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

Requirements Specification

According to the gantt chart (Fig 4.1) the team were supposed to update the requirement specification starting from week 2 and continuing up until week 10. For us it was still the case that there were a clearly identifiable requirement specifications phase. This was primarily from week 2 up to and including week 4. The outcome from this three week process was heavily used in order to establish agreement between us and the customer. This chapter presents the result from this process.

1.1 Purpose and Scope of this Specification

The purpose of the requirement specification document is to specify the objectives for our end product. Requirements are written at different levels of detail. This is to make it easy to communicate the requirements to both business and technical parties. We have mainly written the functional requirements as stories and then broken them into smaller pieces. This makes the requirements easy to communicate to the customer, and succinct for the developers. These stories can be viewed in appendix **??**. It is important to recognize that our project only lasted for a few months. Thus, late changes to requirements were inserted promptly and without revision control. This is a common practice in agile development[1]. The advantage and reason we chose not to perform revision control, is that we could save time in not formally documenting all changes.

The coverage of the requirements is intended to be a complete coverage of the product. This implies that all features available from the application domain is listed in our specification. What the requirements specification does not cover are organizational and external requirements. This follows from the small amount of administrative users and developers involved, and trust between the customer and the developers.

1.2 Process of the Requirement Specification

The customer passed on an initial list of requirements to our group. After a classification and organization of the features, we drafted scenarios and internally discussed the implication to each requested feature. Therein, we saw what features would be infeasible and additional features we

[1] Page 91, Sommersville

would want to introduce to the customer. The modified list of requirements was then presented to the customer, before proceeding with the implementation of the end-product. Throughout the entire development process both we and the customer have been modifying the list of requirements.

1.3 Product/service Description

In this section, you will find our interpretation of the physical user-domain. The reader should note that some members of our group has competed earlier, which has given us helpful empirical insight.

1.3.1 Expected Physical Environment

Our solution is used in different contexts. Table 1.1 has the different application and user-domains.

Table 1.1: Users and their expected stereotypical properties

IDI Open is hosted in P15, Høgskoleringen 3, on Gløshaugen campus every year. Every team participating in the contest get allocated their own computer.	For offsite contestants, javascript must be enabled.
Software is required. A web server(Apache, Nginx), database server(MySQL, PostgreSQL), Python with PyPi package manager.	Linux kernel with ssh enabled, supplemented with a root user.

1.3.2 User Characteristics

Table 1.2 show different stereotypes of expected typical users. While open to deviations from the stereotypes, they highlight important properties required for our solution.

Table 1.2: Users and their expected stereotypes

Judge	**Admin**
Education: Computer Science Age: 20-25 Responsibility: review assignments Frustrations: • Irresponsive interfaces • Incorrect data • User submission system • Response types	Education: Computer Science Age: 20-27 Responsibility: review assignments Frustrations: • Irresponsive interfaces • Node failures • Incorrect data • Backend system • Dataflow
Contestant	**Sponsor**

Education: Computer Science Age: 20-25 Responsibility: Submit and upload assignments, keep track of score Frustrations: • Irresponsive interfaces • Lack of overview • Backend system • Dataflow	Education: Computer Science Age: 27-40 Responsibility: Advertize company Frustrations: • Dissatisfied contestants • No overview • Nothing special
Balloon-functionary Education: Computer Science Age: 20-25 Responsibility: Review scores, hand out balloons and information Frustrations: • Mis-information • Scoreboards, about competition	

It can be seen in table 1.2 that the most prominent trait of our users is that they have a background in computer science. As a consequence, it is assumed a higher level of technical competence from our users. The user profiles also highlight that some features were more important than others, e.g. responsiveness over aesthetics.

1.4 Requirements

Stories can be ambiguous and open for misinterpretation. we felt that a natural language specification of requirements would make it easier to understand our application domain. To reduce miscommunication we made sure to give each specification as short, succinct sentences. The stories were used as a way to communicate with the customer about requirements without them having to read through the table of requirements.

There are three different states for priorities, HIGH, MED and LOW. This ensured strict priorities. Using more states would make it hard to differentiate between the priorities we gave the requirements.

The following definitions make out the guideline for prioritizing the requirements:

- HIGH: The requirement is a “must have”. To have a successfull product, the requirement must be implemented.
- MED: The requirement is a “should have”. The fulfillment of the requirement will benefit the quality system.
- LOW: The requirement is a “nice to have”. This includes functionality not critical to the system.

1.4.1 Functional

The functional requirements are broken down in different categories. Each category corresponds to a user group. The categories are Admin, Judge, Contestant, Functionary, Teams, and Other. Each category has an ID, priority and story. Table X.X shows the complete list of the requirements, while the corresponding stories are given in appendix**??**

The ID system can be interpreted in the following way

- The F stands for Functional
- The second letter determines which category, e.g A stands for admin.

The milestone show when each requirement needs to be met.

1.4.2 Functional requirements for Admin

Table 1.3: Feasible triples for highly variable Grid, MLMMH.

Requrement	ID	Story	Comment	Priority	Milestone
An admin shall be able to create a new contest	FA-01	SA-1	A new contest equals a new web page	HIGH	M-03
An admin can choose whether the site should be published immediately or not	FA-02	SA-1		MED	M-03
An admin can add custom CSS to the web-page	FA-03	SA-1		LOW	M-03
An admin shall be able to choose settings for the contest	FA-04	SA-1	of contestants, maximum number of contestants per team, date, name. Default settings will be provided	HIGH	M-06
An admin shall have access to all modules in the program	FA-05	SA-2		HIGH	M-06
An admin can change permission of a usergroup	FA-06	SA-2		LOW	M-06
An admin can remove/add to a user group.	FA-07	SA-2	This includes promoting new admins	LOW	M-06
An admin can deactivate users	FA-08	SA-2		LOW	M-06
An admin can remove users from the database	FA-09	SA-2		HIGH	M-06
An admin can add a node	FA-10	SA-4	The node must be a privileged user	HIGH	M-06
An admin can remove a node	FA-11	SA-4		HIGH	M-06

Table 1.3 – continued from previous page

Requirement	ID	Story	Comment	Priority	Milestone
An admin can manage a node.	FA-12	SA-4	This requirement is in terms of compiler profiles support	HIGH	M-06
An admin can add more than one node	FA-13	SA-4		MED	M-06
An admin can add news items	FA-14	SA-5		HIGH	M-03
An admin can remove new items	FA-15	SA-5		MED	M-03
An admin can modify news item	FA-16	SA-5		MED	M-03

1.4.3 Functional requirements for Judge

Table 1.4: Feasible triples for highly variable Grid, MLMMH.

Requirement	ID	Story	Comment	Priority	Milestone
A Judge can create a problem	FJ-01	SJ-1	This includes cases with input and output	HIGH	M-06
A judge can upload cases to a problem and name each case	FJ-02	SJ-1		MED	M-06
A judge can set a resource limit on each task	FJ-03	SJ-1		LOW	M-06
A judge can add a solution that gives the right output	FJ-04	SJ-1		HIGH	M-06
A judge can add a solution that gives timeout	FJ-05	SJ-1		MED	M-06
A judge can add a solution that gives wrong answer	FJ-06	SJ-1		MED	M-06
A judge shall be able to view and edit all problems	FJ-07	SJ-1		HIGH	
A judge shall be able to respond to a question from a team	FJ-08	SJ-2	This is about the clarification system.	MED	M-06
A judge shall get a notification when received a question	FJ-09	SJ-2		LOW	M-06
A judge shall be able to respond to a question globally	FJ-10	SJ-2	By globally it is intended that the all teams can view the response and question	HIGH	M-06

Table 1.4 – continued from previous page

Requirement	ID	Story	Comment	Priority	Milestone
A judge shall be able supervise all submissions	FJ-11			MED	

1.4.4 Functional requirements for Contestant

Table 1.5: Feasible triples for highly variable Grid, MLMMH.

Requirement	ID	Story	Comment	Priority	Milestone
A contestant shall be able to edit their own information	FC-01	SC-1		HIGH	M-03
When created a contestant shall receive a confirmation email	FC-02	SC-1		HIGH	M-03
A contestant shall see which teams they are invited to	FC-03	SC-2		HIGH	M-03
A contestant shall see which team they are a member of	FC-04	SC-2		HIGH	M-03
A contestant shall see which teams and contests they have participated in earlier	FC-05	SC-2		MED	M-03
A contestant shall be able to ask a question to a judge	FC-06	SC-3		MED	M-03
A contestant shall have access to global answers from judges	FC-07	SC-3		MED	M-06
A contestant shall be able to change his/her email	FC-02	SC-2		MED	

1.4.5 Functional requirements for Functionary

A functionary shall be able to register a balloon colour to each task/problem	FF-01	SF-1		LOW	M-06	TF-12

A functionary shall have access to information about newly completed problems	FF-02	SF-1		MED	M-06	TF-12

1.4.6 Functional requirements for Teams

Table 1.7: Feasible triples for highly variable Grid, MLMMH.

Requirement	ID	Story	Comment	Priority	Milestone
A user shall be able to register a team	FT-01	ST-1	Whether or not the team is onsite, a team password, and a email for the team leader	HIGH	M-06
A user shall be able to register other team members for the team	FT-02	ST-2	By providing other users' email	HIGH	M-03
If the contestant is already in the system shall recognize personal info	FT-03	ST-2	Personal information like name, gender and so on.	LOW	M-03
A team leader must be able to invite new members	FT-04	ST-2	Input: email	MED	M-03
A team leader should be able to delete the team before the competition	FT-05	ST-2		MED	M-03
When a team leader invites a new member the new member must receive a registration link	FT-06	ST-2	The receiver of this email link must fill in the data specified in: T-3	MED	M-03
If a member's email is already in the database they will receive a confirmation link	FT-07	ST-2	The confirmation link will include automatically filled data. See T-4	LOW	M-03
All team information is editable in the team overview.	FT-08	ST-2		LOW	M-03
A team must be able to deliver submissions to problems	FT-09	ST-3		HIGH	M-06
When a team deliver a submission they shall receive response from the system	FT-10	ST-3	system should give timeout. This is specified by a judge.	HIGH	M-06

1.4.7 Other requirements

Table 1.8: Feasible triples for highly variable Grid, MLMMH.

Requirement	ID	Story	Comment	Priority	Milestone
The system shall be able to gather some statistics	FO-01	SA-3	It is here implied statistics from contestants in accordance with FE-3	HIGH	M-05
The system shall be able to gather a large variety of statistics specified by the admin	FO-02	SA-3		LOW	M-05
The system shall include a clarification system	FO-03	SJ-2	This is according to FJ-8, FJ-9, FJ-10, and FE-14, FE-15, FE-16, FE-17, FE-18	HIGH	M-07
The contest results are to be visible in the form of a highscore list.	FO-04	ST-03		MED	M-07

1.5 Non-functional

The nonfunctional requirements defines what objectives our end product needs to meet. Measure make it easy to agree on whether the requirement is fulfilled or not. Tables X.X can be interpreted in the following way:

- NF in the ID stands for non-functional
- Measure describe what the requirement holds
- Value is a quantitive measure
-

1.5.1 Speed

ID	Measure	Value	Priority	Comment
NF-01	Response from action	< 1.5 sec	MED	E.g. clicking a click
NF-02	Posting news	< 5 sec	MED	
NF-03	Edit user	< 1 min	MED	E.g. change email, password

1.5.2 Size

ID	Measure	Value	Priority	Comment
NF-04	Number of contestants	500	HIGH	

Table 1.10 – continued from previous page

ID	Measure	Value	Priority	Comment
NF-05	Number of teams	200	HIGH	
NF-06	Number of judges	20	HIGH	
NF-07	Number of admins	> 1	HIGH	
NF-08	Limitation of submission size	50kB	HIGH	

1.5.3 Ease of Use

ID	Measure	Value	Priority	Comment
NF-09	Learning time for contestants	< 5 min	MED	The users of the program should be good at computers and therefore know what they are doing.
NF-10	Learning time for admins	< 15 min	MED	
NF-11	Learning time for judge	< 10 min	MED	

1.5.4 Reliability

ID:	Measure:	Value:	Priority:	Comment:
NF-12	Mean time to failure	> 1 week	HIGH	The system should not be down during a contest
NF-13	Availability	$> 99.9\%$	MED	Downtime is not critical after or before a contest

1.5.5 Robustness

ID	Measure	Value	Priority	Comment
NF-14	Time to restart after failure	< 10 min	HIGH	
NF-15	Probability of data corruption on failure	$< 1\%$	MED	This is determined by backup coverage
NF-16	Expected living time	> 10 years	HIGH	
NF-17	Execution node	$= 1$	HIGH	
NF-18	Execution nodes	> 1	MED	It should be possible to utilize addition nodes

1.5.6 Portability/Scalability

ID	Measure	Value	Priority	Comment
NF-19	Extensibility		HIGH	Adding features should be easy
NF-20	Module-based code		HIGH	The code should be easy to maintain

1.5.7 Other

ID	Measure	Value	Priority	Comment
NF-21	Accessibility		HIGH	
NF-22	Open-source	GPL	MED	

1.6 Security

While security requirements are non-functional, we decided to do the security requirements engineering as a separate process. Table can be interpreted in the following way:

- In the ID, S is for security
- Measure describes

1.6.1 Authentication and Authorization

Table 1.16: Security requirements for authentication and authorization

ID	Measure	Priority	Comment
S-01	No user in any given user group shall be able to perform any operation outside of the definition of the requirements	MED	
S-02	An authenticated user shall not be able to perform any operation, as another user	HIGH	
S-03	After an authenticated user performs an action to be logged out, that user will need to log in to re-authenticate	MED	E.g. session-cookies should not remain such that you can still re-login
S-04	No user shall gain administrative rights without manual approval of current admins		Ensure no user is registered as admin by mistake, no scripts that automatically escalates privileges to administrator when conditions are met
S-05	Correct authorization must be required for respective content.	HIGH	
S-06	To authorize, you will either need to provide mandatory usercredentials through an interface, or have a valid session ID.	HIGH	
S-07	Session tokens shall be unique to one computer only	MED	Not possible to simply acquire a session ID and use it on other computers to authenticate

1.6.2 Immunity

Table 1.17: Security requirements for immunity

ID	Measure	Priority	Comment
S-08	No input-field shall be susceptible to injection attacks	HIGH	
S-09	All data that passes the trust zone shall be in plaintext, and validated against code	HIGH	
S-10	Data from non-developers can only be directed saved in databases.	MED	
S-11	Uploaded submissions shall not write to any file	HIGH	
S-12	Uploaded submissions shall not read from any other file than stdin	HIGH	
S-13	Uploaded submissions shall not access network or any other external service not needed to solve a problem.	HIGH	
S-14	Data from a user shall not be modified by non-users	MED	

1.6.3 Non-repudiation

Table 1.18: Security requirements for non-repudiation

ID	Measure	Priority	Comment
S-15	All modifications of data shall be logged	MED	
S-16	All log entries shall contain username(s) and a timestamp with day and current hour	LOW	
S-17	Logs will be backed up	LOW	
S-18	A team's score shall not be affected by anything other than what is given in the contest rules	HIGH	

1.6.4 Privacy

Table 1.19: Security requirements for privacy

ID	Measure	Priority	Comment
S-19	Sensitive user data shall not be stored in plain-text	HIGH	E.g. password, gender
S-20	Every user-field that is stored shall be justified in the requirements specification		This requirement does no longer apply
S-21	No sensitive data shall be exposed publicly, even if it is encrypted	MED	
S-22	User-data for a given user shall not be modified without that user's consent.	LOW	

1.6.5 Auditing

ID	Measure	Priority	Comment
S-23	Database shall be manually/automatically checked/verified for inconsistency or errors before an event.	MED	
S-24	Password that are used in development shall not be publicly available	HIGH	

1.7 Requirements Not Met

Most of the requirements on time. There were some minor requirements not fulfilled mainly due to time constraints. All of them were priority LOW. Here are the requirements we did not complete:

A judge shall get a notification when received a question	FJ-09
A functionary shall be able to register a balloon colour to each task/problem	FF-01
The system shall be able to gather a large variety of statistic specified by the admin	FO-02

The reason they were not completed was due to the their low priority and time constraint. In addition to the unfinished requirements there were also requirements that were not met in an ideal way. This was in agreement with the customer. These are the partially met requirements:

An admin can add a node	FA-10
An admin can remove a node	FA-11
An admin can manage a node.	FA-12
Response from action	NF-01
Logs will be backed up	NR-03

Unfortunately, an admin can only manage the execution nodes through the code. This is planned to be fixed before the next contest. The response time did unfortunately exceed 1.5 seconds during the contest. This was due to a bad implementation of the high score list, detailed in **??**. NR-03 had to be overruled during the contest. This is discussed in detail in chapter TODO *development*.