This is the Title of my Thesis

Your Name

December 2012

PROJECT THESIS
Department of Production and Quality Engineering
Norwegian University of Science and Technology

Supervisor 1: Professor Ask Burlefot
Supervisor 2: Professor Fingal Olsson

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

0.1 Requirements Specification

According to the Gantt diagram (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 3 week process was heavily used in order to establish agreement between us and the customers. This documents indices the result from this process. 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 requirements (functional) 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 is only lasting for a few months. Thus, late changes to requirements were inserted promptly and without revision controlling. This is also a common practise 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 naturally follows from that there is only a small amount of administrative users and developers involved, and trust between the customers and the developers.

0.1.1 Process of the Requirement Specification

The requirements were initially drafted by our customer alone. The customer passed on a list of features 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 would want to introduce to the customer. The modified list of requirements was then presented to for 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. Our process of requirements engineering was similar to what is described in the paragraph above, except that with an increased understanding of the application domain, we spent less time interpreting and agreeing on the implication to implementing each feature.

0.1.2 Product/service description

In this section, you will find our interpretation of the physical user-domain. It is also important to note that some of our developers has competed earlier years, so we had some ethnographic experience.

[1] Page 91, Sommersville

0.1.3 Expected Physical Environment

GentleIDI is used in different contexts and relies on multiple services. In table X.X you can find the different application and usage-domains.

IDIOpen is hosted in P15, Hgskoleringen 3, on Glshaugen campus every year. Every team participating in the contest get allocated their own computer.	For contestants participating offsite, a modern browser with javascript enabled is expected.
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.

User CharacteristicsWe created different stereotypes of those we expected to be our typical user. While open to deviations from the stereotypes, they highlight important properties required for GentleIDI. The description is given in table X.X.

• Irresponsive interfaces • Incorrect data • User submission system • Response types	• Irresponsive interfaces • Node failures • Incorrect data • Backend system • Dataflow
keep track of score • Irresponsive interfaces • Lack of overview • Backend system • Dataflow	• Dissatisfied contestants • No overview • Nothing special
and information • Mis-information • Scoreboards, about competition	

The most prominent trait of our users is that the majority of the end users have a background in computer science. Thus, we could assume a higher level of technological competence from our users. The user profiles also highlighted that some features were more important than others, for example we would prioritize responsiveness higher than aesthetics.

0.1.4 Requirements

The stories can be viewed in appendix,??. While 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.

Functional

We decided to break the functional requirements 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. The ID consists of a letter corresponding to the category, and a number that specifies which requirement it is in the category.

We decided to have three different states for the priority. We used high, med and low because we wanted to have very strict priorities. If we had used numbers ranging from 1-6, it would be hard to differentiate between the priorities we give the requirements.

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

- HIGH: The requirement is a "must have". We need this requirement to have a successful product.
- MED: The requirement is a "should have". The fulfillment of the requirement will be a benefit for the system
- LOW: The requirement is a "nice to have". This may include new functionality.

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

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

Functional requirements for Admin

Requirement	ID	Story	Comments	Priority	Milestone	Test
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	TF-01
An admin can choose whether the site should be published immediately or not	FA-02	SA-1		MED	M-03	TF-01

An admin can add custom CSS to the web-page	FA-03	SA-1		LOW	M-03	TF-01
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	TF-03
An admin shall have access to all modules in the program	FA-05	SA-2		HIGH	M-06	TF-03
An admin can change permission of a usergroup	FA-06	SA-2		LOW	M-06	TF-04
An admin can remove/add to a user group.	FA-07	SA-2	This includes promoting new admins	LOW	M-06	TF-04
An admin can deactivate users	FA-08	SA-2		LOW	M-06	TF-04
An admin can remove users from the database	FA-09	SA-2		HIGH	M-06	TF-04
An admin can add a node	FA-10	SA-4	The node must be a privileged user	HIGH	M-06	TF-06
An admin can remove a node	FA-11	SA-4		HIGH	M-06	TF-06
An admin can manage a node.	FA-12	SA-4	This requirement is in terms of compiler profiles support	HIGH	M-06	TF-06

An admin can add more than one node	FA-13	SA-4		MED	M-06	TF-06
An admin can add news items	FA-14	SA-5		HIGH	M-03	TF-01
An admin can remove new items	FA-15	SA-5		MED	M-03	TF-01
An admin can modify news item	FA-16	SA-5		MED	M-03	TF-01

Functional requirements for Judge

A Judge can create a problem	FJ-01	SJ-1	This includes cases with input and output	HIGH	M-06	TF-05
A judge can upload cases to a problem and name each case	FJ-02	SJ-1		MED	M-06	TF-05
A judge can set a resource limit on each task	FJ-03	SJ-1		LOW	M-06	TF-05
A judge can add a solution that gives the right output	FJ-04	SJ-1		HIGH	M-06	TF-05
A judge can add a solution that gives timeout	FJ-05	SJ-1		MED	M-06	TF-05
A judge can add a solution that gives wrong answer	FJ-06	SJ-1		MED	M-06	TF-05

A judge shall be able to view and edit all problems	FJ-07	SJ-1		HIGH		TF-05
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	TF-07
A judge shall get a notification when received a question	FJ-09	SJ-2		LOW	M-06	TF-07
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	TF-07
A judge shall be able supervise all submissions	FJ-11				M-06	

Functional requirements for Contestant

A contestant shall be able to edit their own information	FC-01	SC-1		HIGH	M-03	TF-08
When created a contestant shall receive a confirmation email	FC-02	SC-1		HIGH	M-03	TF-08
A contestant shall see which teams they are invited to	FC-03	SC-2		HIGH	M-03	TF-09
A contestant shall see which team they are a member of	FC-04	SC-2		HIGH	M-03	TF-02

A contestant shall see which teams and contests they have participated in earlier	FC-05	SC-2		MED	M-03	TF-09
A contestant shall be able to ask a question to a judge	FC-06	SC-3		MED	M-03	TF-07
A contestant shall have access to global answers from judges	FC-07	SC-3		MED	M-06	TF-07
A contestant shall be able to change his/her email	FC-02	SC-2		MED		

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

Functional requirements for Teams

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	TF-02
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	TF-02
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	TF-10

A team leader must be able to invite new members	FT-04	ST-2	Input: email	MED	M-03	TF-02
A team leader should be able to delete the team before the competition	FT-05	ST-2		MED	M-03	TF-10
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	TF-02
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	TF-10
All team information is editable in the team overview.	FT-08	ST-2		LOW	M-03	TF-10
A team must be able to deliver submissions to problems	FT-09	ST-3		HIGH	M-06	TF-11
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	TF-11

Other requirements

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	TF-07
The contest results are to be visible in the form of a highscore list.	FO-04	ST-03		MED	M-07	

0.1.5 Non-functional

The nonfunctional requirements defines, what objectives our end product needs to meet. The qualitative measures make it easier to agree on whether the requirement is fulfilled or not. Table can be interpreted in the following way:

- In ID the NF stands for Non-Functional
- Measure describes how/what the property should measure.

Speed

ID:	**Measure:**	**Value:**	**Priority:**	**Comment:**
NF-01	Response from action	< 1.5 sec	MED	
NF-02	Posting news	< 5 sec	MED	
NF-03	Edit user	< 1 min	MED	Time from submission until the whole system is updated

Size

ID:	**Measure:**	**Value:**	**Priority:**	**Comment:**
NF-04	Number of contestants	500	HIGH	
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 solution size	50kB		

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	

Reliability

ID:	Measure:	Value:	Priority:	Comment:
NF-12	Mean time to failure	> 1 week	HIGH	The program should NOT be down during a contest
NF-13	Availability	> 99.9%	HIGH	How much is the software up and running.

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	What kind of data
NF-16	Expected living time	10-15 years	HIGH	
NF-17	Execution node	= 1	HIGH	
NF-18	Execution nodes	> 1	MED	It should be possible to implement more

Portability/Scalability

ID:	Measure:	Value:	Priority:	Comment:
NF-19	Extensibility		HIGH	adding features, and carry-forward of customizations at next major version upgrade
NF-20	Module-based code		HIGH	The code should be easy to maintain

Other

ID:	Measure:	Value:	Priority:	Comment:
NF-21	Accessibility	Internal and external	HIGH	slightly more important than external
NF-22	Emotional factors	FUN	HIGH	It should be fun
NF-23	Open-source	GPL	LOW	

0.1.6 Security

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

- In ID the first two letters stands for the property of the requirement.
- Measure describes how/what the property should measure.

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, asanother user		
S-03	After an authenticated user performs an action to be logged out, thatuser will need to log in to re-authenticate		E.g. session-cookies should not remain such that you can still relogin
S-04	No user shall gain administrative rights without manual approval ofcurrent administrators		Ensure no user is registered as admin by mistake, no scripts thatautomatically escalates privileges to administrator when conditions aremet
S-05	aprovided interface and provide mandatory credentials.		Sometimesusers a identified by session ID's
S-06	To authorize, you will either need to provide mandatory usercredentials through an interface, or have a valid session ID.		
S-07	Session tokens shall be unique to one computer only		Not possible to simply acquire a session ID and use it on other computers toauthenticate

Immunity

ID	Measure	Priority	Comment
S-08	No front-end exposed input-fields shall besusceptible to injection attacks		
S-09	All data that passes the trust zone shall be in plaintext, andvalidated for injection attacks		
S-10	Except from contest program submissions, data from non-developers canonly be directed/saved in databases.		E.g. you shall not be able to create files in the source directory.
S-11	Uploaded file scripts shall not write to any file		
S-12	Uploaded file scripts shall not read from any other file than stdin		
S-13	Uploaded file scripts shall not access network, threading, or anyother external service not needed to solve a problem.		
S-14	Data from a user shall not be modified		

Non-repudiation

ID	Measure	Priority	Comment
S-15	All modifications of data shall be logged		

S-16	All log entries shall contain username(s) and a timestamp with dayand current hour		
S-17	Logs will be backed up		
S-18	A team's score shall not be affected by anything other thanwhat is given in the contest rules		

Privacy

ID	Measure	Priority	Comment
S-19	Sensitive user data shall not be stored in plaintext		
S-20	Every user-field that is stored shall be justified in therequirements specification		
S-21	No sensitive data shall be exposed publicly, even if it is encrypted		
S-22	User-data for a given user shall not be modified without that user's consent.		

Auditing

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

0.1.7 Requirements Not Met

We met most of the requirements in time to their respective milestone, but there were some minor requirements we had to drop because of 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 we also ended up with some requirements that we technically did meet, but not 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
An admin can add more than one node	FA-13
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 choice, described in **??**. NR-03 had to be overruled during the contest. This is discussed in detail in section *development*.