Analysis of Trajectories of Clarifying Communication of Requirements

by

Jyoti Sheoran
B.Tech., Maharshi Dayanand University, 2011

An Industrial Project Report Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Analysis of Trajectories of Clarifying Communication of Requirements

by

Jyoti Sheoran
B.Tech., Maharshi Dayanand University, 2011

Supervisory Committee

Dr. Daniela Damian, Supervisor
(Department of Computer Science)

Dr. Alex Thomo, Departmental Member
(Department of Computer Science)

Supervisory Committee

Dr. Daniela Damian, Supervisor
(Department of Computer Science)

Dr. Alex Thomo, Departmental Member
(Department of Computer Science)

ABSTRACT

Stakeholders of IBM® Rational Team Concert® project use text-based online tools for communication. Requirements related text-based discussions could be classified into different communication patterns. Identification of ineffective communication patterns in requirement discussions can help management to take immediate actions to ensure completion of requirement within planned time. This report presents results from a study of the trajectories of clarifying communication and the history of user requirements in RTC project. There are statistically significant differences in the six communication patterns - *discordant*, *procrastination*, *textbook-example*, *back-to-draft*, *happy-ending* and *indifferent*. Also, certain requirement attributes such as number of comments and priority significantly affect the communication pattern of a requirement.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

I wish to thank my committee members who were more than generous with their expertise and precious time. A special thanks to Dr. Daniela Damian, my supervisor for her countless hours of reflecting, reading, encouraging, and most of all patience throughout the entire process. I thank Dr. Alex Thomo, my committe member for his invaluable insight and expertise in Data Mining.

I would like to express a special word of thanks to my friends and family who tirelessly listened to my ideas and offered encouragement when it was most needed.

Finally, I would like to thank Dr. Eric Knauss and Dr. Kelly Blinoce who assisted me with this project. Their excitement and willingness to provide feedback made the completion of this research an enjoyable experience.

DEDICATION

I dedicate my report to loving memory of my grandfather Balwant Singh Sheoran.

Chapter 1

Introduction

Software products evolve with changing needs of the customers. Requirements engineering is a human-centric activity and depends on communication between stakeholders. Agile development teams use an iterative approach to facilitate and understand requirements with more emphasis on frequent communication than documentation [4]. User stories are created to depict customers requirements. Developers discuss user stories with customers to understand the requirements and to coordinate their development progress [1]. In large distributed software projects, stakeholders often use online communication tools for elicitation and clarification of requirements. Text-based online communication is effective for asynchronous communication and documentation.

IBM® Rational Team Concert® (RTC)[1] is a collaborative software lifecycle management tool built on IBM Jazz. It integrates tasks across the software life cycle. RTC project has a large distributed team. Stakeholders communicate through text-based online communication tools. The management mandates recording of all decisions in the project repository [9]. Requirements in RTC are represented as user stories and stakeholders use text-based comments to discuss the user stories. Discussions help in understanding the requirements of a user story and consensus among stakeholders. However, discussions can also result into misunderstandings and disagreements which can lead into dead-ends and stagnation of a requirement. Identification of such discussions can help management in taking immediate actions to ensure that the development progresses and finishes within the expected time.

In this report we address the problem of identifying stagnating requirements by

[1] https://jazz.net/products/rational-team-concert/

analysing the user story attributes and communication patterns of user story discussions. A trajectory of claryifying communication can be drawn throughout the lifetime of a requirement to find the communication pattern. The communication pattern together with the presence of certain management factors such as change in user story priority, estimation, completion date or resources can help identify bad progress of a requirement [10]. In our study, we analyzed the trajectories of clarifying communication and the history of user stories to understand the differences in communication patterns in requirement communication. We analyzed the change in important artefacts of a requirement that relates to management factors or decisions. We found statistically significant differences between the communication patterns based on changes in user story attributes such as priority, type, release date, etc. We also found that the number of comments significantly affect the communication pattern whereas user story priority and number of duplicates has some effect on communication pattern. We found no evidence that communication pattern can affect change in major management factors such priority, release date, complexity and allocation of workforce.

The remainder of this report is organised as follows:

Chapter 2 surveys related research in the study of communication in requirements engineering.

Chapter 3 describes our methodology in studying user requirements and user communication for requirements.

Chapter 4 presents our analysis and reports on results.

Chapter 5 discusses the findings, threats to validity of our research and suggestions for future work.

Chapter 6 concludes the research results.

Chapter 2

Literature Review

There is a lot of research about communication in requirements engineering. Al-Rawas et al. [2] found that extensive communication is required in software systems and dialectic process is crucial for clarifying issues. They also found that communication needs are higher during early phases to define terms and understand requirements [2]. Abdullah et al. [1] studied communication and collaboration in co-located agile teams and how it supports requirements activities. They found patterns of communication to manage elicitation, evolution and clarification of requirements. Cao et al. [4] studied best practices in agile requirements engineering. These include intensive communication between stakeholders and an iterative approach to understand the requirements [4]. Damian et al. [8] studied modeling of collaboration of stakeholders during development and management of requirements and the challenges in requirements-driven collaborated software development. Calefato et al. [3] found that group performance is not affected by communication medium and common ground is achieved in requirements elicitation in text-based communication. Hence, text-based communication is preferred than face-to-face communication for requirements communication in certain scenarios, such as better facilitation of task, visibility of decisions and documentation.

There has been some work in studying actual instances and content of requirements communication between stakeholders. Studies by Damian and colleagues [3] found that most common topics of discussion between stakeholders are clarification of requirements and communication of changes. Cleland-Huang et al. developed a technique to automatically detect non-functional requirements in documents and classify them into various types, such as security, performance and usability [6][7]. Kof [12] worked on extraction of semantics from natural language texts to identify ambigui-

ties in requirements specification. Lee at al. [14] studied semi-automatic extraction of ontology from requirements documents. Chantree et al. [5] developed a technique to automatically detect ambiguities in requirements using heuristics. They used word distribution in requirements to train the heuristic classifier.

Knauss et al. [11] build a semi-automatic classifier to identify clarification and other discussion events in requirements discussions in text-based communication. They found 6 patterns of communication in requirements discussions of a user story: indifferent, discordant, back-to-draft, happy-ending, procrastination and text-book example. Knauss et al. build a pattern matcher to automatically identify the pattern of communication in user story discussions [10]. The study found that the type of communication pattern in a user story along with the presence of certain management factors in that user story could help project managers in identifying bad user stories. Our study directly builds up on Knauss and colleagues research [10] to further understand and distinguish between the 6 patterns of communications in requirements discussions by studying the history of a user story and its artefacts.

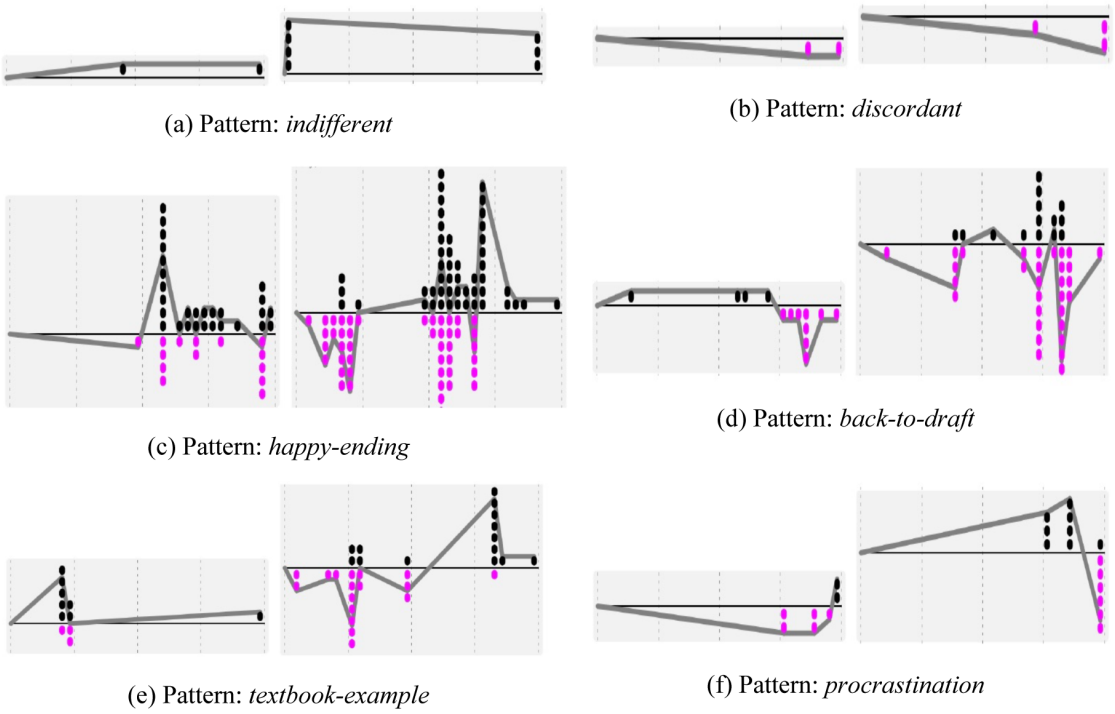

Figure 2.1: The Clarification patterns identified in the IBM® RTC project [11].

Our analysis was guided by following research questions:

***RQ1*:** *What can we say about the patterns by looking at story attributes?*

***RQ2*:** *What are the differences between the communication patterns?*

***RQ3*:** *Apart from communication itself, what user story attributes affect communication pattern?*

***RQ4*:** *Can a user story communication pattern predict the change in story attributes?*

Chapter 3

Methodology

This chapter describes the sources of data considered for better understanding of requirements discussion patterns and methods used for data analysis.

3.1 Data Sources

DiscussionAnalyzer tool and Communication Patterns dataset from Knauss et al. research [11] was used to obtain classification of discussion events of user stories and for identification of communication patterns. To collect history information of the user stories, RTC Reportable API [1] was used.

3.1.1 Communication Patterns

Communication Patterns dataset [11] contained analysis of RTC issue tracking system, a dataset between December 2006 and June 2008. The dataset contained 244 user stories. In RTC, user stories are of 5 types - *planitem*, *story*, *task*, *enhancement* and *defect*. A plan-item type user story is planned to be finished in one iteration. A plan-item is then divided into stories that are planned to be finished in one sprint. Knauss et al. [11] dataset contained a total of 96 stories, 71 tasks, 52 enhancements and 25 defects. Of the 244 user stories, 2 enhancements had plan-item as parent user story and 20 defects had no parent user story. The dataset also contained manual classification of discussion events (comments) about a user story into clarification and other communication, as well as a tool called DiscussionAnalyzer to identify the communication pattern of a user story.

[1] https://jazz.net/wiki/bin/view/Main/ReportsRESTAPI

3.1.2 RTC Reportable API

RTC provides a set of Reportable REST API [2]. These API provides some feature not available in the OSLC APIs [3], such as the ability to get a schema for a given resource and the ability to query (i.e. filter the results based on some criteria and select which fields are returned). The report API provides access to 5 monolithic resources, one of which is the user story. We queried the report API to retrieve various user story artefacts (e.g. priority, type, parent, history items etc.) for all user stories in Knauss et al. [11] dataset.

3.2 Data Processing

IBM RTC dataset containing user story discussion events and their classification was used from Knauss et al. research [11]. Then the rater is selected to classify the requirement discussions into patterns. This information in then exported into a text file. The text file was manually converted into CSV file and imported to PostgreSQL database.

RTCs reportable API returns an XML document. We wanted to save the XML data into a database to make it easier to query individual user story. We used MongoDB database as data can be easily dumped as JSON objects without worrying about the schema. A Java program was written to read the XML file and convert it into JSON object, then each user story object is stored into MongoDB database inside a collection called workitems.

The history of a user story is represented as an array, with each item in the array representing a snapshot of the user story in the timeline. We were interested in the changes in some user stories artefacts such as priority, targets, type, etc. A Java program was build up to sort the history items (array) in descending order (based on modification date) and then to compare adjacent items in the sorted array (e.g. item 0 with item 1, item 1 with item 2, etc.) and record the changes of select user story artefacts. The changes were saved into tables in PostgreSQL database.

[2] https://jazz.net/wiki/bin/view/Main/ReportsRESTAPI
[3] http://open-services.net/bin/view/Main/ReportingHome

3.3 Data Analysis Methods

We used statistical analysis methods to answer our research questions. Below is a description of the analysis methods used for each research question:

RQ1: What can we say about the patterns by looking at story attributes?
To answer this question, we first studied the distribution of user stories into the 6 patterns. We also used descriptive statistics (such as mean, standard deviation, etc.) to analyse the average value of story attributes in each pattern. We then used Kruskal-Wallis Chi-Squared Test [13] to see if the story attributes have any statistically significant difference among the communication patterns. We used Kruskal-Wallis test as it is a non-parametric statistical test and can be used to determine if there are significant differences between two or more independent variables on a dependent variable.

RQ2: What are the differences between the patterns?
To answer this question, we studied difference between individual pairs of communication patterns using Man-Whitney test [15]. We used this test because it is a non-parametric statistical test to determine if there are significant differences between two independent variables on a dependent variable.

RQ3: Apart from communication itself, what user story attributes affect communication pattern?
To answer this question, we examined the impact of user story attributes on communication patterns. We build various logistic regression models to determine whether a story attribute affects a communication pattern.

RQ4: Can a user story communication pattern predict the change in story attributes?
To answer this question, we used a logistic regression to determine the effect of communication pattern on story attributes.

Chapter 4

Analysis and Results

RQ1: What can we say about the patterns by looking at story attributes? To answer this question, we analyzed user story attributes for each communication pattern. Table 4.1 presents a summary of the distribution of user stories in 6 communication patterns. A total of 237 user stories were examined. 47.2% of user stories fall into indifferent pattern and 40% of user stories fall into back-to-draft pattern. Patterns happy-ending and textbook-example represent healthy communication of requirements of a user story. However, in the given dataset, very few user stories have these patterns (5.6% and 2.9% respectively). On the other hand, back-to-draft pattern represents ill-communication and is found in 40% of the studied user stories. Also, Indifferent pattern which shows no evidence of clarification of the requirements, constitute 47.2% of the studied user stories.

S.No.	**Communication Patterns**	**No. of requirements**	**Type of requirements**			
			Story	Task	Enh-ance-ment	Defect
1	Indifferent	112	44	32	26	10
2	Discordant	3	0	0	1	2
3	Happy–Ending	14	7	3	3	1
4	Back–to–draft	95	38	27	20	10
5	Textbook–example	7	0	6	1	0
6	Procrastination	6	6	0	0	0

Table 4.1: Distribution of user stories across communication patterns

Table 4.2 summarises the descriptive statistics of all the attributes of a user story that were studied. Row 1 to 10 analyse the count of respective attributes. For

User Story Attribute	**Mean**	**Median**	**Minimum**	**Maximum**	**Variance**	**Standard Deviation**	**Skewness**
Comments	5.21	3	34	0	30.83	5.55	2.17
Children	1.96	0	0	33	16.84	4.1	3.48
Blocks	0.08	0	0	2	0.09	0.3	3.87
Depends on	0.13	0	0	10	0.5	0.71	11.34
Duplicated by	0.24	0	0	12	0.85	0.92	9
Duplicate of	0.037	0	0	1	0.03	0.18	4.88
Related	1.06	1	0	13	2.86	1.69	3.22
Target	0.9	1	0	1	0.092	0.3	2.6
Approvals	1.13	0	0	23	10.3	3.2	3.68
Subscriptions	5.05	3	1	34	25.82	5.08	2.5
Priority	1.59	1	1	4	0.38	0.61	0.82
Targets	1.74	1	1	7	1.56	1.25	1.82
Type	1.21	1	1	3	0.2	0.44	1.92
Category	1.16	1	1	4	0.19	0.44	3.02
Resolved	0.617	1	0	2	0.34	0.58	0.35

Table 4.2: Discriptive Statistics

example, row 1 analyse the number of comments of a user story and row 3 analyse the number of child user stories. The last five rows (11-15) analyse the number of changes in the respective attribute of the user story during its lifetime. For example, row 11 analyse the number of times the priority of a user story is changed. We observe that every attribute of user story has some degree of skewness. We also observe that the median in rows 2 to 6 is 0. This could be attributed to the fact that these attributes are more relevant to the ***story*** type, which is a higher level of abstraction than other user story types (task, defect and enhancement).

Table 4.3 reports Kruskal-Wallis chi-squared tests. Some properties show significant differences between the six requirement communication patterns. The type of a user story significantly differ between the communication patterns ($W = 17.64$, $p = 0.004$). The number of comments, subscriptions and related user stories also differ significantly between the communication patterns ($p < 0.001$). The number of blocks

Property	Kruskal-Wallis Chi-Squared	P-value
Priority	9.64	0.085
Type of Requirement	17.264	0.004
Number of Comments	167.69	2.20E-16
Number of Child stories	11	0.051
Number of Related user stories	26.2	8.15E-05
Number of Subscriptions	78.36	1.85E-15
Number of Depends On	4.62	0.462
Number of Duplicate Of	5.71	0.334
Number of Duplicated By	9.14	0.103
Number of Blocks	12.7	0.026
Number of Approvals	12.08	0.033
Number of changes in Priority	10.76	0.056
Number of changes in Priority (not counting Undefined)	10.76	0.056
Number of changes in user story Type	1.96	0.854
Number of changes in Target	17.41	0.003
Number of times a user story was reopened	11.99	0.034
Number of changes in Category	8.6	0.125

Table 4.3: Kruskal-Wallis Chi-squared Test

and approvals are also different between the communication patterns ($p < 0.05$). This suggests that the communication patterns of user stories have many differential factors other than the classification of comments.

We also observe that the number of changes in target of the user story and the number of times the user story is reopened significantly differ between the communication patterns ($W = 17.41$ and $W = 11.99$ respectively with $p < 0.05$). A change in target is an example of re-scheduling of a user story (a project management factor), while reopening of a user story represents misunderstanding of the requirements. Hence, presence of both these attributes might be an indication of bad progress of a user story.

RQ2: What are the differences between communication patterns?

To answer this question we examined the differences between communication patterns using Man-Whitney test between each pair of communication pattern. Table 4 reports the results. The number of times a user story is reopened significantly

Pattern	Indifferent	Discordant	Procrast-ination	Back-to-Draft	Happy-Ending
Indifferent					
Discordant	W=244.5 p-value=0.126				
Procrast-ination	W=159 p-value=0.013	W=0 p-value=0.007			
Back-to-Draft	W=5782.5 p-value=0.226	W=96 p-value=0.293	W=447 p-value=0.010		
Happy-Ending	W=840.5 p-value=0.619	W=13.5 p-value=0.310	W=66 p-value=0.024	W=654 p-value=0.916	
Textbook-Example	W=515.5 p-value=0.110	W=9 p-value=0.662	W=39 p-value=0.003	W=398.5 p-value=0.335	W=60 p-value=0.358

Table 4.4: Mann-Whitney Tests: Number of times a user story is reopened

differentiate procrastination pattern with all other patterns ($p < 0.05$).

Also, indifferent pattern differs significantly from back-to-draft and happy-ending patterns in the number of changes in target ($p < 0.01$). Indifferent pattern also differs significantly from back-to-draft and discordant patterns in the number of changes in priority ($p < 0.05$).

The number of changes in target release of a user story also differs in discordant and happy-ending pattern ($W = 37.5$, $p < 0.05$).

RQ3: Apart from communication, what user story attributes affect communication pattern?

We constructed regression models to examine the relative impact of user story attributes on the communication pattern. Table 5 summarises the results of logistic regression. The number of comments on a user story has a significant impact on the user story communication pattern ($p < 0.001$). User storys priority and number of duplicates also has significant impact on the communication pattern ($p < 0.05$). The impact of number of blocks and related user stories is only marginally significant ($p < 0.1$).

RQ4: Can a user story communication pattern predict the change in

Pattern	Indifferent	Discordant	Procrast-ination	Back-to-Draft	Happy-Ending
Indifferent					
Discordant	W=121.5 p-value=0.361				
Procrast-ination	W=350.5 p-value=0.846	W=12 p-value=0.376			
Back-to-Draft	W=6392.5 p-value=0.006	W=211.5 p-value=0.128	W=333 p-value=0.463		
Happy-Ending	W=1164.5 p-value=0.001	W=37.5 p-value=0.034	W=61.5 p-value=0.100	W=851 p-value=0.074	
Textbook-Example	W=515.5 p-value=0.120	W=16.5 p-value=0.156	W=27 p-value=0.389	W=377.5 p-value=0.528	W=43 p-value=0.668

Table 4.5: Mann-Whitney Tests: Number of changes in release date

Pattern	Indifferent	Discordant	Procrast-ination	Back-to-Draft	Happy-Ending
Indifferent					
Discordant	W=285 p-value=0.018				
Procrast-ination	W=309 p-value=0.704	W=2 p-value=0.065			
Back-to-Draft	W=6096 p-value=0.040	W=62 p-value=0.066	W=350 p-value=0.299		
Happy-Ending	W=977.5 p-value=0.084	W=10.5 p-value=0.151	W=56 p-value=0.209	W=729 p-value=0.520	
Textbook-Example	W=419 p-value=0.728	W=4.5 p-value=0.179	W=24 p-value=0.678	W=311 p-value=0.756	W=41.5 p-value=0.562

Table 4.6: Mann-Whitney Tests: Number of changes in priority

Property	**Estimate**	**P-value**
Priority	0.22468	0.0198*
Number of changes in priority	-0.08851	0.5891
Number of changes in type	0.21702	0.2893
Number of changes in target	0.21028	0.5138
Total number of targets	0.11567	0.1511
Number of child stories	0.02166	0.3775
Number of changes in category	-0.07787	0.7075
Number of blocks	0.53054	0.0788 .
Number of stories on which given story depends on	-0.08357	0.4925
Number of stories duplicated by given story	0.03924	0.6788
Number of stories that are duplicate of given story	1.20672	0.0105 *
Number of related stories	-0.10901	0.0724 .
Number of approvals	0.04868	0.1053
Number of times stories is reopened	0.03514	0.8391
Number of subscriptions	0.03659	0.1119
Number of comments	0.14853	5.8e-12 ***
Significance codes: 0 * 0.001 ** 0.01 * 0.05 . 0.1 1**		
Multiple R-squared: 0.4265, Adjusted R-squared: 0.3861		

Table 4.7: Summary of Logistic Regression

story attributes that relate to management factors?

We also constructed regression models to examine the impact of communication pattern on user story attributes (priority, workitem type, target, number of children and reopened count), but did not find any significant impact of communication pattern on the respective user story attributes. This suggests that though there is significant difference between user story communication patterns, the patterns alone cannot determine other important user story attributes (priority, workitem type, target, number of children and reopened count).

Chapter 5

Discussion

5.1 Patterns of Clarifying Communication

The observed differences between communication patterns have consequences of both theoretical and practical interest.

Discordant user stories are the ones that have clarifying communication throughout their lifetime. We found that discordant user stories have high number of changes in priority (mean = 2.33) and it differs significantly from other bad communication patterns (indifferent, procrastination and back-to-draft). However, such user stories dont show any change in their type and targets. This suggests that the even though user stories are finished on time, their priority is not well understood. Managers must monitor such user stories.

Procrastination pattern shows that communication of user story start at the very end of the user story. The other significant difference between Procrastination pattern and other patterns is that these user stories are never reopened. These user stories also have the least average number of changes in priority. Also, these user stories have least average number of changes in targets and type. Hence, even though the communication started at the very end, but with no significant changes in management related artefacts, we conclude that this pattern does not correspond to a bad progress of a user story.

Textbook-example is a classic trajectory of clarification communication. User sto-

ries with textbook-example pattern have highest average number of changes in targets and types and the times a user story is reopened. However, we could not find any statistically significant difference. This suggests that the textbook-example is in fact a good pattern and shows expected progress of the user story.

Happy-ending pattern shows positive communication at the end, which suggests that it might have been recovered from a back-to-draft pattern. Also, the statistically significant difference between number of changes in priority and targets in indifferent pattern and happy-ending as well as discordant and happy-ending pattern also suggests that this pattern is not a bad pattern. We did not found any significant difference between happy-ending and back-to-draft patterns. This supports the claim of Knauss et al. [11] that happy-ending pattern might represent user stories that have recovered from back-to-draft pattern.

Back-to-draft pattern indicates high clarification near the end of a user story. We found statistically significant difference between back-to-draft and other bad communication patterns (indifferent and discordant) in the number of changes in priority as well as in number of changes in targets. This suggests that this pattern is significantly different from indifferent and discordant patterns.

Indifferent pattern does not show any clarification event throughout the lifetime of the user story. We found significant difference between indifferent pattern and all other patterns except textbook-example. Hence, we conclude that indifferent pattern also needs close inspection from the manager.

5.2 Threats to Validity

5.2.1 Conclusion Validity

One of the threat to our study is the conclusion validity because of small sample size. To mitigate this threat, we used nonparametric tests in statistical analysis. Non-parametric tests make no assumptions on the distribution of data and are more robust as compared to parametric tests. We used a significance level of 0.05 to draw conclusions when testing the differences between patterns in communication of a user story. The sample size could not be increased in our case because of the restrictions

posed by Knauss et al. dataset [11]. We can only acknowledge that the small sample of user stories represent a useful scenario in which we furthered our understanding in patterns of communication in requirements engineering. We encourage replications of our study where more resources are available.

Furthermore, our understanding of clarification patterns and their differences is dependent on project and process. However, we believe that our findings are applicable in any iterative development project.

5.2.2 Construct Validity

RTC user stories can have parent-child relationship. This is done for the purpose of dividing a complex user story into smaller units. For example, a user story of type task can be created from a user story of type: *story*. In our study, we only looked at individual user stories as each user story represents a unique work and has its own discussion events. Future work should investigate parent-child relationship of user stories to develop further understanding of clarification patterns.

5.2.3 External Validity

Our results are exclusively based on data from RTC team and Knauss et al. []. Hence, the ability to generalize results to other software projects is limited. Online requirements communication is becoming more common in distributed teams with the advent of online messaging applications that bring all communication together in one place e.g. Slack [1], Flowdock [2], etc. Projects using such applications are suitable to use Knauss et al. [10] technique to identify clarification patterns. Also, projects that archive history of changes in user story artifacts are suitable for the analysis that we performed in this study.

5.3 Future Work

In future work we intend to use our understanding of differences in clarification patterns and Knauss et al. [10] tool in designing a tool to assist managers in finding bad progressing user stories in real-time. We also plan to build a prototype and conduct a field evaluation with project managers to evaluate the usefulness of the designed tool

[1] https://slack.com/
[2] https://www.flowdock.com/

in a real project. Further, we want to investigate how the parent-child relationship of user stories affects clarification patterns.

Chapter 6

Conclusion

In this report, we studied trajectories of clarifying communication and history of user requirements. We collected user requirements using IBM RTC Report API to perform statistical analysis on it. Our results help us to get a better understanding of the patterns of clarifying communication. We found statistically significant differences in the communication patterns based on changes in priority, type, and release date of user requirements. We also found that the number of comments significantly affect the requirement communication patterns. User story priority and the number of its duplicates have some affect on communication patterns as well. Our results show that managers should pay special attention on any requirement that shows Discordant or Back-to-Draft pattern as these patterns relate to lack of understanding of the requirement which could lead to delay in its completion. Requirements with Indifferent pattern may also be inspected, as this pattern does not have any clarifying communication.

Bibliography

[1] Nik Nailah Binti Abdullah, Shinichi Honiden, Helen Sharp, Bashar Nuseibeh, and David Notkin. Communication patterns of agile requirements engineering. In *Proceedings of the 1st workshop on agile requirements engineering*, page 1. ACM, 2011.

[2] Amer Al-Rawas and Steve Easterbrook. Communication problems in requirements engineering: A field study. In *Proc. of Conferemce on Prof. on Awareness in Software Engineering*, pages 47–60, 1996.

[3] Fabio Calefato, Daniela Damian, and Filippo Lanubile. Computer-mediated communication to support distributed requirements elicitations and negotiations tasks. *Empirical Software Engineering*, 17(6):640–674, 2012.

[4] Lan Cao and Balasubramaniam Ramesh. Agile requirements engineering practices: An empirical study. *Software, IEEE*, 25(1):60–67, 2008.

[5] Francis Chantree, Bashar Nuseibeh, Anne De Roeck, and Alistair Willis. Identifying nocuous ambiguities in natural language requirements. In *Requirements Engineering, 14th IEEE International Conference*, pages 59–68. IEEE, 2006.

[6] Jane Cleland-Huang, Raffaella Settimi, Xuchang Zou, and Peter Solc. The detection and classification of non-functional requirements with application to early aspects. In *Requirements Engineering, 14th IEEE International Conference*, pages 39–48. IEEE, 2006.

[7] Jane Cleland-Huang, Raffaella Settimi, Xuchang Zou, and Peter Solc. Automated classification of non-functional requirements. *Requirements Engineering*, 12(2):103–120, 2007.

[8] Daniela Damian, Irwin Kwan, and Sabrina Marczak. Requirements-driven collaboration: Leveraging the invisible relationships between requirements and people. In *Collaborative software engineering*, pages 57–76. Springer, 2010.

[9] Randall Frost. Jazz and the eclipse way of collaboration. *Software, IEEE*, 24(6):114–117, 2007.

[10] Eric Knauss, Daniela Damian, Jane Cleland-Huang, and Remko Helms. Patterns of continuous requirements clarification. *Requirements Engineering*, pages 1–21, 2014.

[11] Eric Knauss, Daniela Damian, Germán Poo-Caamaño, and Jane Cleland-Huang. Detecting and classifying patterns of requirements clarifications. In *Requirements Engineering Conference (RE), 2012 20th IEEE International*, pages 251–260. IEEE, 2012.

[12] L. Kof. *Text Analysis for Requirements Engineering*. PhD thesis, Technische Universität München, 2005.

[13] William H. Kruskal and W. Allen Wallis. Use of ranks in one-criterion variance analysis. *Journal of the American Statistical Association*, 47(260):583–621, 1952.

[14] Seok-Won Lee, Divya Muthurajan, Robin A Gandhi, Deepak Yavagal, and Gail-Joon Ahn. Building decision support problem domain ontology from natural language requirements for software assurance. *International Journal of Software Engineering and Knowledge Engineering*, 16(06):851–884, 2006.

[15] H. B. Mann and D. R. Whitney. On a test of whether one of two random variables is stochastically larger than the other. *Ann. Math. Statist.*, 18(1):50–60, 03 1947.