



The Norwegian University of Science and Technology (NTNU)
Faculty of Information Technology, Mathematics and Electrical
Engineering (IME)
Department of Computer and Information Science (IDI)

Postal address: NO-7491 TRONDHEIM

Telephone: 73 59 34 40

Telefax: 73 59 44 66

TITLE	
Compendium: Introduction to course TDT4290 Customer Driven Project, Autumn 2011	
EMPLOYER IDI, NTNU	
REVISED BY Reidar Conradi, Jon Atle Gulla, Jon Espen Ingvaldsen, Steinar Hagen, Geir Solskinnsbakk, Andreas D. Landmark, Anh Nguyen Duc and Sobah Abbas Petersen	
COURSE TDT4290 Customer Driven Project	
VERSION Wednesday, 31 August 2011	PAGES 62
IDI technical report 2/2011. ISSN: 1503-416X	
http://www.idi.ntnu.no/emner/tdt4290/docs/TDT4290-compendium-2011.pdf	

Table of contents

1	INTRODUCTION	4
1.1	GENERAL INFORMATION	4
1.2	GOAL AND RATIONALE OF THE COURSE	6
1.3	REQUIRED KNOWLEDGE	7
2	MOTIVATION ON PROJECT WORK AND GROUP DYNAMICS	8
2.1	ABOUT THE COURSE	8
2.2	PROJECT WORK IN A DIDACTIC PERSPECTIVE	8
2.3	TRAINING IN GROUP DYNAMICS	9
3	ADMINISTRATIVE INFORMATION	11
3.1	WORK LOAD	11
3.2	TIMELINE	11
3.3	GROUP ASSIGNMENTS	12
3.4	RATING OF PROJECT WORK	13
3.5	SUPERVISION AND MEETINGS	14
3.6	PRE-DELIVERY FOR EXAMINER, WRITING TEACHER AND MEDIA, ON OCTOBER 6TH	15
3.7	FINAL PRESENTATION AND DEMONSTRATION ON NOVEMBER 24	15
3.8	ANTI-PLAGIARISM	16
3.9	COPYRIGHT OR INTELLECTUAL PROPERTY RIGHTS (IPR)	16
3.10	COURSE REFLECTION, EVALUATION AND FEEDBACK	17
	APPENDIX A – THE PROJECT PLAN	18
A1.	OVERALL PROJECT PLAN	18
A2.	CONCRETE PROJECT WORK PLAN	19
A3.	PROJECT ORGANIZATION	20
A4.	TEMPLATES AND STANDARDS	20
A5.	VERSION CONTROL PROCEDURES	20
A6.	DOCUMENTATION OF PROJECT WORK	20
A7.	QUALITY ASSURANCE (QA)	21
A8.	TEST PLAN	22
	APPENDIX B – SUGGESTION FOR APPENDICES IN YOUR PROJECT PLAN	23
B1.	PARTNERS	23
B2.	CONCRETE PROJECT PLAN	23
B3.	DETAILED PHASE PLANS	23
B4.	TABLE FOR HANDLING OF RISKS	23
B5.	TABLE FOR EFFORT REGISTRATION	24
	APPENDIX C – CONTENT OF THE PHASE/SPRINT DOCUMENTS/CHAPTERS	26
C1.	INTRODUCTION	26
C2.	PLANNING	26
C3.	PRE-STUDY OF THE PROBLEM SPACE VS. SOLUTION SPACE	26
C4.	LIFECYCLE MODEL: WATERFALL VS. AGILE?	27
C5.	REQUIREMENTS SPECIFICATIONS	27
C6.	ESTIMATION OF REALIZATION EFFORT OF A USE-CASE MODEL	28
C8.	PROGRAMMING	29
C9.	TESTING	30

C10.	INTERNAL AND EXTERNAL DOCUMENTATION	31
C11.	EVALUATION	31
C12.	PROJECT PRESENTATION AND DEMONSTRATION	32
C13.	APPENDICES	32
4	APPENDIX D – ADMINISTRATIVE AND TECHNICAL RESOURCES	33
D4.1	OFFICE RESOURCES	33
D4.2	TECHNICAL RESOURCES	33
D4.2.1.	<i>Workstations</i>	33
D4.2.2.	<i>CVS / SVN configuration management tool</i>	34
D4.2.3.	<i>Text processing tools</i>	34
D4.2.4.	<i>Use of collaboration technology in the project</i>	34
5	APPENDIX E – SCRUM – A POPULAR AGILE METHOD	35
E5.1.	PRODUCT BACKLOG	35
E5.2.	SPRINT PLANNING MEETING	35
E5.2.1.	<i>Daily Scrum status meeting</i>	36
E5.2.2.	<i>Irregularities</i>	36
E5.2.3.	<i>Sprint review meeting</i>	36
E5.2.4.	<i>How do I prepare a project for Scrum (short tutorial)</i>	36
6	APPENDIX F – USE-CASE BASED EFFORT ESTIMATION	38
F6.1.	INTRODUCTION TO USE-CASE ESTIMATION	38
F6.2.	MORE ON THE ESTIMATION METHOD	38
F6.3.	A MINI-DISCUSSION	39
F6.4.	REFERENCES	39
F6.5.	APPENDIX: A SMALL USE-CASE DIAGRAM, WITH EXTRA COMMENTS	40
F6.6.	ADD-ON TO USE-CASE EXAMPLE	41
7	APPENDIX G – PROJECT ASSIGNMENTS PROPOSED BY CUSTOMERS	44
G7.1	GRIDMEDIA TECHNOLOGIES	44
G7.2	SCHLUMBERGER	45
G7.3	SINTEF ICT, SINTEF T&S	46
G7.4	SINTEF ICT	47
G7.5	INSTITUTE FOR ENERGY TECHNOLOGY (IFE)	48
G7.6	SINTEF IKT AND TRONDHEIM KOMMUNE	49
G7.7	NTNU IDI	50
G7.8	ARTSDATABANKEN	51
G7.9	THALES NORWAY AS	52
G7.10	NORWEGIAN PUBLIC ROADS ADMINISTRATION, ITS	53
G7.11	NORSK TILLITSMANN STAMDATA AS	54
G7.12	Q-FREE ASA, R&D	55
G7.13	VITENSKAPSMUSEET, NTNU	56
8	APPENDIX H – STUDENT LISTS AND GROUPS	57
9	APPENDIX I- KICK-OFF ROOMS FOR GROUPS	60

Any writing error will be rewarded by 10 NOK upon first time repeating, and any logical error by 50 NOK similarly.

Please contact R. Conradi in person at any time.

1 Introduction

1.1 General information

This **master-level** course **TDT4290 Customer Driven Project** deals with a **project assignment** that is **mandatory** for all computer science (“Datateknikk”) students in their 4th study year at NTNU/IDI, typically with 50 and mostly Norwegian students. In addition comes participants from the two-year, international master program in Information Systems at IDI (with 10-15 students from all over the world), plus Erasmus and other guest students (usually a handful of Europeans).




This compendium contains all the necessary information for this course, the assignments (one for each project-group), and a suggested outline for the final project report. In addition come some examples of what a project plan should contain.







Practical information regarding project-group composition, dates etc. can also be found on the web page of the course (<http://www.idi.ntnu.no/emner/tdt4290/>). All students should check this web page for updates. In case of mismatch between information in this compendium, information given during lectures, by email, and on the above web page, the *last updated* information should be regarded as correct. The following IDI people are involved with this course:

Course responsables:

Prof. Reidar Conradi	Course responsible	conradi<##>idi.ntnu.no	tel 918.97029
Prof. Jon Atle Gulla	Course co-responsible	jag<##>idi.ntnu.no	tel 913.47759
Adj. Assoc. Prof. Sobah A. Petersen	Course co-responsible	sap<##>idi.ntnu.no	tel 73551336
PhD-stud. Andreas D. Landmark	Practical coordinator	andreal<##>idi.ntnu.no	tel 988.07021

Group advisors:

Name	# group	Picture	Email contact	Telephone
Prof. Reidar Conradi	1		conradi<##>idi.ntnu.no	918.97029
Prof. Jon Atle Gulla			jag<##>idi.ntnu.no	735.91847
Postdoc Daniela S. Cruzes	1		dcruzes<##>idi.ntnu.no	942.49891

PhD-stud. Andreas D. Landmark	1		andreala<##>idi.ntnu.no	988.07021
PhD-stud. Anh Nguyen Duc	2		anhn<##>idi.ntnu.no	483.48496
PhD-stud. Mohsen Anvaari	2		mohsena<##>idi.ntnu.no	405.70403
PhD-stud. Tosin D. Oyetoyan	2		tosindo<##>idi.ntnu.no	405.65642
PhD-stud. Wei Wei	2		wwei<##>idi.ntnu.no	735.93360
PhD-stud. Muhammad Asif	2		muhamma<##>idi.ntnu.n o	735.93671

Students will be divided in project groups of 5-6 students (detail in Section 3.3). Each group will be allocated a advisor from IDI - either a faculty member, a postdoc, or a PhD student - plus a main customer representative.

The mandate of the group’s advisor:

The group’s advisor serves as a one-person “steering committee” for your project. His/her responsibility is to keep an eye on the main process of the work, and to oversee that sufficient contact with the customer is maintained. The advisor must therefore regularly receive updated status reports, copies of relevant work plans and technical documents from the group, so that all this can be discussed in a **weekly advisor meeting**.

Due to budget restrictions in 2011, the course this year offers only *one* group advisor instead of two. In the previous years, we assigned one main *and* one *assistant* advisor for each group, where the latter usually was a senior student to help with more technical issues. The consequences of this reduction will be evaluated at the end of the course.

In addition, the group should have several, weekly *internal meetings*, and regular *customer meetings* (detail in Section 3.5),

1.2 Goal and rationale of the course

The **goal** of course TDT4290 is to teach *you* and your fellow students – by working in groups – software engineering (SE) skills in the context of a *development project* to make a realistic *prototype* of an Information System (IS) “*on contract*” for a real-world customer.

Each project group is initially given a one-page *project assignment* from an external customer. All the phases of a typical IS/IT project are covered, e.g. project management and planning, pre-study, requirements, design, programming, testing, evaluation and documentation, but no “maintenance”. However, we do *not* accept customers that just want the group to write a “summary and evaluation” of some hot topic, with no ensuing implementation. And inversely, we do *not* want customers that come with pre-made requirements, and just want the group to complete a pre-designed system architecture.

Due to resource constraints, the focus should nevertheless be on the “*early*” *lifecycle phases*, i.e., project planning, pre-study, requirements specifications and system design – but the exact focus and emphasis will be decided by the group, in dialog with their customer. For instance, groups with focus on the early phases should not omit making a working prototype of some system parts. On the other hand, “programming-eager” groups may try to make a rather complete prototype, e.g. by applying agile iterations in Scrum-style (see C.4 and Appendix F). The important issue is that the group clearly justify their decisions, and that there is a logical flow in the project report from start to end of all the phases, and that all the phases and iterations build naturally on each other.

Each group should write a project report in English, and hold a presentation and demonstration of the final *prototyped* product for the customer, while an external examiner (censor) is present.

If a fundamental disagreement with the customer arises, the group has, if needed, “the final word” since the group members gets the credit through a final exam (report) worth 15 Sp. But such a dead-lock situation has hardly happened in the 38 years that this course has been arranged! However, the group and their advisor should do what is possible to resolve any major disagreements. Conflicts are to be explained, negotiated and resolved (managed), as this is part of the real world work. **It is therefore crucial that the group is focused and has a good dialog with the customer.**

1.3 Required Knowledge

Required theories and methods for making large and long-lived SE/IS systems are mostly covered in previous, bachelor-level courses. This knowledge base is supplemented by a double seminar on group dynamics, and four guest lectures on project management, Scrum, IT architecture, and use-case based effort estimation. In addition come a course in presentation techniques and a seminar on technical writing.

Since 2008 in this course, the students have been encouraged to use the Agile software development method so-called “Scrum”. Given the time constraints of this student project, there is hardly time for more than 2-3 increments, called “sprints” in Scrum. Many groups have since 2008 chosen an agile development model and had good results. As mentioned above, the structure of the final project report must reflect the choice of overall lifecycle model, i.e. waterfall vs. iterative/agile development methods.

2 Motivation on Project Work and Group Dynamics

2.1 About the Course

The goal of this course is to teach fundamental software engineering skills through realistic training in software development and project management. You will have the opportunity to apply the knowledge you have gained previously in your studies.

During this course, you will experience situations that will require:

- Decision, solving and design and development of a relatively large and complex system.
- Creative and collaborative problem solving. Earlier in your studies, the tasks have been smaller and more well-defined. In this project, there are (conflicting) decisions to be made. You will have to show creativity, be pragmatic and be capable of solving fuzzy tasks under heavy time and resource constraints; i.e. fast decision making under great uncertainty.
- Coordination of efforts and distribution of work and responsibilities.
- Project management, cooperation, decisions, follow-ups, and a dispute resolution.
- Ability to adapt to non-ideal working situations.
- Planning and execution of plans. This involves creation of project plans and registration and monitoring of effort and resource usage.
- You must handle difficult customers. They can be unreliable and/or unavailable. They might change directions, come up with new ideas, and have an unclear picture of what they really want. An important part of this course is to manage the group project, so that the results match the customer's needs, even though the situation may turn difficult. This requires routines for quality control. Each group should deliver a project plan with resources and milestones, coupled to quantitative measures, and use both "dry" verification (mainly reviews) and "wet" testing.
- Structuring of requirements specifications.
- Documentation. The project documents must be complete, well structured and target the technical knowledge level of the customer.
- Defend decisions that are taken on behalf of the customer. You should document all delays, overruns, and weaknesses, so that they can be explained and argued. Ideally, all decisions should match conditions coming from the customer (the customer has the right to complain on any aberration that is not his/her fault).
- To present (and sell) the final product for the customer / external examiner. Under the final presentation and demonstration, it is important to give the customer a complete and good impression of the system delivered.

2.2 Project work in a didactic perspective

Several evaluations have been carried out of previous versions of TDT4290 (i.e. "Systemering prosjektarbeid" and "Programmering prosjektarbeid"). These evaluations are in general very positive. See Markus Sorge: "Evaluering av prosjektundervisningen ved IDI, NTNU", Program for lærerutdanning, spring 2000 (www.idi.ntnu.no/undervisning/siving/docs/prosjektevaluering.pdf).

"Technology" aims to satisfy human-societal needs in a cost-effectiveness way. Engineering is then the process of constructing such means as houses, food, clothing, roads, bridges, vehicles, machines, and, presently, computer- and information like "systems". That is, an engineer creates new reality, not only studies the existing one(s).

Engineering requires a domain-specific methodology for the actual context, e.g. being farming, bridge-building or banking. An engineer will apply scientific insight (both technical and social), combined with knowledge and experience from many sources. It is often a strong relationship between what is being constructed and available time, budgets, and tools / methods. Because of the

substantial complexity and diversity of the given work and the characteristics of the processes, it is often necessary for several people to work together. That means that engineering has a social dimension, since it is executed as group work. Cooperative and communicative skills are therefore essential.

Project work in teams is an important part of the engineering discipline. Your study program at NTNU is among the ones with most emphasis on project work. Project work means on the one hand that you need to make an agreement with a customer (customer / organization) about what should be constructed. On the other hand, you have to design and implement technical solutions that satisfy the elicited constraints and requirements. You also have to consider changes over time, as most customers are not sure about what they really want, what their ... may want in the future or they may anyway change opinion. As a consequence, the proposed unfinished solution must be modified. The project groups must also be well organized and effective, and try to avoid destructive internal conflicts.

All this means that you will get a hectic work situation – sometimes at the edge of chaos. You have to combine your theoretical knowledge from previous courses to solve specific and practical problems. You have to use a considerable amount of effort in cooperation, communication, planning, and improvisation and show capabilities of working under pressure.

Your project will give you essential training to become a professional software engineer. Feedback from industry says, that it is almost impossible to get more done in 3 months than what such a group of students is capable of. Further, software engineers from NTNU are useful from day one: they possess the theoretical knowledge and know how to work efficiently in teams.

So, the expectations are great from all participants: the IDI department, lecturers, advisors, external customers and of course the students themselves. The project report (written in English) should not be more than 200 pages, exclusive appendices and graphics.

If your group experiences that some of the team members are not participating satisfactorily, you should immediately contact your advisor. If you experience other minor problems, the advisor is the one to contact. However, most (minor) problems are to live with; in fact, it is a part of the course to learn to deal with such issues in a project.

So, welcome to an interesting and hectic semester in this course!

2.3 Training in group dynamics

Good teamwork and group dynamics are essential for the success of any collaborative project. Therefore, "social" skills are of utmost importance to become a successful project co-worker. A seminar on *group dynamics* is planned as a part of this course, to support the project groups to learn more about team work and group dynamics. In addition to the seminar, the following time slots should be used wisely to create a good team atmosphere among your group, particularly at the beginning of the project.

1. Your first internal group meeting at the start of the course, Tuesday 30 August, 13:15-14:00 (for room, see Appendix I). This hour could be used to get to know each other and to find out more about each other's complementary skills that could be utilised for the benefit of the project.
2. The first customer meeting, Tuesday 30 August, 14:15-16:00. Use this hour wisely to get to know your customer as it may be a long time until you meet the customer again. Make an effort to understand the customer's needs and to establish communication means and routines with the customer. It is also important that the customer gets to know you as well.

3. First advisor meeting, Wednesday 31 August, 09:00-13:00 (see room and time as in point 1). During this meeting, the advisors will focus on teamwork and group dynamics and ensure that you have some support to establish a good project group.
4. Mini-seminar in Group Dynamics, by Faveo Prosjektledelse, in auditorium S8 on Friday 2 September, at 14:15-18:00.
5. Follow-up seminar in Group Dynamics by Faveo Prosjektledelse, in S8 on Tuesday 20 September, at 14:15-18:00 Prepare yourselves for these Groups Dynamics seminars and ask the presenter for advise and support for your group.
6. Additional support on group dynamics will be announced later.

3 Administrative information

3.1 Work load

NTNU has officially an autumn semester with 19 weeks, and a spring semester with 21 weeks. Of the former 19 weeks, two are spent on continuation exams (and immatriculation etc.), and three are spent on exams in December - which you don't have. This implies 17 "study weeks" in the autumn semester, each of 40 person-hours (work-hours) per student.

This again corresponds to $170 (17 * 25\% * 40)$ person-hours per student for a "7.5-Sp" course (25% of 30-Sp semester total). And note that an hour has 60, not 45 minutes. :-)

Since this project runs in 13 weeks (really 12.6) instead of 17 weeks, the weekly effort per student then becomes: $40 * 50\% * 17/13 = 26.2$ person-hours - i.e. 11% more than a 48-hour week. We will anyhow try to start one week earlier next year to get a duration of 14, not 13, weeks.

Furthermore, the official web page of the course -

<http://www.ntnu.edu/studies/courses/TDT4290/2011> - specifies 24 weekly person-hours per student, but this is for 14 weeks. **So let us round off to 25 person-hours per week and student, i.e., 325 person-hours per student in totally 13 weeks.**

For a project group of 5-6 students, the available effort per group will lie between **1625** and **1950** person-hours, including own reading, meetings, lectures, and seminars. Earlier projects have shown that it is possible to deliver really good results within that timeframe.

It is important that everyone is honest and registers all effort (as person-hours) spent on the project. This means that the project documents must show the real work load. Effort overruns will result in less sparetime for you personally and less time for other courses. Inflated work effort does not affect the grades given in this course!

3.2 Timeline

This is the preliminary time plan for the activities of this course. For further details and updates check the web page. Activities marked with **(M)** are mandatory.

Table 1 – Timeline of the course

Day	Date	Week	Activity	Place	Responsible/ lecturer	Time
Tuesday	August 30.	36	Customer meeting (M)	ITV-454, plenary	Reidar Conradi/ Jon Atle Gulla/ Sobah A. Petersen/ Andreas D. Landmark, advisors, customers	11:15 - 12:00
Tuesday	August 30.	36	Common kick-off (M)	S1, plenary	Reidar Conradi/ Jon Atle Gulla/ Sobah A. Petersen/ Andreas D. Landmark, advisors.	12:15 - 13:00
				See Appendix I	Only group members.	13:15 - 14:00
				Same as above	Customer, group members, advisor (some of these may share two groups).	14:15 - 16:00
Wednesday	August 31.	36	Follow-up day (M)	See Appendix I	Group members and their advisors.	See Appendix I
Friday	September 2.	36	Mini seminar in group dynamics, part I (M)	KJL1	Faveo Prosjektledelse	14:15-18:00
Tuesday	September 6.	37	Project management	S8	Bearing Point: Michael Sars Norum	14:15-18:00
Tuesday	Sept. 13.	38	Scrum, an agile development method	S8	Torgeir Dingsøy	14:15-16:00

Tuesday	Sept. 20.	39	Seminar in group dynamics, part II (M)	S8	Faveo Prosjektledelse	14:15-18:00
Tuesday	Sept. 27.	40	IT-architecture	S8	Einar Landre, Statoil	14:15-18:00
Tuesday	October 4.	41	Use-case based effort estimation	S8	Reidar Conradi	14:15-15:00
Thursday	October 6.	41	Pre-delivery of project report (M)	TBA	Give via Andreas D. Landmark	12:00
Tuesday	October 11.	42	Course in presentation techniques	S8	Per Kotte, ex-Statoil	14:15-18:00
Tuesday	October 18.	43	Seminar in technical writing (M)	S8	Nancy Lea Eik-Ness, NTNU-Drøgtvoll	14:15-18:00
Thursday	Nov. 24.	48	Project demo (M), Final delivery of project report	Misc. group rooms, put on web later	All student groups and their advisors and customers	09.15-16.00

3.3 Group assignments

We expect about 75 students in total. This gives in total 13 groups with 5-6 students per group. Each group is preallocated to one customer and one group advisor. The groups should therefore have a tight cooperation with their advisor.

Group assignments are essentially made *randomly*. This is done intentionally to create groups where the members generally do not know each other beforehand. This is a typical situation in a real life, especially when working as a consultant. Foreigners will furthermore be allocated as evenly as possible. Also, we put either 2-3 girls in a group, or none.

Since many of the students in the course are foreign, with limited or no knowledge to Norwegian language, the lectures and seminars will be held in English.

Table 2 – Overview of groups, customers and advisors

Num.	Assignment	Customer contact	Group advisor
1	Gridmedia Technologies AS: Geelix 3D Mesh Viewer System	Ole-Ivar Holthe, ole@gridmedia.com	Wei Wei
2	Schlumberger: Spatial Gesture Investigation	Floyd Broussard, fbroussard@slb.com Michael James Moody, mmoody@slb.com	Wei Wei
3	COPD@Home welfare technology for chronically ill at home, connected to intermunicipal health and welfare services	Marius Mikalsen, marius.mikalsen@sintef.no	Andreas D. Landmark
4	SINTEF ICT: Privacy protection for information control	Inger Anne Tøndel, inger.a.tondel@sintef.no Åsmund Ahlmann Nyre, asmund.a.nyre@sintef.no	Tosin D. Oyetoyan
5	Institute for Energy Technology (IFE): ToSS - Tool to Support Traceability of Safety Systems	Vikash Katta, vikash.katta@hrp.no	Mohsen Anvaari
6	SINTEF IKT and Trondheim commune: Social network for elder	Anders Kofod Petersen, akof@sintef.no Bjørn Magnus Mathisen, BjornMagnus.Mathisen@sintef.no	Anh Nguyen Duc
7	IDI at NTNU: Interactive Door Sign	Torstein Hjelle, torstein.hjelle@idi.ntnu.no	Muhammad Asif

8	Artsdatabanken: Mobile application for reporting and use of digital identification keys	Nils Valland, nilsvalland@artsdatabanken.no Askild Olsen, askild.olsen@artsdatabanken.no	Muhammad Asif
9	Thales Norway AS: Wireshark: Automated generation of protocol dissectors	Christian Tellefsen, christian.tellefsen (aa) thalesgroup.com Stig Bjørlykke, stig.bjorlykke (aa) thalesgroup.com	Daniela S. Cruzes
10	Norwegian Public Roads Administration, ITS: Hardware fault monitoring and notification for roadside infrastructure	Gryteselv Kristin, kristin.gryteselv@vegvesen.no Jo Skjermo, jo.skjermo@vegvesen.no	Reidar Conradi
11	Norsk Tillitsmann Stamdata AS: Community driven league tables	Thies Schrader, schrader@trustee.no	Anh Nguyen Duc
12	Q-Free ASA, R&D: Realtime Component Performance Monitoring	Morten Tokle, morten.tokle@q-free.com	Tosin D. Oyetoyan
13	NTNU: Samlingssøk	Seksjonsleder Torkild Bakken, Vitenskapsmuseet, NTNU Seniorrådgiver Carl-Fredrik Sørensen, NTNU IT	Mohsen Anvaari

3.4 Rating of project work

The project work will be evaluated based on the quality of the project report and presentation delivered at the end of the course and the students' reflections on the project work:

- The project report and presentation will be 95% of the marks.
- The reflection report will contribute to 5% of the marks.

The project report must be written in English, and the presentation must be done in English,. Both the project report and the presentation count towards the grade in an integrated way (they are not formally weighted against each other).

How the group actually has worked, technical problems, customer behaviour and availability, etc. are a part of the reflections report . The group is asked to deliver a 1-2 page report, reflecting upon their experiences during the project and reflecting upon what they had learned and how they could have done things differently. *The focus of this part is on the process rather than the product.* This report is due at the end of the course and will be evaluated by the group advisor and the course coordinator.

The following criteria are evaluated in an integrated way:

- Whether the group has solved the given assignment, according to the customer's objectives of the project.
- Reasonable grounds for decisions taken.
- Logical flow in the report.
- Visibility of limitations done.
- Layout and structure readability.
- The students' ability to reflect on the process during the project.

The criterias are not formally weighted against each other.

Note that since the presentation counts towards the grade, it is important that you maintain a functioning version of your program in case you (the group) appeal the result (grade). If an appeal is

made, you will have to make your presentation for the new examiner, including demonstration of the program.

3.5 Supervision and meetings

Your very first *group-internal meeting* is scheduled for Tuesday 30.8 at 13:15-14:00 (for group room, please consult Appendix H), i.e., just after the plenary kick-off. Each of you should introduce yourself to the others in the group, and try get the group organized for the first *customer meeting* the following two hours (14:15-16:00hrs and in the same room). The challenge is yours!

Furthermore, your group should have a main ***advisor meeting with your advisor once a week***, normally lasting one hour. Such meetings will have a group-specific content, but share a template agenda from Appendix A.7. All written documents for such meetings (agenda, weekly status report, phase-specific documents etc.) must be delivered on paper or by email to the advisors before 14:00 the day before. In Appendix A – Project plan, you will find more information about such meetings. Thus, during the *first, pre-planned* advisor meeting between your group and your advisor on Wednesday 31.8 (see room/time in Appendix I), you will have to agree upon *when and where the weekly advisor meetings* shall take place for the rest of the semester. The group is responsible for booking a meeting room for these meetings (possibly helped by the advisor). During this hour, the advisor will also focus on the teamwork and group dynamics aspects and support you to establish a good group atmosphere.

How to book a room for a meeting:

You can book a room by contacting Ellen Solberg at the IDI information desk. It is recommended that you send an email to Ellen.Solberg@idi.ntnu.no or by phone 73 59 34 40.

It is wise to suggest 2-3 alternatives times for the meeting, because many of the rooms may already be booked. Always give Ellen Solberg a note if you not are using a reserved room.

A suggestion of an email:

*Regarding the “TDT4290 - Customer Driven Project” we would like to book a room for X persons.
We would like one of the following times (in prioritised order)*

1. date, from-to
- 2.
- 3.

Best regards,

...

It is also possible to book some rooms through the room reservation site “Romres” (<https://romres.ntnu.no/>). This reservation page is only accessible from users on the NTNU-intranet.

For reservation of rooms in the Electro buildings, use the page <https://eddis2.ime.ntnu.no/romreservasjon/>

Or contact Hilde Berg in the “corner window” in the Electro building hilde.berg@ime.ntnu.no, phone 73594201

Customer meetings are held when needed, starting on Tuesday 30.8 at 14:15-16:00 (see rooms in Appendix I). The next customer meetings arranged in dialog with the customer, but the group is responsible for booking a room and other logistics. We recommend taking more contact with the customer, *before* the second advisor meeting in the following week.

You will probably also need several weekly, internal group meetings. So try to book a fixed room once or twice a week during the semester.

Note: Before the first advisor meeting on 31.8, the group is collectively responsible for making a written resume of the first customer meeting held on Tuesday 30.8 at 14:00-16:00. This resume should be sent by email to the persons involved (group members, advisor, customer) later on the same day (30.8). So take good notes of this first customer meeting!

3.6 Pre-delivery for examiner, writing teacher and media, on October 6th

You are required to submit a copy of the Abstract, Introduction, the Pre-study and the Choice-of-Lifecycle-model chapters to the *external examiner* (censor) and *technical writing* teacher by 6 October. These chapters are normally not in their final stage and will *not* be used as part of the final assessment. It is only to let the external examiner be better prepared. The technical writing teacher will provide feedback on the chapters.

In addition to the chapters mentioned earlier, the groups are required to include the outline of the full report (Table of Contents). The Table of Contents should not be too detailed, but must contain enough detail to understand how each chapter is structured and what the final focus of each section should be.

Remember to include the *time and place for the final presentation and demo*, and also a short description of the project report (cf. *project abstract*) – i.e. one extra page.

Two printed copies of these pre-reports (with an extra page) must be delivered at the IDI front desk by 12:00 on October 6th (exact location for the delivery will be announced on the course web page).

Use also the opportunity to invite media to the November, 24th event:

Remember that you are the “consultants” that invites to a presentation of your work. You should be proud of your product and give it the publicity it deserves. Create a flyer to catch the intention of the audience, and send this to the customer, the advisors, and others that might be interested - such as local TV/radio, Universitetsavisa, Adresseavisen etc. - well ahead of the presentation.

3.7 Final presentation and demonstration on November 24

The projects will be presented and demonstrated at NTNU on Thursday, November 24, during 9.15 till 16.00

A possible agenda stands in Appendix C.12.

All groups have to make their **presentations in English**.

Room: If the project demonstration requires special facilities (such as virtual reality or cave equipment), the groups can also book and have the presentation in other rooms. If your group needs to have the presentation in a specific room, please notify the course coordinator (Andreas D. Landmark). Remember that the room must have space for 10-15 persons. The time and place for each presentation, will be published on the course webpage.

Laptop: Most groups use one of their personal laptops for the demonstration. If your group do not have a suitable laptop for the presentation, please notice drift<##>idi.ntnu.no two weeks before the presentation. A **beamer** will be made available in all presentation rooms.

Copying project documents: We want **four** printed and bound copies of the project report. The costs for copying and binding four complete project reports are covered by IDI. Copying should be carried out, at the latest, one day before the final presentation (e.g. on Nov. 23 or before). Make sure to book the main copier machine in ITV-154 at least 14 days ahead of the copying. The booking can be done at the IDI information desk.

Delivery of final report: Four bounded paper copies of the final report (with implementations on attached CDs / DVDs) should be delivered on the same day as the presentation. The customer should get one of the copies, and the three others should be delivered at the IDI information desk. If the information desk is closed at the time your presentation is finished, contact the practical course coordinator.

The four printed copies of the report will be distributed to:

- Customer
- Examiner
- IDI archive
- Advisor

In addition, the course coordinator should also receive a digital single-file, **.pdf-format** copy of the entire final report by as an e-mail attachment.

On the CDs / DVDs, you should include the final report as a .pdf-file, with relevant implementations enclosed (source code etc.). All this should be documented by a Readme.txt file.

3.8 Anti-plagiarism

The rules for this are very strict, see §36 in "Forskrift om studier ved NTNU" (page 23 in "Studiehåndbok for Sivilingeniørstudiet 2011-12") regarding cheating and <http://www.lovddata.no/all/hl-20050401-015.html#4-7>

See also <http://www.idi.ntnu.no/grupper/su/publ/ese/plagiarism.html>.

3.9 Copyright or Intellectual Property Rights (IPR)

For the entire lifetime of this course, it has been "unclear", although rather frictionless, which "legal person" actually owned the IPR for the produced work, typically a project report and associated software. The Norwegian copyright law stands in LOV-1961-05-12-2 (<http://www.lovddata.no/all/nl-19610512-002.html>), and follows the *Berne Convention for the Protection of Literary and Artistic Work from 1886*.

Note: In Norway there is no need for a © symbol as in USA. Patents on software does not apply in Europe, but can be awarded in USA. Your employer owns the copyrights for software written as part of your employment contract (EØS rule).

IPR can be dealt with in four ways:

1. Just letting **all parties** (students, advisors, companies, anybody) **do whatever they want with the work and with no explicit rules**, just as now and compliant with the principle that scientific results (including implementation) shall be open and free of charge to everybody. However, legally and by default (Berne convention), the copyright (or IPR) belongs to those (the students) that have their names on the front page of the actual work.
2. Modifying NTNU's new IPR policy, aimed at covering **master's theses** - alone or in a group - but not **project (i.e. pre-master) reports**, see <http://www.ntnu.no/studieavd/skjema/standardavtale07e.doc>, 2p, dated 2007-08-28.
3. Using a revised, **liberal and BSD-inspired IPR policy** (about free and open software), formulated and adapted by Reidar Conradi in August 2008: <http://www.idi.ntnu.no/grupper/su/publ/ese/new-standardavtale08e.doc>. It is a combination of point 1) and 2).
4. **Applying NTNU's new IPR policy** (also) for the TDT4290 project course, possibly revising point 2): The original IPR policy proposal was drafted by NTNU's legal advisor Morten Øien, but withdrawn at its board meeting on June 12, 2008, awaiting vital clarifications, see http://www.ntnu.no/styret/saker_prot/12.06.08web/46.08_vedl.pdf, 12p. NTNU's IPR policy

was finally **approved** at its **board meeting on June 6, 2010**; see NTNU's S-sak 36/10: http://www.ntnu.no/styret/saker_prot/09.06.10web/36.10.pdf, 12p.

NTNU generally tries to coordinate its IPR policy with that of the University of Oslo from October 19, 2010: <http://www.uio.no/for-ansatte/arbeidsstotte/fa/kontraktinngaaelse/ipr-politikk-191010.pdf>, 16p. See also proposed extensions from the "Sejersted-II committee" at University of Oslo from May 20, 2011: http://www.hf.uio.no/imv/om/dok/2011/instituttstyret/SAK192011Høringsnotat_fra_rektor.pdf, 5p.

Check also what law professor Olav Torvund from University of Oslo has written in his interesting blog on IPRs and other issues: <http://blogg.torvund.net/>. See finally some overall comments from June 2008 by Reidar Conradi in: <http://www.idi.ntnu.no/~conradi/IT-debate/ip-politikk-ntnu-26jun08.html>, 6p.

3.10 Course reflection, evaluation and feedback

We intend to do a systematic evaluation of this project course. For this purpose, a "student reference group" must be established among the course participants. The course will be evaluated in the following ways:

- **Student surveys:** individual students will be asked to fill in a questionnaire at the beginning and at the end of the course. The questionnaires at the beginning of the course will be used to gather data on the students' expectations and the questionnaires at the end will be used to gather data on if the students' expectations have been met and other relevant feedback from the students.
- **Mid-term assessment by advisors:** the advisors will be asked to provide a mid-term assessment on how their project group(s) are progressing and notify if they foresee any challenges with their groups.
- **Customer feedback:** Customers will be asked to fill in a short questionnaire at the beginning and at the end of the course. These will be used to gather data on if and how the customers' expectations have been met and to gauge customer satisfaction with their project group and the course in general.

The feedback received from the different parties will be used to improve the course for the future students.

Appendix A – The project plan

This section gives an example of how to structure a project plan. The project plan is a dynamic document that will evolve and change throughout the whole project. The project plan regulates the administrative part of the project and guides the project.

Depending on the type of lifecycle model you use you will have to structure the project plan differently.

A1. Overall project plan

Recommended content of the project plan (“project directive”):

- **Project name**
- **Project sponsor** (customer)
- **Partners including responsible third party providers**
- **Background for the project:** software system development
- **Measurement of project effects, i.e. goals like:**
 - Reduce the time it takes to create a daily production report with 3 hours...
 - 30% cost reduction of...
 - 40% increase in sales...
 - etc.

The effect measures are typically stated by the project sponsor, but it is likely that the group has to take the initiative to specify these in detail.

- **General terms.** What are your limitations, tool selections, organizational demands from the customer, resources etc.?
- **Based on the planned effort:** How many person-hours are to be used?
- **Schedule of results.** When should deliverables be available as milestones or sprints/iterations?

A2. Concrete project work plan

Recommended content:

- phases/sprints
- activities
- milestones
- person-hours per activity and phase + lectures + project management

The project plan (in form of a Gantt diagram) can be attached as appendices.

It is also recommended to attach the detailed plan of the phases as appendices.

The workload (measured in %) of this course is normally scattered out on different project phases, as shown in table 1. A suggestion of a relative workload is found in column “Norm”, while the experiences from groups 1 to 7 in 1997 are found in the subsequent columns. As you can see from the table, the relative workload varies a lot from group to group. This is a normal variation and is caused by the different assignments, the groups working differently, and that it is not strictly defined what each phase should contain. The suggestions given in “Norm” are a good starting point for the project plan.

For projects based on iterative development like Scrum, you will have to incorporate the phases which are needed into each sprint into the documents for each sprint.

Phase\ Share in %	Norm	Gr.1	Gr.2	Gr.3	Gr.4	Gr.5	Gr.6	Gr.7
Project management	10	-	-	-	-	-	-	-
Lectures and self study	10	-	-	-	-	-	-	-
Planning	7	9	5	9	9	4	5	6
Pre study	15	24	14	26	21	17	22	38
Requirements specification	20	26	34	25	25	18	24	24
Design	15	27	19	15	18	22	19	9
Programming and documentation	13	6	25	11	14	31	23	14
Project evaluation	5	2	2	7	5	3	2	4
Presentation and demonstration	5	6	1	7	8	5	5	5

Table 1: Relative workload for waterfall-like project.

Ideas to the content of the different phases are found in Appendix B. It is also recommended to look at previous project reports, which can be found at the web site of the course.

A3. Project organization

Recommended content:

- An organizational diagram of how the group is organized
- Roles, i.e., project leader, system analysis, system architect, system designer, test leader, customer contact, QA responsible, etc. Try to be inventive in role allocation!
- Responsibilities of the different roles
- Weekly schedule

A4. Templates and standards

The group should create templates for all relevant document types. Even though it will take some time to create these in the beginning, the group will benefit from these in two ways: 1) the layout will be correct when creating project documents and 2) reduction of irritation and stress within the group.

Templates ought to be made for:

- phase documents
- agenda for meetings
- weekly status reports for the advisor meetings
- etc.

The group should also create pragmatic standards for:

- organisation of files
- naming of files
- coding style
- etc.

A5. Version control procedures

The group must create a *systematic procedure* for version control for *all* textual documents, source code, etc., see Appendix D.4.2.3 on actual tools like CVS, SVN, Make etc.

A6. Documentation of project work

Internal project meetings

Try to have internal meetings at least once per week. In these meetings you should present the status, coordinate activities, divide tasks, and check the “mood” of the project. Set up an agenda and write precise minutes for each meeting.

Internal reports

A typical internal report:

- Person-hours - used and remaining (according to plan)
- Activities - done and remaining
- Achieved and not achieved milestones. These are important progress indicators.

Reporting of person-hours used should be done in written form to a specified time, e.g. as a part of the weekly reports.

A7. Quality Assurance (QA)

QA assumes that the relevant product qualities have been identified, so that the development process can be tailored to achieve these, e.g. reliability, performance, usefulness etc.

There exists an ISO-standard for this (ISO 9126). More information can be found on Wikipedia, see http://en.wikipedia.org/wiki/ISO_9126

Time of response

Make agreements with the customer. There should be time of response on:

- Approval of minutes of customer meeting (e.g. 24 hours)
- Feedback on phase documents the customer would like for review (max 48 timer)
- Approval of phase documents (max 48 hours)
- Answer to a question (e.g. 24 hours)
- To get agreed documents etc (e.g. 24 hours)
- Other

Routines for producing high quality internally

This has something to do with how you organize the specification and programming work, e.g. user involvement, “pair programming”, design examination, etc. The number of people involved should be weighed against available resources.

Routines for approval of phase documents

Specify how you are going to approve the phase results (deliverable), which mainly consists of the phase documents. It is natural to involve the customer in the approval of pre-studies and requirements specification (or whatever you might call these documents). You must, as stated earlier, agree upon a time of response with the customer.

Calling for a meeting with the customer

For all the meetings with the customer you should send a call for the meeting, specifying time, place, intention (result), agenda, and background documents. It is vital to specify what preparations you expect of the customer and the group before the meeting.

You have to agree with the customer how long in advance the calling for meeting should be sent, e.g. at 12:00 two working days before the meeting is going to take place.

Minutes of a customer meeting

You must write a summary of the meetings with the customer. It is vital that you write down decisions, actions (what, who, and deadline), clarifications etc. that are important for further work in the project. The customer must approve the minutes of the meeting, to make sure there where no misunderstanding of decisions made etc. The minutes of meetings are part of the “contract” with the customer. In normal working life it is not uncommon that the minutes of meetings are part of the contract document with the customer.

In the project plan you should specify when the summary of the meeting should be done, when it is to be given to the customer for approval, how to distribute (e-mail, fax etc.) and expected time of response from the customer. It ought to be written by 12:00 o'clock the following day and should be distributed as soon as possible when you are done with internal approval within the group. It is vital that you get an approval as soon as possible to avoid misunderstandings.

Calling for the weekly advisor meeting between the group and its advisors

12:00 o'clock the day before the meeting, with print out of all needed paper.

Agenda for the weekly meeting with the advisor - a template that is to be followed

1. Approval of agenda
2. Approval of minutes of meeting from last advisor meeting
3. Comments to the minutes from last customer meeting or other meetings
4. Approval of the status report, which may be structured as follows:
 - 4.1 Summary
 - 4.2 Work done in this period
 - Status of the documents that are being created
 - Meetings
 - Other activities
 - 4.3 Problems – what is interfering with the progress or taking resources? Problems are often risks that have taken effect.
 - 4.4 Planning of work for the next period
 - Meetings
 - Activities
 - 4.5 Other
5. Review/approval of attached phase documents
6. Other issues are listed here...
7. Other issues

The status report (see Section 4.1-4.5 above) should be handed in as a separate document.

Minutes of the weekly meeting with the advisors

Is attached to the next calling for meeting and is a fixed subject on the agenda.

A8. Test plan

The project has to have an overall test plan, which either can be part of the project plan or as a test document (the latter is recommended, see Appendix C6).

Appendix B – Suggestion for appendices in your project plan

B1. Partners

Owners, target audience, customer representative(s), project group, advisors. For each person, record:

- name
- address
- phone number
- e-mail
- etc.

B2. Concrete project plan

The current project plan and old project plans. By also keeping the old plans the group can see how they have evolved and also possible learn from previous experience.

B3. Detailed phase plans

A detailed description of what each phase consists of.

When ending a phase, the next phase is fine planned in detail. The detailed plans for each phase are put here and not in the end of last phase document.

B4. Table for handling of risks

Project nn

Nr	Activity	Risk factor	Consequents	Probability.	Strategy and actions	Deadline	Responsible
	Which of the activities of the project are affected	Catching the name of the risk factors	Start with H , M or L before describing the consequences	H, M or L	Select strategy: Avoid, Transfer, Reduce, or Accept. Then on the next lines describe the measures	Set a clear deadline for ...	Give one person the responsibility
1	All	Hans is involved in UKA	H: The quality of the project results will decrease	M	Reduce Assign delimited tasks to Hans with clear deadlines	Continues	Project leader

L = Low, M = Medium, H = High

B5. Table for effort registration

All projects need to register the effort spent by each project participant on the different activities (e.g. Prestudy, Programming etc.) and in what period (week 1, week 2 etc.). This is needed to ensure that the project is on track according to the project plan. A weekly registration or periodization is common.

So *each* of you must *weekly* report - in a so-called *time sheet* – *seven data items per relevant activity and period: project group no, person name, date of registration, period no, activity name or id, your effort spent and given in person-hours* (possibly zero).

Make a *template* time-sheet for this information as soon as possible (a textual email-message format will do), and establish reporting procedures from the very project start. The reported effort data should be delivered ca. two days before the weekly advisor meeting. The project manager (or a delegated person) should be responsible to collect and synthesize the individual effort data into an updated *project effort-matrix* on a spread-sheet. The matrix data will be used to regularly monitor the planned (or estimated) effort vs. the actual one for the whole project. This matrix has *time* (period number) as the horizontal dimension, and *activity* as the vertical dimension. Each *matrix cell* contains a number measured in person-hours (ph).

So very early in the project, as part of making a Project Plan, you must *break down* the project's total available or estimated effort (ca. 1700 person-hours) into a dozen main activities or phases, which again are allocated to periods (week no 1-13), cf. Appendix A2. Naturally, activities belonging to the last part of the project cannot be broken down in detail in the start. Thus the project plan must be adjusted over time.

Example: Assume that we have a software project with *three estimated activities (A1-A3) over three time periods (T1-T3)*:

- A1. Prestudy, whose estimated effort is 40 ph (person-hours).
- A2. Requirements, with 40 ph.
- A3. Implementation, with 20 ph.
- A. Total of 100 ph.

The project has an unspecified number of participants, so our project manager must keep track of the total resource usage (effort, time).

Version 1: Initial effort-matrix with very uneven effort estimates in the three periods:

Group no: ...				
Date: ...				
<i>Activity \ Period</i>	T1	T2	T3	Activity sums
<i>A1. Pre-study</i>	40			40
<i>A2. Requirement</i>		40		40
<i>A3. Implementation</i>			20	20
<i>Period sums</i>	40	40	20	100

Comment: It makes sense to overlap the three activities a bit, to get a more even effort distribution over the three periods.

Version 2: Reconciled matrix version, where the three “**diagonal**” ph-estimates (40, 40, 20) are spread out to get a more even effort distribution over time - please discuss the revised ph-estimates

Group no: ...				
Date: ...				
Activity\ Period	T1	T2	T3	Activity sums
<i>A1. Pre-study</i>	20	10	10	40
<i>A2. Requirement</i>	10	20	10	40
<i>A3. Implementation</i>	0 (OK)	5	15	20
Period sums	30	35	35	100

Version 3: Now introducing **estimated** (E:) vs. **actual** (A:) effort per period (T1-T3), both per **running** period (as above) and **accumulated** over several periods (see after the “/”-symbol in the below effort-matrix):

Group no: ...						
Date: ...						
Activity\Period	Start	T1	T2	T3	Activity sums	Activity comments
<i>A1. Pre-study</i>	E:20 A:0	E: 20/20 A: 13/13	E: 10/30 A:12/25	E: 10/40 A: ??/??	E: */40 A:
<i>A2. Requirement</i>	E=10 A=0	E: 10/10 A: 11/11	E: 20/30 A: 19/30	E: 10/40 A: ??/??	E: */40 A:
<i>A3. Implementation</i>	E=0 A=0	E: 0/ 0 A: 2/ 2	E: 5/ 5 A: 7/ 9	E: 15/20 A: ??/??	E: */20 A:
Period sums	E=30 A=0	E:30/30 A: 26/26	E: 35/65 A: 38/64	E:35/100 A: ??/??	E: */100 A:
Period comments		A1 delayed A3 before	A1 delayed A3 before	

* means irrelevant

Let us assume that two time periods (T1-T2) have passed, with T3 just about to start.

Observation: in activity A1 after time T2 the Estimated running effort is 10 ph and the estimated accumulated effort (i.e. including T1) is 20+10 = 30 ph. However, the Actual effort for A1/T2 is 12 ph, and the accumulated effort is 13+12 = 25 ph. So it seems that A1 is a bit behind the estimated effort (“plan”) – but that can have many valid reasons. We are only measuring resource usage (effort, time), not the actual state of the software under development!

Ex. what advice will you give to all the activities A1-A3 for the last T3 period?

Appendix C – Content of the phase/sprint documents/chapters

This appendix contains information about what the different phase documents (or report chapters) should include. See also former project reports for more details.

That is, it is common to divide a software project into the following 13 (or so) phases, whose documentation then becomes a chapter in your final project report:

- C1. Introduction
- C2. Planning
- C3. Pre-study of the problem space vs. solution space
- C4. Choice of lifecycle-model: waterfall vs. agile?
- C5. Requirements specifications
- C6. Estimation of realization effort for use-case model
- C7. Construction/ design
- C8. Programming
- C9. Testing
- C10. Documentation
- C11. Evaluation
- C12. Presentation and demonstration
- C13. Appendices

If an agile lifecycle model is chosen, like in Scrum, we recommend one “phase” (or report chapter) per Scrum-sprint, typically 2-3 such. These “Scrum phases” will then replace the waterfall phases C5-C10.

Below you will find some more information on what each phase document is expected to contain. Note that the group can also select another phasing.

C1. Introduction

Write a good, one-page *abstract* early, and explain the overall context, motivation, demands and results.

C2. Planning

See appendix A – Project plan.

C3. Pre-study of the problem space vs. solution space

The preliminary studies are vital for the group to obtain a good understanding of the total problem. Here, you will have to describe the problem at hand. You should describe the current system and the planned solutions (text, workflow, use-case scenarios, information flow, and other graphical presentations you can use). It is all about getting a good understanding of the challenges ahead!

The group should investigate if existing and potentially competing solutions exists on the market. If such solutions exist, they should be described. You should also describe alternative solutions that fully or partially require custom implementations. The group must also set up evaluation criteria which forms the basis for choice of a solution. Software by 3rd party software providers (as OSS or COTS) should be actively pursued as candidates for *implementation* of large parts of your software system, see <http://sourceforge.net>

In cases where existing components can be applied as modules in the project solution, a simple cost-benefit analysis should be carried out.

Summary:

- Describe the main business requirements, both functional and non-functional, that will constitute the requirements for the final solution and its functionality. These requirements will later form the base for later formalization of requirements. Try also to make *use-case* diagrams to express the major functional requirements, cf. the simple effort-estimation method in Appendix F.
- Describe the situation and solutions of today (“as-is”)
- Describe the wanted situation and its possible solutions (“to-be”)
- Evaluation criteria
- Market investigations
- Description of alternative solutions
- Evaluation of alternative solutions, including adjusted requirements and potential costs and benefits.
- Choice of solution, in dialog with customer.

To conclude, the pre-study should have two main deliveries:

- A (partly) prioritized set of *requirements* – cf. the Scrum “backlog”.
- A proposed *system architecture* to enable a fast break-down (modularization) of the technical work

This is also the time to revise your project plan, possible due to use-case based estimation or clear choices of 3rd party software.

C4. Lifecycle model: waterfall vs. agile?

For the realization of the recommended requirements above, the groups are fairly free to choose between a waterfall or an agile lifecycle model to make a prototype for this project. However, some of the assignments do require use of agile development methods, and the customer will anyhow like to have his/her hand on the steering wheel.

The concept of waterfall development and its properties are considered known from previous courses, and are not covered here. See also phase descriptions in sections C5-C10 below.

Use of the agile methodology called Scrum is covered briefly in Appendix E and by the lecture on September 17th.

C5. Requirements Specifications

In the requirements specification phase, it is important to explicitly state the system requirements and link them to the business requirements from the pre-study phase. Typically, requirements are divided into functional and non-functional requirements. Structure the requirements such that the presentation is well organized.

Some persons like to enumerate requirements (R1, R2 ...), which may create “boring” reading where it is easy to lose track of the content. The advantage with numbering is that it is then easy to separate the requirements from the rest of the text, each becomes explicit, and you achieve traceability and structure.

Use a lot of figures! Good figures say more than a thousand words. We strongly recommend making *use-case diagrams* here, also because we then can make quick and reliable *estimates* of the ensuing design, programming and test effort (Appendix F).

Before requirements are stabilized, you should give an overview of the software architecture. This means including figures that show how separate modules are related.

Prototype-diagrams of the user interfaces are often very helpful for communicating with the customer.

Suggested outline of a more detailed Requirements specifications

The IEEE (Institute of Electrical and Electronics Engineers) have made a good recommendation paper on recommended practices for writing requirements specifications which can be found on; (<http://ieeexplore.ieee.org/iel4/5841/15571/00720574.pdf>). If you are using this recommendation, please note that it is designed as a general purpose software requirement specification plan and that some parts may not be relevant for your project.

The outline of the IEEE software requirement specification is:

Table of Contents

1. Introduction
 - 1.1 Purpose
 - 1.2 Scope
 - 1.3 Definitions, acronyms, and abbreviations
 - 1.4 References
 - 1.5 Overview
 2. Overall description
 - 2.1 Product perspective
 - 2.2 Product functions
 - 2.3 User characteristic
 - 2.4 Constraints
 - 2.5 Assumptions and dependencies
 3. Specific non-functional requirements (e.g. performance requirements, database requirements, security, reliability etc.)
- Appendices
Index
Glossary

C6. Estimation of realization effort of a use-case model

For TDT4290 Customer-driven Project, IDI, NTNU - by Reidar Conradi, Aug. 2011.
See also Appendix F for an extended presentation; here comes the short one.

Use-case based effort estimation is a very simple and cheap method to estimate the remaining *implementation effort* of a *requirements specification*, when the latter is expressed as a set of "upgraded" *use-case diagrams*. This means that a *textual specification* (resembling "pseudo-code") must be added to the graphical actor and use-case models. The actors and use-cases must then be categorised as Simple, Average, and Complex - by you.

An *effort estimate* in *person-hours* can now be calculated, by you or by a spread-sheet tool (www.idi.ntnu.no/grupper/su/publ/reidar/uc-ProjectEstimateMethod-2011-v2.xls). It typically takes 30 minutes for you to execute the entire method, given that "upgraded" use-case diagrams are available. *And only six "numbers" need to be given by you!*

Example: Appendix F contains a small use-case diagram with 3 actors and 10 use-cases. All 3 actors are termed Complex. Further, 2 of the use-cases are termed Simple, 3 are Average, and 5 are Complex. We then multiply the actor numbers with the “cost points” 1,2,3 for respectively Simple, Average, Complex actors, i.e. $1*0 + 2*0 + 3*3$ - or totally 9 actor points. We similarly multiply the use-case numbers with the cost points 5,10 and 15 for respectively Simple, Average, Complex use-cases, i.e. $5*2 + 10*3 + 15*5$ or totally 115 cost points. The sum is $9 + 115 = 124$ cost points, which should be multiplied by a *Productivity* factor to get real person-hours. For small systems like yours, this factor can be set to 10. Thus, the estimated implementation effort becomes $124*10 = ca. 1250$ person-hours (+- 20%)!

C7. Construction / Design

Construction is all about getting a vague system description, to a specific and detailed description that can be implemented and realized.

A client-server software system typically includes three tiers:

1. User interface
2. Business or application logic
3. Database or file system

Pseudo-code and UML are useful for describing solutions in the construction phase.

Regardless of which type of development strategy is chosen, (waterfall, incremental) most software implementation projects start with **system architecture** and a sketch of the desired design, in order to ease later division into parts. The project most likely will be further developed later, so a “**modular**” design is to prefer. Designing a modular system also makes testing much more easily, since defects can be tracked down to individual modules.

C8. Programming

In the very beginning of the project a lot of important technical decisions have to be made. This is vital for the remainder of the project, but there are certain practices that make it all easier.

General knowledge about programming is expected to be covered in previous courses. Depending on the actual project commitment, this project might require that the team members learn new programming language(s), new concepts of programming, various technical skills etc. The group have to plan how to obtain this knowledge, maybe in cooperation with the customer and the advisor.

Best practices

The code you write might be used as a base for further development, and may also be used by the other team members.

Inside the group, it will also be practical to have common design and code conventions that all group members understand and practice. If the customer has a coding convention, they probably would like you to use it also.

Knowledge to and use of patterns is also useful when it comes to programming. This is briefly covered in previous courses, but examples of patterns from architecture and coding are found practically anywhere on the internet. A good book resource on patterns is “Design Patterns: Elements

of Reusable Object-Oriented Software”, see also:
[http://en.wikipedia.org/wiki/Gang_of_Four_\(software\)](http://en.wikipedia.org/wiki/Gang_of_Four_(software))

Legal issues

Please observe that some freeware or trialware licenses of code editors etc. states that is it prohibited to use them to write code for commercial use. Check the license of the software that you decide to use in the project, and discuss it with the customer if there are such clauses that you might be in conflict with.

Technical support tools

Versioning control and backup CVS / SVN tools should, as mentioned in Appendix A5, be used for all technical artefacts (UML-diagrams, code, test data, etc.) and documents in the project. When you set up a development environment, make sure to set up a functional versioning system with backup as well. You never know when you might need it! It is possible to use CVS/SVN on IDI, for more info, contact the Gurutjenesten in the P-15 building.

Coding style

How to write source code should also be specified. Such documentation should typically contain:

- Programming conventions, e.g. in use by the customer
- Standards for commenting source code.
- Show examples of source for how the programming conventions look like in practise.

The source code should be commented and documented so well, that the customer easily can make modifications and build on your work after the project is finished.

At the end of your project, the source code and necessary resources should be included on the CD / DVD attached to the project report, supplemented with a Readme.txt file.

C9. Testing

Testing is usually planned and carried out in five parts:

1. Overall test plan – This should be created as the last part of the requirement specification phase.
2. A plan for each test that need to be carried out. – This should be done in the end of the construction phase.
3. Creation of detailed test specifications or checklists for each test. – This should be done in the end of the construction phase.
4. Execution of tests, including correction of defects, re-testing and documentation of test results.
5. Approval of test results

When you create a test plan it is important to specify:

- Which tests should be carried out?
- Which tests should contain checklists? (Checklists are most common for entity and module testing)
- Which tests should contain detailed specifications? (Detailed test specifications are common for testing systems, integrations, usability and acceptance)
- Who are the test persons? (Project, customer, others,..)
- When should the tests be carried out?
- Who are responsible for carrying out detailed test specifications and check lists?

The level of detail should fit the nature of your project.

The detailed test specifications should contain:

- Test descriptions (the operations that should be carried out)
- Data that will be tested (input and expected output)

Tests carried out	Description
Unit test (programming phase)	Testing of the smallest units in the project, i.e., user interface, methods, stored procedures, objects, classes, etc.
Modul test (design phase)	Entities integrated into bigger software components. Modules are tested to assure that the coordination and communication between the entities are as expected.
System test (requirement phase)	All modules that together form a complete version of the system should be tested. The systems are tested to assure that the coordination and communication between models are as expected.
Integration test (design pahse)	This is a complete test of the system and its interfaces to the world around. The last defects should be found and it should be verified that the system behaves well according to the requirement specifications. In some projects integration and system tests are merged.
Tests carried out by the end users	
Usability tests (non functional)	These are tests that assure that the interaction between users and the system is as expected. The goal is to get user friendly applications.
Acceptance tests (non functional)	Here, the end users should test if the system and its user interface to its environment are as expected. Based on this acceptance test, the management or customer make decisions on whether the product should be used or not.

When testing, you should perform the test after completing a phase/design/component etc. This is very close to the V-model used in software development. For more information about this, the Wikipedia article gives a good introduction ([http://en.wikipedia.org/wiki/V-Model_\(software_development\)](http://en.wikipedia.org/wiki/V-Model_(software_development)))).

C10. Internal and external documentation

A user- and installation-guide for the final product must also be created. It should include an installation guide, which describe the installation process step by step.

Note, that your system and its installation will be tested by your advisor.

Hint: start on the documentation as soon as possible, as it describes the current state of the project for the project team.

C11. Evaluation

The groups decide themselves what to include in the project evaluation, but we recommend including the following elements:

1. The internal process and results: How have you worked together as a team? What have you done well? What have you not done so well? What would you have done differently? Conflicts that arose and how these were handled? Did you reach the project goals? What did you learn?

2. The customer and the project task: How was the communication with the customer? How did you experience the project assignment?
3. The advisors: How was communication with the advisors? Was the supervision good enough? How could the course be improved to next year?
4. Further work: Give an estimate for how much effort that is necessary to complete the product / project.
5. Suggestions for improvement. What is missing to make this course better for both students, customers and advisors.

It is important to describe hidden problems that can have affected the work but is not shown in the project report. Make sure that you also describe any additional work that is not shown in the project report.

C12. Project presentation and demonstration

To be used in Section 3.7.

The start-up is divided into three parts:

1. Presentation

Explain the project assignment and goals

Problems and priorities

Which solution did you choose? What were the alternatives? Why did you choose the final solution?

A description of the final solution

Some final reflections

2. Demo

Show your implemented prototype and its main functionalities.

3. Questions: max 5 minutes

Total: 45 minutes

Remember to give a hand-out of the presentation to your advisor, customer and external examiner (censor) at the presentation.

C13. Appendices

These may - for instance – include the following, but see also section **C10** above:

1. User and installation guides (cf. external documentation?)
2. Technical/internal documents (cf. internal documentation?)
3. Other, e.g. special material provided by the customer.
4. Possible contracts and non-disclosure agreements.

4 APPENDIX D – Administrative and Technical Resources

This chapter presents an overview over the resources available during the project work.

D4.1 Office Resources

Printouts	The group has a quota of 500 pages each.
Photo Copying	All the documents you deliver during the project period can be copied for free at the main copier in IT-154. The secretaries at the information desk will give you the password to operate the copier.
Telephone	If you need to use a phone as a part of you project work, please contact the course coordinator or your advisor.
Telefax	You have access to a fax machine (73 59 44 66) at IDI (ITV-254). Ask at the information desk.
E-mail	<p>Electronic mailing lists have been created for each group to simplify communication.</p> <p>Each group has their own mailing list, named: "kpro1<##>idi.ntnu.no", "kpro2<##>idi.ntnu.no", etc. according to the group number.</p> <p>A mailing list for all students and advisors is also created. This list is named 'kpro-alle<##>idi.ntnu.no'.</p> <p>These lists will simplify communication, and a lot of information will be distributed through these mailing lists. The lists will be in operation from week 2 of the course.</p> <p>The e-mail addresses of all the students (and advisors and lecturers) is listed in this compendium as well.</p>
Homepage	The course webpage contains a lot of important information. It should therefore be checked frequently, see www.idi.ntnu.no/emner/tdt4290/

D4.2 Technical Resources

D4.2.1. Workstations

Fourth year computer science students have access to the 5th floor in the P15 building. Additionally IDI technical group offers virtual servers for all the groups to host/run any special software necessary that is not available on the normal servers available to students.

If technical problems occur or you have particular needs regarding the computers, contact the IDI technical group, drift<##>idi.ntnu.no

D4.2.2. CVS / SVN configuration management tool

CVS and SVN are client /server – systems for versioning control of all types of files. It makes it easy to go back to former versions (with change control), to let multiple persons work on the same document at the same time. Eclipse is an open-source development framework with excellent support for CVS and SVN and makes file sharing and version control easy. If you have questions on using the CVS / SVN capabilities on the IDI servers please contact drift<##>idi.ntnu.no.

It is worth to keep in mind that CVS and SVN both adopts a centralised repository design, a lot of developers are shifting towards distributed repositories due to weaknesses in this design. Alternatives to look at would be Git, Mercurial, Arch and Bazaar (all open source).

D4.2.3. Text processing tools

The groups are free to use any text processing tool for writing the report, typically using MS-word, Open Office or Latex. For those who choose to use Word, the following link could be helpful:
<http://www.ntnu.no/adm/it/dokumenter/brukerstotte/word.pdf>

Although it applies to an older version of Word, the principles are basically the same.

For those who choose to use LaTeX, the following page could be of good help:
<http://www.stud.math.ntnu.no/kurs/htmlkurs/kurs.html>

The course webpage contains also links to useful resources.

D4.2.4. Use of collaboration technology in the project

The groups are encouraged to use some kind of collaboration technology (file sharing, defect registration etc.) to coordinate the different tasks among group the members. There exist several tools for this. The following is a list with some of the more well-known software packages for collaboration and information gathering:

Gathering of static information on the web:

Mediawiki: Open-source program for gathering static information on the web.

Example of use: wikipedia.org

Management software for defects etc:

Trac: Great for registration of defects, things to be done etc. Open-source.

Example of use: <http://trac.edgewall.org/report>

Sharing of files:

Microsoft Sharepoint: For sharing of files, documents etc. Commercial, but is available to NTNU students for free, under special licensing rules.

Example of use: <http://office.microsoft.com/nb-no/sharepointserver/HA101672721044.aspx>

Dropbox is another popular example, <http://www.dropbox.com>

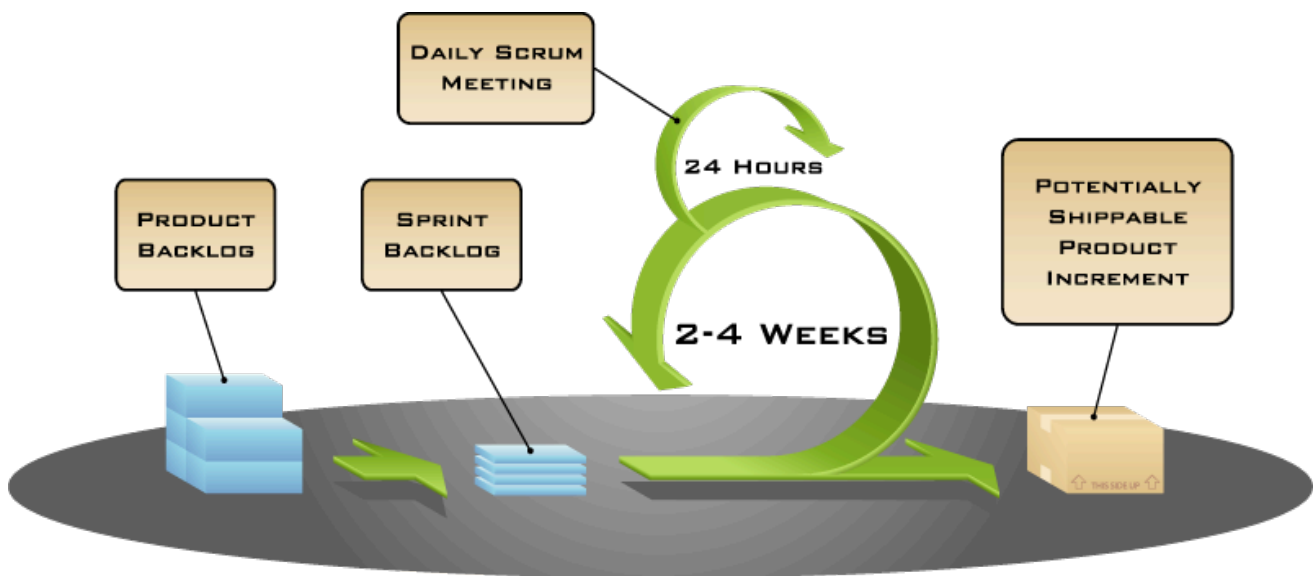
Additionally Stud.ntnu.no offers group areas: <http://www.stud.ntnu.no/kundesenter/>

5 Appendix E – SCRUM – a popular agile method

Written by Torgeir Dingsøy, PhD, SINTEF/IDI; Aug. 2008.

Scrum is an iterative planning and execution method for software development, is based on the “Agile Manifesto” (<http://agilemanifesto.org/>).

Scrum is based a series of increments of **Sprints**, being focused efforts for a limited period toward concrete goals. The concept originates from the mechanical engineering industry. Due to limitations in time in this project, there are not room for more than just a few sprints. Most textbooks and tutorials state that the sprints should take approximately 30 days, sprints of 14 days are becoming more widespread. For a small enough set of tasks, sprints of shorter duration also goes well.



COPYRIGHT © 2005, MOUNTAIN GOAT SOFTWARE

The basic workflow in Scrum development

Drawing taken from: <http://www.mountaingoatsoftware.com/Scrum>

E5.1. Product backlog

The other key concept of Scrum is the product backlog, or just the backlog.

The product backlog is the list containing all functionality in the product. This should be made in the initial phase of the project, but it can be altered and changed throughout the project as the requirements evolve. The backlog must not be mistaken with the requirements specifications.

Consider the backlog as a more informal document.

E5.2. Sprint planning meeting

Before beginning a new sprint you should have a sprint planning meeting. This meeting is attended by the product owner (customer), the Scrum master and the rest of the Scrum team. The meeting could also be open for any interested.

During the meeting the Scrum team and the product owner should come to agreement over which features and functions that have the highest priority. Based on this, the Scrum team should be able to determine which task they will move from the product backlog to the sprint backlog. The Scrum

team and the product owner should collectively define a sprint goal, a short description of what the sprint will achieve. This goal will later be discussed during the sprint review meeting. After the sprint planning meeting, the Scrum team will have to discuss how much they are able to commit during the sprint. This might lead to renegotiation with the product owner, but it will always be up to the team how much they can achieve during the sprint.

The Scrum team organization is well suited for a group of students, because none of the traditional software engineering roles like programmer, architect, designer or tester exist. Instead the Scrum team is focusing on collectively complete the tasks within the sprint.

E5.2.1. Daily Scrum status meeting

The daily Scrum(-meeting) should be held every day during a sprint. Usually these meetings are held in the morning, so that the team members can plan the rest of the day. Anyone can attend these meetings, but others than the team members are only allowed to listen.

The daily Scrum should not be used as a meeting for problem solving; this should rather be discussed after the meeting only by the involved team members.

During the daily Scrum every team member should answer the three following questions:

1. What did you do yesterday?
2. What will you do today?
3. Are there any impediments in your way?

The daily Scrum is not a status update, it is more like a commitment to the other team members of what you will do till the next day, and what you have done since last meeting. The good thing about having a daily Scrum meeting is that it helps the team to see how important these commitments are to themselves and the team.

Since this is a student project, and you all have other courses to attend (and students are not known to work from 08-16 either) it is not always possible to have a daily Scrum meeting every day or in the morning. The team have to work out a solution that every team member feel comfortably and in the same time makes it possible to monitor the progress of each team member.

E5.2.2. Irregularities

Most likely there will sometimes be impediments between the scheduled plans and what is actually committed from the team members. Impediments could have many and various causes but it is the Scrum masters responsibility to resolve them as soon as possible. In cases where the impediments regard the Scrum master, he or she takes responsibility so that someone else is solving them. In more extreme cases (one on the team is not doing his workload, serious illness etc.) the team should contact the advisors of the course.

E5.2.3. Sprint review meeting

After each sprint a sprint review meeting is held. This is an informal meeting, which typically can consist of a demo of the new features made during the sprint. As for the other meetings this one is also open for everyone interested, but the Scrum team, Scrum master and the customer should normally participate this meeting.

During the sprint review meeting the project is assed against the sprint goal.

E5.2.4. How do I prepare a project for Scrum (short tutorial)

First you transcribe the requirements of the requirements specification to a list called the Product Backlog

Then, for each sprint, you make a prioritized list called a Sprint Backlog.

E5.2.4.1. The team

The product owner is usually the contact person with the customer. The product owner administrates the product backlog

The Scrum master is similar to a project manager, but is not the same as a project manager.

E5.2.4.2. During sprints

The first day of a new sprint is used to create and analyse the sprint backlog. This is revised with the product owner before the work begins.

Every day during the sprint, the group have a meeting called the daily sprint, or sometimes the daily stand-up. This meeting should not take long, and should for all members of the group answer the following three questions:

1. What have you done since the last meeting?
2. What will you do between now and the next meeting?
3. Are there any threats, or anything preventing you from doing what you have planned?

E5.2.4.3. Recommended online resources about Scrum

Tutorial on agile development by Geir Ketil Hansen, SINTEF IKT

Software Process Improvement Conference (EuroSPI 2006), October 2006

<http://www.idi.ntnu.no/grupper/su/publ/geirkjetil/eurospi06-slides-agile-hanssen.pdf>

Mountain Goat Software is a training and consultant company that provide several free articles about Scrum:

http://www.mountangoatsoftware.com/Scrum_articles

A brief tutorial on how to prepare a project for Scrum:

http://www.softhouse.se/Uploades/Scrum_eng_webb.pdf

Wikipedia article with several links and citations:

[http://en.wikipedia.org/wiki/Scrum_\(development\)](http://en.wikipedia.org/wiki/Scrum_(development))

6 Appendix F – Use-case based effort estimation

Written for NTNU course TDT4290 Project-Driven Project.

by Bente C. D. Anda, UiO and adapted by Reidar Conradi, IDI in 2001-2011.

www.idi.ntnu.no/grupper/su/publ/reidar/tdt4290-usecase-estim-final-23aug11-rc.doc

F6.1. Introduction to use-case estimation

It is quite hard to get a useful (i.e. reliable) *cost or effort estimate* (measured in system size or work effort) to implement a set of high-level requirements, when almost no design decisions have been made. IBM et al. developed in the 1970s a *Function Point (FP)* method (<http://www.functionpoint.com/>) to convert ER- and flow-diagrams into Function Points (FPs), which can be converted into Lines-Of-Code (LOC), and then person-hours.

The FP method has recently been adapted for use-cases by Univ. of Linköping (Karner 1993), Univ. of Oslo (Anda et al., 2001) and NTNU (Mohagheghi et al., 2005). Use-case based effort estimation aims to predict the *implementation effort - in person-hours - of a requirements specification*, given as a set of textually “upgraded” (also called “structured”) use-case diagrams. Such an estimate covers all remaining *design and programming work, plus unit and module testing*.

Our industrial results have so far demonstrated satisfactory precision (+-20 %) for systems of moderate size, i.e. about 3000 person-hours or 20,000 Java-LOC.

We need your help: to try out and improve this estimation method, which only takes 30 minutes per “upgraded” use-case-diagram. The added text (“pseudo code”) in the diagrams will anyhow be useful during remaining implementation work!

Some background: other estimation methods include the Delphi approach, i.e. letting 3-5 "human experts" try to agree upon a common prediction - or simply use the median. There is also the Work Breakdown into Subsystems, which at least assumes a high-level design specification. Furthermore, estimates done by human experts often fares better than estimates built on formal models. Lastly, Prof. Magne Jørgensen at the Simula Research Laboratory outside Oslo claims that the industrial overrun of cost- or schedule estimates in software projects is about 30% - and has remained so for 30 years (Jørgensen 2005)!

Findings from the literature: An industrial software developer spends 40% of his time on Requirements and Design, 20% on Programming, and 40% on Testing and documentation. Given a work year of 1500 person-hours, he/she will annually produce 4 lines of C-code (C-LOC) per person-hour for mega-LOC systems. This means 6000 C-LOC or 3500 equivalent Java-LOC, or 60 Function Points per year per person. The number of defects "injected" by programmers in this software is about 4 per Function Point, or 40 defects per 1000 C-LOC, or 240 (6*40) defects totally per year. Of these defects, 204 (85 %) are pre-release ones, each costing 1-3 person-hours to correct. However, 36 (15%) of the defects will survive into the first release, and each will later cost 10-30 person-hours to correct (Jones 2007). This means that almost half the programming effort goes to defect correction!

F6.2. More on the estimation method

The idea behind use-case based estimation is to add textual specifications, written in "pseudo-code", to document and thus upgrade the graphical actor and use-case models. This extra textual information is used to classify each upgraded actor and use-case model as either Simple, Average, or

Complex - respectively with "cost points" 1, 2, 3 for actors and 10, 15, 20 cost points for use cases. From the aggregated and weighted cost points, a total effort estimate in real person-hours can be produced.

The estimation method takes only 30 minutes to perform, based upon upgraded use-case diagrams. The estimation method may also use a spreadsheet (www.idi.ntnu.no/grupper/su/publ/reidar/uc-ProjectEstimateMethod-2011-v2.xls), or you can do the conversion yourself with a pen and pencil in the same time.

You should later *compare* ("cross-check") the *effort estimate* against the person-hours *actually* consumed in the total design, programming and testing of each use-case, or accumulated for all the use-cases.

F6.3. A mini-discussion

Challenges: How to achieve incremental maintenance - i.e., not only for the first release? How to handle reuse from 3rd party software providers, such as Open Source Software (OSS) and Commercial-Off-The-shelves Software (COTS)? And how to scale up to larger systems: - student software has Productivity of 10-12, the smallish "office" systems in the Anda paper have a Productivity of 25-30, while the Ericsson telecom systems in the Mohagheghi paper have a Productivity of 60-70.

F6.4. References

(Anda et al., 2001) Bente Anda, Hege Dreiem, Dag I. K. Sjøberg, and Magne Jørgensen: "Estimating Software Development Effort Based on Use Cases-Experiences from Industry", In Martin Gogolla and Cris Kobryn (Eds.): Proc. *4th International Conference on The Unified Modeling Language, Modeling Languages, Concepts, and Tools (UML'2001)*, Toronto, Canada, Oct. 1-5, 2001, Springer Verlag Lecture Notes in Computer Science, Vol. 2185, ISBN 3-540-42667-1, pp. 487-502, DOI: http://dx.doi.org/10.1007/3-540-45441-1_3.

(Jones 1997) Capers Jones: *Software Quality – Analysis and Guidelines for Success*. 1997, Boston, MA: International Thomson Computer Press (a pearl of industrial data for mega-LOC systems).

(Jørgensen 2005) Magne Jørgensen: A review of studies on expert estimation of software development effort. *Journal of Systems and Software* 70(1-2): 37-60 (2005).

(Karner 1993) Gustav Karner: Metrics for Objectory. Diploma thesis, University of Linköping, Sweden. No. LiTHIDA-Ex-9344:21. December 1993.

(Mohagheghi et al., 2005) Parastoo Mohagheghi, Bente Anda, Reidar Conradi: "Effort estimation of use cases for incremental large-ware Development Effort Based on Use Cases-Experiences from Industry", In Gruia-Catalin Roman, William G. Griswold, and Bashar Nuseibeh (Eds.): Proc. 27th International Conference on Software Engineering (ICSE'2005), 15-21 May 2005, St. Louis, Missouri, USA, ACM Press, pp. 303-311, DOI: <http://doi.acm.org/10.1145/1062455.10516>.

F6.5. Appendix: A small use-case diagram, with extra comments

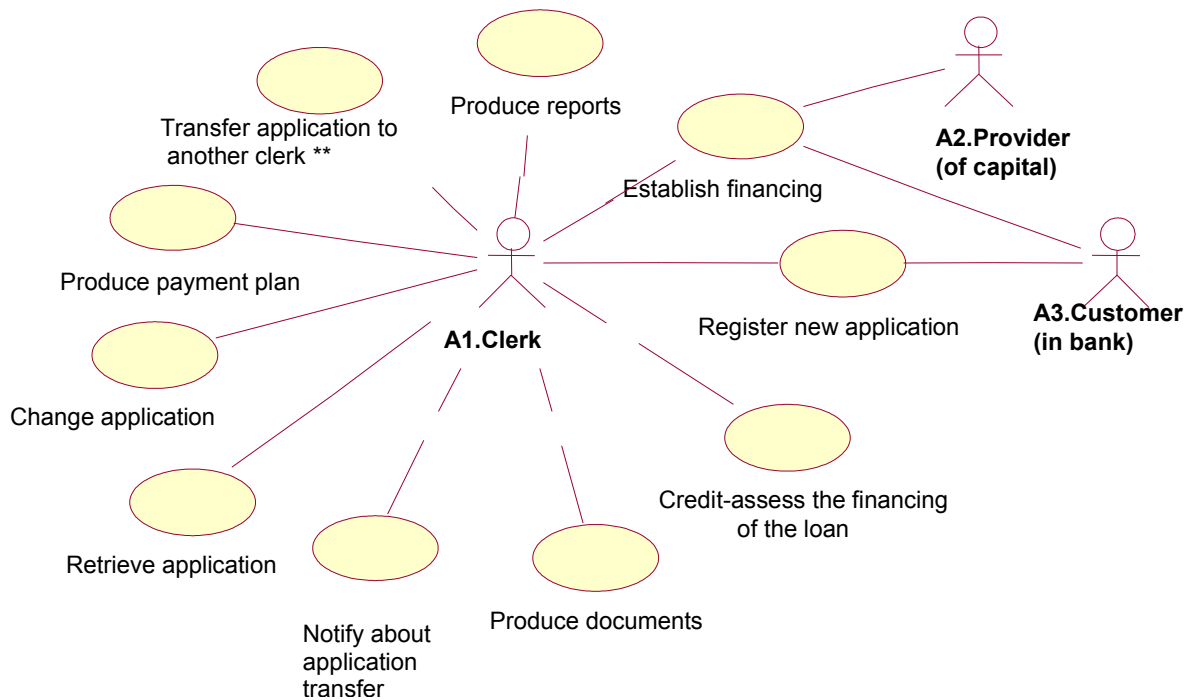
A more detailed explanation of some points in the estimation method stands below

1. *Make first a well-structured, textual requirement specification*, with requirements numbered e.g. as Ra-b.y (a,b:letters, y:digit). This is assumed made on beforehand.
2. *Make a graphical use-case diagram in the usual way*. That is, an actor is depicted as a “match-stick” figure, and a use-case as an “oval”. Lines or arrows between these symbols show relations between the underlying entities.
3. Upgrade the graphical diagrams with textual specification for the actors and users. The actual use-case diagram has 10 use-cases and 3 actors (see example later). You must therefore make textual descriptions for totally 13 entities, see below.
4. For each participating *actor*, categorize him/her as easy, average, or complex:
 - a. A *simple* actor relates to a single system with a defined programming interface (API).
 - b. An *average* actor relates to either another system through a communication protocol, e.g. TCP / IP, or communicates to another actor via a textual interface.
 - c. A *complex* actor communicates via a graphical interface.
 - d. Count up the number of actors in each category.
Ex. Here, the 3 actors are all persons who shall communicate with the system through a graphical interface. Thus, we have 3 *complex actors* (all are complex, so we do not need textual actor descriptions to clarify this).
5. Similarly, for each *use-case*, categorize it as simple, average, or complex, depending on the number of major or alternative transactions in the event flow. A transaction is defined as an event, that occurs between an actor and a system:
 - a. A *simple* use-case has 3 or fewer transactions.
 - b. An *average* use-case has 4 to 7 transactions.
 - c. A *complex* use-case has 8 or more transactions.
 - d. Count up the number of use-cases in each category.
Ex. The use-case "*Transfer application to another clerk*" is one of 10 such, and is marked in the diagram with **. It contains totally 10 transactions -- 7 in normal event flow and 3 in its only variation flow (5a1-5a3). This use-case must therefore be categorized as *complex*. The other 9 use-cases are up to you to categorize. The spread-sheet has tentatively been filled up with 5 *complex* use-cases, 3 *average* and 2 *simple*.
6. *A total of 21 (!!) context factors to express the effect of technical and environmental issues*. These must be provided by you in the general case. In the spread-sheet, all these are just pre-set to 3, i.e. “neutral”, giving an overall multiplicative context factor of 1. So just *drop these!*
7. *Summing up*: We first multiply the actor numbers (0,0,3) with “cost points” 1,2,3 for respectively Simple,Average,Complex actors, i.e. $1*0 + 2*0 + 3*3$ - or totally 9 such points. We then multiply the use-case numbers (2,3,5) with cost points 5,10,15 for respectively Simple,Average,Complex use-cases, i.e. $5*2 + 10*3 + 15*5$ or totally 115 points. The total is $9+115 = 124$ so-called Use-case-Cost-Points (UCP). *Congratulations, you are now done!! That is, only six “numbers” on actors and use-cases must be provided by you!*

8. *Productivity per UCP*: At last, a Productivity factor must be multiplied with UCP above to get an effort estimate in person-hours. With a Productivity of 10, we will get $10 \cdot 124 = \text{ca. } 1250 \text{ person-hours!!}$
9. *Apply* the calculated effort estimate to either:
 - a. *Down-scale the requirements*: Your group project will have a maximum budget of about 1800 person-hours, with perhaps 1/3 or 600 person-hours for implementation. Thus, your requirements must be trimmed to almost half. And this is definitely useful to know beforehand!
 - b. *Start implementing, record the actual effort – and compare with the estimate!*

F6.6. Add-on to use-case example

Graphical use-case diagram, with text specification added for 3 Actors and one Use-case (**).



Example of added textual specification for actor A1.Clerk:

Actor ID:	A1.Clerk
Description: <A short text to characterize the actor>.	Decides who will have a bank loan, specified by a loan Application.
Examples of actions: <...> .	Has a full graphical interface, to help with getting all the details of a loan Application in place.

Example of added textual specification for actor A2.Provider:

Actor ID:	A2.Provider (of capital)
Description: <A short text to characterize the actor>.	The capital part of a bank that actually pays out accepted loans.
Examples of actions: <...>.	Has a full graphical interface, to help with the legal parts of an accepted loan Application.

Example of added textual specification for actor A3.Customer:

Actor ID:	A3.Customer (in bank)
Description: <A short text to characterize the actor>.	Wants a bank loan, as requested by an emitted loan Application.
Examples of actions: <...>.	Has a full graphical interface, to help an actual Customer to fill in the application.

Example of added textual specification for a simple use-case:

Use-case name:	Transfer loan Application to another clerk. **
Actors related:	Clerk (only one actor).
Trigger: <Event that starts this uses-case>.	Some parts of an Application need to be discussed by other clerks.
Pre-conditions: <that must be satisfied <i>before</i> the use-case can <i>start</i> execution>.	The actual clerk must be logged on the system.
Post-conditions: <that must be satisfied <i>before</i> the use-case can <i>finish</i> execution>.	The Application must be stored in the database with a valid and consistent state, and has been assigned to a specific clerk or a group of similar clerks.
Normal event flow: <A list of <i>transactions</i> that will be executed in a <i>normal</i> event flow of the use-case>.	<ol style="list-style-type: none"> 1. The clerk announces that he/she will transfer an Application under treatment to another clerk. 2. The system presents the name of the applicant and the reference number of his/her Application. 3. The clerk checks all this information. 4. The system presents a list over suitable groups of clerks and the customers now assigned to each group. 5. The clerk proposes another clerk, and possibly a second one just in case. 6. The (previous) clerk asks the system to transfer the Application to the proposed clerk. 7. The system transfers the Application to the proposed (new) clerk.
Variations in the event flow: <A list with descriptions of possible <i>transaction variations</i> (not necessarily errors) in the normal flow of the use-case>.	<ol style="list-style-type: none"> 5a. The previous clerk may order the system to generate an email message to notify the new clerk about the transferred Application. <ol style="list-style-type: none"> 5a1. The previous clerk asks for such notification. 5a2. The system generates an email message to the new clerk. 5a3. The use-case continues from point 7.
Related information	

Comments on the description of event flow:

- The event flow shall be described as a list of transactions on the form:
<no> <event description>.
- An <event description> shall be on the form:
<actor> <verb> <direct object> <prepositional sentence>
- Such an <event description> may be one or more of the following:
 - An actor sends a request or data to the system.
 - The system validates the received data.
 - The system changes its internal state.
 - The system responds to the actor with the result.

7 Appendix G – Project assignments proposed by customers

G7.1 Gridmedia Technologies

Title: Geelix 3D Mesh Viewer System

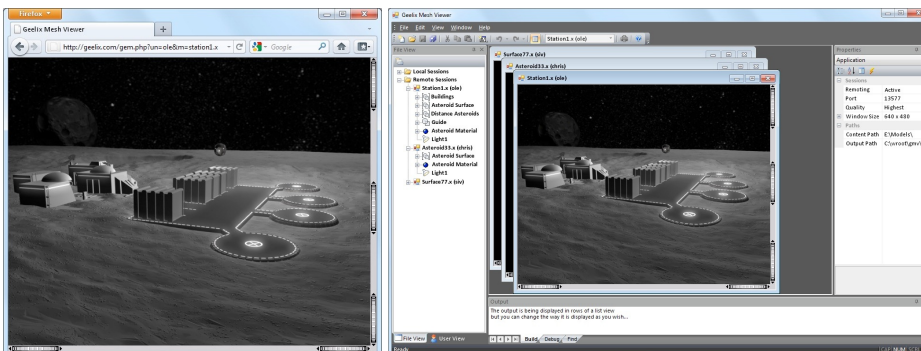
Customer: Gridmedia Technologies AS

Address: PO Box 2642 California City, CA 93504-0642, USA

We wish to develop a system for interactively viewing 3D models over the Internet. The idea is a system that consists of a software application, running on the server-side, that can receive socket commands on a port (i.e. port 13577) to open 3D models, render 3D models and output JPEGs of the rendered 3D models in the web server file directory. Figure 2 illustrates this app.

The system will also consist of a web browser application, as illustrated in Figure 1, that show the pictures generated by the server application. The web viewer (Fig. 1) communicates with the server application using sockets to open a specific 3D model, and view the 3D model from different 3D directions using the rudder-controls. The top-right rudder moves the camera up and down. The center-right rudder moves in and out. The bottom-right rudder rotates up and down. The bottom-left horizontal rudder moves left and right. The bottom-right rudder rotates left and right. URL parameters specify the user name and what 3D model to display.

The server application will be implemented using C++/MFC in Microsoft Visual Studio 2010. The user interface framework, as it appears in Fig. 2, is easily implemented in Visual Studio. The students must implement: loading 3D models (in DirectX x/sdkmesh formats), rendering the models in DirectX 11, 3D camera navigation, and processing socket operations. If there is time, it would be nice to be able to edit the properties of the internal parts of the 3D models, such as textures, shaders, etc. (ref. File View). It would be nice to be able to experiment with using different shaders on the 3D models. The web viewer will be implemented using PHP on the server; and DHTML and JavaScript on the client side. The web viewer will not use any web plug-ins (e.g. Flash). JPEGs will be used for visualizing the 3D model. A new JPEG will be generated when using the rudders.



Contact details:

Name: Ole-Ivar Holthe

Tlf: +1 661 350 1359

Mobile: +1 661 350 1359

E-mail: ole@gridmedia.com

G7.2 Schlumberger

Title: Spatial Gesture Investigation

Customer: Schlumberger ltd.

Address: Risabergveien 3, 4050 Sola

At Schlumberger information solutions (SIS) we develop a large desktop application for the oil and gas industry, namely Petrel (<http://www.slb.com/content/services/software/geo/petrel/.aspx>). This application is used in the exploration process, and helps the users interpret, model and simulate oil and gas scenarios, and more. As it is a modeling application there is a huge focus on interactivity, usability and precision. Per today the application functions in a standard desktop application manner, where the main interfaces are keyboard and mouse. We would like to have a closer look at expanding to the use of touch technology.

Background: Multitouch technology has become widely used in smartphones, tablets and personal media devices. Its prevalence has brought about a change in user perceptiveness for the way they expect to interact with a device. Multitouch in a software product can attract new users; not having it could risk losing existing users. Beyond multitouch is spatial recognition. It has been dramatised by films such as the Minority Report and has been most popularly realised by Microsoft's Kinect controller. It could be the next game changer for software in both corporate and consumer environments.

Goal: We would like to investigate the viability of spatial gestures and their usefulness in a corporate environment.

- Implementing the technology in the Petrel software using the Ocean framework.
- Analyse ergonomic scenarios and evaluate their viability.
- Explore gestures similar to those popularised by multitouch, such as pinch and inertial swipe.
- Demonstrate spatial gesture manipulation of several key objects, such as seismic slices, well control points and the 2D user interface.

Contact details:

Name: Floyd Broussard

Tlf: 1 713 513 8878

Mobile: 1 832 814 1655

E-mail: fbroussard@slb.com

Name: Michael James Moody

Tlf: 44 1235 858 028

Mobile: 44 7557 009 635

E-mail: mmoody@slb.com

G7.3 SINTEF ICT, SINTEF T&S

Title: COPD@Home – welfare technology for chronically ill at home, connected to intermunicipal health and welfare services

Customer: SINTEF ICT, SINTEF T&S

Address: S.P. Andersensvei 15b, 7465 Trondheim

COPD@Home is a cooperation project between SINTEF, Trondheim municipality, InnoMed, and St. Olavs Hospital. COPD ("Kols") patients belong to the lung department at St Olavs hospital, are a lot in and out of the hospital, and consumes much healthcare services. The goal of COPD@Home is to improve the homebased care of COPD patients, so that the number of hospital stays decrease and the patients receive better care at home and increase quality of care at home. The project shall:

- Reduce follow up needed for disease monitoring: The current treatment model in COPD@Home must be enhanced with usable technology that supports the patient in monitoring his own disease, and at the same time reducing the follow needed by homecare services. Homecare services then can focus on providing care.
- Focus on usability: Use of technology for home-based advanced follow up of diseases such as COPD must be suited to the particular needs of the user group for a simple, intuitive and usable user interface. This is necessary to motivate the user so that they themselves master the use of the solutions, so to avoid assistance from the homecare services on this, so that they can focus on more important tasks.
- Support learning: In addition to follow up and monitor diseases, the solution must have functionality for patient/next of kin training and learning.
- Support social networking. In addition to follow up of the disease, the solutions must support social networking, alarms in case of emergency, and medication reminders.
- Run on SMART equipment: Solutions for COPD care at home must run on different hardware, including PC, Smartphones (iPad, Android) and smart TVs. The solution should be interoperable with standardised sensors (e.g. using Continua)
- Generic solution: A solution developed for COPD must be generic so that the same solution can be used for different user groups (e.g. strokes, diabetes etc).
- Secure communication and storage: Communication of data from patient to the Norwegian Health network must be enabled

Contact details:

Name: Marius Mikalsen

Tlf: +47 970 34 099

E-mail: marius.mikalsen@sintef.no

G7.4 SINTEF ICT

Title: Privacy protection for information control
Customer: SINTEF ICT
Address: SP Andersensveg 15B, 7465 Trondheim

SINTEF ICT is currently investigating new approaches to privacy protection of end-users. While there are numerous policy languages, algorithms, models and to some degree implementations of privacy enhancing technology to protect personal information, we have yet to see widespread adoption by end-users.

This project will implement a privacy enhancing technology utilising case-based reasoning, expert privacy knowledge and anonymous community support. Possible client devices include both laptops and smartphones. Focus will be on end-user requirements and usability.

The project does not require particular knowledge of privacy enhancing technologies, but a basic understanding of case-based reasoning would be beneficial. Further, as we see the community support realised through the use of web services, the concept should at least be familiar. The students will get the opportunity to work closely with researchers at SINTEF.

Contact details:

Name: Inger Anne Tøndel
Tlf: 73592948
Mobile: 97088476
E-mail: inger.a.tondel@sintef.no

Name: Åsmund Ahlmann Nyre
Tlf:
Mobile: 92442422
E-mail: asmund.a.nyre@sintef.no

G7.5 Institute for Energy Technology (IFE)

Title: ToSS - Tool to Support Traceability of Safety Systems
Customer: Institute for Energy Technology (IFE)
Address: Postbox 173, 1751 Halden

Exercise: Web-based risk management

Traceability is the ability to describe and follow the life of a system artifact, in both forwards and backwards direction. The software developers should be able to trace from requirements through design decisions to code implementation, and the other way around. In safety domains such as aviation and nuclear power, traceability is considered as an important activity that should be performed during safety systems development. Traceability in the context of safety systems has a much broader scope than the general view on traceability, i.e. need to trace additional artifacts. For instance, while developing safety systems it is essential to trace the hazards to their respective safety objectives and safety requirements, and to trace the safety requirements to their design and implementation. Such traceability information is required and must be documented in order to justify and demonstrate the safety of systems. However, due to lack of adequate methods and tools, especially for safety systems, traceability is time-consuming and often poorly performed.

Requirements traceability and management is one of the research focus areas of the ICT RID department at the Institute for Energy Technology (IFE). In collaboration with NTNU, a PhD project was initiated looking into the topic of a traceability based approach for safety documentation and argumentation. One of the initial outcomes of the research is the conceptual traceability model of traceability for safety systems – see appendix. The conceptual traceability model is the initial solution which attempts to present the data (also called artifacts) generated during system development and safety assessment processes, and the relations between the data. Next step of the research is to refine the conceptual model with well-defined syntax and semantics, and also to provide a tool support. We propose a student project to develop a computer-based tool support to the traceability model. The developers (TDT 4290 students) will systematically elicit and describe the requirements, and thereafter design and implement the tool.

Contact details:

Name: Vikash Katta
Tlf: 45 46 43 23
E-mail: vikash.katta@hrp.no

G7.6 SINTEF IKT and Trondheim Kommune

Title: Social network for elder

Customer: SINTEF IKT og Trondheim kommune

Address: S.P. Andersensvei 15b, 7465 Trondheim

There is a clear association between boredom amongst elder and their mental and physical health. As a part of an international research project SINTEF and Trondheim Kommune is researching the possibilities for using modern ICT to motivate older people to keep up and expand their social network; especially with a focus on getting people “out of the door”. This problem is based on building a mechanism for preference matching in social networks and delivery of communication via e.g. iPad, Android and/or SMS. The test case is most likely Hornemansgårdens services for elderly people ein Trondheim.

The tasks is based on knowledge and problem solving within the following areas:

- Service oriented architectures (SOA)
- Web-services
- iPhone and iPad
- Android

Contact details:

Name: Anders Kofod-Petersen

Tlf: 73 59 29 55

Mobil: 91 89 72 87

Fax: 73 59 29 77

Mail: akof@sintef.no

Name: Bjørn Magnus Mathisen

Tlf: 73 59 28 22

Mobil: 41 47 44 23

Fax: 73 59 29 77

Mail: BjornMagnus.Mathisen@sintef.no

G7.7 NTNU IDI

Title: Interactive Door Sign
Customer: Torstein Hjelle (NTNU IDI)
Address:

The goal of this project is to design and implement a functional prototype of an interactive door sign to be used outside for instance office doors.

Given a reasonably priced tablet running Android OS with a 7 inch touch screen, Wi-Fi, built-in camera, microphone, speakers, etc. a system for using this as a door sign is to be developed. When mounted outside an office the display should show information like name of person who use the office, title and a picture. There should also be room for a short message (like “In a meeting”, “Do not disturb”, “Back at 13:30”, etc.). This message should be customizable and it should be possible to update it remotely via mobile phone (SMS) and a web-based interface.

The unit should also function as door phone with both audio and video. When visitors press a button on the unit a video call with the owner of the office should be initiated using the built-in camera, microphone and speakers. This video call should be possible no matter where in the world the owner of the office is located as long as he/she is connected to the Internet. If the owner is not connected to the Internet, visitors should be able to leave a message in form of a video and audio recording.

Focus:

1. Usability – Easy to use, both for office owner and visitor
2. Security – Unwanted access to the unit should be limited

What to make:

1. Software running on the Android-based tablet
2. Client software running on the office owner’s computer
3. (Depending on solution) Server software that both the tablet and the client software uses to retrieve information, save recordings and coordinate connections

Contact details:

Name: Torstein Hjelle
Mobile: 917 02 358
E-mail: torstein.hjelle@idi.ntnu.no

G7.8 Artsdatabanken

Title: Mobi application for reporting and use of digital identification keys

Company: Artsdatabanken

Address: Erling Skakkesgt. 47, 7491 Trondheim

Artdatabankens web application for reporting species (www.artsobservasjoner.no) is to be released in version 2.0 in September. It's based on state-of-the-art technology and .Net. However, it still lacks a mobile interface for offline reporting. Artsobservasjoner is in use both in Norway and Sweden and has so far in excess of 15 000 users and 35 million reported observations. There is a pilot adapted for smart-phones and tablets on <http://touch.artsdatabanken.no> .

Artsdatabanken would like a mobile interface for reporting into Artsobservasjoner 2.0. This requires the GPS and camera on smart phones. It must be possible to registrations offline and enable synchornising/reporting when the phone comes back online. In offline-mode, the determination of species must use elements from the pre-existing keys. The solution would ideally be for both Android and iOS, but should be designed that it can be ported to other platforms. The business logic should be the same for different platforms.

Contact details:

Name: Nils Valland

Tlf: 73592301

Mobile: 92412037

Fax: 73 59 22 40

E-mail: nilsvalland@artsdatabanken.no

Name: Askild Olsen

Tlf: 73 59 21 93

Mobile: 91 78 34 89

Fax: 73 59 22 40

E-mail: askild.olsen@artsdatabanken.no

G7.9 Thales Norway AS

Title: Wireshark: Automated generation of protocol dissectors

Company: Thales Norway AS

Address: Strindv 1, 7030 Trondheim

Wireshark is the world's foremost network protocol analyzer, and is the de facto (and often de jure) standard across many industries and educational institutions. Wireshark development thrives thanks to the contributions of networking experts across the globe. It is the continuation of an open source project that started in 1998. (www.wireshark.org) Lua is a powerful, fast, lightweight, embeddable scripting language. Lua has been used in many industrial applications (e.g. Adobe's Photoshop Lightroom) with an emphasis on embedded systems, as well as games (e.g. World of Warcraft). Lua is currently the leading scripting language in games. (www.lua.org)

In Wireshark, Lua provides an easy way to write or extend protocol dissectors, the part of Wireshark responsible for decoding the captured network data. The goal of this project is to design a tool that is able to generate Lua code for dissecting the binary representation of C/C++ structs, allowing Wireshark to show, filter, and search through such data.

The project should result in a stand-alone utility program, script or library, which can be used as a supporting tool for Wireshark. The project will introduce the students to Lua-scripting, parsing of C/C++ code, open source projects and network communication issues. The students will need to work with the customer to define the technical and functional requirements, gain knowledge of the problem domain, and design and implement a solution to the problem.

The task may be adjusted according to the student group's interests and skills. Thales will provide technical support for Wireshark and Lua, and participates with a member from the Wireshark core development team.

Contact details:

Name: Christian Tellefsen

Mobile: 959 98 765

Fax: 73 93 87 11

E-mail: christian.tellefsen (aa) thalesgroup.com

Name: Stig Bjørlykke

Mobile: 982 29 806

E-mail: stig.bjorlykke (aa) thalesgroup.com

G7.10 Norwegian Public Roads Administration, ITS

Title: Hardware fault monitoring and notification for roadside infrastructure

Company: Norwegian Public Roads Administration, ITS

Address: Abels Gate 5, 7030 Trondheim

The Norwegian Public Roads Administration has a large number of installations at the side of the Norwegian road network that performs vehicle registration and counting. The data from these installations is used for multiple purposes, included deciding on future infrastructure needs.

As of today there is no overall system for detection or notification of hardware failure at these installations, even if the hardware is able to perform some self-diagnostic. Because of this the collected data has to undergo a manual and somewhat labor-intensive process before it can be of further use. With better notification and logging of errors this process can hopefully be reduced.

Our wish is a design and prototype for a system that gather information on both hardware (Datarec7 or newer) and data communication state (given from our telecom provider), and display this information in a clear interface. We wish for a web-based interface where we can check status, analyze faults and read out state logs. We also wish to examine if it is possible to automatically estimate undetected hardware errors from lack of expected vehicle traffic, and display this in the interface. Automatic notification of hardware faults to the correct instances using sms or email could also be considered. Finally, it is also a wish that the system should be easy to integrate into existing systems and databases at the Norwegian Public Roads Administration.

Contact details:

Name: Gryteselv Kristin

Tlf: 73954694

Mobile: 90929565

E-mail: kristin.gryteselv@vegvesen.no

Name: Jo Skjermo

Tlf: 73954646

Mobile: 92236618

E-mail: jo.skjermo@vegvesen.no

G7.11 Norsk Tillitsmann Stamdata AS

Title:

Company: Norsk Tillitsmann Stamdata AS

Address: Haakon VII gt 1 0161 OSLO

Funding of a large company is often done in two tears, where the first tear is stocks and the second tear being loans. When these loans are not given by banks, but by private persons, funds or other companies they are typically in the form of Bonds. The Norwegian Bond market is over 1500 billion NOK in size. Stamdata is the leading provider of reference data regarding such bonds in Norway - and is used daily by most banks, funds and brokers to gain detailed information on bonds in portfolios and to be traded. The data is made available though web-pages and through a data feed, making data available in trading and fund management systems. Stamdata's offices are based in Oslo, and meetings will be conducted over audio/video-conference, weekly/bi-weekly/as-required. A representative from Stamdata will be present in Trondheim at project kick-off and evaluation.

Project suggestion: Community driven league tables Brokers report sales to the market in league tables (high score lists) that are used to rate brokers against each other. The reports are based on reports made by the brokers themselves, and are as reliable as the broker that enters the data, and the discipline between the brokers (=high error rates). We estimate that between 2000 and 3000 reports are entered per year. Double reporting, lack of reporting and mis-reporting is common, and is typically corrected by Stamdata by making calls, requesting details and resolving conflicts between brokers - a process that is high manual and time consuming. This reporting can be vastly improved by creating a community system, where reports can be verified, marked as suspicion and corrected by the community themselves. Users with high levels of incorrect entries are given less possibilities to edit content, whereas those with high correctness are allowed to edit not even their own, but also other's reporting. The reporting can also be crossed check after official sales to the market are released (typically 2 weeks after initial reporting) and notifications of discrepancies between official sales and initial reporting can be created along with executing corrective actions.

The project is to create such a community driven league table.

Project suggestion: Bond statistics. The project is to develop a statistics web-site based on the data available in the reference database. This includes defining and understanding requirements from basic users like newspaper reporters to advanced users like market analysts that analyze different data sources on a daily basis. Tasks involve creating the interaction design and user interface along with databases/cubes required to give quick responses to user requests. Defining simple statistical templates for new users, but also allowing tuning and downloading of data for advanced users. Visualization of graphs and drilldown/ drill-up interaction are also see as methods to simplify and understand the raw data in new ways.

Contact details:

Name: Thies Schrader

Tlf: + 47 22 87 94 16

E-mail: schrader@trustee.no

G7.12 Q-Free ASA, R&D

Title: Realtime Component Performance Monitoring

Company: Q-Free ASA, R&D

Address: Thoning Owesens gt 35c, 7044 Leangen

Q-Free implements and delivers systems for Road Tolling. A system for Road Tolling normally consists of both roadside components and back office systems. For our customers it is important that the system has high performance with respect to vehicle identification. It is therefore essential that all the sensors installed on the roadside are able to detect vehicles correctly.

There can be many approaches to meet the performance requirement. We would like to explore the possibility to use a statistical analysis of a live data stream from the roadside to discover abnormalities. The statistical analysis must be backed by a historical data set, a norm, which says something about the expectancy at a certain time of year. Some examples of statistics worth looking at can be:

- Images of vehicles are processed to find the number plate. The process is referred to as Automatic Number Plate Recognition (ANPR). ANPR will give a confidence, and this can be used to say something about the image quality/camera performance.
- The number of passages (vehicle passing a tolling point) needing ALPR (no tag read), can be used to say something about the performance of the tag reader.

The assignment consists of three parts:

1. Establish a norm for statistical analysis
2. Perform a statistical analysis of the live data stream
3. Create rules and trigger events when signs of degraded performance are detected

The expected outcome of the project is to have some sort of algorithm/description of how to detect a decline of performance at a roadside sensor. Also it is necessary to describe some rules for triggering events. The preferred way to prove/document the algorithm and rules would be to implement a prototype that demonstrates the statistical analysis and triggered events.

Contact details:

Name: Morten Tokle

Tlf:

Mobile: +47 900 56902

E-mail: morten.tokle@q-free.com

G7.13 Vitenskapsmuseet, NTNU

Title: Samlingsøk

Company: Vitenskapsmuseet, NTNU

Address:

NTNU Vitenskapsmuseet would like to establish an internet-based "Samlingsøk"-solution to communicate information from their natural history scientific collections. The collections consists of approximate 1.3 million Norwegian and foreign objects, whereof 90 % is digitised.

The botanical data is stored in Oracle-databases that are administered by MUSIT (Universitetsmuseenes felles IT organisasjon) at USIT/UiO. The zoological data is stored in a homemade Access 2003-solution that is administered locally at Vitenskapsmuseet. These zoological data will in time be migrated from Access to a new national Oracle-solution to be administered by MUSIT.

The intention behind the is that the solution that will be developed can be used as a platform for other similar services for natural and cultural historical collections and that development and administration should not be dependent on key personnel in the organisation. The solution should be scalable for many concurrent users and could tie in information from several sources. It is sought that the solution has an architecture and design that is partially independent on the information sources, but at the same time establish standardised interfaces against these so that the solution will be independent of the underlying database technology. WebServices (SOAP/REST) can be a possibility for search and retrieval of source information.

The objects in the collection partially contain location information, and integrations with map solutions and location searches would be of interest.

For certain objects it would also be interesting to offer mobile applications (Android/iPhone) that connect new observations with previously registered objects in the collection. Collection/registration of new information would not be a part of the collections from Vitenskapsmuseet, but can be developed/established/integrated and be used with the collections where it makes sense. Other integrations can also be of interest where samlingsøk could offer relevant facts about which objects are available at Vitenskapsmuseet and where different findings are registered.

Contact details:

Name: Torkild Bakken

Tlf:

Mobile:

E-mail: torkild.bakken@vm.ntnu.no

Name: Carl-Fredrik Sørensen

Tlf:

Mobile:

E-mail: carl-fredrik.sorensen@ntnu.no

8 Appendix H – Student lists and groups

The *final student/group allocations will be done at the end of the common course kick-off in the S1 auditorium at the NTNU-Gløshaugen campus on Tuesday 30.8 at 12:15-13:00*. The final allocations will also be announced on the web as soon as possible.

So please note, that the below allocation lists are preliminary, i.e. they may be incomplete and/or incorrect. The *final allocation* of students to groups may be adjusted, due to updated information about students' whereabouts, being revealed as late as during the 30.8 kick-off! Because the deadline for enrolling is after the start of the course, this list might change.

Table 2: Group assignment

Name	@stud.ntnu.no	Group
Martin Belgau Ellefsrød	ellefsro	1
Aleksander Aanesl. Elvemo	aleksael	1
Andreas Helmich Hunderi	andreaahu	1
Erlend Børslid Haugsdal	erlehau	1
Nicolaj Broby Petersen	nicolajs	1
Cato Olsen Strand	catoolse	1
Vegard Djuvslan	djuvslan	1
Eirik Emanuelsen	eirikem	2
Helge Johan Fosse	helgejfo	2
Vegard Gamnes	vegardga	2
Håvard Malm Geithus	havardge	2
Carl Joachim Rørvik	carljoac	2
Steffen Rendahl Stenersen	steffenr	2
Yann Colliva	ylcolliv	2
Ellen Wiig Andresen	ellenwii	3
Malin Berg	malinbe	3
Lars Greger Nordland Hagen	larsgreg	3
Ingvild Indrebø	ingvilin	3
Lars Espen Strand Nordhus	larsesn	3
Martin Vagstad	martivag	3
Håvard Schei	havardsc	3
Einar Nour Afiouni	afiouni	4
Neshahavan Karunakaran	neshahav	4
Amanpreet Kaur	amanprek	4
Henrik H Knutsen	henrikkn	4
Dimitry Kongevold	dimitry	4
Nicholas Gerstle	nicholge	4
Ulf Nore	ulfno	4
Lars Andersen	larsan	5
Vidar Lillebø	vidarli	5
Dean Lozo	dean	5
Thomas Marstrander	thomm	5
Bhuwan Krishna Som Poudel	bkpoudel	5

Fredrik Klokk Holst	frederho	5
Marko Radosavljevic	markor	5
Emil Grunt	emilg	6
Simen Kind Gulbrandsen	simenkin	6
Kjetil Mehl	kjetime	6
Safoura Shamsolketabi	safouras	6
Jaspreet Singh	jaspreet	6
Eirik Fikkan	eirikfi	6
Miso Vrucinic	misov	6
David Andrassy Eilertsen	davidae	7
Knut Esten Melandsø Nekså	knuteste	7
Andreas Nordahl	andrnord	7
Sibel Ayper Tasdemir	tasdemir	7
Igoris Trimailovas	igorist	7
Elise Boinnot	emboinno	7
Michal Krajicek	michalk	7
Dag-Inge Aas	dagingaa	8
Muhsin Günaydin	gunaydin	8
Stian Liknes	stianlik	8
Anders Søbstad Rye	anderrye	8
Andreas Berg Skomedal	andrskom	8
Nikola Djoric	nikoladj	8
Yonathan Redda	redda	8
Erik Bergersen	eribe	9
Sondre Johan Mannsverk	sondrejo	9
Terje Snarby	snarby	9
Even Wiik Thomassen	evenwiik	9
Lars Solvoll Tønder	larssot	9
Sigurd Wien	sigurdw	9
Jaroslav Fibichr	jaroslaf	9
Sonik Shrestha	soniks	10
Roar Bjurstrøm	roarbju	10
Kato Stølen	kato	10
Sondre Løberg Sæter	sondrelo	10
Bjørnar Valle	bjornava	10
Robert Versvik	robertve	10
Eirik Stene	eirikast	10
Fatemeh Salehian Kia	fatemehs	11
Thomas Bruun	thombru	11
Martin Rudi Holaker	marthola	11
Nikolas Jansen	nikolas	11
Christian Gjerde Jensen	chrjens	11
Anders Røsæg Pedersen	andeped	11
Ole Andreas Rydland	oleandry	11
Majid Navaii	majidn	12
Kenneth Børtveit	kennetbo	12

Yu He	yuh	12
Karl Johan V Heimark	heimark	12
Christian Overvåg Hjorth	christhj	12
Leif Julian Øvrelid	leifjuli	12
Petr Dvorak	petrd	12
Jose de Jesu Gonzales Ibanez	jdgonzal	13
Jonas Eikli	jonasei	13
Kent Robin Haugen	kentrobi	13
Tormund Sandve Haus	tormunds	13
Ole-Petter Olsen	olepetol	13
Marius Qvam Wollamo	wollamo	13
Nikola Aleksic	nikolaal	13

9 Appendix I- Kick-off rooms for groups

Detailed group allocation will be done in the information meeting 30.8 and documented on the course webpage.

Group 1, Gridmedia Technologies AS

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITV-054
30.08	First Customer Meeting	14.15-	ITV-054
31.08	First advisor meeting	08.15-09.00	ITV-054

Group 2, Schlumberger

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITS-236
30.08	First Customer Meeting	14.15-	ITS-236
31.08	First advisor meeting	09.15-10.00	ITV-054

Group 3, Sintef T&S – COPD@Home

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITV-354
30.08	First Customer Meeting	14.15-	ITV-354
31.08	First advisor meeting	13.15-14.00	ITV-464

Group 4, Sintef ICT

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	E204
30.08	First Customer Meeting	14.15-	E204
31.08	First advisor meeting	09.15-10.00	ITS-236

Group 5, IFE

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	R40
30.08	First Customer Meeting	14.15-	R40
31.08	First advisor meeting	14.15-15.00	ITV-054

Group 6, Sintef ICT & Trondheim Kommune

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITV-464
30.08	First Customer Meeting	14.15-	ITV-464
31.08	First advisor meeting	09.15-10.00	ITV-464

Group 7, NTNU IDI

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITV-242
30.08	First Customer Meeting	14.15-	ITV-242
31.08	First advisor meeting	08.15-09.00	ITV-354

Group 8, Artsdatabanken

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	F204
30.08	First Customer Meeting	14.15-	F204
31.08	First advisor meeting	09.15-10.00	ITV-354

Group 9, Thales Norway AS

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	F404
30.08	First Customer Meeting	14.15-	F404
31.08	First advisor meeting	08.15-09.00	ITV-242

Group 10, Norwegian Public Roads Administration

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	iArbeid
30.08	First Customer Meeting	14.15-	iArbeid
31.08	First advisor meeting	09.15-10.00	ITV-242

Group 11, Norsk Tillitsmann Stamdata AS

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	G138
30.08	First Customer Meeting	14.15-	G138
31.08	First advisor meeting	08.15-09.00	ITV-464

Group 12, Q-Free ASA

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	ITV-454
30.08	First Customer Meeting	14.15-	ITV-454
31.08	First advisor meeting	08.15-09.00	ITS-236

Group 13, NTNU VM

Date	Activity	Time	Location
30.08	Kick off	12.15-13.00	S1
30.08	First Group Meeting	13.15-14.00	R41
30.08	First Customer Meeting	14.15-	R41
31.08	First advisor meeting	15.15-16.00	ITV-054