IT2901 - Informatics Project II

IDI Open
Programming Contest System

Haakon Konrad William Aasebø
Håkon Gimnes Kaurel
Tino Hakim Lazreg
Filip Fjuk Egge
Anders Sildnes
Eirik Fosse

May 2014

Norwegian University of Science and Technology

Supervisor: Hong Guo

Foreword

Originally inspired by the Nordic Collegiate Programming Contest (NCPC), it has been held at NTNU every spring since 2007. The format is a five-hour contest with competing teams consisting of one, two or three contestants. A team of volunteer judges write the problems and answer clarification requests during the contest, while another team hands out balloons for each solved problem. Usually a rather hectic affair, it is extremely important that everything is well prepared. The number of teams is often more than 100, with the record being 162 teams in 2011

The contest system that verifies solutions is at the heart of the contest when it is in progress, and needs to be working perfectly at all times. The system must handle several submissions per second, while verifying that each one is correct and runs within the set resource limits. Submissions must show up on the high score list, and when problems are solved the team handing out balloons must be notified. In addition to this there were a lot of other functional requirements having to do with the bureaucracy of organizing the contest

A requirement was that new features could be easily added in the future, and the code was written with this in mind. The project will now become open source, and all programming contest enthusiasts will soon be able to request and implement their desired features

All aspects of this project have been pleasing and delightful for us. The team has exceeded all our expectations and their system will be used for years to come.

Preface

Before there were computers, there were algorithms. But now that there are computers, there are even more algorithms, and algorithms lie at the heart of computing. Designing a system for eager students to hone their skill in the heart of computing has been a true joy

Our group never wanted to settle for adequacy and mere requisiteness. For the past few months, weve taught ourselves a new programming language and framework and used advanced development frameworks - while tackling many social and technical conflicts.

We have ve proven how Ambition is a dream with a V8 engine, as Elvis Presley once said.

The group would like to thank our eager customers, Finn Inderhaug Holme, Christian Chavez and Christian Neverdal Jonassen for their time to meet us and provide constructive feedback. We also owe a big thanks to our supervisor, Hong Guo, for constructive criticism and reflections; without which, we would not ascertain the peak of our own potential

Contents

Chapter 1

Project Management

This chapter will go through the different project roles we deemed important. We will explain our development method, which tools we use and give an overview of how we planned the project. Furthermore, in section 1.4.1. We also provide a structured overview of how we organized our time.

1.1 Project Roles

We wanted to ensure that all developers had an even workload and experience in all components of our project. We maintained a flat organizational structure where all decisions were made in groups. No member would work alone on a task for a longer period of time. Some tasks and delegations were easier to assign only once.

The most central role is that of the scrum master. The role mainly consists of setting up meeting agendas and keeping control of what team members are working on. In addition, the scrum master should act as a buffer between the team and other distractions. The scrum master had a casting vote whenever there was a disagreement. The group elected Haakon to be scrum master because of his well established authority and organization.

We also assigned the role of a transcriptionist. His job consists of writing a short summary of every meeting, and making this available to the rest of the group. This includes meetings with the customer and supervisor. This job was performed by Anders, who volunteered for the position. We assigned Håkon to be customer contact, and Tino as responsible for room reservations.

1.2 Development Method (Scrum)

Scrum focuses on having daily meetings, and constantly adjusting to changes by iterative development. This makes it easier to predict and to adjust for problems that may occur. It was hard to predict what would happen in our project, therefore our sprints were short, lasting at most two weeks. The transition between two sprints was done during a prolonged meeting on Wednesdays. During this meeting we evaluated the latest sprint and planned the upcoming one. Every team member were requested to say three good things and three bad things regarding the last sprint. This was followed by a discussion of how to plan the next sprint better. Lastly we showed what had been completed, to the other members of the group, before setting up the next sprint. Scrum also

focuses on having finished versions of the systems on each iteration, and to finish all packages in the given iteration. In order to take advantage of the best in everyone's abilities we worked in pairs where this was efficient. Working in pairs is common in agile development. This was to improve code quality and reduce errors.

1.3 Tools/Framework

The customer wanted our end product to be easy to maintain for future developers. Therefore we have chosen tools that are well known and easy to learn. Some of the most important are:

- Django, a framework written in Python.
- VIM and Eclipse for editing
- Google Drive and latex for documentation
- Git as version control, with github as hosting service
- Email lists, IRC and Facebook for communication
- Bootstrap and Grappelli for user interface design

A lot of different tools were considered for this system. A full list of all tools and frameworks used and considered can be viewed in appendix *Tools and Frameworks.*

1.4 Project-Level Planning

After our initial requirements elicitation we began to plan our development process. The purpose of the plan was to verify that we had enough time to complete the requirements, and to avoid unforeseen risks. This section will present the various components we introduced to structurize the project.

1.4.1 Work Breakdown Structure

WBS is a decomposition of the project into phases, deliverables and work packages. Each package was further broken down into different tasks. The benefits from the WBS are as follows:

- Planning out the entire process prevents bottlenecks
- Clearly defining the scope of a package prevents excess or insufficient time usage
- It is easy for supervisors and other parties to evaluate and understand our process

Table 1.1 shows the work breakdown structure created. These high-level packages were later broken down into activities, which are in the product backlog, see appendix TODO

Table 1.1: Work breakdown structure

1. Project management
    (a) Write timesheet template
    (b) Look at the reflection notes
    (c) Meetings
        i. Internal
        ii. Customer
        iii. supervisor
    (d) Report
        i. Preliminary version
        ii. Mid-semester version
        iii. Final version
    (e) Risk assessment
    (f) WBS
    (g) Status report
    (h) Activity plans
2. Pre-study
    (a) Install and learn tools
    (b) Learn language/framework
    (c) Course
3. Design
    (a) Requirement Specification
        i. Functional
        ii. Non-functional
    (b) System architecture
    (c) Database modeling
    (d) User Interface
        i. Prototyping
        ii. Usability Testing
    (e) Admin interface
4. Development
    (a) Backend
        i. Execution-node(s)
            A. Web-page
            A. User
            B. Usergroups
            C. Team management
        ii. Statistics
        iii. Contest management
        iv. Clarification system
        v. Balloons system
        vi. Unit testing
    (b) Testing
        i. User-test
        ii. System-test
        iii. Final test
    (c) Implementation
        i. Deploy to production
        ii. Installation
        iii. Turn in to stakeholder
    (d) Implementation
        i. Verify
        ii. Document

We also created a gantt chart. Here, each package was assigned an estimated time period, over how long time we expected to use. For ease of comprehension, not every package was included from the WBS. The gantt chart is shown in figure 1.2

Table 1.2: Gantt chart

WP Name	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Project management															
WBS															
Pre-study															
Install and learn tools															
Learn language/framework															
Course															
Design															
Requirement specification															
System architecture															
Database modelling															
Tests															
User-interface															
Development															
Execution node															
Implement single node															
Implement several nodes															
Content Management System															
Front end															
Testing															
Unit testing															
Integration testing															
System test															
Production															
Post-implementation															

The gantt chart was revised several times during the first four sprints, mainly due to new deadlines set by the customer.

1.4.2 Milestones

Throughout the project, the supervisor, customer, and the project group set deadlines. Some of the milestones marks the completion of work packages. We have four of these milestones, M-03, M-05, M-06 and M-07. The other milestones represents events with deadlines that were given by the course stakeholders. These are M-01, M-02, M-04, M-08. The group used the milestones in order to determine if the project is on schedule and to monitor the progress. The reader can view what requirements that were met for each milestone in TODO REF MILESTONE X.X.

Preliminary report M-01 Preliminary report is the delivery of the first version of the report. This was to help us get started with important aspects of the project work.

Mid-semester report M-02 This version of the report should present all of the analysis and most of the design of our system. The delivery date for the mid-semester report is 16th of March. We wanted to complete this one week earlier, 9th of March, focus on M-03.

First release M-03 This milestone marks the groups first delivery to the customer. In summary this release should make it possible for contestants to sign up for a competition. Three days prior to the release the group will meet up with the customer and overlook that all the requirements are met. This meeting will also act as an introduction on how to manage the system.

Presentation M-04 The main purpose of the presentation is for the class to share their experiences and learn from other groups.

Beta-release M-05 The beta release should contain most of the essential features. This version of the program should only be a release to a selected group of people.

IDI Open test event M-06 On April 26th there will be a test event where everybody could test the system. This means that leading up to this event the system should be a release candidate.

IDI Open M-07 This is the day of the competition and the system should be a release version.

Final report M-08 This milestone marks the final date for delivering the report and the end of our project. Based on feedback received from the competition the group might choose to implement some changes to the system.

1.4.3 Meetings

Our meetings can be categorized in three categories: internal, supervisor and customer meeting. We established some meetings rules:

- All meetings follow "the academic quarter", meaning that the time of start was XX:15
- Members that were late had to bring a cake to the next meeting
- All members may at any time propose a coffee break, a proposal that has to be followed.
- No laptop should be open during the meetings

Internal meetings

We had three internal meetings each week. Two of which were daily scrum meetings. These were primarily set to be on Mondays and Thursdays. During these meetings each group member would answer three questions:

- What have you done since the last meeting?
- What are you planning to do until the next meeting?
- Do you have any problems regarding the completion of your task?

The group would usually continue to work together after these meetings.

On Wednesday we had longer meetings marking the end of one sprint and the beginning of the next. This meeting would consist of a sprint review meeting and a sprint retrospective, where we discussed

- What was good/bad with the last sprint?
- What should we try to improve during the next sprint?

After that we held a sprint planning meeting and created a new sprint backlog. Our official meeting structure for this meeting can be viewed in the appendix TODO

Supervisor meeting

Meetings with the supervisor was generally held at a biweekly basis. During these meetings we talked about what we had done, what we were going to do and received feedback on what we had done. Before each meeting we had to deliver status reports and activity charts. These activity diagrams were early on replaced by sprint backlog and burndown charts to facilitate the development process.

Customer meeting

Customer meetings were held whenever we felt that a certain part of the requirements specification was unclear to us, and when we wanted approval of a newly completed feature. Throughout the semester there were a lot of meetings. As we never decided upon a fixed interval between customer meetings, the frequency varied a lot. The couple of days leading up to a release date often contained customer meetings in order to get everything right before starting on the next release. During our periods of focusing on writing this report, the frequency of these meetings naturally went down as the product did not progress, and as a consequence we had little to discuss with the customer.

1.4.4 Resources

This section contains the available resources for the project. We intended to use a minimum of 20/25 hours per person each week, but prepared for more work as we approached the deadline. This estimate was later scaled up to a minimum of 25/30 two weeks before easter. During easter, the amount of hours per week scaled up higher.

Planned work

Table 1.3 shows our first initial draft of sprints.

Table 1.3: Initial sprint overview

Sprint	Range (week)	Days	Hours
1	3 - 4	7	15
2	4 - 5	7	20
3	5 - 6	7	20
4	6 - 7	7	20

5	7 - 8	7	20
6	8 - 9	7	20
7	9 - 10	7	20
8	10 - 11	7	20
9	11 -12	7	20
10	12 -13	7	20
11	13 - 14	7	20
12	14 - 15	9	33
Easter	15 - 17	12	-
13	17 - 18	7	35
14	18 - 19 (Leading up to event)	9	35
After	19 - 22	21	50
Total:		91	368

Actual work

Table 1.4 shows the actual sprints and work done. The hours are for each person, during that sprint.

Table 1.4: Actual work

Sprint	Week	Days	Hours
1	3-4	7	15
2	4-5	7	15
3	5-6	7	20
4	6-7	7	20
5	7-8	7	27
6	8-9	7	31
7	10-11	7	35
8	11-12	7	30
9	12-13	7	30
10	14-15	9	40
11	15-17 (starting 16.04, ending 26.04, easter)	10	90
12	18-19	6	35
After	19-22	21	65
Total		100	453