidelines are only met, or partly not met.

The Eight Golden Rules of Interface Design were retrieved from the following site: <http://faculty.washington.edu/jtenenbg/courses/360/f04/sessions/schneidermanGoldenRules.html>

4.1 System Requirements Groups

Rule:	--	-	-/+	+	++	Comment:
Requirement 1. Two-way communication with users operating external mobile devices during collaboration sessions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 2. Sharing of view with mobile devices during collaboration sessions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 3. Possibilities for registering information, as for example keywords, during meetings.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 4. The software must take the existing meeting roles into account by being designed for both meeting participants, meeting leaders, etc.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 5. Multiple modes for varying system flow based on current state in the meeting process. Examples include an editor, a meeting mode and a report mode.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 6. Possibilities for editing and prepare a future meeting's setup.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 7. Implementing options for presenting real-time data.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 8. Implementing options for presenting fixed data.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 9. Implementing an own mode active during meetings, especially designed for such a stage in the planning process.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 10. Implementing an own mode active after meetings, able of producing minutes or other kinds of reports.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 11. Implementing modes for different types of devices to make the interface user friendly, regardless of device/monitor used.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 12. Collaboration participants should see the same platform view regardless of physical location or monitors in use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Rule:	--	-	-/+	+	++	Comment:
Requirement 13. Implementing possibilities for sharing the cursor during meetings.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 14. Implementing agents for decision-support.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 15. Implementing page hierarchy with maximum of two levels.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 16. Implementing possibilities for registering and timing meeting attendance.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 17. Implementing functionality for opening other systems.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 18. Present clear navigation information during interaction with system, as well as when other systems are active inside the software's context.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 19. Having an overview of personnel competence, and possibly also use the competence as decision-support.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 20. Easy to use system stripped of unnecessary functionality.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 21. Understandable and intuitive automation.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 22. Clear system feedback, including graphical feedback.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 23. Measures for focus as warm and cold colors for inter alia setting priority.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 24. Implementing functionality capable of giving readability feedback based on the current adjustments.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 25. A readability test to check if current adjustments satisfies the users' needs.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 26. Implementing possibilities for readability adjustments.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 27. Pre-selection of options when possible, to make it easier for users to follow ordinary system flow.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Rule:	--	-	-/+	+	++	Comment:
Requirement 28. Support for display reduction, if the current meeting makes it hard for viewers to see the whole screen.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 29. Different complexity levels enabling both expert and novice use.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 30. Notifications about important news.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 31. Local display settings, as brightness, color and contrast.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 32. A system capable of taking available hardware in current meeting rooms into account.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 33. A system capable of sharing the required information in morning meetings.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
Requirement 34. Whiteboard functionality.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Further comments:

4.2 The Eight Golden Rules of Interface Design

Rule:	--	-	-/+	+	++	Comment:
1 Strive for consistency. Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent commands should be employed throughout.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
2 Enable frequent users to use shortcuts. As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. Abbreviations, function keys, hidden commands, and macro facilities are very helpful to an expert user.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
3 Offer informative feedback. For every operator action, there should be some system feedback. For frequent and minor actions, the response can be modest, while for infrequent and major actions, the response should be more substantial.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
4 Design dialog to yield closure. Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives the operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
5 Offer simple error handling. As much as possible, design the system so the user cannot make a serious error. If an error is made, the system should be able to detect the error and offer simple, comprehensible mechanisms for handling the error.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
6 Permit easy reversal of actions. This feature relieves anxiety, since the user knows that errors can be undone; it thus encourages exploration of unfamiliar options. The units of reversibility may be a single action, a data entry, or a complete group of actions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Rule:	--	-	-/+	+	++	Comment:
7 Support internal locus of control. Experienced operators strongly desire the sense that they are in charge of the system and that the system responds to their actions. Design the system to make users the initiators of actions rather than the responders.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
8 Reduce short-term memory load. The limitation of human information processing in short-term memory requires that displays be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequences of actions.	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Further comments:

5 ”Usability Inspection”

The future application is designed to fit into the IO context in the oil and gas industry. Please jointly discuss the advantages and disadvantages of the current design with regard to a future implementation in an IO context.

Appendix E

Test of Current Systems' Features

Morning Meetings in the Oil and Gas Industry - a Survey of Existing Products

Introduction

My name is Amund Lågbu. I am a student at Østfold University College in Halden, Norway. I am currently working on a master's degree in informatics. My thesis aims for producing a prototype of a system supporting specifically morning meetings in the oil and gas industry.

This survey is a small part of the master thesis and aims for investigating what functions supporting such meetings that are available in existing software. A set of loosely defined requirements for a future system have been defined in cooperation with Institute for Energy Tecnology in Halden, Norway. This survey's questions are based on these requirements.

The results will be published in a table in the thesis. Here systems will be compared based on their implementations of the defined requirements. The questionnaire will be presented in the thesis' appendix. If you wish that your system's name should be exempted from public, please let me know.

Answering the questionnaire

This survey contains several questions. Each question may be answered by selecting a ranged score, or selecting the option "NA" if it is considered not relevant. Additionally proposed functionality may be rated important or not by selecting a scale to the right of the previously mentioned one. Here "+ +" stands for highly important, while "--" means not important at all.

Note that most of the questions have a wording best suited for describing a tool specialized for use in morning meetings.

Page 3

Your software may be used as a platform during a morning meeting session.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 4

Your system is especially designed for being used in collaboration rooms where two large-screen monitors are used for displaying information.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Your system is able to communicate with personnel outside traditional morning meeting sessions, operating mobile equipment.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 5

Your system has an interface to contact personnel not participating in a particular morning meeting session, but who are available via a mobile device.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Your system has functionality for sharing the view of the meeting platform used in a collaboration session with external users operating mobile devices.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 6

Your system has integrated possibilities for registering actions, decisions and key-information points during morning meetings. Examples on such interaction tools may be text boxes.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 7

If your system has possibilities for registering information in text boxes, these have maximum text limits preventing users from registering long text strings.

This functionality should prevent users from entering too much text during a meeting.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 8

The system is designed for different user roles as the meeting leader, participants, mobile application users, etc.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

The system has implemented modes or similar functions letting the users prepare for a morning meeting and evaluate results afterwards.

To make the different types of interactions with the system more structural - before after and during a morning meeting, and with different types of devices.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 9

The system has an editor, or similar functionality, to tailor the agenda of an upcoming morning meeting.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

The system has implemented several meeting templates, making it easy for the user to create an agenda for an upcoming morning meeting.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 10

While being used as a morning meeting platform, your system has possibilities for displaying real-time information generated by third-party systems.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

While being used as a meeting platform, your system has possibilities for displaying static information generated by third-party systems.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 11

In oil and gas organizations morning meetings typically lasts for 15 minutes. Your system highly supports walk-throughs designed for morning meetings - avoiding unnecessary time-consuming actions.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

After a meeting was held, your software is capable of producing a report describing changes made and new actions and decisions. Examples may be rejections of notifications and movements of jobs.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 12

Your system has modes adapted to different types of monitors/devices

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

During a collaboration session your system makes the participants view the exactly same interface, although they are located at different places.

To prevent misunderstandings due to differently displayed views during a collaboration session.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 13

Your system makes it possible for participants to share the pointer, or similar objects, during a morning meeting session.

To make it possible for participants to get directly involved during meeting sessions.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Your system has implemented automatic agents for decision support during meeting sessions.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 14

Your system has implemented an agent that may be used to test different option alternatives during an ongoing meeting session.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Your system has implemented possibilities for registering who attended a morning meeting session.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 15

In addition to registering, your system is also capable of timing the participants' attendance during a session.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

It is easy to open other systems, and to interact with them, from your platform during meeting sessions.

	True	Partly true	Partly not true	Not true	NA	Weighting				
						--	-	+	++	
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 16

When operating other systems during a meeting session, your system clearly displays the current navigation.

Such means may be an own navigational bar above and outside the third-party system's view, displaying the current navigation.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

During a morning meeting session hot permit jobs may be approved. Your system has implemented possibilities for informing of required competence when a hot permit job is to be approved.

To prevent hot permit jobs to be approved by personnel lacking required competence.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 17

If your system contains several sites, the current navigation is always explained clearly.

Measures as back-buttons, page titles, etc. could be such means.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Functionality, which is not necessary for the meeting participants to perform their job, is not implemented in your system.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 18

Complex functions are never part of the ordinary system flow during morning meeting sessions.

To prevent inexperienced users from making mistakes, complex functions may be accessible from for example drop-down menus.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

New meeting leaders could operate the system without having read instructions or performed training prior to a meeting session.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 19

Automation of system functions is never leading users to believe that manual functions are automated.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

It is implented functions for setting priorities before and during a meeting. This may include marking a text in a specific color.

Making it possible for the system users to display text with different priorities during a meeting session. Functions may include marking text in a specific color, or to gray out displayed text to remove focus.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 20

It is possible for personnel participating in a meeting to adjust text sizes and information layout on the fly by changing system settings.

To increase readability.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 21

The system is capable of automatically adjusting text sizes to display readable text.

By for example calculating text sizes based on information about distance from the viewers to the monitors in meeting rooms, etc.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

The system displays warnings, etc. if text on screen is found not to be sufficiently readable during a meeting.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 22

The system is capable of displaying a readability test for all users (before a session starts).

If all users share the same view of an interface, during a collaboration session, some may find text hard to read. Such a test may find optimal text sizes.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

The system is capable of operating optimally if the displayed view has to be shrunk to fit the users' needs.

Height from for example floor to ceiling may vary in different meeting rooms. This may affect the participants capability to view the screens. In some occasions the displayed interface may have to be cropped to ensure that the participants may view information.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Page 23

If relevant and important information has been generated after a meeting has been prepared, and possibly also during a meeting, the system is capable of notifying about it.

If information is generated, affecting a morning meeting, the personnel in the meeting could be notified about the new information. For example subscriptions to information topics may be thought to be used for such purposes.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

The system has a whiteboard functionality which has possibilities to store the drawn screens.

In some meeting rooms, whiteboards are used for displaying information. A whiteboard functionality may fulfill the same purpose.

	True	Partly true	Partly not true	Not true	NA	Weighting			
						--	-	+	++
Answer:	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Ideas

What do you think?

This survey contained questions linked to some of the requirements found. Please add your remarks or proposals for system improvements in the text area below.

End

Thank you for your patience!

Yours sincerely,

Amund Lågbu
amundlag@hiof.no

» **Redirection to final page of Online Undersøkelse**

Appendix F

XML DTD Schemas

F.1 Menu DTD

```
<!DOCTYPE menu
[
<!ELEMENT menu (schedule, tools)>
<!ELEMENT schedule (meeting_stage*)>
<!ELEMENT meeting_stage (topic*)>
<!ELEMENT topic (item*)>
<!ELEMENT item (heading?)>
<!ELEMENT tools (tool*)>
<!ELEMENT tool (heading?, img?, func, id?)>

<!ELEMENT heading (#PCDATA)>
<!ELEMENT img (#PCDATA)>
<!ELEMENT func (#PCDATA)>
<!ELEMENT id (#PCDATA)>

<!ATTLIST item type (participants | external | whiteboard |
txt | map) #REQUIRED>
<!ATTLIST item url CDATA #IMPLIED>
<!ATTLIST tool type (final | mutable) #REQUIRED>
<!ATTLIST topic n CDATA #IMPLIED>
<!ATTLIST tool url CDATA #IMPLIED>
<!ATTLIST meeting_stage n CDATA #IMPLIED>
]
>
```

F.2 Map DTD

```
<!ELEMENT map (item*)>
<!ELEMENT item (id, title, description, start, finish,
modified, xcord, ycord)>
<!ELEMENT id (#PCDATA)>
<!ELEMENT title (#PCDATA)>
```

```
<!ELEMENT description (#PCDATA)>
<!ELEMENT start (#PCDATA)>
<!ELEMENT finish (#PCDATA)>
<!ELEMENT modified (#PCDATA)>
<!ELEMENT xcord (#PCDATA)>
<!ELEMENT ycord (#PCDATA)>

<!ATTLIST map src CDATA #REQUIRED>
<!ATTLIST map created CDATA #REQUIRED>
<!ATTLIST item type (hot_work | entry | work_over_sea |
hydro_carbon_work | equipment | hazard | other) #REQUIRED>
```