UiO : Department of Informatics
University of Oslo

Does Limit on Work-In-Progress (WIP) in Software Development Matter?

Truls Skeie

Master's Thesis Spring 2014

Summary

Background: In software engineering there are several principles that got an impact on a software project. If these principles are applied the wrong way, or not considered, it can starve a software project. WIP-limit is of one those principles. WIP-limit is used to limit number of tasks people can work on. As of today, there is little evidence proving the impact of WIP-limit for software development.
Aim: The aim of this work is to investigate the impact that WIP-limits have on software development.
Methods: The methods used to investigate the research question were a case study of an in-house software development company. The case study was based on a data set with meta-data about each of the tasks that the software company worked on from 2008 to 2013. The data set was analyzed using an application developed for, and later described in this work. From the data set, the application measured variables such as *WIP, throughput, bugs, lead time* and *churn* for each team. The data produced by the application was interpreted with correlations and case summaries in statistical application. Correlation is a statistical method that measures how two variables change in relation to each other. Case summaries is a statistical method for grouping variables and calculate descriptive statistics. The correlation between variables is used to investigate the impact of WIP-limits.
Results: Some of the results of this work were a mean correlation of 0.4 between *WIP* and *throughput*, a mean correlation of 0.2 between *WIP* and both *bugs* and *lead time* and a mean correlation of -0.1 between *WIP* and *churn* across the teams.
Conclusion: Based on the data presented in this work, the conclusion is that WIP-limits matter in software development.

Contents

List of Figures

List of Tables

Listings

Preface

First of all I would like to thank my supervisor, *Dag Sjøberg*, for valuable discussions and great guidance throughout this work as well as Software Innovation for providing the unique data set. I would also like to thank my dad, *Nils-Olav Skeie* for great guidance and for helping me correct spelling errors and for giving me useful tips. I would also like to thank my friends *Michael Gunnulfsen* and *Brynjar Rongved* for helping me correct countless spelling errors and for giving me useful tips. I would also like to thank *Henrik Skifjeld* for helping me with Python. Last but not least, I would like to thank my girlfriend *Vilde Fjeldstad*, my mom, *Eva Solesvik*, my sister *Hilde* and all my friends for your love, support and friendship.

Chapter 1

Introduction

This work focuses on Work In Progress (WIP)-limit, which is one of the principles in Kanban. Kanban is a software development method defined as a WIP-limited pull system visualized by a Kanban board (D. Anderson et al., 2011). The Kanban method is further explained in Chapter 2. The focus of this work will be to evaluate what kind of impact WIP-limit has in a development process. In order to do so, a data set gathered by an in-house software company in Norway called Software Innovation (SI), was used. SI is a Scandinavian software company that delivers Enterprise Content Management applications.

1.1 Motivation

In software development, processes and methods are important in order to deliver the right product on time and one rarely solves two identical problems for different stakeholders. The problems in software development are becoming larger and more complex, which means that new processes and methods are introduced. And the already existing processes and methods need to be adapted to solve the complex problems in the most efficient ways. The number of popular software development methods (e.g. Extreme programming, Spiral, Scrum and Kanban) emerged in the recent years, proves this assumption (Gandomani et al., 2013) (Marko Ikonen et al., 2010).

This is the reason why this work will focus on software development methods. The methods in each development project are such a key element to make a project successful. The main focus of this work will be the Kanban method and the principle WIP-limit. In Kanban is the WIP-limit used to limit the number of tasks each developer can work on at each workflow state, to prevent bottlenecks and to ensure flow of tasks through the development cycle (Gandomani et al., 2013) (Marko Ikonen et al., 2010).

There are published various literature on Kanban in software development. (D. J. Anderson, 2010), (Kniberg, 2010), (Middleton and Joyce, 2012). Although there is various literatures, there is no information on how to apply WIP-limit, even though most of the experienced Kanban enthusiasts agree that limiting WIP is an important principle. There is no research backing up this statement. The literature states that one should experiment with WIP-limits in order to find the best WIP-limit for one's case (M. Ikonen et al., 2011) (Kniberg, 2010).

Because there is lack of available research on WIP-limit, the motivation of this work will be to investigate WIP-limit in software development.

1.2 Research Question

In this work the overall research question will be to study the effects of WIP-limits for an in-house software company, in particular:

- Does WIP-limit in software development matter?
- If so, how can one find the optimal WIP-limit?

1.3 Approach

This work will a use case study as an approach to answer the research questions. A data set from an in-house software company will be used to conduct the case study. The data set will be evaluated at team level. The software company consists of ten teams, all of them will be investigated.

A software application that was developed for this work will evaluate the data set. The software application will convert the data set into more deliberate data. The new data is interpreted by SPSS. SPSS is a statistic analysis program that was used in this work to compute correlation and descriptive statistics. A figure representing the work flow is presented in Section 4.2.

1.4 Chapter overview

Chapter 2: Background:

Chapter 2 introduces background information and relevant concepts and methods in software development.

Chapter 3: Research Methods:

Chapter 3 introduces and explains the research methods used in this work as well as complementary information about Software Innovation and why the data set from Software Innovation is used in this work.

Chapter 4: The calculation of WIP and the remaining the variables:

Chapter 4 gives information about the data set and the calculations. Complementary information about how the developed program operates is given, as well as information about how the output data from the program is measured using SPSS.

Chapter 5: Results:

Chapter 5 presents the result produced by the developed software application and SPSS. The result is presented with descriptive statistics and correlation tables.

Chapter 6: Discussion:

Chapter 6 presents a discussion on the results from the case study against the finding from prior research and the research questions from this work.

Chapter 7: Conclusion:

Chapter 7 provides the conclusion to the research questions as well as recommendations for future work.

Chapter 2

Background

In this chapter there will be a brief introduction to software development process/methods as Waterfall (Section 2.1), Scrum (Section 2.2), Lean (Section 2.3) and Kanban (Section 2.4) with affiliated tools.

2.1 Waterfall

"The waterfall model is the classical model of software engineering. This model is one of the oldest models and is widely used in government projects and in many major companies" (Munassar and Govardhan, 2010). The main goal of the waterfall model is to plan in early stages to ensure design flaws before coding is started. Since planning is so critical in the waterfall method it fits projects where quality control is a major concern (Munassar and Govardhan, 2010).

The waterfall method consists of several non-overlapping stages as shown in Figure 2.1. The figure is an example of the waterfall model with a life cycle of establishing system requirements and software requirements and continues with architectural design, detailed design, coding, testing and maintenance (Munassar and Govardhan, 2010). One of the main principles of the waterfall method discourages return to an earlier phase. For example returning from detailed design to architectural design. However, if returning to an earlier phase is needed, it involves costly rework. When a phase is completed, the phase requires formal review and extensive documentation development. Therefore, if something is missed out an earlier phase, it is expensive to correct it later (Munassar and Govardhan, 2010)

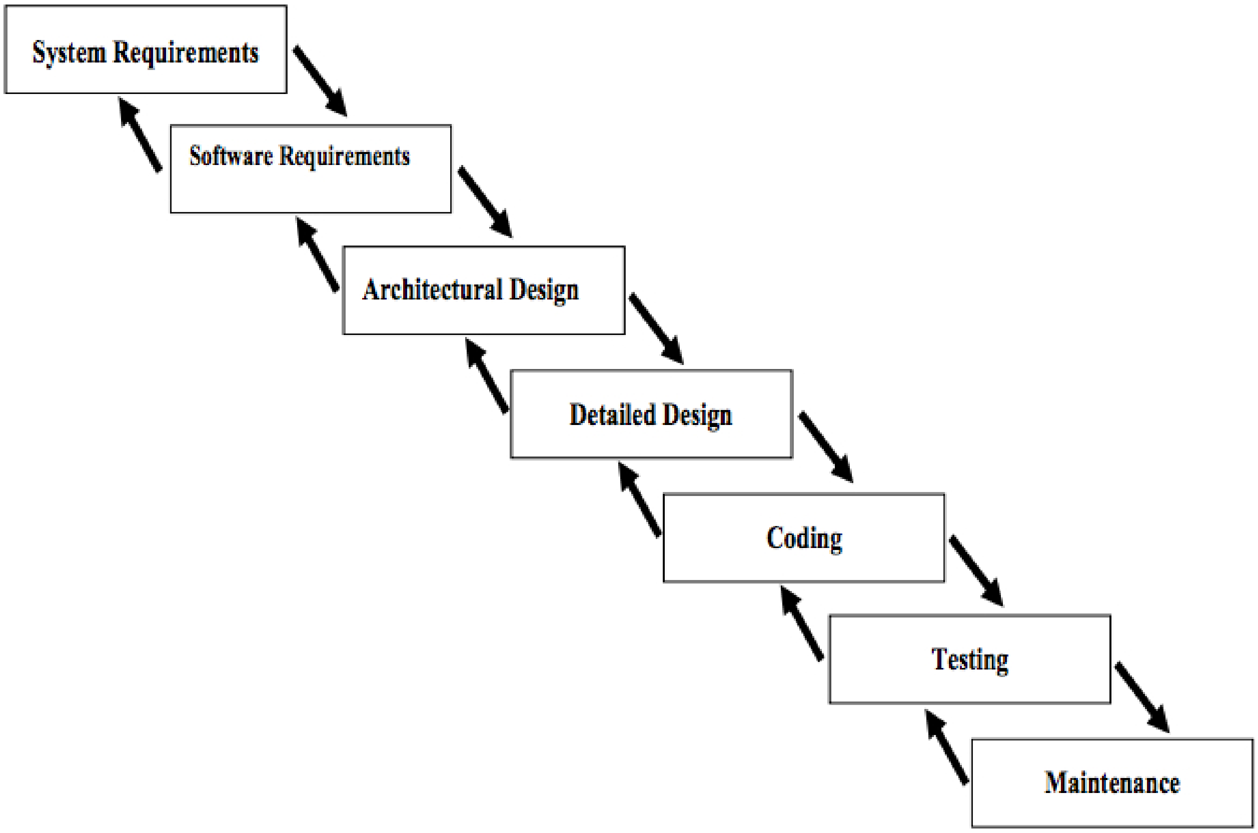


Figure 2.1: Waterfall model

2.2 Scrum

"Scrum is the best-known of the Agile frameworks. It is the source of much of the thinking behind the values and principles of the Agile Manifesto"(Alliance, 2012). The values of the Agile Manifesto are (Alliance, 2012):

Individuals and interactions *over processes and tools*
Working software *over comprehensive documentation*
Customer collaboration *over contract negotiation*
Responding to change *over following a plan*

These principles of Scrum and Agile manifesto are not so rigid as the principles of the Waterfall method. Some says that Scrum is the opposite of the Waterfall method considering the rigidness. (Cocco et al., 2011).

Scrum have three main roles; the Product Owner, the Scrum Master and the members of the development team. The Product Owner in collaboration with the Scrum Master decides which work to be prioritized in the backlog. The backlog represents the tasks to be done in order to complete the project. The Scrum Master acts like a team leader and helps the development team and the organization to take best advantages of Scrum. The development team works on tasks specific for current sprint (Alliance, 2012).

Sprint is a time-boxed interval over a given time. The Scrum framework suggests duration of sprints to be from one to four weeks. Before each sprint, a sprint planning meeting is conducted with all the team members attending. A Sprint planning meeting is held so the team can discuss tasks from the backlog and come to an agreement of which tasks to be put in the minimal backlog (Alliance, 2012).

In each sprint, a minimal backlog is created so the developer knows which tasks to work on in the current sprint. The Product Owner and the team members discuss and decide which tasks from the backlog to be added to the minimal backlog. After the minimal backlog is complete, the Product Owner and the team members discuss each task in order to get a better and shared understanding of what is required to complete the tasks (Alliance, 2012).

One of the main principles in Scrum is that it requires that at least one new feature is ready for release after each sprint. The feature should be a visible part of the product in order to get feedback from end-users. So all the tasks in the minimal backlog combined should be a visible part of the product (Alliance, 2012).

2.3 Lean

"Lean is all about getting the right things to the right place at the right time the first time while minimizing waste and being open to change" (Raman, 1998). The Lean approach was introduced around 1948 in manufacturing for Toyota. In 1975 was Toyota able to create almost 50 more production units per employee than in 1948 due to the Lean approach (Manning, 2013). Lean strives to maximize the value produced by an organization and delivered to costumer. This is done by finding and eliminating waste, controlling variability and maximizing the flow of delivered software all within the culture of continuous improvements (D. Anderson et al., 2011). In 2003, Mary and Tom Poppendieck first introduced Lean thinking to software development by publishing a book (M. Poppendieck and T. Poppendieck, 2003). In the book, Poppendieck stated that an important concept is to manage workflow with the concept of pull-systems, which means that tasks are put in production only when a costumer asks for it (M. Poppendieck and T. Poppendieck, 2009). The pull based method Kanban has in recent years been introduced more and more to software development, and is becoming one of the keys to Lean practice in software development (D. Anderson et al., 2011). In Lean there are eight fundamental principles (M. Poppendieck, 2003);

1. **Start Early:** Do not wait for details. As soon as enough information is gathered start the development activity. Get everyone involved in figuring out the details. Do not build any walls between people, make people collaborate and start a two-way communication as soon possible. This will start the learning cycle as well.
2. **Learn Constantly:** Start with a breadth-first approach, explore multiple options. The system is expected to change, so focus on creating simplicity code and robustness so the system is easy to change
3. **Delay Commitment:** In order to delay commitment, automated testing and refactoring are essential for keeping code changeable.
4. **Deliver Fast:** Deliver fast mark of excellent operational capability. The whole idea of **delaying commitment** is to make every decision as late as possible when one have the most knowledge.
5. **Eliminate Waste:** The only thing worth doing is deliver value to the costumer anything else is waste. Discover waste and eliminate it is the first key of Lean. Lean suggests using a value stream map for removing waste. A Value Stream Map (VSM) is a map over the whole company chain. VSM helps visualize where waste is located within the company.
6. **Empower The Team:** When one is going to deliver fast, there is no room for central control. The work environment should be structured so work and workers are self-directing.
7. **Build Integrity In:** Lean software is build with integrity. That's why one of the principles in Lean suggests that tests are integrated into software development just as any code, so it becomes a part of the delivered product.
8. **Avoid Sub-Optimization:** In software development it is normal to break down a complex problem into small parts of the problem in order to minimize the complexity. If some of the parts are sub-optimized, bottlenecks can occur. For example, if ten developers are hired to work on tasks, but only three testers are hired. The development process is sub-optimized since the developers will likely produce more than the tester can test and that could cause bottleneck.

2.4 Kanban

Toyota production system introduced Kanban as a scheduling system for Lean and just-in-time (JIT) production during late 1940's and in the early 1950's in order to catch up with the American car industry. The Kanban method combined with the Lean approach was a success for Toyota. The success was noticed by the software development industry among others (Conboy, 2009), (Ohno, 2001). In the recent years, the software industry has seen an increasing amount of project that applies Kanban and Lean principles (D. Anderson et al., 2011).

”One can define Kanban software process as a WIP-limited pull system visualized by the Kanban board” (D. Anderson et al., 2011). One of the most important people in Kanban software development, David Anderson also referred to as ”father of Kanban in the software development industry” (Gupta, 2013) and author of the book "Kanban: Successful Evolutionary Change for Your Technology Business"(D. J. Anderson, 2010) stated ”If you think that there was Capability Maturity Model Integration, there was Rational Unified Process, there was Extreme Programming and there was Scrum, Kanban is the next thing in that succession.” (Leonardo Campos, 2013) .

In software development, Kanban splits the major problem into many small pieces of problems. When the small pieces are defined by the team, the problems are put up on the Kanban board to visualize the problems, track what others are working on and see potential bottlenecks during development. Shinkle stated that when people start to understand Kanban, they easily discover where the bottlenecks are (Shinkle, 2009). In short, Kanban systems focus on (D. Anderson et al., 2011):

- continuous flow of work,
- no fixed iterations or sprints,
- work is delivered when it is done,
- teams only work on few tasks at the time specified by the WIP-limit and
- make constant flow of released features throughout the development.

Contrary to Scrum, Kanban does not use the principles of sprints or estimations. In Kanban the tasks do not need to be estimated or finished within a certain time. In one paper (Concas et al., 2013), the authors let the developers work with small tasks and without being interrupted with estimation of tasks or Scrum meetings, which showed that the developers become more effective. The authors found out that Scrum was too rigid for the development team. The estimation and sprint meetings worked counterproductive in their case. The authors made the developers change to Lean-Kanban. The change implied the removal of sprints and estimation. After removing sprints and estimation the teams increased the ability to perform work, lower the lead time and meet the production dates.

In the paper by Sjøberg (Sjøberg, Johnsen and Solberg, 2012), the company also felt that the Scrum approach was too rigid. The paper reported positive results when the team changed to Kanban. The company almost halved its lead time, reduced the number of weighted bugs by 10 percent, and improved productivity. Other papers also state that Scrum maybe too rigid and that's Kanban's advantages over Scrum (Beedle et al., 1999), (Brekkan and Mathisen, 2010).

2.4.1 Kanban Board

”The Kanban board makes it clear to all the team members the exact status of progress, blockages, bottlenecks and they also signal possible future issues to prepare for”(Middleton and Joyce, 2012). The Kanban board is one of many tools in Kanban, it is used to control WIP, increase the information flow with visualization (Concas et al., 2013). A Kanban board is illustrated in Figure 2.2. Each column in the figure has a intuitive name in order to describe itself so the developers easily can track where each task is.

The columns are named *Backlog*, *In progress* and *Done*. The columns can have a WIP-limit to specify how many items in progress there are allowed in the column (Middleton and Joyce, 2012). In Figure 2.2, the WIP-limit is stated under the column name. The *Backlog* column has a WIP-limit of 4, *In progress* has 5 and *Done* does not need a WIP-limit.

The yellow stickers represent the tasks. Some development teams follow the path to mark stickers with different colors representing the severities or by using one color for feature and another color for bug. In the paper by Seikola (Seikola, Loisa and Jagos, 2011), the stickers have three different colors; green, yellow and red depending on how close to overdue the tasks are. If the sticker is red, the task is already overdue, if the tasks are soon-to-overdue, then stickers are yellow. In another project, they used yellow sticky notes for scenarios, blue for bugs, pink for issues (Shinkle, 2009).

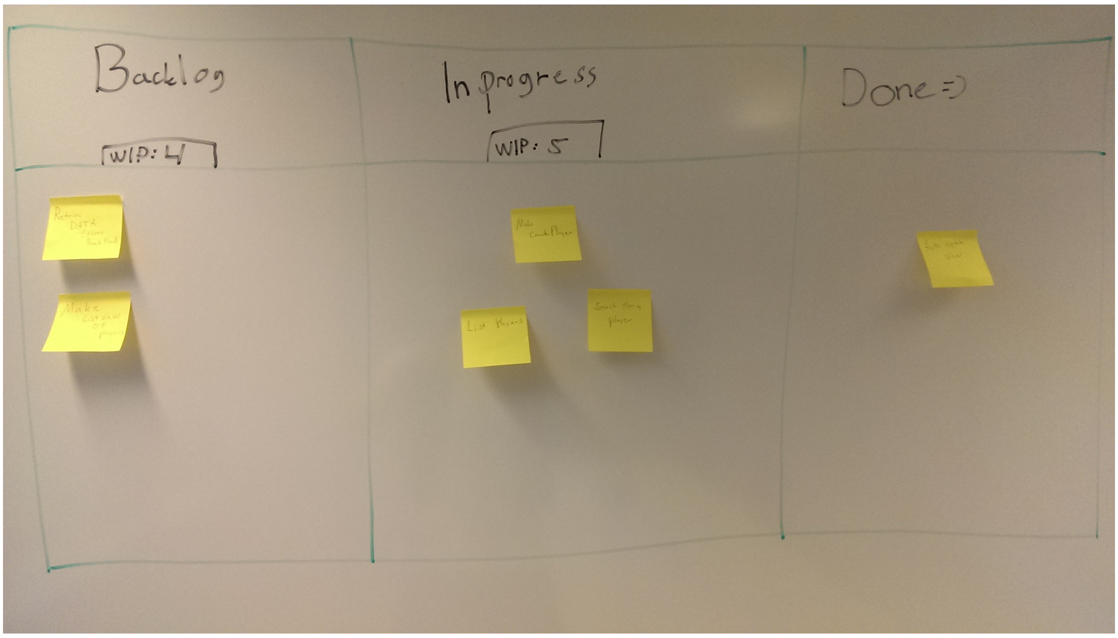


Figure 2.2: Example of a Kanban board

2.5 WIP-limit

”WIP-limits seem to be the worst understood part of the Kanban system. When used properly, it exposes bottlenecks and reduces lead time for individual work items. Used improperly, it can starve developers for work or result in too many people working on the same work items.” (Shinkle, 2009)

WIP-limit is one of the core principles in Kanban (Seikola, Loisa and Jagos, 2011). WIP-limit helps to reduce overhead by limit task switching for each developer and make constant flow of tasks throughout the development (D. Anderson et al., 2011). One way to explain WIP and the asserted impact of WIP-limit is to use cars and roads as analogy. All roads have a maximum capacity of cars. When this limit is reached, traffic jam occurs and the throughput of cars decreases and lead time increases. The same can be said about software development teams. A software team has a maximum number of tasks they can perform, if the team is pushed over the maximum limit, the throughput of tasks may decreases and lead time may increases.

When first implementing Kanban, Shinkle explained that the users often do not care about WIP or setting a WIP-limit, but rather the visibility of Kanban through the Kanban board. When users gain more experience with Kanban, they start to attempt the principles of WIP-limit (Shinkle, 2009). Srinivasan, Ebbing and Swearing said that setting the WIP-limit is not easy. They suggest that the WIP-limit is set, and then observe throughput, and adjust after that . The papers by Kniberg and Ikonen (Kniberg, 2010), (M. Ikonen et al., 2011), suggests that one start by limiting WIP, then experiment with it. David Anderson says that the WIP-limit is a policy choice. A WIP-limit of one per developer should be the starting point and it can be modified later, but the WIP-limits should be from one to three per developer. Anderson also said that it is a mistake not to set a WIP-limit (D. J. Anderson, 2010). The conclusion of the prior studies are to keep the WIP-limit low, one per developer for instance and experiment by slowly increase the WIP-limit until the throughput decreased and lead time increased, then you know that the previous WIP-limit was a good one.

On *how to determine WIP-limit*, one paper by Sienkiewicz was found (Sienkiewicz, 2012). If one implements Kanban with sprints or uses Scrum, Łukasz proposes to use the effectiveness metric to help determine the WIP-limit. The effectiveness metric shown in formula 2.1 should be applied after end sprint according to Sienkiewicz. After each sprint, one can apply the effectiveness metric and the result could be used as a guideline for WIP-limit for the next sprint. The effectiveness metric takes the number of bugs found *(ai)* and the number of bugs found by external people (e.g. lawyers, accountants, coaches, consultants, translators, internal and external service providers etc.) *(ei)*, and minus *ai* and *ei*, then divide the result by *ai* and multiply it by 100% as shown in formula 2.1.

$$Ei = \frac{(ai - ei)}{ai} * 100\% \tag{2.1}$$

Section 2.5.1 shows a summary of the the papers by Giulio Concas, Hongyu Zhang (Concas et al., 2013) and David Anderson, Giulio Concas, Maria Ilaria Lunesu, and Michele Marchesi (D. Anderson et al., 2011). The papers researched the difference between WIP-limit and unlimited WIP. Section 2.5.2 shows the importance of WIP-limit, stated by various researches.

2.5.1 WIP-limit vs. Unlimited WIP

Giulio Concas and Hongyu Zhang (Concas et al., 2013) simulated two different software maintenance processes. The first process was based on 4 years of experience with a Microsoft maintenance team. The second process was from a Chinese software firm. The simulation executed 10 runs and one of the results was the average of closed tasks were 4145 when the WIP was limited and 3853 when the limit was not limited (about 7% less). The paper concludes findings as; developers are more focused on fixing few issues rather than multi-task between tasks. When developers do not multi-task they are more likely to continue on the issue from the day before, rather than starting on another issue. This reduces overhead, because when developers start on a new issue, they need time to familiarize themselves with the code and the issue. That could create unnecessary overhead if some developer already has done it, but that developer is now working on another issue.

The study also showed that WIP-limit could improve throughput and work efficiency, because WIP-limits prevented task switching. The authors did a simulation of a process that was originally without WIP-limits, with WIP-limits. The study showed the simulated process with WIP-limits out performed the original process. (Concas et al., 2013).

David Anderson (D. Anderson et al., 2011) did a simulation of a lean-kanban approach with the impact of WIP-limit vs. no WIP-limit on developers with skills in different activities. The four skill activities from the paper were design, development, testing and deployment.

The paper did four different simulations. A simulation with WIP-limits and seven developers with skill in two of the four activities. A simulation with no WIP-limit and seven developers with skilled in two of the four activities. A simulation with WIP-limits and seven developers with skill in all of the activities. A simulation with no WIP-limits and seven developers with skill in two of the four activities.

The paper concluded that the last two is unlikely in the real world, because there is rarely a whole team with developers skilled in all activities. When the developers had skill in two out of four activities, the WIP-limit simulation used 100 days, but the simulation without WIP-limit used 120 days. The simulation with WIP-limit showed an almost constant flow of features that completed, while in the same simulation with no WIP-limit, the flow of features was much more irregular (D. Anderson et al., 2011).

2.5.2 Benefits with setting WIP-limit

This subsection contains excerpt from papers with various authors that have done study on WIP-limit.

- Lowering the WIP-limit will help people avoid task switching. When switching tasks, it is more difficult to be able to fully concentrate and it creates overhead. (M. Ikonen et al., 2011).
- There's stated when using short-cycle times and Kanban board to WIP-limit, the software development team's learning is increased (Middleton and Joyce, 2012):
- WIP-limit increases productivity (Middleton and Joyce, 2012).

- WIP-limit reduce cycle time (Birkeland, 2010)
- When WIP was too high, lead times grew and as a result so did the bugs and rework (Shinkle, 2009).
- WIP-limits are important to reduce lead times (The-Kanban-Way, 2011).

Both the studies on WIP-limit vs. no limit and the papers shows the importance of WIP-limit. If Sienkiewicz's effectiveness equation 2.1 is discarded , there is no clear rule on how to determine WIP-limit even though WIP is supposed to be a crucial principle in order to take full advantage of Kanban.

2.6 Lead time

"Lead time is the total elapsed time from when a customer requests software to when the finished software is released to the customer" (Middleton and Joyce, 2012). Lead time is measured to track how quickly software is delivered to customers (Middleton and Joyce, 2012). Lead time could be an essential ingredient when you look for the optimal WIP, if there is one. Often in a project, lead time is split into pieces, so every task has its own lead time. This gives the development teams the advantages to experiment with different WIP-limits in order to see the different lead times, then measure which WIP that suits this project the best.

According to the paper by Dag Sjøberg (Sjøberg, Johnsen and Solberg, 2012) the citation by Middleton and Joyce above is best suited for consultancy companies with costumers who requests tailored software solutions. The paper defined lead time as the amount of time that passed from the moment that the development team receives a request to the moment that it completes the work item. The reason why the paper disapproves the definition by Middleton and Joyce is because: "The amount of time a work item remains in the backlog queue before it is put on the board is a function of priority, not whether the company uses Scrum, Kanban or other development methods. Furthermore, companies that develop and sell products to many customers might propose new features themselves and put them on the backlog before any customers request them. Second, given a policy of two or three releases a year, the result of a work item isn't delivered to the customer immediately after it is finished" (Sjøberg, Johnsen and Solberg, 2012).

2.7 Just-In-Time

"Just-In-Time is based on delivering only the necessary products, to the necessary time and the necessary quantity" (Lai, Lee and Ip, 2003). Just-In-Time (JIT) was introduced in the 1970s by Toyota in combination with Lean (Javadian Kootanaee, Babu and Talari, 2013). JIT has been introduced to increase productivity through waste reduction and increase the value added in the production processes. To explain the JIT principle, Mary and Tom Poppendieck use the picture shown in Figure 2.3 (Lai, Lee and Ip, 2003) (M. Poppendieck and T. Poppendieck, 2006). The stream reflects the inventory. Under the stream are the rocks located in different sizes. The rocks illustrate waste and problems that can occur. If the stream level is lowered, the rocks are more visualized. At this point you have to clear out rocks (remove waste and problems) in order to make the boat continue it is journey, or it will crash into the rocks. After the rocks are cleaned out, one can lower the stream level again and continue the procedure until there are only pebbles left. Then the boat can float without problems.

If one lowers the stream (inventory), problems and waste will become visible. Lean wants to lower inventory in order to make problems and waste occur, because when problems and waste occurs, one are able to fix the problems and remove the waste. Fixing the problem and removing the waste have several benefits such as; your process could be optimized and you are one step closer to have zero problems and zero waste. (Lai, Lee and Ip, 2003) (M. Poppendieck and T. Poppendieck, 2006).

In Software development the JIT principle means one should not deliver anything before it is demanded. For example, a development team adds two new features to a product without the stakeholders asking for it and it turns out the stakeholders do not want it. Then the team has produced waste.

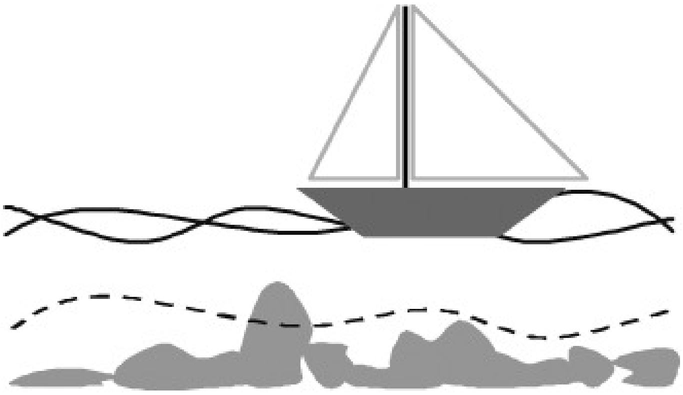


Figure 2.3: JIT example

2.8 Throughput

”The output of a production process (machine, workstation, line plant) per unit time (e.g., parts per hour) is defined as the systems throughput or sometimes throughput rate” (Adams and Smoak, 1990). The main concept of throughput is to measure how productive teams, people or companies are. Throughput is measured in number of finished delivered tasks or units per hour, day, week, month, quarter or year. A key factor in successfully measuring throughput in software development is to specify a standard size for each task. If the standard is not specified there is little use in throughput measurements (Rouse, 2005). To illustrate throughput with different task sizes an example is provided:

Team x had a throughput of eighteen tasks after the first quarter, twenty after the second, fifteen after the third and twelve after the last quarter. Team x used Scrum the first two quarters and Kanban the last two as illustrated in table 2.1. It will look like team x benefits most from Scrum. But if the task during the Kanban time was twice the size of Scrum, Kanban would suite team x the best. So, to get valid result from throughput measurements, the size of tasks has to be agreed upon by the teams or company.

Table 2.1: The throughput table for team x

Quarter	Throughput	Method
1	18	Scrum
2	20	Scrum
3	15	Kanban
4	12	Kanban

2.9 Code churn

"Churn is defined as the sum of the number of lines added, deleted, and modified in the source code" (Sjøberg, Johnsen and Solberg, 2012). Churn is a measure that is not as familiar as lead time, throughput or WIP in the software development. Churn is a term used as surrogates for effort in software engineering. Many studies in software development use code churn or revisions as surrogate measure of effort (D. Sjøberg, Anda, Mockus et al., 2012). Emam stated that "analysts should be discouraged from using surrogate measures, such as code churn, unless there is evidence that they are indeed good surrogates" (El-Emam, 2000). The study by Sjøberg, Johnsen and Solberg, showed that churn could be used as a surrogate for tasks size (D. Sjøberg, Anda, Mockus et al., 2012).

Chapter 3

Research Methods

In this chapter the research methods used in this work will be introduced, complementary information about SI and the reason why the data set from Software Innovation was chosen are explained. Section 3.1 gives a brief introduction to the research method *Case Study*. Section 3.2 is about SI, Section 3.3 is about the choice of case and Section 3.4 is about the correlation method used.

3.1 Case study

To answer the research questions, a case study was conducted. A case study is used to explore causation in order to find underlying principles (Shepard and Greene, 2002)(R. K. Yin, 2008). But which methods one can use in a case study or how the case study is conducted is ambiguous. Case study may be qualitative or quantitate. A case study might utilize a particular type of evidence (for example ethnographic, participant observation or field research). Platt stated: "Much case study theorizing has been conceptually confused because too many different themes have been packed into the idea *case study*" (Gerring, 2006). Gerring stated: "A case study may be understood as the intensive study of a single case where the purpose of that study is – at least in part to shed light on a larger class of cases (Gerring, 2006). As one can see, there is no clear rule for how to conduct a case study or what it is.

In this work, the case study is used to explore WIP-limits effect in software development. The purpose is to shed light on WIP-limit in software development and if it matters.

3.2 Software Innovation

Software Innovation[1] is a Scandinavian software company. SI develops and delivers Enterprise Content Management applications that helps organizations improve and increase efficiency in document management, case handling and technical document control. SI builds products around the Microsoft Sharepoint platform. (Sjøberg, Johnsen and Solberg, 2012), (*Software Innovation* 2013). SI has approximately 300 employees in Oslo, Copenhagen, Stockholm and Bangalore (*Software Innovation* 2013).

Figure 3.1 shows the size of the ten teams vs. quarter. The team size is used as a variable to compute the result presented in Chapter 5. Team seven, shown in Table 3.1g contributes data from 2010 to 2012. After 2012, team seven was shut down.

Year	Quarter	Team Size
2010	3	6
2010	4	3
2011	1	16
2011	2	28
2011	3	2
2011	4	38
2012	1	35
2012	2	34
2012	3	32
2012	4	29
2013	1	24
2013	2	37
2013	3	23
2013	4	23
Total		330

(a) Team size - team one

Year	Quarter	Team Size
2010	3	10
2010	4	15
2011	1	13
2011	2	12
2011	3	15
2011	4	14
2012	1	15
2012	2	7
2012	3	8
2012	4	9
2013	1	10
2013	2	7
2013	3	7
2013	4	8
Total		150

(b) Team size - team two

Year	Quarter	Team Size
2010	3	6
2010	4	9
2011	1	7
2011	2	10
2011	3	9
2011	4	10
2012	1	11
2012	2	11
2012	3	13
2012	4	13
2013	1	13
2013	2	7
2013	3	8
2013	4	8
Total		135

(c) Team size - team three

Year	Quarter	Team Size
2010	3	3
2010	4	8
2011	1	4
2011	2	4
2011	3	4
2011	4	4
2012	1	4
2012	2	2
2012	3	3
2012	4	5
2013	1	7
2013	2	5
2013	3	5
2013	4	5
Total		63

(d) Team size - team four

Year	Quarter	Team Size
2010	3	5
2010	4	13
2011	1	14
2011	2	25
2011	3	21
2011	4	23
2012	1	25
2012	2	19
2012	3	24
2012	4	18
2013	1	31
2013	2	29
2013	3	27
2013	4	11
Total		285

(e) Team size - five

Year	Quarter	Team Size
2010	3	5
2010	4	6
2011	1	6
2011	2	6
2011	3	5
2011	4	5
2012	1	4
2012	2	6
2012	3	6
2012	4	9
2013	1	9
2013	2	9
2013	3	9
2013	4	14
Total		99

(f) Team size - team six

Year	Quarter	Team Size
2010	3	10
2010	4	8
2011	1	8
2011	2	6
2011	3	8
2011	4	9
2012	1	10
2012	2	5
2012	3	9
2012	4	3
Total		76

(g) Team size - team seven

Year	Quarter	Team Size
2010	4	2
2011	1	8
2011	2	8
2011	3	13
2011	4	9
2012	1	10
2012	2	2
2012	3	25
2012	4	11
2013	1	22
2013	2	21
2013	3	23
2013	4	8
Total		162

(h) Team size - team eight

Year	Quarter	Team Size
2010	4	5
2011	1	8
2011	2	7
2011	3	7
2011	4	9
2012	1	10
2012	2	8
2012	3	10
2012	4	12
2013	1	8
2013	2	9
2013	3	8
2013	4	8
Total		109

(i) Team size - team nine

Year	Quarter	Team Size
2010	3	3
2010	4	11
2011	1	12
2011	2	9
2011	3	4
2011	4	17
2012	1	20
2012	2	17
2012	3	18
2012	4	13
2013	1	17
2013	2	9
2013	3	10
2013	4	10
Total		170

(j) Team size - team ten

Figure 3.1: Caption of team size for teams in SI

[1] http://www.software-innovation.com/

3.2.1 Software Innovation's development process

From 2001 to 2006 SI used the Waterfall process with a life cycle of (Sjøberg, Johnsen and Solberg, 2012):

1. Design
2. Implementation
3. Testing
4. Deployment for each new release

In 2007, SI examined their development process, which resulted in a decision to change to Scrum. Scrum was implemented with the standard elements of Scrum (Sjøberg, Johnsen and Solberg, 2012):

- Cross functional teams
- Sprint planning meetings
- Estimation of work items using planning poker
- Daily standup meetings
- Sprints

SI implemented three weeks sprint and after each sprint a fully tested shippable system was ready. In 2010, SI went from Scrum to Kanban. SI felt that Scrum was too rigid and did not fit their purpose, they also feared that inaccurate estimation and time boxing gave them longer lead time. SI also saw Scrum planning meetings as waste that reduced productivity and quality (Sjøberg, Johnsen and Solberg, 2012).

SI decided to implement Kanban in the following manner. When a work item is pulled from the backlog, SI tries to make the item flow through all the stages until it is ready for release. This procedure happens as quickly as possible. In order for an item to be ready for release, it has to be at a satisfactory quality level, which is defined by SI. SI also implemented WIP-limits. If the WIP-limit is reached, no new tasks are started until another task is finished, which is based on the principle of just-in-time (Sjøberg, Johnsen and Solberg, 2012).

3.3 Choice of case

The data set from SI contains information about each task that SI has worked on from 2008 to 2013. The data set is represented with help of Microsoft Team Foundation Server (TFS) (Microsoft, 2013). An excerpt of some of the columns from SI's TFS is shown in Table 3.1. Although the data set contains items from 2008-2013, data from year 2008, 2009 and the two first quarters of 2010 will be excluded. The dates will be excluded partially because the transition between processes and it was inaccurate measurements when SI first started with TFS.

The reason SI and the data set from SI is analyzed in this work is because a prior research done by inter alia Sjøberg (Sjøberg, Johnsen and Solberg, 2012) used the data set. Since Sjøberg is the supervisor of this work and had access to the data set, so it was convenient to use the same data set.

Table 3.1: Excerpt from the data set

ID	Type	Created Date	From Day	Date To	Lead Time	Team
3027	Bug	2008-10-07	2008-10-09	2008-10-16	20	Team one
3028	Bug	2008-10-07	2008-10-07	2008-10-08	10	Team six
3029	Feature	2008-10-07	2008-12-30	2008-12-30	105	Team two
3030	Feature	2008-10-07	2008-10-07	2008-10-07	1	Team three
3035	Bug	2008-10-08	2008-11-20	2008-11-28	17	Team five
3037	Feature	2008-10-08	2008-10-19	2008-10-19	7	Team three
3040	Bug	2008-10-10	2008-11-19	2008-11-19	48	Team one

The data set contains thirty columns with different data for each task, most of these columns are irrelevant for this study, but the important columns are stated in Table 3.2.

Table 3.2: Variables from the SI dataset

Variable	**Description**
Created Date	When a task is put in backlog
Date From	When a given task is pulled out from the backlog
Date to	When a task is finished and ready for release.
Lead Time	The amount of days elapsed from the date the task was created until the task has finished
Type	The value of the type column is *feature* or *bug*.
Lines added	Number of lines added to a task
Lines modified	Number of lines modified when working on a task
Lines deleted	Number of lines deleted from a task
Team	States the team who has been working on the task.

The **Created date** column consists of dates for when tasks were created. The **Date from** column contains date from the tasks was pulled from the backlog. The **Date to** column consists of dates when tasks were marked as finished. The **Lines added**, **Lines Modified** and **Lines Deleted** column contains the amount of lines added, modified or deleted in order to finish the task. The **Type** column consists of a string that has the value as either *bug* or *feature*. The **Lead time** column consists of the lead time value, measured in days. The **Team** column consists of the team who is assigned to the task.

The data from SI was analyzed on team level using the developed software application and SPSS. The software application computed the variables shown in Table 3.3 for all of the teams.

Table 3.3: Relationship between variable and columns from SI

Computed variable	Description	Columns from SI
WIP	Tasks in progress on the given day	Date From and Date To.
Throughput	Number of tasks finished on a given day	Date To
Churn	Lines added, lines modified and lines deleted added together	Lines Added, Lines Modified, Lines Deleted and Date To
Bugs	The number of tasks created labeled as bug	Type and Date to
Lead time	The time used on a task, measured in days	Lead time and Date To
Bugs finished, quarter	Number of bugs finished, per quarter	Created date, Date to and Type
Avg days backlog, bug	Mean days in backlog for bugs, per quarter	Created date, Date from and Type

Both the variables churn and throughput were split to two moderating variables with suffix of *feature* and *bug*. The variable with suffix of *feature* means tasks labeled with type *feature* are the only one that counted and the same for variables with suffix *bug*. These variables are referred to as moderating variables in this work. The *Bugs finished, quarter* variable represents how many tasks labeled *bug* that are finished within the same quarter as it was created. The *Avg days backlog, bug* variable represent the average number of days bugs were in backlog before it was pulled out.

3.4 Correlation

The correlation coefficient between two variables is used to reflect the linear relationship between these two variables. The most common is Pearson correlation. The range of the correlation is [-1, +1], where +1 represents a perfect positive relationship and -1 represents a perfect negative relationship (L. Yin, Xiao and Xu, 2013). In this work it is chosen to look at the linear relationship between two variables of interest, based on this, the Pearson correlation is used.

Chapter 4

The calculation of WIP and the remaining the variables

This chapter introduces how the software application's algorithm works as well as a brief introduction to SPSS. The first section gives a short introduction to the statistical analyzes program SPSS (Section 4.1). The next section, Section 4.3 introduces the algorithm of how the developed program measures WIP for each day. The subsection 4.3.4 provides a comprehensive example of how the program measures WIP per day. The consecutively sections reveal the algorithms of how the program measures throughput (Section 4.4.1), churn (Section 4.4.2), lead time (Section 4.4.3), the moderating variables (Section 4.4.5), number of bugs finished per quarter (Section 4.4.6) and mean days for bugs in the backlog (Section 4.4.7).

Table 4.1 shows how quarters, dates and days are represented in this work.

Table 4.1: The standard of the data set

- The date standard is specified as YYYY-MM-DD.
- All seven days in the week are considered when the software application calculates.
- Quarter of a year is defined as (Investopedia, 2013):
  - January, February and March (Q1),
  - April, May and June (Q2),
  - July, August and September (Q3),
  - October, November and December (Q4).

4.1 SPSS

"IBM®SPSS®Statistics is a comprehensive system for analyzing data. SPSS Statistics can take data from almost any type of a file and use them to generate tabulated reports, charts and plots of distributions and trends, descriptive statistics, and complex statistical analyses." (IBM, 2014). SPSS will be used to analyze the derived data from the developed software application by using two statistics method; correlation and case summaries.

4.2 The workflow

The workflow for creating the result in Chapter 5 is showed in Figure 4.1. First is the data converted from Microsoft TFS server to an excel document. The excel document is then converted to a .csv file. The .csv file is measured by the developed software application. The application produce documents containing WIP, throughput, lead time etc based on the .csv file. Then the documents produced by the application are used by SPSS to compute correlation tables and descriptive statistics.

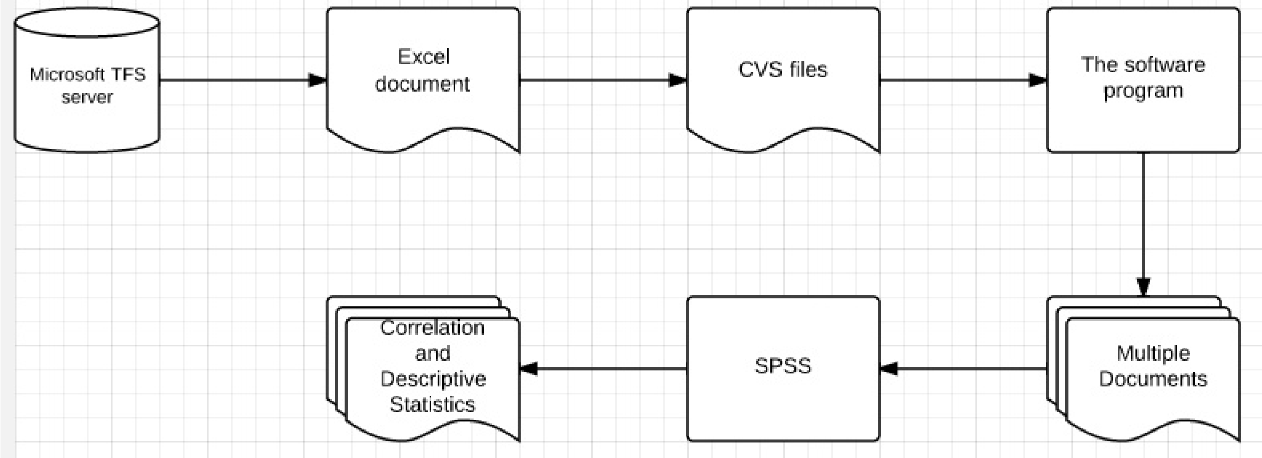


Figure 4.1: The workflow

4.3 WIP measurement per day

4.3.1 Step 1: Gather all rows into a Arraylist

The first step of this WIP algorithm is to create a WIP object with the attributes in Table 4.2. The values that are assigned to the object are gathered from the data set, by reading on row at the time as shown by Listing 4.1. After the values are assigned to the WIP object, the program puts the WIP object into the right Arraylist[1] based on the team variable as shown in Listing 4.2.

Table 4.2: Variables of the WIP objects

Type	Variable name
Date	start
Date	end
String	team
String	processType
int	WIP

[1] Arraylist is a resizable array implementation of a list. The Arraylist class provides function for manipulating the size of the array, check the size of the list and convert the list to an array (Oracle, 2013).

```
While inputFile != EOF // EOF = End Of file
     WIP = New WIP()
     WIP.start = inputFile.start
     WIP.end = inputFile.end
     WIP.team = inputFile.team
     WIP.processType = inputFile.processType
     WIP.WIP = 1
     FindTeam(WIP)

```

Listing 4.1: Gather all unique dates into Arraylist

```
void FindTeam (WIP w)
     if w.team EQUALS "TeamOne"
        TeamOne.add(w)
     if w.team EQUALS "TeamTwo"
        TeamTwo.add(w)
     if w.team EQUALS "TeamThree"
        TeamThree.add(w)
/* And so on for the rest if the seven teams */

```

Listing 4.2: Gather WIP object to the right data structure

4.3.2 Step 2: Gather the remaining dates

There were some dates missing from the data set. The software application has to create those. To create the remaining dates, the program takes the first date and the last date from each of the teams' Arraylist, presented in line 1 and 2 of Listing 4.3. Each of the Arraylists are sorted by date. Then the program checks if all the dates between the first date and the last date are in the team's Arraylist. If the date is not in the Arraylist, the program will generate the date and put it into the right place in the Arraylist, which is done by the method addToArraylist showed in the lines 10-13, as presented in Figure 4.3. In order to keep the pseudocode simple, the generateWIP method stated in line 12 was omitted. The generateWIP method creates a new WIP object and returns it.

```
1  WIP first = Arraylist.get(0)
2  WIP last = Arraylist.get(Arraylist.size()-1)
3  Next_date
4  Next_date = first.getDate() // Next_date assigned before iteration
5  while Next_date NOT EQUALS last.getDate()
6    New_date = Next_Date + 1 //Compute the next date
7    AddToArraylist(New_date, first.getTeam())
8    Next_date = New_date
9
10 void addToArraylist(Date d, String team)
11   if d NOT CONTAINS IN Arraylist
12     WIP = generateWIP(d, team)
13      Arraylist.add(WIP)
14
```

Listing 4.3: Gather the remaining dates.

4.3.3 Step 3: Measure WIP

The each teams' Arraylists from section 4.3.1 and 4.3.2 now contains a WIP object for each date. In this step, the program will loop through each of the teams Arraylists. During the iteration each WIP object is extracted from the Arraylist and the WIP is measured. The two methods stated in line 10 and 17 in Listing 4.4 respectively gather how many tasks there were in process on that date (method in line 10) and finds how many tasks are finished on that date (method in line 18) and returns the result. The result is used in line 6 to compute the current WIP. The conditional statement on line 4 assures that only one instance of each date is measured.

```
void measureWIP()
  lastWIP = 0
  for WIP Object IN Arraylist
    if(DateNotMeasured(WIP.getStartDate()) == true)
      WIP_for_this_date = get_current_WIP(WIP.getStartDate())
      WIP_measured = WIP_for_this_date − Nr_of_finishedDates(WIP.getStartDate
    ) + lastWIP
      WIP.setWIP(WIP_measured)
      lastWIP = WIP_measured

int get_current_WIP(Date date)
  current_WIP = 0
  for WIP in Arraylist
    if date EQUALS WIP.getStartDate()
      Nr_of_dates_to_decrement++
return current_WIP

int Nr_of_finished_dates(Date date)
  Nr_of_dates_to_decrement = 0
  for WIP in Arraylist
    if date AFTER WIP.getEndDate() DO
      if date not picked
        Nr_of_dates_to_decrement++
        dateIsPicked(WIP)
return Nr_of_dates_to_decrement

```

Listing 4.4: WIP measurement

4.3.4 Example of the step 1, 2 and 3 of WIP measurement

This section will provide a comprehensive example of how the WIP algorithm works. Figure 4.2 shows task ids on the y-axis and dates on the x-axis. These task ids and dates are the same as the one in Table 4.3. The green line in Figure 4.2 indicates the duration of the task. The figure helps visualize how many WIPs there are in progress for a given date. For example on the date 2010-10-12, tasks 3, 5 and 6 are in progress, which means the WIP is 3 for 2010-10-12. The data in Table 4.3 represents data from TFS. The table will be used to illustrate how the algorithm measures WIP. The first section, Section 4.3.4.1, explains how the program gather each row from Table 4.3 into the Arraylist, the next section, Section 4.3.4.2, explains how the program measure the remaining dates. The last section, Section 4.3.4.3 explains how the WIP per day is calculated.

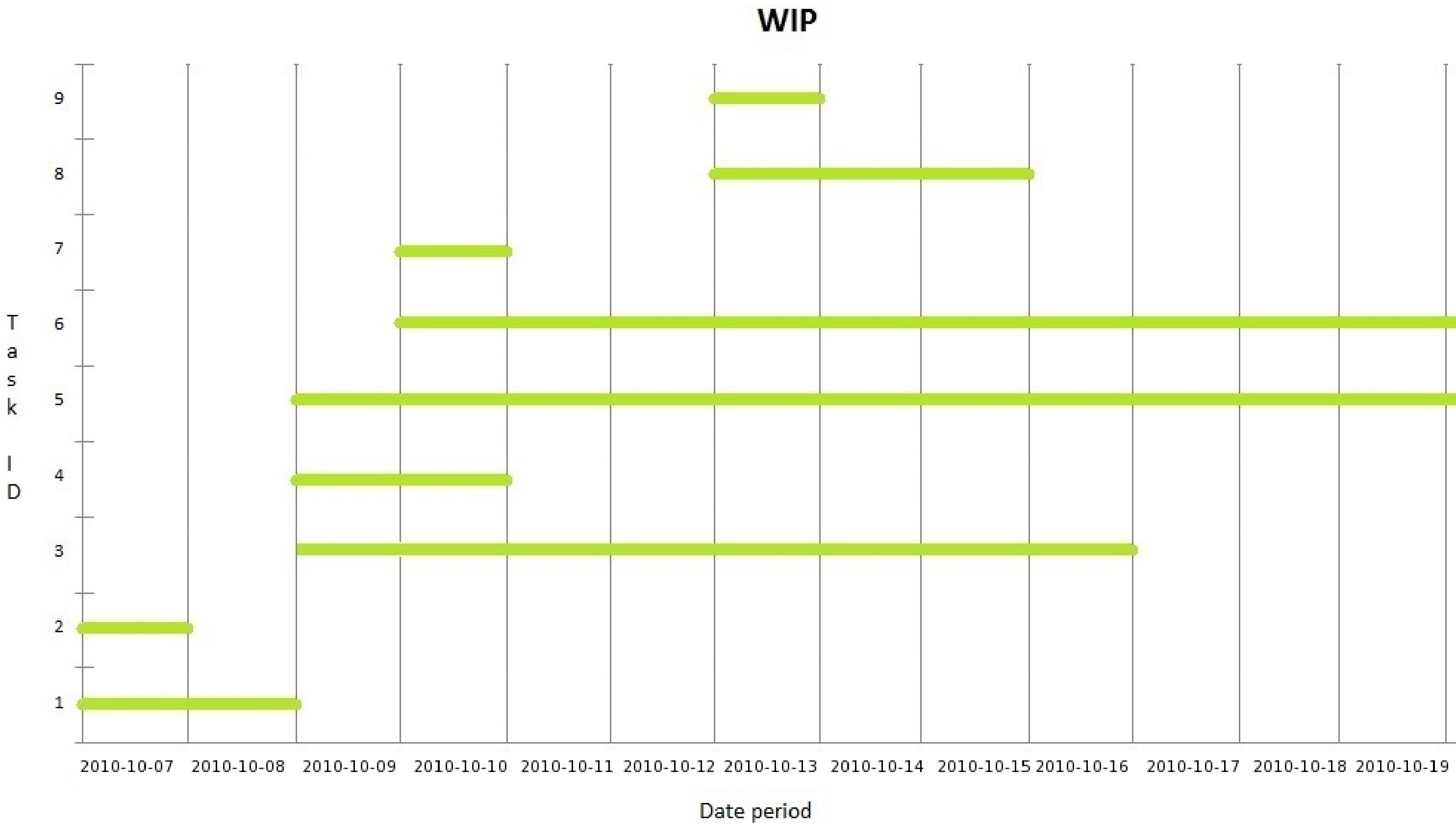


Figure 4.2: Illustrating the WIP timeline. The x-axis represents the dates and the y-axis represents the tasks id from Table 4.3

Table 4.3: Showing Task ID, Date From and Date to

Task ID	Date From	Date To	Team	Process Type
1	2010-10-07	2010-10-08	Team One	Kanban
2	2010-10-07	2010-10-07	Team One	Kanban
3	2010-10-09	2010-10-16	Team One	Kanban
4	2010-10-09	2010-10-10	Team One	Kanban
5	2010-10-09	2010-11-04	Team One	Kanban
6	2010-10-10	2010-11-05	Team One	Kanban
7	2010-10-10	2010-10-10	Team One	Kanban
8	2010-10-13	2010-10-15	Team One	Kanban
9	2010-10-13	2010-10-13	Team One	Kanban

4.3.4.1 Step 1: Gather all rows into a Arraylist

The program will first read in line 1 with task id 1, showed in Table 4.3. The program creates the WIP-object for line 1, shown by Listing 4.5. After the WIP-object is created it is put in team one's Arraylist. The program will follow the exact same procedure until all the dates are read.

```
WIP = new WIP()
WIP.start = 2010-10-07
WIP.end = 2010-10-08
WIP.team = "Team One"
WIP.processType = "Kanban"
WIP.WIP = 1
```

Listing 4.5: Creating WIP-object

4.3.4.2 Step 2: Gather the remaining dates

Now that the whole set has been read and saved, the next thing to do is to create the remaining dates. The Arraylist contains all the dates from Table 4.3. The program will now extract the first and the last date from the Arraylist. Before this step, the objects in the Arraylist are sorted by date. The first date is 2010-10-07 and the last date is 2010-10-13. The program will check if the date after 2010-10-07 contains in the set, which it does not. The program then generates a WIP object for the date 2010-10-08 and adds it to the Arraylist. After the date is created, the program will see if the date 2008-10-09 exists and will do so for the rest of the dates.

4.3.4.3 Step 3: Measure WIP

The Arraylist now contains the dates from 2010-10-07 to 2010-10-13. The last step is to measure WIP for each date. The program will now loop through the Arraylist. The first date in the arraylist is 2010-10-07. The get_current_wip method from line 9 in Listing 4.4 will be called with the date 2010-10-07 as parameter. The method will return two, because both tasks one and two were started at 2010-10-07 as shown by Figure 4.2. The next thing to do is to find out how many tasks to decrement the current WIP with. The method Nr_of_finished_dates in line 17 is called with the date 2010-10-07. The Figure 4.2 shows that there was no task finished at the date 2010-10-07, so the method returns 0. The program then updates the WIP objects' counter to be two and saves the WIP value in the lastWIP variable. The next date is 2010-10-08, which the program made in Subsection 4.3.4.2. There is no task started at 2010-10-08, but task 1 is finished at the date. So the Nr_of_finished_dates returns one and flags the current date as shown in Listing 4.4 by the line 23. The result of WIP_measure in line 5 is 1, because the WIP from previous date was 2 ($0 - 1 + 2 = 1$). Therefore the WIP at 2010-10-08 1, as shown by Figure 4.2. The program will continue this procedure until all the dates are measured. The reason why the date is flagged in Nr_of_finished_dates method is to be sure that each date is only evaluated ones.

4.4 The Remaining variables

To compute *throughput*, *churn*, their moderating variables, *lead time*, *churn*, *bugs finished*, *quarter* and *avg days backlog*, *bug* a new algorithm is required. The first part of the algorithm for the remaining variables is identical. The identical algorithm reads the data set from SI. For each of the lines in the data set, the program creates an object and saves the needed information from the data set in the object. Then each object is saved in an Arraylist based on team association, as showed in Listing 4.6. After all the lines have been read and all objects have been put in the right data structure the algorithms differ in respect of what is going to be measured.

```
void addBug(Bug b)
  if b.team EQUALS "TeamOne"
    if dateExists(b.date, TeamOne) EQUALS false
      // if date does not exists, then add the bug
      TeamOne.add(b)

  if b.team EQUALS "TeamTwo"
    if dateExists(b.date, TeamTwo) EQUALS false
      // if date does not exists, then add the bug
      TeamTwo.add(b)

  if b.team EQUALS "TeamThree"
    if dateExists(b.date, TeamThree) EQUALS false
      // if date does not exists, then add the bug
      TeamThree.add(b)

  if b.team EQUALS "TeamFour"
    if dateExists(b.date, TeamFour) EQUALS false
      // if date does not exists, then add the bug
      TeamFour.add(b)

      /* And so on for the rest of the teams */
```

Listing 4.6: Pseudocode example of how objects are added

4.4.1 Throughput

When the steps described in Section 4.4 are finished, the program takes the teams data structures and compute throughput. To compute throughput, a counter representing the throughput for each date is created. The method dateExists in Listing 4.7 does the actual computation. The method is called for each object from the data set. The method starts of with a test. If the date of the object is in the data structure, the corresponding counter is incremented. If the date is not in the data structure, the new object is added to the data structure.

```
void dateExists(Throughput tp, Arraylist list)
  for Throughput t in list
    if t.date EQUALS tp.date
      t.counter++
      return

  structure.add(tp);
```

Listing 4.7: Pseudocode example of how throughput is measured

4.4.2 Churn

As stated in Section 2.9, in order to take churn into account, one has to know its good surrogates. SI has gathered churn with help of TFS (Sjøberg, Johnsen and Solberg, 2012). The TFS system automatically records data such as churn and lead time. Based on the TFS, one knows that churn for SI is a good surrogate.

To measure churn the data set from SI contains three columns (*Lines added*, *Lines modified* and *Lines deleted*) shown in Table 4.4. These three variables are summed together and saved in a variable called *churn*. For example; for task id 1 the churn is 2028 ($352 + 307 + 1369 = 2028$), as presented in Table 4.4. Some tasks has zero churn, for example task with id 6, these tasks do not need code in order to be finished such tasks need technical support to be finished. The churn algorithm is shown in Listing 4.8

```
void updateChurn(Churn c, Arraylist list)
  for Churn ch in list
    if ch.date EQUALS c.date
      ch.churn += c.linesAdded()+c.linesModified()+c.linesDeleted()
      return
  structure.add(c);
```

Listing 4.8: Pseudocode example of how throughput is measured

Table 4.4: How churn is presented in the TFS document

Task id	Lines added	Lines modified	Lines deleted	Churn
1	352	307	1369	2028
2	314	31	15	360
3	314	31	20	365
4	62	327	153	542
5	21	3	0	24
6	0	0	0	0

4.4.3 Lead time

The program does not need to compute the lead time for each task. The lead time for each task is recorded by TFS. The lead time is represented in the data set by a column called *lead time* as shown in Table 4.5. The program will gather all the tasks that are started on the same date and belong to the same team and add up their lead time together as showed in code Listing 4.9.

Table 4.5: How lead time is recorded in the TFS document

ID	Type	Lead time
84096	Feature	1
84118	Bug	25
84096	Feature	7
84118	Bug	13

```

addLeadTime(Leadtime t, Arraylist list)
  for lead_time in list
    if lead_time.date = t.date
      lead_time.value = t.value
      return
  structure.add(t)
```

Listing 4.9: Pseudocode example of lead time is measured

4.4.4 Lead time and churn

As stated in the paper by Sjøberg, Johnsen and Solberg (Sjøberg, Johnsen and Solberg, 2012) to prevent outliers from having a large effect on the results, the top and lowest ten percent of lead time and churn are removed from the data set. Churn is removed because a module or a feature, which consists of hundreds or thousands of lines of code could be removed without much work. Lead time is removed because some tasks could be given low priority due to lack of manpower in a given period. Or, tasks could be labeled as not critical and the lead time of these tasks will effect the result.

4.4.5 Moderating variables

To measure the moderating variables for *churn* and *throughput*, the developed software application and SPSS were used. The application will generate *throughput* and *churn* as described in Sections 4.4.1 and 4.4.2, the output from the software application will look like Table 4.6. The output from the program will be used by SPSS. SPSS will use a function called case summaries. The case summaries function groups variables based on a common value. With the case summaries function the variables *team name*, *quarter* and *type* will group the variables *churn* and *throughput*. The result from case summaries provides the moderating variables for *churn* and *throughput* for each quarter.

Table 4.6: A excerpt from the result data produced by the program

Team name	**Churn**	**Throughput**	**Date**	**Quarter**	**Type**
Team one	25	10	2011-12-20	2011-4	Feature
Team two	3	5	2012-04-19	2012-2	bug
Team one	7	2	2010-08-06	2010-3	Feature

4.4.6 Bugs finished, quarter

To get the statistics on number of bugs finished the same quarter as it was recorded, the software application and SPSS were used. The program extracts the *created date* and the *date to* values from each task and checks if their quarter and year match. If they do, a boolean value represented in the task is set to true, otherwise it is set to false. The output produced by the software application is used by SPSS, where the boolean value will be grouped by *team name*, *type* and *quarter*.

4.4.7 Average days backlog, bug

To get the statistics on the average number of days bugs are in backlog per quarter, the software application measures the number of days between the *created date* and the *date from* value. The result is saved together with the task. SPSS is used on the output of the program to measure the average days for bugs per quarter.

The moderating variables, bugs finished, quarter and mean days backlog, bug is used as help variables. A correlation table for these variables will not be provided in Chapter 5.

4.5 Summary

This chapter presented the algorithm and an example of how WIP is computed in this work as well as how the other variables are computed.

Chapter 5

Results

The first result was conducted without considering team size. This result showed a mean correlation of 0.6 between *team size* and *WIP*, a mean correlation of 0.5 between *team size* and *throughput*, a mean correlation of 0.4 between *bugs* and *team size* and a mean correlation of 0.3 between *lead time* and *team size*. Based on these mean values, it was hard to find any evidence that WIP-limit matters in software development. This resulted in a new analyze where each variable for each quarter was divided by the corresponding team size.

The correlation and descriptive statistic tables for *churn*, *lead time*, *throughput* and *bugs* are represented in this chapter, so the mean correlation between these variables can be investigated with respect of WIP-limits in Chapter 6.

Each section except Sections 5.6, Section 5.7 and Section 5.8 are presented with two correlation tables and two corresponding descriptive statistics tables. One correlation table and descriptive statistic table for the first analysis and the same two tables for the second analysis. Section 5.6 is presented with one correlation table and one descriptive statistic table. Sections Section 5.7 and Section 5.8 are presented with one WIP and throughput figure for each team. The content of the sections will consist of highlighting the variables with a significant correlation, describe the descriptive statistic tables and describe the team figures.

5.1 Correlation result for WIP

Table 5.1 displays the correlation tables for *WIP* when team size is **not** considered. The variables are listed vertically in the correlation table. Horizontally are the corresponding teams. The team names are shortened, team one is shortened to T1, team two is shortened to T2 and so on. In the table one can see **team one** has a positive correlation between *WIP* and *throughput*, *throughput feature*, *bugs*, *lead time*, *churn feature* and *team size*. **Team two** has a significant negative correlation between *WIP* and *churn* and both *churn bug*. **Team three** has a positive correlation between *WIP* and all *throughput* variables, *bugs* and *team size*. **Team four** has a significant positive correlation between *WIP* and all variables except *team size*. **Team five** has a positive correlation between *WIP* and *throughput bug*, *lead time* and *team size*.

Table 5.1: Correlation for WIP. Team size is **not** considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
Throughput	0.74**	0.21	0.76**	0.83**	0.52	0.64*	0.67*	0.47	0.89**	0.61*
Throughput Feature	0.73**	-0.14	0.83**	0.82**	0.25	0.68**	0.63*	0.56*	0.82**	0.20
Throughput bug	0.02	0.25	0.73**	0.56*	0.54*	0.07	0.55	0.15	0.88**	0.63*
Bugs	0.72**	0.20	0.60*	0.74**	0.50	0.46	0.62	0.04	0.58*	0.18
Bugs finished, quarter	0.35	0.10	-0.07	0.56*	0.19	0.19	0.85**	0.23	0.52	0.35
Avg days in backlog, bugs	-0.03	0.44	0.42	0.54*	-0.18	0.02	0.10	0.14	-0.20	-0.18
Lead time	0.75**	0.46	0.49	0.70**	0.57*	0.29	0.68*	0.16	0.23	0.72**
Churn	0.47	-0.71**	-0.32	0.66*	0.03	-0.30	0.15	0.16	-0.09	0.16
Churn feature	0.72**	-0.25	-0.34	0.72**	0.06	-0.36	0.10	0.20	-0.12	0.32
Churn bug	0.15	-0.60*	-0.52	0.62*	0.11	0.77**	-0.05	-0.22	-0.30	-0.10
Team size	0.68**	0.35	0.78**	0.06	0.57*	0.77**	0.62	0.65*	0.54	0.76**

Team six has a positive correlation between *WIP* and *throughput, throughput feature, churn bug* and *team size*. **Team seven** has a positive correlation between *WIP* and *throughput, throughput feature, bugs finished, quarter* and *lead time*. **Team eight** has a positive correlation between *WIP* and *throughput feature* and *team size*. **Team nine** has positive correlation between *WIP* and the three *throughput* variables and *bugs*. **Team ten** has a positive correlation between *WIP* and *throughput, throughput bug, lead time* and *team size*.

Throughput feature, *throughput bug*, *churn feature* and *churn bug* are subset of respectively *throughput* and *churn*. It is natural that these moderating variables have a significant positive correlation to *WIP*, when either *throughput* or *churn* has. That's not the case for all teams. There is a gap in the relationship between the *churn* variables and *throughput* variables for both **team one** and **six**. The *throughput* variables for **teams five, seven, eight** and **ten** also have a gap, the same goes for the *churn* variables for **team two**, showed by Table 5.1. The relationship between these variables is explained in Section

6.7.

The descriptive statistics tables in each section are based on correlation. The tables show number of values measured (N), mean, median, standard deviation (Std.dev), maximum (Max) and minimum (Min) values from the correlation tables. In Table 5.2, the mean correlation between *WIP* and both *throughput* and *team size* are 0.6. The mean correlation between *WIP* and *lead time, throughput feature, throughput bug and bugs* are 0.5 and *bugs finished, quarter* have a mean correlation of 0.3 between *WIP*. Rest of the values has a mean value of ± 0.2 or less.

Table 5.2: Descriptive Statistic for WIP correlation. Team size is **not** considered

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
Throughput	10	0.6	0.7	0.2	0.9	0.2
Throughput ft	10	0.5	0.7	0.3	0.8	-0.1
Throughput bug	10	0.5	0.5	0.3	0.9	0
Bugs	10	0.5	0.5	0.2	0.7	0
Bugs finished, quarter	10	0.3	0.3	0.3	0.9	-0.1
Avg days backlog, bugs	10	0.1	0.1	0.3	0.5	-0.2
Lead time	10	0.5	0.5	0.2	0.7	0.2
Churn	10	0	0.1	0.4	0.7	-0.7
Churn ft	10	0.1	0.1	0.4	0.7	-0.4
Churn bug	10	0	-0	0.4	0.8	-0.6
Team size	10	0.6	0.6	0.2	0.8	0.1

The Table 5.3 shows the correlation from when team size was considered. **Team two** has a significant positive correlation between *WIP* and *throughput*, *throughput bug*, *bugs finished, quarter* and *lead time*. **Team three** has a significant correlation between *WIP* and *throughput*, *throughput feature* and *bugs finished, quarter*. **Team four** has a positive correlation between *WIP* and all variables except *throughput bug, bugs, avg days in backlog, bugs* and *churn bug*.

Team seven has a significant positive correlation between *WIP* and *bugs*, *bugs, finished quarter* and *lead time*. **Team eight** has a significant correlation between *WIP* and *bugs finished, quarter* and *churn bug*. **Team nine** has a significant positive correlation between *WIP* and all *throughput* variables. **Team ten** has a significant negative correlation for *avg days in backlog, bugs*.

In Table 5.3, **teams two** and **three** have a significant correlation for two of the three *throughput* variables. **Team four** has a positive correlation for two of the three *churn* and *throughput* variables and **team eight** has a significant correlation to one of the three *churn* variables. The relationship between these moderating variables is explained in Section 6.7. The descriptive statistic Table 5.4 shows *throughput* has a mean correlation

Table 5.3: Correlation for WIP. Team size is considered

Variables	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
Throughput	0.37	0.59*	0.57*	0.86**	0.11	0.08	0.49	0.28	0.66*	-0.21
Throughput Feature	0.31	0.47	0.71**	0.85**	0	0.14	0.46	-0.26	0.60*	-0.16
Throughput bug	0.09	0.65*	0.52	0.27	0.11	0.07	0.57	0.37	0.58*	-0.22
Bugs	0.10	0.49	0.25	0.25	0.11	0.25	0.75*	0.32	-0.05	-0.28
Bugs finished, quarter	-0.28	0.71**	-0.62*	0.74**	-0.24	-0.04	0.85**	0.82**	0.32	-0.28
Avg days in backlog, bugs	0.03	-0.31	0.51	0.10	-0.14	0	0.16	-0.17	-0.40	-0.61*
Lead time	-0.09	0.67**	-0.03	0.87**	0.03	0.32	0.77*	-0.09	-0.18	-0.05
Churn	-0.27	0.16	-0.29	0.77**	-0.09	-0.35	-0.17	0.39	-0.34	-0.37
Churn feature	0.37	0.03	-0.38	0.78**	-0.15	-0.39	-0.17	-0.04	-0.17	0.08
Churn bug	-0.29	0.21	-0.39	0.26	-0.07	0.12	-0.17	0.66*	-0.49	-0.43

* Correlation is significant at the 0.05 level (2-tailed).

** Correlation is significant at the 0.01 level (2-tailed).

of 0.4 between *WIP*. The rest of the variables except *throughputs* moderating variables have a mean correlation of ± 0.2 or less.

Table 5.4: Descriptive Statistic for WIP correlation. Team size is considered

Variables	N	Mean	Median	Std.Dev	Max	Min
Throughput	10	0.4	0.4	0.3	0.9	-0.2
Throughput ft	10	0.3	0.4	0.4	0.9	-0.3
Throughput bug	10	0.3	0.4	0.3	0.7	-0.2
Bugs	10	0.2	0.2	0.3	0.8	-0.3
Bugs finished, quarter	10	0.2	0.1	0.6	0.8	-0.6
Avg days backlog, bugs	10	-0.1	-0.1	0.3	0.5	-0.6
Lead time	10	0.2	0	0.4	0.9	-0.2
Churn	10	-0.1	-0.2	0.4	0.8	-0.4
Churn ft	10	0	-0.1	0.4	0.8	-0.4
Churn bug	10	-0.1	-0.1	0.4	0.7	-0.5

5.2 Correlation result for lead time

Table 5.5 displays the first correlation table for *lead time*, which shows the correlation between *lead time* and the variables when team size is **not** considered. **Team one** has a positive correlation between *lead time* and all variables except *throughput bug*, *avg days in backlog, bugs* and *churn bug*. **Team two** has a positive correlation between *lead time* and both *throughput* and *throughput bug*. **Team three** has a significant correlation between *lead time* and both *bugs* and *churn bug*. **Team four** has a significant positive correlation between *lead time* and *WIP*, *throughput*, *throughput feature*, *churn* and *churn feature*. **Team five** has a significant correlation between *lead time* and *WIP*.

Table 5.5: Correlation for lead time. Team size is **not** considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	0.75**	0.46	0.49	0.70**	0.57*	0.29	0.68*	0.16	0.23	0.72**
Throughput	0.70**	0.67**	0.49	0.68**	0.36	0.13	0.47	0.54	0.42	0.32
Throughput Feature	0.73**	0.09	0.44	0.64*	0.14	0.10	0.41	0.62*	0.41	-0.05
Throughput bug	-0.30	0.60*	0.52	0.31	0.42	-0.01	0.61	-0.17	0.28	0.37
Bugs	0.77**	0.50	0.54*	0.22	0.32	-0.23	0.69*	-0.13	0.44	0.04
Bugs finished, quarter	0.70**	-0.14	0.20	0.23	-0.09	-0.27	0.73*	0.37	0.53	0.19
Avg days in backlog, bugs	0.06	0.40	0.07	0.07	-0.08	-0.03	0.57	-0.12	-0.48	-0.52
Churn	0.70**	-0.42	-0.45	0.97**	0.18	-0.34	0.37	0.91**	-0.37	-0.04
Churn feature	0.86**	0.20	-0.27	0.96**	0.11	-0.31	0.39	0.79**	-0.46	0.32
Churn bug	0.26	-0.39	-0.64*	0.20	0.24	0.28	0.16	-0.12	-0.08	-0.27
Team size	0.61*	0.38	0.44	-0.30	0.36	-0.11	0.59	0.22	0.38	0.53

Team seven has a significant positive correlation between *lead time* and *WIP*, *bugs* and *bugs finished, quarter*. **Team eight** has a significant correlation between *lead time* and *throughput feature*, *churn* and *churn feature*. **Team ten** has a significant correlation between *lead time* and *WIP*.

Table 5.5 displays the variances between the *throughput* variables for **teams one, two, four** and **eight**. There is also variance for the *churn* variables for **teams one, three, four** and **eight**. The relationship between these variables is explained in Section 6.7. Table 5.6 show the mean correlation between *lead time* and both *WIP* and *throughput* are 0.5. *Throughput feature* has the mean correlation of 0.4 between *lead time* and *throughput bug*, *bugs*, *churn feature* and *team size* have the mean correlation of 0.3 between *lead time*. The rest of the values have a correlation of ± 0.2 or less between *lead time*.

Table 5.6: Descriptive Statistic for lead time correlation. Team size is **not** considered

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
WIP	10	0.5	0.5	0.2	0.7	0.2
Throughput	10	0.5	0.5	0.2	0.7	0.1
Throughput ft	10	0.4	0.4	0.3	0.7	-0.1
Throughput bug	10	0.3	0.4	0.3	0.6	-0.3
Bugs	10	0.3	0.4	0.3	0.8	-0.2
Bugs finished, quarter	10	0.2	0.2	0.3	0.7	-0.3
Avg days backlog, bugs	10	0	0	0.3	0.6	-0.5
Churn	10	0.2	0.1	0.6	1	-0.5
Churn ft	10	0.3	0.3	0.5	1	-0.5
Churn bug	10	-0.1	-0.1	0.3	0.3	-0.6
Team size	10	0.3	0.4	0.3	0.6	-0.3

Table 5.7 shows the correlation between *lead time* and the variables when team size is considered. The table shows **team one** has a significant positive correlation between *lead time* and *bugs*, *bugs finished, quarter*, *churn* and *churn bug*. **Team two** has a significant positive correlation between *lead time* and all values except *avg days in backlog, bugs, churn* and *churn feature*. **Team four** has a significant positive correlation between *lead time* and *WIP*, *throughput*, *throughput feature*, *churn* and *churn feature*. **Team five** has a significant correlation between *lead time* and all variables except *WIP*, *throughput feature* and *churn feature*.

Table 5.7: Correlation for lead time. Team size considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	-0.09	0.67**	-0.03	0.87**	0.03	0.32	0.77*	-0.09	-0.18	-0.05
Throughput	0.06	0.88**	0.26	0.90**	0.80**	0.69**	0.33	0.24	0.32	0.90**
Throughput Feature	0.01	0.75**	0.16	0.89**	-0.05	0.61*	0.34	0.87**	0.20	-0.26
Throughput bug	0.51	0.84**	0.29	0.22	0.84**	0.41	0.56	-0.27	0.25	0.91**
Bugs	0.83**	0.72**	0.48	0.44	0.72**	0.18	0.85**	-0.23	0.54	0.88**
Bugs finished, quarter	0.88**	0.79**	0.32	0.52	0.80**	0.36	0.77**	-0.04	0.45	0.57*
Avg days in backlog, bugs	0.41	-0.52	-0.04	-0.07	0.85**	-0.09	0.51	-0.20	-0.17	0.10
Churn	0.72**	0.49	0.16	0.96**	0.94**	0.04	0.08	0.71**	-0.18	0.01
Churn feature	-0.52	0.38	0.22	0.96**	-0.21	0.11	0.05	0.86**	-0.28	-0.24
Churn bug	0.73**	0.55*	0.07	0.13	0.95**	-0.17	-0.06	-0.12	0.24	0.03

Team six has a significant correlation between *lead time* and both *throughput* and *throughput feature*. **Team seven** has a significant correlation between *lead time* and *WIP*, *bugs* and *bugs finished, quarter*. **Team eight** has a significant correlation between *lead time* and *throughput feature*, *churn* and *churn feature*. **Team ten** has a significant positive correlation between *lead time* and *throughput*, *throughput bug*, *bugs* and *bugs finished, quarter*.

The *throughput* relationship for **team four, five, six, eight** and **ten** show variances according to Table 5.7. *Churn* variables also show variances for **team one, two, four, five** and **eight**. The relationship between these variables is explained in Section 6.7. The Table 5.8 displays a mean correlation of 0.5 between *lead time* and *throughput*, *throughput bug*, *bugs* and *bugs finished, quarter*. *Churn* and *throughput feature* have the mean correlation of 0.4 between *lead time*. The rest of the values have a mean correlation of ± 0.2 or less between *lead time*.

Table 5.8: Descriptive Statistic for lead time correlation. Team size is into account

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
WIP	10	0.2	0	0.4	0.9	-0.2
Throughput	10	0.5	0.5	0.3	0.9	0.1
Throughput ft	10	0.4	0.3	0.4	0.9	-0.3
Throughput bug	10	0.5	0.5	0.4	0.9	-0.3
Bugs	10	0.5	0.6	0.4	0.9	-0.2
Bugs finished, quarter	10	0.5	0.5	0.3	0.9	-0
Avg days backlog, bugs	10	0.1	-0.1	0.4	0.8	-0.5
Churn	10	0.4	0.3	0.4	1	-0.2
Churn ft	10	0.1	0.1	0.5	1	-0.5
Churn bug	10	0.2	0.1	0.4	1	-0.2

5.3 Correlation result for bugs

This section contains information about the correlation tables between the variables and *bugs*. In the first correlation table, Table 5.9 the correlation between *lead time* and the variables, when team size is **not** considered is presented. In the table one can see **team one** has a significant correlation between *bugs* and all the variables except *throughput bug*, *bugs finished, quarter*, *avg days in backlog, bugs*, and *churn bug*. **Team two** has a significant correlation between *bugs* and both *throughput* and *throughput bug*. **Team three** has a significant correlation between *bugs* and *WIP*, the *throughput* variables and *lead time*. **Team four** has a significant correlation between *bugs* and *WIP*, *throughput bug*, *bugs finished, quarter*, *avg days in backlog, bugs* and *churn bug*. **Team five** has a significant correlation between *bugs* and the *throughput* variables and *team size*.

Table 5.9: Correlation for Bugs. Team size is **not** considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	0.72**	0.20	0.60*	0.74*	0.50	0.46	0.62	0.04	0.58*	0.18
Throughput	0.69**	0.81**	0.88**	0.51	0.97**	0.27	0.53	0.41	0.70**	0.56*
Throughput Feature	0.74**	0.01	0.82**	0.53	0.88**	0.30	0.56	0.22	0.60*	-0.14
Throughput bug	-0.17	0.83**	0.87**	0.54*	0.96**	0.69**	0.50	0.92**	0.65*	0.59*
Bugs finished, quarter	0.50	-0.18	0.12	0.87**	0.17	0.76**	0.79**	0.18	0.70**	0.05
Avg days in backlog, bugs	0.52	0.38	0.43	0.53*	0.18	0.24	0.23	0.28	0.21	0.13
Lead time	0.77**	0.50	0.54*	0.22	0.32	-0.23	0.69*	-0.13	0.44	0.04
Churn	0.62*	-0.27	0.10	0.12	-0.06	-0.12	0.11	-0.16	-0.48	0.04
Churn feature	0.77**	0.01	0.09	0.22	0.43	-0.10	0.11	-0.25	-0.62*	0.07
Churn bug	0.42	-0.19	-0.19	0.71*	-0.11	0.42	0.12	0.65*	-0.04	0
Team size	0.80**	0.26	0.27	0.06	0.71**	0.41	0.41	0.42	0.41	0.16

Team six has a significant correlation between *bugs* and both *throughput bug* and *bugs finished, quarter*. **Team seven** has a significant correlation between *bugs* and both *bugs finished, quarter* and *lead time*. **Team eight** has a significant correlation relationship between *bugs* and both *throughput bug* and *churn bug*. **Team nine** has a significant correlation relationship between *bugs* and *WIP*, the *throughput* variables, *bugs finished, quarter* and *churn feature*. **Team ten** has a significant correlation between *bugs* and both *throughput* and *throughput bug*.

Teams one, four and **eight** have variances in both the *throughput* and *churn* variables according to Table 5.9. **Teams two, six** and **ten** also have variances for the *throughput* variables, while **team nine** has a variances between the *churn* variables. The relationship between these variables is explained in Section 6.7. In Table 5.10, one can see that *throughput* and *throughput bug* have the mean correlation of 0.6 between *bugs*, *throughput feature* has the correlation of 0.5 between *bugs*, *WIP* has a mean correlation of 0.4 between *bugs*. *Bugs finished, quarter*, *avg days backlog, bugs*, *lead time* and *team size* have the mean correlation of 0.3 between *bugs*. The *churn* variables have the mean values of ± 0.2 or less between *bugs*.

Table 5.10: Descriptive Statistic for bugs correlation. Team size is **not** considered

Variables	N	Mean	Median	Std.Dev	Max	Min
WIP	10	0.4	0.5	0.2	0.7	0
Throughput	10	0.6	0.6	0.2	1	0.3
Throughput ft	10	0.5	0.6	0.3	0.9	-0.1
Throughput bug	10	0.6	0.7	0.3	1	-0.2
Bugs finished, quarter	10	0.3	0.3	0.3	0.9	-0.2
Avg days backlog, bugs	10	0.3	0.3	0.1	0.5	0.1
Lead time	10	0.3	0.4	0.3	0.8	-0.2
Churn	10	0	-0	0.3	0.6	-0.5
Churn ft	10	0.1	0.1	0.4	0.8	-0.6
Churn bug	10	0	0	0	0.7	-0
Team size	10	0.3	0.4	0.3	0.6	-0.3

The second correlation table for bugs, displayed in Table 5.11 shows correlation when team size is considered. In the table one can see **team one** has a significant correlation between *bugs* and *throughput bug*, *bugs finished, quarter*, *avg days in backlog, bugs* and *lead time*. **Team two** has a significant correlation between *bugs* and all the *throughput* variable and *lead time*. **Team three** has a significant correlation between *bugs* and all the *throughput* variables and all the *churn variables*. **Team five** has a significant correlation between *bugs* and all variables except *WIP*, *throughput feature*, *bugs finished, quarter* and *churn feature*.

Table 5.11: Correlation with Bugs - Team size is considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
WIP	0.10	0.49	0.25	0.27	0.11	0.25	0.75*	0.32	-0.05	-0.28
Throughput	0.05	0.90**	0.81**	0.32	0.97**	-0.02	0.46	0.82**	0.57*	0.96**
Throughput Feature	-0.12	0.62*	0.66**	0.33	0.50	-0.10	0.48	-0.52	0.36	-0.25
Throughput bug	0.79**	0.92**	0.79**	0.29	0.95**	0.80**	0.60	0.98**	0.64*	0.96**
Bugs finished, quarter	0.59*	0.50	0.44	0.17	0.52	0.77**	0.86**	0.70**	0.19	0.59*
Avg days in backlog, bugs	0.76**	-0.07	0.38	0.18	0.75**	0.30	0.37	0.38	0.37	0.39
Lead time	0.83**	0.72**	0.48	0.22	0.72**	0.18	0.85**	-0.23	0.54	0.88**
Churn	0.47	0.29	0.67**	0.22	0.73**	0.20	-0.05	0.34	-0.02	0.19
Churn feature	-0.40	0.12	0.62*	0.27	-0.10	0.27	-0.07	-0.26	-0.24	-0.19
Churn bug	0.48	0.35	0.55*	0.44	0.73**	-0.19	-0.08	0.78**	0.43	0.23

Team six has a significant correlation between *bugs* and both *throughput bug* and *bugs* *finished, quarter*. **Team seven** has a significant relationship between *bugs* and *WIP*, *bugs finished, quarter* and *lead time*. **Team eight** has a significant correlation between *bugs* and *throughput*, *throughput bug*, *bugs finished, quarter* and *churn bug*. **Team nine** has a significant correlation between *bugs* and both *throughput* and *throughput bug*. **Team ten** has a significant correlation between *bugs* and *throughput, throughput bug*, *bugs finished, quarter* and *lead time*.

Throughput and *churn* variables for **team five** and **eight** shows variance according to Table 5.11. **Team one, six** and **nine** shows variance between the *throughput variables*. Their relationship is explained in Section 6.7. Table 5.12 displays a mean correlation of 0.8 for *throughput bug* between *bugs*, 0.6 for *throughput* between *bugs*, 0.5 for *bugs finished, quarter* and *lead time* between *bugs* and 0.4 for *avg days backlog, days* and *churn bug* between *bugs*. *Churn* has the correlation of 0.3 between *bugs*, while the rest has a mean correlation of ± 0.2 or less between *bugs*.

Table 5.12: Descriptive Statistic for bugs correlation. Team size is considered

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
WIP	10	0.2	0.2	0.3	0.8	-0.3
Throughput	10	0.6	0.7	0.4	1	0
Throughput ft	10	0.2	0.3	0.4	0.7	-0.5
Throughput bug	10	0.8	0.8	0.2	1	0.3
Bugs finished, quarter	10	0.5	0.6	0.2	0.9	0.2
Avg days backlog, bugs	10	0.4	0.4	0.2	0.8	-0.1
Lead time	10	0.5	0.6	0.4	0.9	-0.2
Churn	10	0.3	0.3	0.3	0.7	-0.1
Churn ft	10	0	-0.1	0.3	0.6	-0.4
Churn bug	10	0.4	0.4	0.3	0.8	-0.2

5.4 Correlation result for throughput

This section shows the correlation table between *throughput* and the variables. The first correlation Table 5.13 shows the correlation when team size is **not** considered. In the table one can see that *throughput* has a significant correlation to either *throughput feature* or *throughput bug* for each of the teams. The **teams three, four, five, seven** and **nine** have a positive correlation to both the *throughput* moderating variables. For **teams one, six** and **eight** are the variance between *throughput feature* and *throughput*. For **teams two** and **ten** are the variance between *throughput bug* and **throughput**, according to Table 5.13. The relationship between the *throughput* variables that have a variance is explained in Section 6.7.

The moderating variables of *throughput* are highlighted above, so the moderating variables will be left out of the highlighting further in this section. **Team one** has a significant correlation between *throughput* and *WIP, bugs, lead time, churn feature* and *team size*. **Team two** has a significant correlation between *throughput* and *bugs* and *lead time*. **Team three** has a significant correlation between *throughput* and both *WIP* and *bugs*. **Team four** has a significant correlation between *throughput* and *WIP, bugs, lead time, churn* and *churn feature*. **Team five** has a significant correlation between *throughput* and both *bugs* and *team size*.

Table 5.13: Correlation for throughput. Team size is **not** considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	0.74**	0.21	0.76**	0.83**	0.52	0.64*	0.67*	0.47	0.89**	0.61*
Throughput Feature	0.96**	0.09	0.93**	1**	0.85**	0.99**	0.91**	0.94**	0.88**	0.43
Throughput bug	0.03	0.97**	0.99**	0.59*	0.99**	0.04	0.91**	0.44	0.96**	0.98**
Bugs	0.69**	0.81**	0.88**	0.51	0.97**	0.27	0.53	0.41	0.70**	0.56*
Bugs finished, quarter	0.16	0.12	0.23	0.39	0.12	0.12	0.58	0.33	0.70**	0.59*
Avg days in backlog, bugs	0.16	0.12	0.45	0.37	0.14	-0.17	0.21	-0.17	-0.41	-0.09
Lead time	0.70**	0.67**	0.49	0.68**	0.36	0.13	0.47	0.54	0.42	0.32
Churn	0.37	-0.43	-0.18	0.72**	-0.06	-0.40	0.60	0.59*	-0.14	0.02
Churn feature	0.78**	-0.10	-0.20	0.81**	0.41	-0.40	0.43	0.43	-0.29	-0.20
Churn bug	-0.06	-0.21	-0.33	0.49	-0.10	0.57*	-0.10	0.03	-0.29	-0.06
Team size	0.70**	0.05	0.52	0.16	0.69**	0.86**	0.62	0.75**	0.53	0.57*

Team six has a significant correlation between *throughput* and *WIP, churn bug* and *team size*. **Team seven** has a significant correlation between *throughput* and *WIP*. **Team eight** has a significant correlation between *throughput* and both *churn* and *team size*. **Team nine** has a significant correlation between *throughput* and *WIP, bugs* and *bugs finished, quarter*. **Team ten** has a significant correlation between *throughput* and *WIP, bugs, bugs finished, quarter* and *team size*.

For **teams one, four, six** and **eight** are there a variance between the *churn* variables, the relationship between these variables is explained in Section 6.7. The Table 5.14 displays *throughput feature* has a mean correlation of 0.8 between *throughput*, *throughput bug* has a mean correlation of 0.7 between *throughput*, *WIP* and *bugs* have a mean correlation of 0.6 between *throughput*, *lead time* and *team size* have the correlation of 0.5 between *throughput* and *bugs finished, quarter* has a mean of 0.3 between *throughput*. The rest of the values have a mean correlation of ± 0.2 or less between *throughput*.

Table 5.14: Descriptive Statistic for throughput correlation. Team size is **not** considered

Variables	N	Mean	Median	Std.Dev	Max	Min
WIP	10	0.6	0.7	0.2	0.9	0.2
Throughput ft	10	0.8	0.9	0.3	1	0.1
Throughput bug	10	0.7	1	0.4	1	0
Bugs	10	0.6	0.6	0.2	1	0.3
Bugs finished, quarter	10	0.3	0.3	0.2	0.7	0.1
Avg days backlog, bugs	10	0.1	0.1	0.3	0.5	-0.4
Lead time	10	0.5	0.5	0.2	0.7	0.1
Churn	10	0.2	-0	0.4	0.7	-0.4
Churn ft	10	0.2	0.2	0.5	0.8	-0.4
Churn bug	10	0	-0.1	0.3	0.6	-0.4
Team size	10	0.5	0.6	0.3	0.9	0.1

The second correlation table, Table 5.15 shows the correlation between *throughput* and the variables when team size is considered. In the table one can see *throughput* has a significant correlation to either *throughput feature* or *throughput bug* for each of the teams. The **teams two, three** and **nine** have a positive correlation to both the *throughput* moderating variables. For **teams five, eight** and **ten** there are a variance between *throughput feature* and *throughput*. For **teams one, four, six** and **seven** are there a variance between *throughput bug* and *throughput*, according to Table 5.15. The relationship between the *throughput* variables that have a variance is explained in Section 6.7.

Table 5.15: Correlation for throughput. Time size is considered.

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
WIP	0.37	0.59*	0.57*	0.86**	0.11	0.08	0.49	0.28	0.66*	-0.21
Throughput Feature	0.71**	0.72**	0.90**	1**	0.40	0.98**	0.90**	-0.02	0.82**	-0.11
Throughput bug	0.16	0.98**	0.98**	0.32	0.99**	0.31	0.56	0.82**	0.95**	1**
Bugs	0.05	0.90**	0.81**	0.50	0.97**	-0.02	0.46	0.82**	0.57*	0.96**
Bugs finished, quarter	-0.11	0.69**	0.04	0.56*	0.57*	0.07	0.31	0.71**	0.55	0.67**
Avg days in backlog, bugs	0.02	-0.33	0.44	-0.12	0.80**	-0.35	-0.06	0.09	-0.27	0.27
Lead time	0.06	0.88**	0.26	0.90**	0.80**	0.69**	0.33	0.24	0.32	0.90**
Churn	-0.14	0.37	0.23	0.88**	0.79**	-0.28	0.46	0.75**	-0.14	0.06
Churn feature	0.05	0.20	0.18	0.90**	-0.11	-0.16	0.12	0.18	-0.29	-0.35
Churn bug	-0.17	0.47	0.14	0.18	0.80**	-0.17	0.01	0.70**	-0.11	0.11

Team two has a significant correlation between *throughput* and all variables except *avg days in backlog, bugs* and the *churn* variables. **Team three** has a significant correlation between *throughput* and both *WIP* and *bugs*. **Team four** has a significant correlation between *throughput* and *WIP, bugs finished, quarter, lead time, churn* and *churn feature*. **Team five** has a significant correlation between *throughput* and all variables except *WIP* and *churn feature*.

Team six has a positive correlation between *throughput* and *lead time*. **Team eight** has a correlation with *bugs, bugs finished, quarter, churn* and *churn bug*. **Team nine** has a significant correlation between *throughput* and both *WIP* and *bugs*. **Team ten** has a significant correlation between *throughput* and *bugs*, *bugs finished, quarter* and *lead time*. The *churn* variables for **teams four, five** and **eight** have a variance, according to Table 5.15. The *churn* relationship is explained in Section 6.7.

Table 5.16 displays a mean correlation of 0.8 between *throughput* and *throughput bug*, a correlation of 0.6 for *throughput feature* and *bugs* between *throughput*, a correlation of 0.5 for *lead time* between *throughput*, a correlation of 0.4 for *WIP* and *bugs finished, quarter* between *throughput* and a correlation of 0.3 for *churn* between *throughput*. The rest of the values have a mean correlation of ± 0.2 or less between *throughput*.

Table 5.16: Descriptive Statistic for throughut correlation. Team size is considered

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
WIP	10	0.4	0.4	0.3	0.9	-0.2
Throughput ft	10	0.6	0.8	0.4	1	-0.1
Throughput bug	10	0.8	1	0.3	1	0.2
Bugs	10	0.6	0.7	0.4	1	0
Bugs finished, quarter	10	0.4	0.6	0.3	0.7	-0.1
Avg days backlog, bugs	10	0	0	0.4	0.8	-0.3
Lead time	10	0.5	0.5	0.3	0.9	0.1
Churn	10	0.3	0.3	0.4	0.9	-0.3
Churn ft	10	0.1	0.1	0.4	0.9	-0.3
Churn bug	10	0.2	0.1	0.3	0.8	-0.2

5.5 Correlation result for churn

This section contains information about the correlation table between the variables and churn. Table 5.17 shows the correlation when team size was **not** considered. The table shows all teams have either one or both moderating variables with significant positive correlation with *churn*. **Teams four, five, six, seven, eight, nine** and **ten** do not have a positive correlation between both the *churn* moderating variables according to Table

5.17. The relationship between the *churn* variables is explained in Section 6.7.

In this section will *churn* variables be left out of the highlighting for the same reason as in Section 5.4. Table 5.17 shows **Team one** has positive correlation between *churn* and *bugs*, *bugs finished*, *quarter* and *lead time*. **Team two** has a significant correlation between *churn* and *WIP*. **Team four** has a significant correlation between *churn* and all variables except *bugs*, *bugs finished*, *quarter*, *avg days in backlog*, *bugs*, *team size* and *throughput bugs*, but not *throughput*. The *throughput* relationship is explained in Section 6.7. Team five has a negative correlation between *churn* and *team size*.

Table 5.17: Correlation for churn. When team size is **not** considered.

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	0.47	-0.71**	-0.32	0.66*	0.03	-0.30	0.10	0.16	-0.09	0.16
Throughput	0.37	-0.43	-0.18	0.72**	-0.06	-0.40	0.43	0.59*	-0.14	0.02
Throughput Feature	0.36	0	-0.12	0.69**	-0.03	-0.37	0.45	0.63*	0.02	-0.17
Throughput bug	-0.52	-0.42	-0.22	0.27	-0.03	-0.03	0.54	-0.17	-0.20	0.07
Bugs	0.62*	-0.27	0.10	0.12	-0.06	-0.12	0.11	-0.16	-0.48	0.04
Bugs finished, quarter	0.80**	-0.22	-0.11	0.15	-0.31	0.04	0.17	0.49	-0.05	0.31
Avg days in backlog, bugs	0.19	-0.12	-0.06	0	0.15	0.56*	0.60	-0.17	-0.01	-0.11
Lead time	0.70**	-0.42	-0.45	0.97**	0.18	-0.34	0.39	0.91**	-0.37	-0.04
Churn feature	0.57*	0.58*	0.90**	0.99**	0.22	0.98**	0.91**	0.84**	0.62*	0.14
Churn bug	0.80**	0.70**	0.85**	0.13	0.94**	-0.02	0.13	-0.07	0.39	0.94**
Team size	0.42	-0.16	-0.51	-0.24	-0.54*	-0.18	0.36	0.14	0.11	0.12

Team six has a significant correlation between *churn* and *avg days in backlog*, *bugs*. **Team eight** has a significant correlation between *churn* and *throughput*, *throughput feature* and *lead time*, but not *throughput bug*. The *throughput* relationship for **team eight** is explained in Section 6.7. The Table 5.18 shows that there do not exists variables with a mean correlation of ± 0.2 between *churn* without the *churn* moderating variables

Table 5.18: Descriptive Statistic for churn correlation. Team size is **not** considered

Variables	**N**	**Mean**	**Median**	**Std.Dev**	**Max**	**Min**
WIP	10	0	0.1	0.4	0.7	-0.7
Throughput	10	0.1	0	0.4	0.7	-0.4
Throughput ft	10	0.1	0	0.4	0.7	-0.4
Throughput bug	10	-0.1	-0.1	0.4	0.5	-0.5
Bugs	10	0	0	0.3	0.6	-0.5
Bugs finished, quarter	10	0.1	0.1	0.3	0.8	-0.3
Avg days backlog, bugs	10	0.1	-0	0.3	0.6	-0.2
Lead time	10	0.2	0.1	0.6	1	-0.5
Churn ft	10	0.7	0.7	0.3	1	0.1
Churn bug	10	0.5	0.5	0.4	0.9	-0.1
Team size	10	0	0	0.3	0.4	-0.5

The second correlation table for *churn*, Table 5.19 shows the *churn* correlation table when team size is considered. The table shows all teams have either one or both moderating variables with significant positive correlation with *churn*. **Teams one, four, five, six, seven, eight** and **ten** do not have a positive correlation between both the *churn* moderating variables according to Table 5.17, the relationship between these variables is explained in Section 6.7.

Team one has a significant positive correlation between *churn* and *bugs finished, quarter* and *lead time*. **Team two** has a significant correlation with *throughput feature*, *bugs finished, quarter* and *avg days in backlog, bugs*. **Team three** has a significant correlation between *churn* and *bugs* and *bugs finished, quarter*. **Team four** has a significant correlation between *churn* and *WIP*, *throughput*, *throughput feature* and *lead time*. **Team five** has a significant correlation between *churn* and *throughput*, *throughput bug*, *bugs*, *bugs finished, quarter*, *avg days in backlog, bugs* and *lead time*

Table 5.19: Correlation for churn. Team size considered

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10
WIP	-0.27	0.16	-0.29	0.77**	-0.09	-0.35	-0.17	0.39	-0.34	-0.37
Throughput	-0.14	0.37	0.23	0.88**	0.79**	-0.28	0.46	0.75**	-0.14	0.06
Throughput Feature	-0.39	0.68**	0.11	0.87**	-0.02	-0.39	0.56	0.48	0.04	0.08
Throughput bug	-0.03	0.30	0.20	0.22	0.83**	0.28	0.08	0.34	-0.10	0.07
Bugs	0.47	0.29	0.67**	0.27	0.73**	0.20	-0.05	0.34	-0.02	0.19
Bugs finished, quarter	0.95**	0.68**	0.76**	0.34	0.90**	0.33	-0.16	0.61*	-0.12	0.44
Avg days in backlog, bugs	-0.11	-0.55*	0.15	-0.12	0.88**	0.56*	0.39	-0.18	0.21	0.76**
Lead time	0.72**	0.49	0.16	0.96**	0.94**	0.04	0.08	0.71**	-0.18	0.01
Churn feature	-0.40	0.79**	0.95**	0.99**	-0.10	0.98**	0.78**	0.66*	0.56*	0.33
Churn bug	1**	0.93**	0.95**	0.09	1**	-0.19	0.25	0.51	0.56*	0.99**

Team six has a significant positive correlation between *churn* and *avg days in backlog, bugs*. **Team eight** has a significant positive correlation between *churn* and *throughput, bugs finished, quarter* and *lead time*. **Team ten** has a significant positive correlation between *churn* and *avg days in backlog, bugs*. **Teams two, four, five** and **eight** have a variance in their *throughput* relationship, according to Table 5.19. The relationship between these variables is explained in Section 6.7. The Table 5.20 displays a mean correlation of 0.6 for both *churn* moderating variables between *churn*, 0.5 for *bugs finished, quarter* between *churn*, 0.4 for *lead time* between *churn* and 0.3 for *throughput, throughput bug* and *bugs* between *churn*. The rest of the values have a correlation of ± 0.2 or less between *churn*.

Table 5.20: Descriptive Statistic for churn correlation. Team size is considered

Variables	N	Mean	Median	Std.Dev	Max	Min
WIP	10	-0.1	-0.2	0.4	0.8	-0.4
Throughput	10	0.3	0.3	0.4	0.9	-0.3
Throughput ft	10	0.2	0.1	0.4	0.9	-0.4
Throughput bug	10	0.3	0.2	0.3	0.8	-0.1
Bugs	10	0.3	0.3	0.3	0.7	-0.1
Bugs finished, quarter	10	0.5	0.5	0.4	0.9	-0.2
Avg days backlog, bugs	10	0.2	0.2	0.5	0.9	-0.6
Lead time	10	0.4	0.3	0.4	1	-0.2
Churn ft	10	0.6	0.7	0.5	1	-0.4
Churn bug	10	0.6	0.7	0.4	1	-0.2

5.6 Correlation result for team size

The team size correlation Table 5.21 displays that **team one** has a significant correlation between *team size* and *WIP, throughput, throughput feature, bugs, lead time* and *churn feature*. **Team two** has a correlation between *team size* and *avg days in backlog, bugs*. **Team three** has a significant correlation between *team size* and both *WIP* and *churn bug*. **Team five** has a significant correlation between *team size* and *WIP, throughput, throughput bug, bugs, churn* and *churn bug*.

Table 5.21: Correlation for Team size

* Correlation is significant at the 0.05 level (2-tailed).
** Correlation is significant at the 0.01 level (2-tailed).

Variables	**T1**	**T2**	**T3**	**T4**	**T5**	**T6**	**T7**	**T8**	**T9**	**T10**
WIP	0.68**	0.35	0.78**	0.06	0.57*	0.77**	0.62	0.65*	0.54	0.76**
Throughput	0.70**	0.05	0.52	0.16	0.69**	0.86**	0.62	0.75**	0.53	0.57*
Throughput Feature	0.74**	-0.22	0.53	0.20	0.48	0.89**	0.47	0.74**	0.48	0.18
Throughput bug	-0.10	0.06	0.51	0.17	0.67**	0	0.71*	0.40	0.48	0.64*
Bugs	0.80**	0.26	0.27	0.06	0.71**	0.41	0.41	0.42	0.41	0.16
Bugs finished, quarter	0.42	-0.53	0.25	-0.19	0.28	0.30	0.71*	0.05	0.38	0.34
Avg days in backlog, bugs	0.48	0.84**	0.04	0.44	0.03	-0.03	0.49	0.03	0.07	-0.03
Lead time	0.61*	0.38	0.44	-0.30	0.36	-0.11	0.59	0.22	0.38	0.53
Churn	0.42	-0.16	-0.51	-0.24	-0.54*	-0.18	0.33	0.14	0.11	0.12
Churn feature	0.79**	0.41	-0.42	-0.17	0.32	-0.23	0.36	0.07	0.01	0.36
Churn bug	0.26	-0.44	-0.61*	0.27	-0.55*	0.74**	-0.32	-0.14	-0.16	-0.10

Team six has a significant correlation between *team size* and *WIP, throughput, throughput feature* and *churn bug*. **Team seven** has a significant correlation between *team size* and *throughput bug* and *bugs finished, quarter*. **Team eight** has a significant correlation between *team size* and *WIP, throughput* and *throughput feature*. **Team ten** has a significant positive correlation between *team size* and *WIP*, *throughput* and *throughput bug*.

Teams one, five and **six** have a variance for both *throughput* and *churn* variables, according to Table 5.21. **Teams seven, eight** and **ten** have a variance between the *throughput* variables, while **team three** has it between the *churn* variables. The relationship between the moderating variables is explained in Section 6.7. The Table 5.22 display a mean correlation of 0.6 for *WIP* between *team size*, 0.5 for *throughput* between *team size*, 0.4 between *team size* and the *throughput* moderating variables and *bugs*. *Lead time* has the mean correlation of 0.3 between *team size*. Rest of the variables has a correlation of ± 0.2 or less between *team size*.

Table 5.22: Descriptive Statistic for team size correlation.

Variables	N	Mean	Median	Std.Dev	Max	Min
WIP	10	0.6	0.6	0.2	0.8	0.1
Throughput	10	0.5	0.6	0.3	0.9	0.1
Throughput ft	10	0.4	0.5	0.3	0.9	-0.2
Throughput bug	10	0.4	0.4	0.3	0.7	-0.1
Bugs	10	0.4	0.4	0.2	0.8	0.2
Bugs finished, quarter	10	0.2	0.3	0.3	0.7	-0.5
Avg days backlog, bugs	10	0.2	0.1	0.3	0.8	-0
Lead time	10	0.3	0.4	0.3	0.6	-0.3
Churn	10	0	-0	0.3	0.4	-0.5
Churn ft	10	0.1	0.2	0.4	0.8	-0.4
Churn bug	10	-0.1	-0.1	0.4	0.7	-0.6

5.7 WIP-limit per team

To investigate if WIP-limit matter, the WIP-limit for each of the teams were measured. *Team one* has a mean WIP-limit of 0.73. *Team two* has a mean WIP-limit of 2.31. *Team three* has a mean WIP-limit of 1.78. *Team four* has a mean WIP-limit of 3.21. *Team five* has a mean WIP-limit of 1.93. *Team six* has a mean WIP-limit of 3.02. *Team seven* has a mean WIP-limit of 2.03 .*Team eight* has a mean WIP-limit of 0.71. *Team nine* has a mean WIP-limit of 2.19 and *team ten* has a mean WIP-limit of 1.08, as showed in Figure 5.1. This gives a mean WIP-limit of 1.9 per developer.

Year - Quarter	WIP-limit
2010-3	0.61
2010-4	0.23
2011-1	0.21
2011-2	0.47
2011-3	0.89
2011-4	0.38
2012-1	0.63
2012-2	0.89
2012-3	1.12
2012-4	1.2
2013-1	1.37
2013-2	1.81
2013-3	0.32
2013-4	0.22
Mean	0.73

(a) Size of T1

Year - Quarter	WIP-limit
2010-3	1.44
2010-4	1.43
2011-1	2.09
2011-2	2.48
2011-3	2.18
2011-4	2.15
2012-1	1.34
2012-2	3.62
2012-3	3.03
2012-4	2.4
2013-1	1.97
2013-2	4.0
2013-3	2.68
2013-4	1.66
Mean	2.31

(b) Size of T2

Year - Quarter	WIP-limit
2010-3	1.55
2010-4	1.55
2011-1	2.19
2011-2	2.35
2011-3	2.3
2011-4	2.33
2012-1	2.26
2012-2	2.18
2012-3	2.16
2012-4	2.28
2013-1	1.24
2013-2	0.99
2013-3	0.88
2013-4	0.7
Mean	1.78

(c) Size of T3

Year -Quarter	WIP-limit
2010-3	0.89
2010-4	0.56
2011-1	2.61
2011-2	3.45
2011-3	3.51
2011-4	4.11
2012-1	4.0
2012-2	5.84
2012-3	4.67
2012-4	4.12
2013-1	2.78
2013-2	3.2
2013-3	3.09
2013-4	2.11
Mean	3.21

(d) Size of T4

Year-Quarter	WIP-limit
2010-3	1.68
2010-4	1.44
2011-1	0.56
2011-2	0.85
2011-3	1.28
2011-4	1.21
2012-1	1.78
2012-2	2.7
2012-3	0.82
2012-4	1.11
2013-1	4.03
2013-2	7.97
2013-3	0.78
2013-4	0.87
Mean	1.93

(e) Size of T5

Year - Quarter	WIP-limit
2010-3	1.9
2010-4	1.72
2011-1	1.63
2011-2	1.73
2011-3	3.9
2011-4	4.58
2012-1	3.91
2012-2	2.92
2012-3	2.54
2012-4	2.92
2013-1	3.62
2013-2	4.86
2013-3	3.4
2013-4	2.67
Mean	3.02

(f) Size of T6

Year - Quarter	WIP-limit
2010-3	1.41
2010-4	2.88
2011-1	1.85
2011-2	2.13
2011-3	2.15
2011-4	1.94
2012-1	1.44
2012-2	3.31
2012-3	2.65
2012-4	0.56
Mean	2.03

(g) Size of T7

Year - Quarter	WIP-limit
2010-4	0.19
2011-1	0.46
2011-2	0.74
2011-3	0.86
2011-4	0.88
2012-1	0.97
2012-2	2.14
2012-3	0.36
2012-4	0.51
2013-1	0.38
2013-2	0.94
2013-3	0.36
2013-4	0.5
Mean	0.71

(h) Size of T8

Year - Quarter	WIP-limit
2010-4	0.9
2011-1	1.48
2011-2	1.61
2011-3	1.82
2011-4	1.78
2012-1	1.62
2012-2	4.43
2012-3	3.26
2012-4	1.82
2013-1	2.68
2013-2	2.95
2013-3	1.99
2013-4	2.14
Mean	2.19

(i) Size of T9

Year - Quarter	WIP-limit
2010-3	0.9
2010-4	1.19
2011-1	0.68
2011-2	0.66
2011-3	0.22
2011-4	0.98
2012-1	1.23
2012-2	2.03
2012-3	0.7
2012-4	1.84
2013-1	0.96
2013-2	0.98
2013-3	1.25
2013-4	1.54
Mean	1.08

(j) Size of T10

5.8 Throughput per team

To try to explain why the relationship between *throughput* and *WIP* are positive, the mean throughput per team were measured. **Team one** has a mean *throughput* of 0.42. **Team two** has a mean *throughput* of 0.44. **Team three** has a mean *throughput* of 0.35. **Team four** has a mean *throughput*of 1.4. **Team five** has a mean *throughput* of 0.30. **Team six** has a mean *throughput* of 0.63. **Team seven** has a mean *throughput* of 0.21. **Team eight** has a mean *throughput* of 0.22. **Team nine** has a mean *throughput* of 0.30 and **Team ten** has a mean *throughput* of 0.20, as showed in Figure 5.2.

Year - Quarter	TP
2010-3	0.5
2010-4	0.33
2011-1	0.65
2011-2	0.34
2011-3	0.5
2011-4	0.39
2012-1	0.24
2012-2	0.33
2012-3	0.17
2012-4	0.49
2013-1	0.81
2013-2	0.58
2013-3	0.39
2013-4	0.27
Mean	0.42

(a) Throughput for T1

Year - Quarter	TP
2010-3	0.43
2010-4	0.27
2011-1	0.35
2011-2	0.58
2011-3	0.25
2011-4	0.26
2012-1	0.29
2012-2	0.58
2012-3	0.73
2012-4	0.58
2013-1	0.46
2013-2	0.47
2013-3	0.55
2013-4	0.4
Mean	0.44

(b) Throughput for T2

Year - Quarter	TP
2010-3	0.55
2010-4	0.39
2011-1	0.32
2011-2	0.43
2011-3	0.44
2011-4	0.47
2012-1	0.59
2012-2	0.26
2012-3	0.26
2012-4	0.39
2013-1	0.25
2013-2	0.23
2013-3	0.25
2013-4	0.2
Mean	0.35

(c) Throughput for T3

Year - Quarter	TP
2010-3	1.0
2010-4	0.36
2011-1	0.94
2011-2	1.58
2011-3	1.58
2011-4	1.87
2012-1	1.69
2012-2	1.97
2012-3	2.0
2012-4	1.15
2013-1	1.23
2013-2	1.65
2013-3	1.6
2013-4	0.99
Mean	1.40

(d) Throughput for T4

Year - Quarter	TP
2010-3	0.72
2010-4	0.33
2011-1	0.26
2011-2	0.29
2011-3	0.27
2011-4	0.22
2012-1	0.25
2012-2	0.28
2012-3	0.27
2012-4	0.26
2013-1	0.51
2013-2	0.23
2013-3	0.17
2013-4	0.18
Mean	0.30

(e) Throughput for T5

Year - Quarter	TP
2010-3	0.89
2010-4	0.55
2011-1	0.39
2011-2	0.46
2011-3	0.53
2011-4	0.65
2012-1	0.83
2012-2	0.87
2012-3	0.67
2012-4	0.67
2013-1	0.7
2013-2	0.49
2013-3	0.52
2013-4	0.65
Mean	0.63

(f) Throughput for T6

Year - Quarter	TP
2010-3	0.5
2010-4	0.19
2011-1	0.16
2011-2	0.13
2011-3	0.14
2011-4	0.14
2012-1	0.5
2012-2	0.1
2012-3	0.15
2012-4	0.17
Mean	0.21

(g) Throughput for T7

Year - Quarter	TP
2010-3	0.5
2010-4	0.19
2011-1	0.16
2011-2	0.13
2011-3	0.14
2011-4	0.14
2012-1	0.5
2012-2	0.1
2012-3	0.15
2012-4	0.17
2013-1	0.17
2013-2	0.15
2013-3	0.44
Mean	0.22

(h) Throughput for T8

Year - Quarter	TP
2010-3	0.35
2010-4	0.22
2011-1	0.32
2011-2	0.22
2011-3	0.22
2011-4	0.24
2012-1	0.42
2012-2	0.33
2012-3	0.24
2012-4	0.36
2013-1	0.39
2013-2	0.32
2013-3	0.3
Mean	0.30

(i) Throughput for T9

Year - Quarter	TP
2010-3	0.53
2010-4	0.2
2011-1	0.21
2011-2	0.21
2011-3	0.25
2011-4	0.15
2012-1	0.09
2012-2	0.16
2012-3	0.09
2012-4	0.19
2013-1	0.17
2013-2	0.21
2013-3	0.18
2013-4	0.18
Mean	0.20

(j) Throughput for T10

Figure 5.2: Caption of throughput for teams in SI

Chapter 6

Discussion

The first sections present discussion about the research questions and what other literature has stated against the correlation from this study. The last Section 6.7 will consist of an explanation of the relationship between the moderating variables. To back up any assumptions about the relationship between the moderating variables, correlation graphs and descriptive statistic tables listed in Appendix A will be used.

6.1 WIP and Team size

The results in Chapter 5 show the importance of taken team size into account when a case study like this is conducted. The results showed that the mean correlation between *WIP* and *throughput* were 0.6 and the mean correlation between *WIP* and both *bugs* and *lead time* were 0.5. These three mean correlations are decreased by 0.2 and 0.3 when team size was considered. The results also showed the mean correlation between *WIP* and *team size* were 0.6, the mean correlation between *team size* and *throughput* were 0.5, the mean correlation between *team size* and *bugs* were 0.4 and the mean correlation between *lead time* and *team size* were 0.3. Based on these variables, the results from when team size was **not** considered are discarded. Because the data showed that *team size* had a great impact on the correlation tables and possible gave biased results.

6.2 WIP and throughput

In this work, throughput is a measure to see how productive teams are. Section 2.5 states when lowering WIP-limit the throughput increases. The paper by David

Anderson (D. Anderson et al., 2011) simulated a lean-kanban process. The simulation showed a higher throughput when WIP-limit was used. The paper by Concas (Concas et al., 2013) showed that WIP-limit could lead to improvement in throughput as well. The paper also showed that when taking a process without WIP-limit and simulating it using WIP-limit would outperform the process without WIP-limit. These papers were mentioned in Section 2.5.

The literature says that there should exist a negative correlation between *WIP* and *throughput*. The correlation from this work shows that almost each team has a positive correlation between *WIP* and *throughput*. Which is reflected in the mean correlation of 0.4 between *WIP* and *throughput*, as shown by Tables 5.3 and 5.4. The reason *WIP* and *throughput* have a positive correlation could be because the WIP-limits is set too low or too high. If the WIP-limits are too low, bottlenecks will not occur and the teams are able to produce more, but they will not be allowed to by the WIP-limit. If WIP-limits are too high, it can make people work on too many tasks at a time, which increase lead time for each task.

Section 2.5 states that WIP-limits should be in the range from 1 to 3 tasks per developer. The mean WIP-limit for SI is 1.9 per developer, as stated by Section 5.7. This shows that the WIP-limit could be either too low or too high. The rest of the discussion in this section and Sections 6.3, 6.5 and 6.4 will explore the option that the WIP-limits may have been set too low or too high, and discuss if WIP-limits matter at all. The correlation between *WIP* and *throughput, lead time, churn* and *bugs* will be discussed as well as the correlation between the variables with a mean correlation higher than ± 0.2. Based on correlation tables and descriptive statistic tables, there is hard to find any pattern between the variables with mean correlation from [0.2, -0.2], because these relationship have few significant correlations were the WIP-limit can be investigated in.

There are two teams with a mean WIP-limit of 3 tasks per developer. These teams are respectively **team four** and **team six**, shown by Figures 5.1d and 5.1f. These two teams had the total number of 53 and 99 employees from 2010-4 to 2013-4, which gives a mean of 4 (53/14) and 7 (99/14) persons per quarter, showed by Tables 3.1d and 3.1f. There are no other teams with fewer employees per quarter than **team four** and **team six**, although these two teams have a mean throughput of 6.2 and 4.6, as showed by the total mean row in the descriptive statistic Figures A.16b and A.26b. There are only two teams with a higher mean throughout than these two teams. The teams with a higher mean throughput are **teams one** and **five**, as shown by the total mean row in Figures A.1b and A.21b. In comparison to **team four** and **six**, **team one** had in average 24 (330/14) employees per quarter and **team five** had in average 20 (285/14) employees per quarter.

Team four has a mean throughput of 1.4 per employee and **team six** has a mean throughput of 0.63 per employee, as shown by Tables 5.2d and 5.2f. The team with the third highest throughput mean per employee is **team two** with a mean of 0.44 and a mean WIP-limit of 2.31, showed by Figures 5.2b and 5.1b. **Team one** and **five** have a mean throughput of 0.42 and 0.30 which can be seen in Figures 5.2a and 5.2e. This shows that **team four** outperform the other teams on throughput per developer and **team six** is second by a wide margin. These data shows that WIP-limit may have been too low, because the three teams with the highest throughput per employee also have the highest mean WIP-limits.

But, the *WIP* correlation table, Table 5.3 shows **team two** with a correlation of 0.59 between *throughput* and *WIP*, **team four** with a significant correlation of 0.86 between *throughput* and *WIP* and **team six** with a correlation of 0.08 between *WIP* and *throughput*. These correlations show that the WIP-limit may not be too low, because **team two** and **team four** have a positive significant correlation. And in this discussion one is looking for a negative correlation. The teams with the lowest correlation between *throughput* and *WIP* are; **team ten** with a correlation of -0.21 and a WIP-limit of 1.08, **team five** with a correlation of 0.11 and a WIP-limit of 1.93 and **team six** with a correlation of 0.08 and a WIP-limit of 3.02, as showed in Table 5.3 and Figures 5.1j, 5.1e and 5.1f. Based on these values it will look like WIP-limits do not have an impact on the correlation between *throughput* and *WIP*, because teams with both high and low WIP-limit have both high and low correlation between *WIP* and *throughput*.

The mean correlation between *lead time* and *throughput* are 0.5, as showed by Table 5.16. A positive correlation between these variables states that when *throughput* increases, the time from when a task is taken from the backlog until it is finished also increases and vice versa. This correlation relationship should be negative. Because if there are produced more tasks, the total time to produce tasks should be minimized.

There are five teams with a positive significant correlation, as showed by 5.15. The mean WIP-limits for these five teams are 2.31, 3.21, 1.93, 3.02 and 1.08, as showed by Figure 5.1. These WIP-limits and correlations show no pattern that point towards the fact that WIP-limits matter or that they are too high or low. Because the teams with a significant correlation have both high and low WIP-limits.

The four teams with the lowest WIP-limits have the correlation of 0.06, 0.26, 0.24 and 0.90, showed by 5.3 and Figure 5.1. Three of the five teams with the significant correlations are also the three teams with the highest mean WIP-limits. Three of the four with the lowest WIP-limits also have the lowest correlation between *lead time* and *throughput*. This data shows that most of the teams with highest mean WIP-limits have a significant correlation. And the teams with the lowest mean WIP-limits are the one closes to have a negative correlation between *throughput* and *lead time*. Although the data show evidence that WIP-limits reduce the correlation between *lead time* and *throughput*, there is no evidence that lowering WIP-limits even more will result in a negative correlation between *lead time* and *throughput*. But, based on these data, it looks like WIP-limits matter, because it decrease the correlation between *lead time* and *throughput*. Based on this as well, it looks like the WIP-limits may have been set too high.

The mean correlation between *bugs* and *throughput* are 0.6, seen in Table 5.4. A positive correlation between *bugs* and *throughput* states that if SI produces more, then they produce more bugs and vice versa. This correlation relationship is standard. When one produces more tasks, there are likely to produce more bugs. If one produces zero tasks, one also produce zero bugs. In an optimal process, the correlation between *bugs* and *throughput* should be a negative correlation, so teams can produce more without producing bugs. This is the principle of *JIT*, explained in Section 2.7.

There are six teams with a significant positive correlation between *throughput* and *bugs* as showed by Table 5.3. These six teams have a mean WIP-limit of 2.31, 1.78, 1.93, 0.71, 2.19 and 1.08, as showed by Figure 5.1. These values indicates no pattern between WIP-limits and the correlations between *bugs* and *throughput*. **Team one** and **six** are the only teams with a correlation lower than 0.45. **Team one** has a correlation of 0.05 and a WIP-limit of 0.73. **Team six** has a correlation of -0.02 and a mean WIP-limit of 3.02, as showed by Table 5.3 and Figure 5.1. These data strengthens the assumption that there is no pattern between WIP-limits and the correlation between *throughput* and *bugs*, because teams with both high and low mean WIP-limits have significant correlation between *throughput* and *bugs*.

The mean correlation between *throughput* and *bugs finished, quarter* are 0.4. The mean correlation between *throughput* and *throughput bug* are 0.8 and the mean correlation between *throughput* and *throughput feature* are 0.6, as showed by Table 5.16. The positive correlation between *throughput* and *bugs finished, quarter* states that when SI produces more, they are able to fix more bugs within the same quarter, which shows the same as the correlation between *bugs* and *throughput*. The positive correlation between *throughput* and both the moderating variables state that SI produces more bugs than feature. These correlations also show the same as the correlation between *throughput* and *bugs*.

The mean correlation between *throughput* and *churn* are 0.3, as showed by Table 5.4. A positive correlation states that if the tasks require more effort, they are able to produce more and vice versa. Based on Kanban, one should have small tasks, as stated in Section 2.4. So the correlation between *throughput* and *churn* should be negative based on the principle in Kanban. There are three teams with a significant correlation, as showed by Table 5.3. These three teams have the mean WIP-limit of 3.21, 1.93 and 0.71, showed by Figure 5.1. There are three teams with a negative correlation between *throughput* and *churn*. These negative correlations are -0.14, -0.14 and -0.28, as showed by Table 5.3. The teams with these negative correlations have the mean WIP-limits of 0.73, 3.02 and 2.19, as showed by Figure 5.1. Both teams with a high mean WIP-limit and low WIP-limit have both a significant positive correlation and negative correlation. This states that there is no pattern between WIP-limits and the correlation relationship between *throughput* and *churn*.

The data in this section showed that WIP-limits had an impact on the correlation between *throughput* and *lead time*. The correlation between *throughput* and *lead time* also showed evidence that the WIP-limits could be too high. The other correlations presented showed no evidence of WIP-limits impact. Based on this should SI possibly lower their WIP-limits.

6.3 WIP and lead time

As stated in Section 2.6, lead time could be used to track how quickly software is delivered to customers. Each development process would like to get their lead time as low as possible. Section 2.5 states that WIP-limits are important to reduce lead times. The correlation between *lead time* and *WIP* shows a mean correlation of 0.2, shown by Table 5.4. A positive correlation between *lead time* and *WIP* states that *lead time* increases when *WIP* increases and vice versa. Based on former research, the correlation between *WIP* and *lead time* should be positive.

The four teams with the highest correlation are **team two, four, six** and **seven**. They have the correlations of 0.67, 0.87, 0.32 and 0.77 between *WIP* and *lead time*. The correlations 0.67, 0.87 and 0.77 are significant correlations. **Team two, four, six** and **seven** have the mean WIP-limits of 2.31, 3.21, 3.02 and 2.03 The four teams with the lowest correlation are **teams one, three, eight, nine** and **ten**. They have the correlations of -0.09, -0.03, -0.09, -0.18, -0.05 and the mean WIP-limits of 0.73, 1.78, 0.71, 1.08 and 2.19, showed by Table 5.3 and Figure 5.1. As stated, the correlation between *WIP* and *lead time* should be positive, so when WIP is decreased, the *lead time* decrease as well. These data points towards the fact that WIP-limit may be too low, because four of the five teams with a WIP-limits of 2 or higher have a positive correlation of 0.32 or higher. The team with the highest WIP-limit also has the highest WIP, but the team with the second highest WIP-limit has the correlation of 0.32. The two other teams with the significant correlation between *WIP* and *lead time* have the third and five highest WIP-limits. This shows that a WIP-limit of 2 or higher for four of the five teams gives at least a moderate correlation. This shows that WIP-limits matters based on the correlation between *WIP* and *lead time* and that the WIP-limits may have been set too low.

The mean correlation between *lead time* and *bugs* are 0.5, as showed by Table 5.8. The correlation means that if SI produces fewer *bugs*, the *lead time* for each task will decrease and vice versa. One should try to have a positive correlation. Five teams have a positive significant correlation and these teams have a WIP-limit of 0.73, 2.31, 1.93, 2.03 and 1.08, showed by Table 5.7 and Figure 5.1. There is no pattern between these variables, because the teams WIP-limit are both high and low. The two teams with a correlation lower than 0.44 between *bugs* and *lead time* have the WIP-limits of 3.02 and 0.71, showed by Table 5.7 and Figure 5.1. The WIP-limits 3.02 and 0.71 are almost the highest and the lowest WIP-limits for SI. This data shows that teams with high and low WIP-limits have both high and low correlation between *lead time* and *bugs*. Based on these data the WIP-limits do not make an impact on the correlation between *bugs* and *lead time*, because there is no pattern between the teams with high WIP-limit or low WIP-limit and the correlation between *lead time* and *bugs*.

The mean correlation between *lead time* and *churn* are 0.4, showed by Table 5.8. The teams should have a positive correlation between *lead time* and *churn*, so the developers can work on small tasks and delivery quickly, which is stated in Section 2.4. **Teams one, four, five** and **eight** have a significant positive correlations and these teams have mean WIP-limits of 0.73, 3.21, 1.93 and 0.71, showed by Table 5.7 and Figure 5.1. These values show no pattern, there are both teams with high and low WIP-limits with significant correlation. The three teams with the lowest correlations are the **teams six, nine** and **ten**. These teams have the WIP-limits of 3.02, 2.19 and 1.08, showed by Table 5.7 and Figure 5.1. Based on the WIP-limits and the correlation, WIP-limits cannot be used to regulate the relationship between *lead time* and *churn*, because there is no pattern between WIP-limits and the correlation.

The mean correlation between *lead time* and *bugs finished quarter* and are 0.5, as showed by Table 5.8. This relationship means that if *lead time* decreases, the number of bugs closed in each quarter decreases and vice versa. The relationship between these variables should be negative, so one can close more bugs and minimize *lead time*. **Teams one, two, five, seven** and **ten** have a significant positive correlation and these teams have the mean WIP-limits of 0.73, 2.31, 1.93, 2.03 and 1.08, showed by Table 5.7 and Figure 5.1. There is one team with a negative correlation, which is **team eight**. **Team eight** has the correlation of -0.04 and the mean WIP-limit of 0.71. However, the two other teams with the lowest correlations have the correlations of 0.32 and 0.36. These two teams have the WIP-limits of 1.78 and 3.02, showed by Table 5.7 and Figure 5.1. These data points towards the fact that WIP-limits have no impact on the correlation between *lead time* and *bugs finished quarter*, because there is no pattern between high or low WIP-limits and the correlations.

The former research states that WIP-limits have an impact on *lead time*, the result from this work shows the same. The correlation between *lead time* and *WIP* showed evidence that WIP-limit matter. These data also showed evidence that the WIP-limit may have been set too low. The other correlations with a mean higher than ± 0.2 were investigated and no one of them showed a pattern towards the fact that WIP-limit helps *lead time*.

6.4 WIP and bugs

The paper cited by Shinkle stated (Shinkle, 2009); when WIP-limits were too high, the number of bugs increased. The mean correlation between *WIP* and *bugs* are 0.2, showed by Table 5.12. A significant positive correlation states that if WIP decreases, so does bugs and vice versa. A positive correlation close to 1 between *WIP* and *bugs* should be the optimal goal. There is one team with a significant correlation and that is **team seven** with a correlation of 0.75 as showed in Table 5.3. **Team seven** has a mean WIP-limit of 2.03 shown by Figure 5.1g. The second highest correlation is 0.49 and belongs to **team two**. **Team two** has the WIP-limit of 2.31, showed by Table 5.3 and Figure 5.1b. The teams with the lowest correlation between *WIP* and *bugs* are **team nine** and **team ten**. **Team nine** has the correlation of -0.05 and a mean WIP-limit of 2.19 and **team ten** has the correlation of -0.28 and a mean WIP-limit of 1.08, showed by Table 5.3 and Figures 5.1i and 5.1j. The team with the highest WIP-limit, **team four** has the correlation of 0.27 between *WIP* and *bugs*. The team with the lowest WIP-limit, **team eight** has a correlation of 0.32, showed by Table 5.11 and Figure 5.1.

These values show that the two teams with the highest correlations have the WIP-limits of 2.03 and 0.49. The teams with the lowest correlation have the WIP-limits of 2.19 and 1.08. The team with the highest WIP-limit has the correlation of 0.27. The team with the lowest WIP-limit has the correlation of 0.32. Based on these values, there is no pattern between WIP-limits and the correlation between *WIP* and *bugs*.

The mean correlation between *bugs* and *bugs finished, quarter* are 0.5, showed by Table 5.12. This correlation states if SI produce more bugs, they are also able to close more bugs within each quarter and vice versa. The correlation between *bugs* and *bugs finished, quarter* should be positive. **Team one, six, seven, eight** and **ten** have a significant positive correlation between *bugs* and *bugs finished, quarter*. These teams have a mean WIP-limit of 0.73, 3.02, 2.03, 0.71 and 1.08, as shown by Table 5.11 and Figure 5.1. These values show no pattern between WIP-limit and the correlation between *bugs* and *bugs finished, quarter*. There are two teams with a correlation lower than 0.20 and these teams have the WIP-limit of 3.21 and 2.19. The three remaining teams have the correlation of 0.44, 0.50 and 0.52 and the WIP-limits of 2.31, 1.78 and 1.93, as showed by Table 5.11 and Figures 5.1b, 5.1c and 5.1e. The WIP-limit for the teams with the correlation of 0.44, 0.50 and 0.52 shows some sort of a pattern, these WIP-limits are close. The scattered WIP-limits for the significant correlation between *bugs* and *bugs finished, quarter* shows evidence towards the fact that the WIP-limits do not matter for the correlation relationship between *bugs* and *bugs finished, quarter*. It is not much to state with the two teams with the lowest correlation, since there are only two. But, the conclusion is that it looks like the WIP-limits do not matter in the correlation relationship between *bugs* and *bugs finished, quarter*, because the teams with significant correlations have a high scattering between the mean WIP-limits.

The *avg days in backlog, bugs* has a mean correlation of 0.4 between *bugs*, showed by Table 5.12. This means that the more bugs produced, the longer the bugs stays in the backlog and vice versa. The optimal correlation should be a positive one, so they can produce fewer bugs, and the average days bugs are in backlog decreases. There is one team with a negative correlation. That team is **team two** with a correlation of -0.07 between *avg days in backlog, bugs* and *bugs* and a WIP-limit of 2.31, showed by Table 5.11 and Figure 5.1b. There are two teams with a significant correlation, these teams are **team one** and **team five**. These two teams have the correlation of 0.76 and 0.75 and WIP-limits of 0.73 and 1.93, showed by Table 5.11 and Figures 5.1a and 5.1e. One cannot conclude anything with these three values, one need to take more values into account. There are six teams with close correlations. These correlations are 0.38, 0.30, 0.37, 0.38, 0.37 and 0.39. These teams have the WIP-limits of 1.78, 3.02, 2.03, 0.71, 2.19 and 1.08. These values show no pattern between WIP-limit and the correlation between *avg days in backlog, bugs* and *bugs*, because there are both high and low WIP-limits when the correlations is almost the same.

The literature states that when WIP-limit was too high, the number of bugs produced decreases. Based on these data, there is no evidence proving this statement. The data from this section shows that WIP-limits have no impact on the correlation between *bugs* and *bugs finished, quarter*, *WIP* and *avg days in backlog, bugs*.

6.5 WIP and Churn

Churn is used as surrogate for effort in this work. No literature was found where the relationship between *churn* and *WIP* are investigated. Kanban, as stated in Section 2.4, suggests small tasks and constant flow of released features throughout the development. In order to have frequent releases, the correlation between *churn* and *WIP* should be a positive one. The mean correlation between *churn* and *WIP* are -0.1, as showed by Table 5.4. There is one team with a positive significant correlation. That team has correlation of 0.77 and a WIP-limit of 3.21 as shown by Table 5.3 and Figure 5.1d. The two other teams with the second and third highest correlations have the values of 0.39 and 0.16. These two teams have the WIP-limits of 2.31 and 0.71, showed by Table 5.3 and Figures 5.1b and 5.1h. The correlations between these three teams are scattered and so are the WIP-limits. The three teams with the lowest WIP-limits have the correlation of -0.27, 0.39 and -0.37. The three teams with the highest WIP-limit have the correlation of 0.16, 0.77 and -0.35, showed by Table 5.3 and Figure 5.1. Based on these values, there is no pattern in the correlation between *WIP* and *churn*, because both high and low WIP-limits have high and low correlations between *WIP* and *churn*.

The mean correlation between *bugs finished, quarter* and *churn* are 0.5, which means that if the task effort increases, the number of bugs closed whiten the same quarter increases and vice versa. Based on the principle of Kanban, the task size should be minimized in order to deliver continuously. So the correlation between *bugs finished, quarter* and *churn* should be a negative one. The two **teams seven** and **nine** have a negative correlation of -0.16 and -0.12. These two teams have a mean WIP-limit of 2.03 and 2.19, as showed by Table 5.19 and Figure 5.1. The two teams with the closest correlation to **team seven** and **team nine** have correlation of 0.34, 0.33 and WIP-limits of 3.21 and 3.02, showed by Table 5.19 and Figure 5.1. The five teams with a positive significant correlations have WIP-limits of 0.73, 2.31, 1.78, 1.93 and 0.71, showed by Table 5.19 and Figure 5.1. The teams with low WIP-limits and teams with high WIP-limits showed a pattern in their correlations, but the significant correlations have a scattered WIP-limits, this points towards the fact that the WIP-limits does not matter for the correlation between *bugs finished, quarter* and *churn*.

The correlations between *WIP* and both *churn* and *bugs finished, quarter* showed that WIP-limits had no impact on the correlation.

6.6 Summary

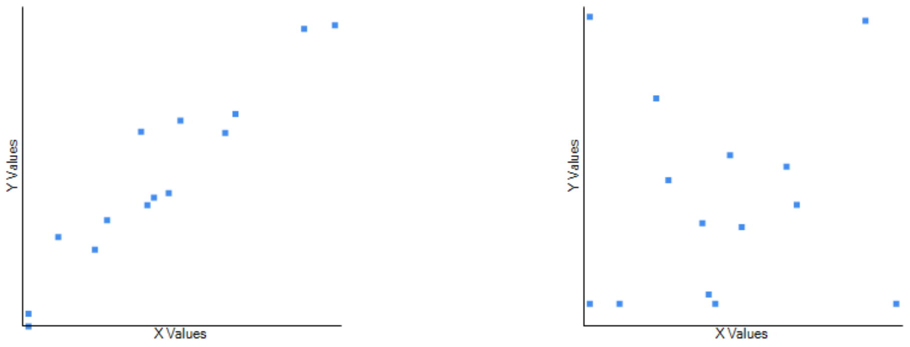
The correlation between *lead time* and *throughput* and *lead time* and *WIP* showed evidence that WIP-limits have an impact in software development. It is not certain that WIP-limit can control the correlation between *lead time* and *throughput* and *lead time* and *WIP*, there could be coincidences, because there is two correlation who showed a pattern between the correlation and WIP-limits. The correlation between *lead time* and *throughput* showed that the WIP-limits may be too high and the correlation between *lead time* and *WIP* showed that the WIP-limits may be too low. Based on this data, the WIP-limit should be between 2 and 3 in order to have a positive correlation between *lead time* and *WIP*. And in order to have lower the correlation between *lead time* and *throughput* the WIP-limit should be lower than 1. Based on this results there could also be other variables that have an impact on the correlation between *lead time* and *throughput*, it is not sure that WIP-limit is the only variable that has an impact on this correlations, but this should be further investigated.

6.7 Discussion of the moderating variables

Chapter 5 highlighted the variance between moderating variables for *churn* and *throughput*. The reason for the teams' variance is explained in the two following subsections.

6.7.1 Throughput

Team one has a significant correlation for both *throughput* and *throughput feature*, but not for *throughput bug*, as showed in the WIP correlation Table 5.1. *Throughput* and *throughput feature* have correlation of 0.74 and 0.73, while *throughput bug* has the correlation of 0.02. The possible cause *throughput bug* does not has a significant correlation with *WIP*, while *throughput* does, might be because *throughput bug* consist of 37% (108/290) of the *throughput* dates, as shown in the total rows in Figures A.1b and A.2b. It is possible to have a close relationship although, since the correlation is based on the mean values. *Throughput feature* has a total mean of 13.7, *throughput bug* has the total mean of 6.4 and *throughput* has the total mean of 11, as shown by Figures A.1b, A.2a and A.2b. The mean values point towards the fact that *throughput feature* represents most of the *throughput* variable. The correlation graphs in Figure 6.1 and *throughput* correlation table in Section 5.4 confirms it. The pattern of dots in Figure 6.1a indicates a clear positive correlation, while the dots in Figure 6.1b has no specific pattern, which reflect the correlation of 0.96 for *throughput feature* and 0.03 for *throughput bug*.

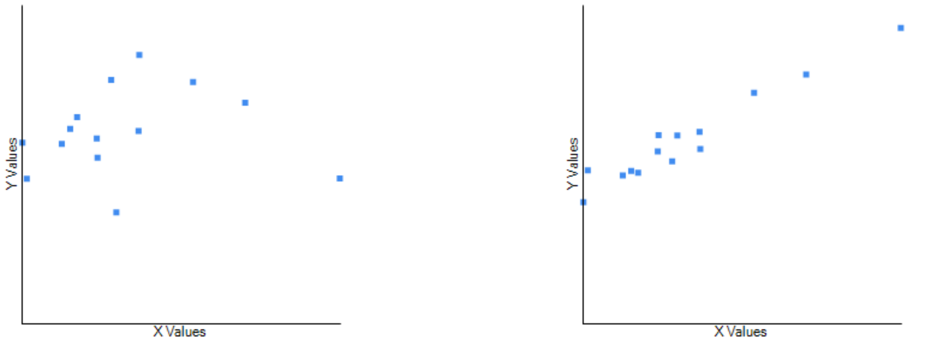


(a) Throughput and throughput feature

(b) Throughput and throughput bugs

Figure 6.1: Correlation graphs between throughput (X-axis) and the throughput moderating variables (Y-axis) for team one.

Team two's *throughput feature* differ from *throughput* based on the correlation from the *bugs* correlation Table 5.9. One could believe the reason is because *throughput bug* consists of 2/3 of *throughput's* (460/690) dates, as shown in the descriptive statistic Figures A.6b and A.7b. But the two tables and Table A.7a show that the total mean of *throughput* is 4.4, while for *throughput bug* it is 4.8 and 3.7 for *throughput feature*. These three variables are quiet close, which could reflect that these three variables could be close based on correlation. But, the correlation graphs in Figure 6.2 and the throughput correlation table in Section 5.4 shows otherwise. Figure 6.2b shows a significant positive correlation, while Figure 6.2a shows dots that are more randomly placed. The throughput correlation Table 5.13 represents the same result with a value of 0.97 for *throughput bug* and 0.1 for *throughput feature*.



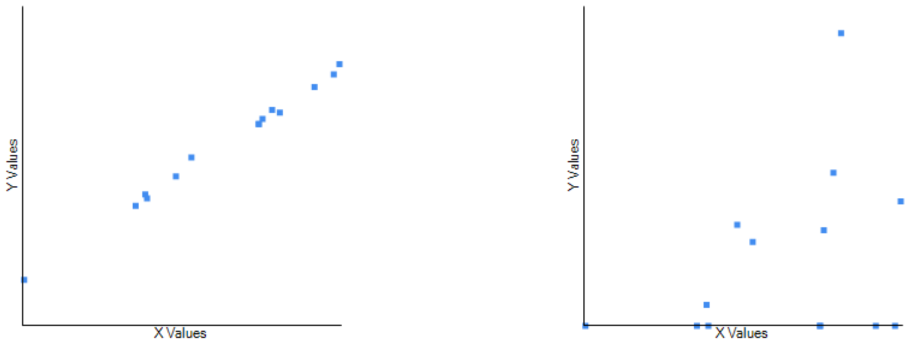
(a) Throughput and throughput feature

(b) Throughput and throughput bug

Figure 6.2: Correlation graphs between throughput (X-axis) and the moderating variables (Y-axis) for team two.

Team three has the correlation of 0.52 for *throughput bug*, 0.57 for *throughput* and 0.71 for *throughput feature*, showed in WIP correlation Table 5.3. Based on the correlation, it looks like *throughput bug* represents most of the *throughput* variable. The descriptive statistic tables empower the assumption. *Throughput* contains 542 dates and has a total mean value of 3.7. *Throughput feature* represents 200 of these dates and has a mean value of 3.3, while *throughput bug* represents the remaining 342 dates and has a total mean value of 4, shown in tables A.11b, A.12a and A.12b. The *throughput* correlation Table 5.15 shows both moderating variables contribute, but *throughput bug* contribute a little more with a correlation of 0.98, while *throughput feature* has a correlation of 0.90.

Team four has the correlation of 0.86 for *throughput*, 0.85 for *throughput feature* and 0.27 for *throughput bug*, showed in WIP correlation Table 5.3. These values indicate that *throughput feature* represents the majority of *throughput*. The dates in the descriptive statistic Figures A.16b, A.17a and A.17b empower this assumption. The dates show that *throughput* consists of 674 dates of which *throughput feature* represents 644 and *throughput bug* represents 30. The mean values on the other hand are 6.2 for *throughput*, 6.2 for *throughput feature* and 5.6 for *throughput bug*. The mean values indicate that *throughput feature* represents the most of *throughput*, but *throughput bug* contributes as well. The correlation graphs in Figure 6.3 and the correlation of 1 for *throughput feature* and 0.32 for *throughput bug* represented by Table 5.15 show otherwise. The correlation table and the graphs show *throughput feature* represents most of the *throughput* variable.



(a) Throughput and throughput feature

(b) Throughput and throughput bug

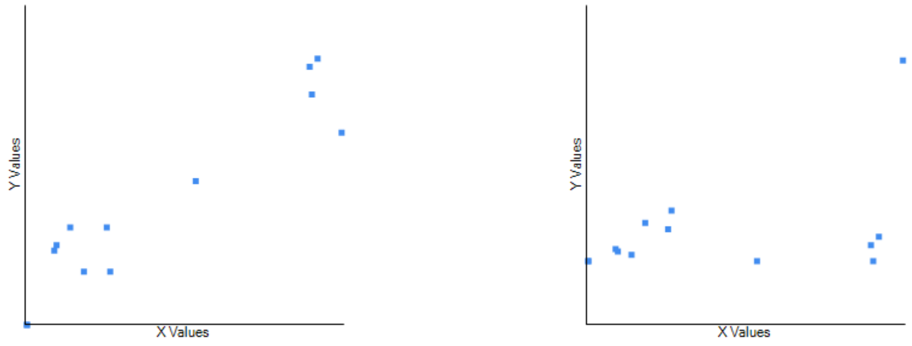
Figure 6.3: Correlation graphs between throughput (X-axis) and the moderating variables (Y-axis) for team four.

Team five has a significant correlation for *throughput bug* with a correlation of 0.54, *throughput* has a significant correlation of 0.52, but *throughput feature* does not has a significant correlation. *Throughput feature* has a correlation of 0.25, as showed in WIP correlation Table 5.1. Based on these values, one can assume that *throughput bug* represents most of *throughput* for team five. The descriptive statistic Figures A.21b, A.22a and A.22b show *throughput* consist of 657 dates. Out of the 657 dates, represents *throughput feature* 108 dates, and *throughput bug* 556 dates. These values also point towards the fact that *throughput bug* represents most of *throughput*. The overall mean for *throughput* is 6.3, for *throughput feature* it is 5.7 and for *throughput bug* it is 6.4. Based on these values, it looks like both the moderating variables contribute. The throughput correlation Table 5.13 proves with the values 0.85 for *throughput feature* and 0.99 for *throughput bug*. This shows that both the moderating variables contribute, but *throughput bug* contribute most.

Team six has a significant correlation to *throughput* and *throughput feature* showed in WIP correlation Table 5.1, while *throughput bug* does not. *Throughput*, *throughput feature* and *throughput bug* have the correlation of 0.64, 0.68 and 0.07. Based on these values, one can assume *throughput feature* contribute a greater proportion to *throughput* than *throughput bug*. The total row in Figures A.26b, A.27a and A.27b show *throughput feature* consist of 609 dates and has a mean value of 4.8, while *throughput bug* consist of 82 dates and a mean value of 3.3. *Throughput* consists of 691 dates and has a mean value of 4.58. With the mean values and the number of dates, the assumption of *throughput feature* represents more of *throughput* than *throughput bug* is empowered. The throughput correlation Table 5.13 proves the assumption with a *throughput feature* correlation of 0.99 and *throughput bugs* value of 0.04.

Team seven has a significant correlation for *throughput*, *throughput feature*, but not *throughput bug*, as showed in WIP correlation Table 5.1. The table showed a correlation of 0.67 for *throughput*, 0.63 for *throughput feature* and 0.55 for *throughput bug*. The difference between the correlations is small, which also can be assumed by the total row in Figures A.31b, A.32a and A.32b. The total rows show *throughput* has a mean of 2.7, while *throughput feature* has a mean correlation of 2.8 and *throughput bug* has a mean correlation of 2.6. *Throughput feature* contributes 156 dates to *throughput* and *throughput bug* contributes 172 dates. Based on these numbers, it looks like both the moderating variables contribute. The *throughput* correlation table in Section 5.4 proves the assumption with correlation of .91 for both *throughput feature* and *bug*.

Team eight has a significant correlation for *throughput bug*, but not *throughput*, as showed in bugs correlation Table 5.9. The descriptive statistic Figures A.36b, A.37a and A.37b show *throughput bug* consist of 99 dates and has a total mean of 1.5, while *throughput feature* consist of 92 dates and a total mean of 3.2. *Throughput* has mean of 2.3 and contains of 191 dates. On the basis of these numbers it will look like both of the moderating variables of *throughput* contributes and both of them should have close relationship to *throughput*. The correlation graphs in Figure 6.4 shows otherwise. The Figure 6.4a shows dots in an upward direction, hence positive correlation, while in Figure 6.4b almost all the dots are all gathered around the low values of Y. The Figures in 6.4 reflects the correlation of 0.94 for *throughput feature* and 0.44 for *throughput bug*. This shows that total dates and total mean can be used as an indicator of the relationship between variables, but it cannot prove it.



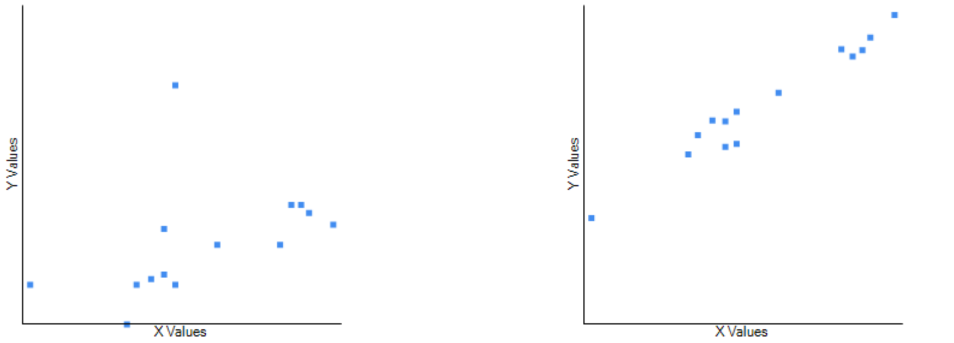
(a) Throughput and throughput feature

(b) Throughput and throughput bug

Figure 6.4: Correlation graphs between throughput (X-axis) and the moderating variables (Y-axis) for team eight.

Team nine has a significant positive correlation for *throughput* and *throughput bug*. The reason *throughput feature* does not has a significant correlation while *throughput* has it, in respect of the bugs correlation Table 5.11, could be because *throughput bug* represents most of the *throughput* variable, as shown in Figures A.41b, A.42a and A.42b. The *throughput* variable contains 521 dates and has a mean correlation of 2.6. *Throughput* *feature* represents 214 of these dates and has a total mean of 2.4, while *throughput bug* represents the remaining 307 dates and has a total mean of 2.8. These variables indicate that both the *throughput* moderating variables contribute. The throughput correlation Table 5.15 proves that both the attributes contributes, but *throughput bug* contributes the most with a correlation of 0.95, while *throughput feature* has a correlation of 0.82.

Team ten has a significant positive relationship for *throughput* and *throughput bug*, but not for *throughput feature*, as showed in WIP correlation Table 5.1. *Throughput* for team ten consists of 404 dates as showed in the total row in Table A.31b. *Throughput bug* represents 335 of these dates, while *throughput feature* represents the remaining 69 dates as shown in Figures A.32a and A.32b. But the overall mean for *throughput*, *throughput feature* and *throughput bug* are 2.2, 2.3 and 2.2, which could reflect a close relationship between these three variables. The throughput correlation table in Section 5.4 and the correlation graphs in Figure 6.5 disproves that assumption. The Figure 6.5a shows a vague significant positive correlation, while Figure 6.5b indicates a clear positive correlation. The correlation table shows *throughput bug* with a correlation of 0.98 and *throughput feature* with a correlation of 0.43. Which proves that *throughput bug* represents most of *throughput*.



(a) Throughput and throughput feature (b) Throughput and throughput bug

Figure 6.5: Correlation graphs between throughput (X-axis) and the moderating variables (Y-axis) for team ten.

6.7.2 Churn

The WIP correlation Table 5.1 shows the three churn variables for **team one** with scattered correlation. *Churn* has a correlation of 0.47, *churn feature* has a correlation of 0.72 and *churn bug* a correlation of 0.15. Judging from these variables, it will look both *churn feature* and *churn bug* contribute to *churn*. The descriptive statistic Figures A.3b, A.4aand A.4b empowers the assumption. The total mean of *churn* is 20.2, while *churn feature* has total mean of 24.5 and *churn bug* has a total mean of 16.8. The descriptive statistic tables also shows that *churn feature* contribute 150 dates to *churn*, while *churn bug* contribute 189 dates. The churn correlation table in Section 5.5 proves the assumption. Both *churn feature* (0.57) and *churn bug* (0.80) has a significant positive correlation to *churn*. Judging from the correlation table from Section 5.4, one can assume that correlation is not transitive. The paper "The Non-Transitivity of Pearson's Correlation Coefficient: An Educational Perspective" (Vesaliusstraat, n.d.) proves the assumption.

Churn and *churn bug* have a significant negative correlation for **team two**. *Churn feature* on the other hand has a correlation of -0.25, showed in WIP correlation Table 5.1. According to the correlation, one can assume that *churn bug* represents most of *churn*. The Figures A.8b, A.9a and A.9b empower the assumption. *Churn bug* contribute 521 dates and has a total mean correlation of 36.3. *Churn feature* contribute 257 dates and has a mean value of 100.6. The total churn contains 778 dates and has a total mean of 57.6. The *churn* correlation table in Section 5.5 shows both the *churn* variables contribute with a correlation of 0.7 for *churn bug* and 0.58 for *churn feature*.

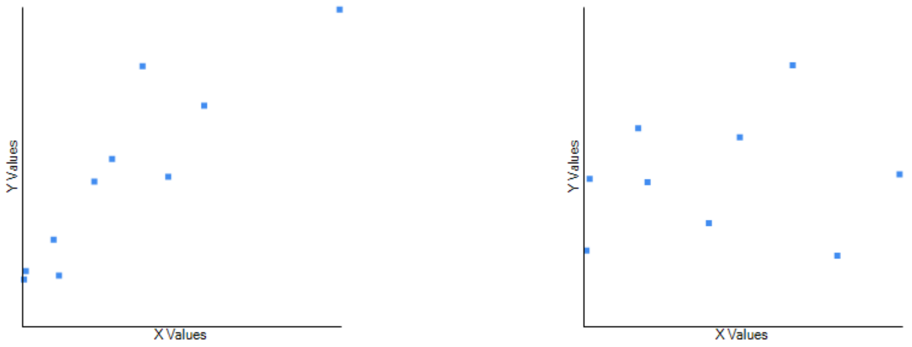
The three *churn* variables for **team three** have the correlation of -0.45 for *churn*, -0.27 for *churn feature* and -0.64 for *churn bug*, as showed in the correlation Table 5.5, for lead time. These values indicate more contribution from *churn bug* than *churn feature*. The total dates and the mean from Figures A.13b, A.14a and A.14b empower the assumption. The total *churn* consists of 576 dates and a total mean of 61.8. *Churn feature* represents 205 of these dates and has a total mean of 98.9. *Churn bug* answers for the remaining 371 dates and has a total mean of 41.4. The *churn* correlation table in Section 5.5 proves the assumption. Still, *churn bug* has a correlation of 0.85, while the correlation between *churn feature* and *churn* is 0.90.

Churn for **team four** has the correlation of 0.97, *churn feature* has the correlation of 0.96 and *churn bug* has the correlation of 0.2, as showed in the correlation table for lead time, Table 5.5. The descriptive statistic Figures A.18b, A.19a and A.19b shows that *churn* consist of 574 dates. *Churn bug* represents 78 of these dates and *churn feature* represents the remaining 496 dates. *Churn features* has the total mean of 8.4, *churn bug's* mean is 1 and *churn's* mean is 7.4. These variables clearly indicate the strong relationship between *churn feature* and *churn*. The churn correlation Table 5.17 verifies the theory with *churn feature* has the correlation of 0.99 and *churn bug* has the correlation of 0.13.

Churn bug for **team five** has a significant positive correlation of .94, while *churn feature* has the correlation of 0.22, as shown in *churn* correlation Table 5.17. This proves that *churn bug* represents most of *churn*. The Figures A.23b, A.24a and A.24b show the same result. *Churn* consists of 698 dates and has a total mean of 33.4, while *churn feature* and *churn bug* represents has 123 dates and 575 dates. The total mean of *churn feature* is 52.1 and for *churn bug* it is 29.4.

Team six has a significant correlation of 0.77 for *churn bug*, while both *churn* and *churn feature* has the correlation of -0.30 and -0.36, showed in WIP correlation Table 5.1. Based on these values, one can assume *churn feature* represents most of *churn*. The Figures A.28b, A.29a and A.29b back this theory. The tables show *churn feature* contribute 576 dates to *churn* and has a total mean of 105.9. *Churn bug* contains of 180 dates and has a total mean value of 73.8. *Churn* has 756 dates and a total mean value of 98.3. The churn correlation table in Section 5.5 proves the assumption of *churn feature* represent most of *churn*, with the correlation 0.98 for *churn feature* and -0.02 for *churn bug*.

Churn feature for **team seven** has the correlation of 0.91 while *churn bug* has the correlation of 0.13, as showed in churn correlation Table 5.17. The descriptive statistic Figures A.33b, A.34a and A.34b shows that *churn* contains 359 dates and has a total mean of 77.3. *Churn feature* represents 141 of these dates, and has a total mean of 121.2. *Churn bug* represents the 218 remaining dates and has a total mean of 48.9. Based on these values, one could assume that both *churn feature* and *churn bug* contribute to *churn*, but the churn correlation table and the graphs in Figure 6.6 disapproves that.

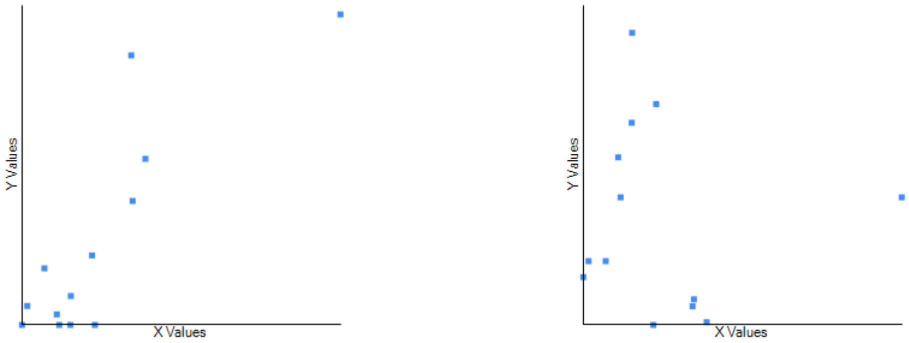


(a) Churn and churn feature

(b) Churn and churn bug

Figure 6.6: Correlation graphs between the churn (X-axis) and the moderating variables (Y-axis) for team seven.

Team eight has a significant correlation to *churn* and *churn feature*, but not *churn bug*, as showed in lead time's correlation Table 5.5. The correlation for *churn* is 0.91, for *churn bug* it is -0.12 and for *churn feature* it is 0.79. Based on these variables, it looks like *churn feature* represents a greater part of the *churn*. The Figures A.38b, A.39a and A.39b indicate otherwise. *Churn* is composed of 137 dates, and has a total mean of 13.4. *Churn feature* represents 79 of these tasks and has a total mean of 17.4. *Churn bug* represents the remaining 58 tasks and has a total mean of 8. The values indicate both *churn feature* and *churn bug* contribute to *churn*. The Figure 6.7 and correlation Table 5.17 disapproves the assumption. The *churn* correlation Table 5.17 shows that *churn feature* has the correlation of 0.84, while *churn bug* has the value of -0.1. This is the same as shown in the correlation graphs. Figure 6.7a indicates a clear positive correlation, while Figure 6.7b shows no pattern and shows a correlation close to 0.

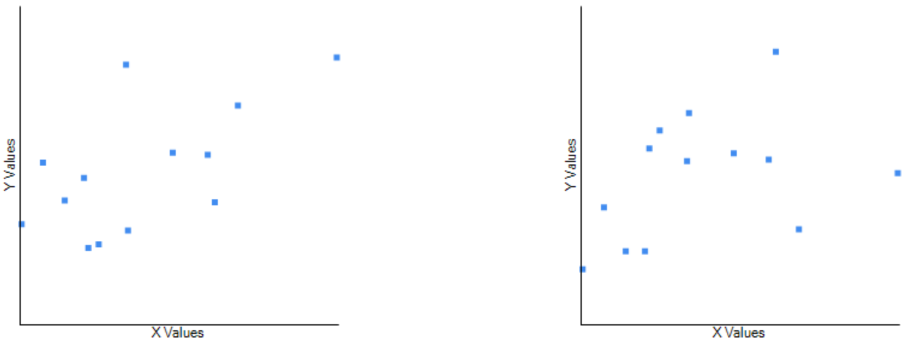


(a) Churn and churn feature

(b) Churn and churn bug

Figure 6.7: Correlation graphs between the churn (X-axis) and the moderating variables (Y-axis) for team eight.

Team nine has a significant correlation of -0.62 to *churn feature*, while *churn* has the correlation of -0.48 and *churn bug* has the correlation of -0.04, as showed in bugs correlation Table 5.9. Based on the correlation, it looks like *churn feature* represents most of *churn*. The descriptive statistic data show otherwise. *Churn* consists of 548 dates, *churn feature* represents 201 of these dates and *churn bug* represents the remaining 347 dates, as shown in Figures A.43b, A.44a and A.44b. The total mean of *churn feature* is 115.9, the total mean for *churn* is 72.2 and the total mean for *churn bug* is 46.1. Judging from these numbers, both *churn feature* and *churn bug* contribute to *churn*. In the correlation graphs in Figure 6.8, one can see Figure 6.8a represents a positive correlation. The Figure 6.8b represents a positive correlation, but not as high as Figure 6.8a. The *churn* correlation states the same, Table 5.17 shows *churn feature* correlation is 0.60 and *churn bug* correlation is 0.39. This proves that *churn feature* represents most of *churn*.



(a) Churn and churn feature

(b) Churn and churn bug

Figure 6.8: Correlation graphs between the churn (X-axis) and the moderating variables (Y-axis) for team nine.

Team ten has a significant correlation of 0.94 for *churn bug* and *churn feature* has the correlation 0.14, as showed in churn correlation Table 5.17. This shows that churn bug represents most of *churn*. Based on the Figures A.48b, A.49a and A.49b, one can see that *churn* contains 361 dates and has a total mean value of 45.4. *Churn feature* represents 69 of these dates and has a total mean of 75.5, while *churn bug* represents 292 of *churn's* dates and has a total mean of 38.3. These data also shows the relationship between *churn* and *churn bug*.

Chapter 7

Conclusion

In this work the main goal was to investigate the research question: *Does WIP-limit in software development matter?* If so: *How can one find the optimal WIP-limit?* To answer the research questions, a data set from a software company called Software Innovation was interpreted. The data set is based on metadata about tasks from 2010 to 2013. More than ten mean correlations were investigated based on the WIP-limits. The correlation between *throughput* and *lead time* and *WIP* and *lead time* showed evidence that WIP-limits have an impact, but the rest of the correlations showed no impact of WIP-limits.

One can see a pattern in the correlation between *throughput* and *lead time* and *WIP* and *lead time*. In the correlation between *throughput* and *lead time* one can see most of the teams with low mean WIP-limits have also low correlation, and most of the teams with high mean WIP-limit have high correlations. Although, there is no evidence showing that lowering WIP-limits even more will produce a negative correlation between *throughput* and *lead time*, but investigating this is suggested as further work. In the correlation between *WIP* and *lead time* the four highest correlations belonged to four of the five teams with a WIP-limit higher than 2, and three of these correlation were significant. The correlation between *WIP* and *lead time* do show that most of the teams with a mean WIP-limit of 2 or higher have at least a moderate correlation.

Based on these data, there is no optimal WIP for any given context. It depends if one wants to optimize the correlation between *lead time* and *throughput* or the correlation between *lead time* and *WIP*. So to answer: *How can one find the optimal WIP-limit?* one should take the correlations of *lead time* and *throughput* and *lead time* and *WIP* into consideration when looking for the optimal WIP-limit based on these data.

7.1 Future work

The conclusion from this work is made on one case study. It is recommend doing the same calculation as in this work with another data set and comparing the outcome. It is also suggested a different approach as looking more deeply into the relationship between *WIP* and *team size*, one could measure the number of employees working on each task instead of take the number of employees per quarter and divide on the mean of each variables value.

There is also done another research on the data from SI, it is recommended comparing the result from this work against the other research. It is also recommended to try to decrease the WIP-limit to see if their current WIP-limit is too high, based on the correlation between *lead time* and *throughput*. It is also recommended to check the cycle time relationship for SI against the *lead time* and *WIP* correlation. It is also recommend to investigate if there is any other variables that plays a part in the correlation between *lead time* and *throughput*. It is also recommended using other statistical methods to falsify or verify the findings from this work.

Appendices

Appendix A

Descriptive statistics (DS) for the ten teams

The following tables show the descriptive statistics for the ten teams before team size was considered.

A.1 Team 1 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	25	3.6	4	0.6	5	3
2010-4	92	0.7	1	0.7	3	0
2011-1	90	3.4	1	6.9	30	0
2011-2	91	13.2	4	14.5	51	2
2011-3	92	1.8	2	0.6	3	1
2011-4	92	14.3	4	22.7	97	1
2012-1	91	22.2	21	14.5	67	4
2012-2	91	30.3	23	29	107	9
2012-3	92	36	38.5	13.6	65	18
2012-4	92	34.7	28.5	16.9	99	25
2013-1	90	32.8	25	13.7	85	25
2013-2	91	67.1	54	44.3	178	3
2013-3	92	7.4	3	8.8	31	1
2013-4	76	5	1	8.1	35	1
Total	1197	20.5	12	26.2	178	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	3	3	1	3.5	7	1
2010-4	3	1	1	0	1	1
2011-1	7	10.4	11	8.1	25	1
2011-2	32	9.4	10	6.7	26	1
2011-3	2	1	1	0	1	1
2011-4	25	14.9	10	14.6	49	1
2012-1	49	8.6	5	8.1	33	1
2012-2	45	11.2	3	16	56	1
2012-3	34	5.5	3	6.3	23	1
2012-4	17	14.2	14	13.7	44	1
2013-1	13	19.5	17	17	58	1
2013-2	26	21.6	18	16.9	60	1
2013-3	17	9	7	7.7	27	1
2013-4	17	6.3	3	7.5	24	1
Total	290	11	6	12.5	60	1

(b) DS - Throughput

Figure A.1: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	7	7	-	7	7
2011-1	7	10.4	11	8.1	25	1
2011-2	24	10.1	10	5.6	24	1
2011-3	1	1	1	-	1	1
2011-4	11	16.6	13	11	35	4
2012-1	16	15.2	15	9.3	33	1
2012-2	26	16.1	5	17.7	56	1
2012-3	23	6	4	6.6	23	1
2012-4	14	15.1	14.5	14.3	44	1
2013-1	10	23.3	20	17.4	58	3
2013-2	21	23.6	24	18.1	60	1
2013-3	16	9.5	7.5	7.7	27	1
2013-4	12	8.3	7.5	8.1	24	1
Total	182	13.7	10	13.2	60	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	2	1	1	0	1	1
2010-4	3	1	1	0	1	1
2011-2	8	7.5	2	9.3	26	1
2011-3	1	1	1	-	1	1
2011-4	14	13.6	5	17.3	49	1
2012-1	33	5.3	5	5	21	1
2012-2	19	4.5	1	10.5	47	1
2012-3	11	4.4	3	5.8	21	1
2012-4	3	10	3	12.1	24	3
2013-1	3	7	3	8.7	17	1
2013-2	5	13.4	13	7.1	21	3
2013-3	1	1	1	-	1	1
2013-4	5	1.4	1	0.9	3	1
Total	108	6.4	3	9.6	49	1

(b) DS - Throughput bug

Figure A.2: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	13	13	-	13	13
2010-4	2	8.5	8.5	9.2	15	2
2011-2	28	13.1	7.5	16.5	78	1
2011-3	1	5	5	-	5	5
2011-4	28	15.7	14.5	11.2	45	1
2012-1	66	12.5	9	11.5	49	1
2012-2	47	18.7	12	19	107	1
2012-3	32	9.9	7	11.3	49	1
2012-4	26	18.1	5.5	58.3	303	1
2013-1	19	18.7	6	27	103	2
2013-2	48	27.9	8.5	75.8	508	1
2013-3	25	15.6	5	25.1	110	1
2013-4	16	14.5	4.5	24	76	1
Total	339	16.7	8	36.2	508	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	13	13	-	13	13
2010-4	2	30	30	41	59	1
2011-2	28	20.1	13	18	74	1
2011-3	1	2	2	-	2	2
2011-4	28	22.9	17.5	19.9	86	0
2012-1	66	18.6	12.5	19.8	97	0
2012-2	47	20.9	17	20	103	0
2012-3	32	13.9	5	20.6	75	0
2012-4	26	24	9	58.8	302	0
2013-1	19	17.8	9	25.3	99	0
2013-2	48	27.9	9.5	73.5	495	0
2013-3	25	14.7	5	23	99	0
2013-4	16	15.1	4.5	23.2	72	0
Total	339	20.2	10	36.8	495	0

(b) DS - Churn

Figure A.3: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-2	8	23.2	22	21.2	49	1
2011-4	8	24.5	14.5	28	86	4
2012-1	20	17.9	17	12.4	48	0
2012-2	21	23.8	16	25.6	103	0
2012-3	20	11.2	3	19.4	75	0
2012-4	16	30.9	9.5	74.1	302	0
2013-1	11	24.7	9	31.8	99	0
2013-2	23	42.6	7	104.9	495	0
2013-3	17	16.6	4	26.7	99	0
2013-4	6	30.3	16	31.3	72	0
Total	150	24.5	10	51.6	495	0

(a) DS - Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	13	13	-	13	13
2010-4	2	30	30	41	59	1
2011-2	20	18.9	13	17	74	2
2011-3	1	2	2	-	2	2
2011-4	20	22.2	19	16.5	65	0
2012-1	46	18.9	10.5	22.4	97	0
2012-2	26	18.6	18	14.2	43	0
2012-3	12	18.4	6.5	22.7	63	1
2012-4	10	12.9	8.5	15.2	52	0
2013-1	8	8.2	8.5	4.4	16	1
2013-2	25	14.4	13	9.6	34	2
2013-3	8	10.8	5.5	12.6	38	0
2013-4	10	5.9	3	10.1	33	0
Total	189	16.8	11	17.2	97	0

(b) DS - Churn bug

Figure A.4: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	1	1	-	1	1
2010-4	4	1	1	0	1	1
2011-2	32	4.2	3.5	3.6	14	1
2011-3	5	1	1	0	1	1
2011-4	36	4.9	2.5	5.4	22	1
2012-1	43	3.5	3	2.3	10	1
2012-2	33	5.4	3	5.5	21	1
2012-3	16	2.4	1.5	1.8	6	1
2012-4	13	2.8	2	1.8	6	1
2013-1	8	3.5	3	2.5	7	1
2013-2	27	5.8	4	4.8	17	1
2013-3	11	1.3	1	0.5	2	1
2013-4	10	1.7	1	1.9	7	1
Total	240	4	2	4	22	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	1	0	1	100	0
2010-4	4	0	4	100	0
2011-2	130	3	133	97.7	2.3
2011-3	1	4	5	20	80
2011-4	156	22	178	87.6	12.3
2012-1	146	4	150	97.3	2.7
2012-2	176	3	179	98.3	1.7
2012-3	37	2	39	94.9	5.1
2012-4	33	3	36	91.7	8.3
2013-1	24	4	28	85.7	14.3
2013-2	157	0	157	100	0
2013-3	13	1	14	92.9	7.1
2013-4	17	0	17	100	0
Mean	63.9	3.3	67.3	83.3	16.7

(b) DS - Bugs per quarter

Figure A.5: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.2 Team 2 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	25	14.4	15	6.2	23	6
2010-4	92	21.4	20	7.2	41	9
2011-1	90	27.2	27.5	4.9	38	17
2011-2	91	29.7	27	14.4	62	12
2011-3	92	32.6	30	9.2	56	18
2011-4	92	30.1	30	10.1	46	13
2012-1	91	20	19	4.6	31	8
2012-2	91	25.3	26	10.3	51	6
2012-3	92	24.2	22.5	7.9	45	11
2012-4	92	21.6	23	10.5	47	3
2013-1	90	19.7	20	5.8	35	8
2013-2	91	28	27	4.4	37	15
2013-3	92	18.7	19	4.5	28	9
2013-4	87	13.3	14	6.6	29	2
Total	1208	23.8	23	9.8	62	2

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	16	4.2	3	4	16	1
2010-4	54	4.1	3	3.9	21	1
2011-1	57	4.6	4	3.6	17	1
2011-2	41	6.9	5	5.7	25	1
2011-3	52	3.8	2	3.6	15	1
2011-4	52	3.7	3	2.7	11	1
2012-1	55	4.3	3	3.4	12	1
2012-2	51	4.1	3	3.5	21	1
2012-3	57	5.8	5	4.3	18	1
2012-4	52	5.2	4.5	3.7	15	1
2013-1	51	4.6	3	3.6	16	1
2013-2	50	3.3	3	2.4	9	1
2013-3	55	3.9	4	2.9	16	1
2013-4	47	3.2	3	2.7	13	1
Total	690	4.4	3	3.7	25	1

(b) DS - Throughput

Figure A.6: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	5	4.6	3	3.2	10	2
2010-4	22	3.1	2.5	2.8	11	1
2011-1	15	5.1	4	4.4	17	1
2011-2	5	3.2	3	1.3	5	2
2011-3	25	3.7	2	3.8	14	1
2011-4	10	3.4	3	2.9	11	1
2012-1	9	2.1	1	2	7	1
2012-2	16	3.5	3.5	2.3	8	1
2012-3	12	4.2	3.5	1.9	8	1
2012-4	25	4.6	3	3.9	13	1
2013-1	11	3.6	3	3.3	11	1
2013-2	27	2.7	2	2	9	1
2013-3	29	3.9	3	2.7	11	1
2013-4	19	3.4	2	3	10	1
Total	230	3.7	3	3	17	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	11	4.1	3	4.4	16	1
2010-4	32	4.8	4	4.4	21	1
2011-1	42	4.4	4	3.3	13	1
2011-2	36	7.4	5.5	5.9	25	1
2011-3	27	3.9	3	3.5	15	1
2011-4	42	3.7	3	2.7	11	1
2012-1	46	4.7	3.5	3.5	12	1
2012-2	35	4.3	3	4	21	1
2012-3	45	6.3	5	4.6	18	1
2012-4	27	5.8	5	3.3	15	1
2013-1	40	4.8	4	3.7	16	1
2013-2	23	3.9	3	2.8	9	1
2013-3	26	3.8	4	3.2	16	1
2013-4	28	3.1	3	2.5	13	1
Total	460	4.8	4	3.9	25	1

(b) DS - Throughput bug

Figure A.7: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	19	15	9	14.2	55	1
2010-4	53	13.7	9	13.2	55	1
2011-1	67	14.4	11	11.3	67	2
2011-2	41	19.4	13	17.5	79	2
2011-3	55	15.6	11	14	55	1
2011-4	49	14.5	10	13.9	61	1
2012-1	63	11.4	8	10.3	41	1
2012-2	58	11.2	10	8.8	38	1
2012-3	83	15.4	13	12	66	1
2012-4	70	12.5	9	12.1	68	1
2013-1	70	12.6	9.5	10.7	44	1
2013-2	40	11.8	7.5	11.1	44	1
2013-3	59	11.3	6	12.1	49	1
2013-4	51	12	10	12.3	71	1
Total	778	13.5	10	12.3	79	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	19	69.6	14	106.2	352	3
2010-4	53	78.6	26	120.7	493	1
2011-1	67	35.1	20	57.3	407	1
2011-2	41	40.5	21	64.5	383	2
2011-3	55	57.8	30	86.3	379	1
2011-4	49	46.9	28	55.7	294	2
2012-1	63	58.4	23	81.3	377	0
2012-2	58	58.1	19	99.3	408	0
2012-3	83	43.4	20	68.6	433	0
2012-4	70	69.8	20	112.9	513	0
2013-1	70	47.4	14.5	94.1	467	0
2013-2	40	43.1	11.5	76.3	310	0
2013-3	59	77.7	26	114.5	459	0
2013-4	51	91.3	32	138.8	474	0
Total	778	57.6	22	94.2	513	0

(b) DS - Churn

Figure A.8: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	6	148	93.5	152.8	352	12
2010-4	18	178.4	118	164.1	493	10
2011-1	19	48.9	37	53.2	214	1
2011-2	4	134.5	70.5	168.8	383	14
2011-3	16	128.1	91	128.1	379	1
2011-4	11	106.3	120	87.6	294	12
2012-1	16	112.8	54.5	125.1	377	0
2012-2	21	90.1	34	122.4	408	0
2012-3	29	52.7	19	67.2	226	0
2012-4	32	103.7	27	151.4	513	0
2013-1	23	93.7	32	150.3	467	0
2013-2	14	82.1	25	117.4	310	0
2013-3	28	97	20	142	459	0
2013-4	20	125.1	37.5	163.5	463	0
Total	257	100.6	38	131.3	513	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	13	33.5	12	52	193	3
2010-4	35	27.2	19	28.8	153	1
2011-1	48	29.6	18	58.5	407	1
2011-2	37	30.3	19	34	152	2
2011-3	39	29	20	34.3	196	1
2011-4	38	29.7	19.5	24.6	95	2
2012-1	47	39.9	20	49.3	237	1
2012-2	37	39.9	15	79.6	380	0
2012-3	54	38.4	22.5	69.5	433	0
2012-4	38	41.2	19.5	52.3	226	0
2013-1	47	24.7	12	29.5	127	0
2013-2	26	22.1	11	24.5	91	0
2013-3	31	60.2	26	80.9	296	0
2013-4	31	69.5	29	118.1	474	4
Total	521	36.3	19	58.4	474	0

(b) DS - Churn feature

Figure A.9: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	20	2.6	2	3.5	17	1
2010-4	40	2.5	2	1.7	9	1
2011-1	47	2.4	2	1.8	8	1
2011-2	40	3.8	3	2.5	13	1
2011-3	43	2.6	2	2.4	13	1
2011-4	47	2.5	2	1.6	8	1
2012-1	35	3.3	3	3	16	1
2012-2	34	2.3	2	1.5	7	1
2012-3	43	3.6	2	2.6	10	1
2012-4	33	3.9	3	3.1	14	1
2013-1	38	2.2	2	1.2	6	1
2013-2	32	1.9	1.5	1.2	5	1
2013-3	35	1.8	1	1.1	5	1
2013-4	37	1.9	1	1.3	7	1
Total	536	2.7	2	2.2	17	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	30	23	53	56.6	43.4
2010-4	65	34	99	65.7	34.3
2011-1	101	13	114	88.6	11.4
2011-2	142	8	150	94.7	5.3
2011-3	87	24	111	78.4	21.6
2011-4	90	29	119	75.6	24.4
2012-1	94	23	117	80.3	19.7
2012-2	70	9	79	88.6	11.4
2012-3	146	7	153	95.4	4.6
2012-4	101	27	128	78.9	21.1
2013-1	78	5	83	94.0	6.0
2013-2	58	3	61	95.1	4.9
2013-3	62	2	64	96.9	3.1
2013-4	69	0	69	100	0
Mean	66.4	12.3	78.8	74.4	25.6

(b) DS - Bugs per quarter

Figure A.10: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.3 Team 3 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	24	9.3	10	6.4	23	1
2010-4	92	13.9	13	3.9	25	5
2011-1	90	15.3	15.5	3.9	23	7
2011-2	91	23.5	24	4.2	37	13
2011-3	92	20.7	20	5.8	34	9
2011-4	92	23.3	23	6.9	36	9
2012-1	91	24.9	24	6.6	42	13
2012-2	91	23.9	23	3.4	34	19
2012-3	92	28	29	5.1	38	21
2012-4	92	29.6	28.5	5.3	44	22
2013-1	90	16.2	15	5	27	9
2013-2	91	7	6	3.2	13	2
2013-3	92	7	7	2	14	3
2013-4	67	5.6	5	2.6	13	2
Total	1187	18.5	19	9.1	44	1

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	16	3.3	3	2.9	12	1
2010-4	54	3.5	3	3	15	1
2011-1	42	2.2	2	1.4	7	1
2011-2	45	4.3	3	3.8	20	1
2011-3	51	4	3	3.4	15	1
2011-4	50	4.7	3	5.2	27	1
2012-1	46	6.5	5	5.7	20	1
2012-2	40	2.9	2	3.2	15	1
2012-3	36	3.4	2.5	3.1	13	1
2012-4	51	5	4	4.4	22	1
2013-1	42	3.2	2	2.9	10	1
2013-2	22	1.6	1	1	5	1
2013-3	29	2	1	2	11	1
2013-4	18	1.6	1	1.1	5	1
Total	542	3.7	3	3.8	27	1

(b) DS - Throughput

Figure A.11: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	3	2.3	3	1.2	3	1
2010-4	19	2.9	2	2.4	9	1
2011-1	29	2.3	2	1.5	7	1
2011-2	24	4	3.5	2.4	8	1
2011-3	23	4.3	3	3.5	13	1
2011-4	19	4	3	4.5	21	1
2012-1	10	5.4	2	5.6	16	1
2012-2	18	2.2	1.5	1.5	6	1
2012-3	12	3.9	1.5	4.3	13	1
2012-4	17	4.8	3	4.5	17	1
2013-1	8	2.1	1.5	1.7	6	1
2013-2	3	1.7	2	0.6	2	1
2013-3	8	1.8	1.5	0.9	3	1
2013-4	7	1.3	1	0.5	2	1
Total	200	3.3	2	3.2	21	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	13	3.5	3	3.2	12	1
2010-4	35	3.9	3	3.3	15	1
2011-1	13	2.1	2	1	3	1
2011-2	21	4.8	3	4.9	20	1
2011-3	28	3.8	3	3.5	15	1
2011-4	31	5.2	3	5.5	27	1
2012-1	36	6.8	5	5.8	20	1
2012-2	22	3.5	2	4.1	15	1
2012-3	24	3.2	3	2.4	9	1
2012-4	34	5.2	4	4.5	22	1
2013-1	34	3.4	2	3	10	1
2013-2	19	1.6	1	1.1	5	1
2013-3	21	2.1	1	2.2	11	1
2013-4	11	1.7	1	1.3	5	1
Total	342	4	3	4.1	27	1

(b) DS - Throughput bug

Figure A.12: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	21	10.6	11	6.4	24	1
2010-4	59	11.6	9	8.9	34	1
2011-1	27	8.7	8	5.9	18	1
2011-2	51	13	11	9.2	34	1
2011-3	48	14.4	10.5	11.3	49	2
2011-4	62	17.8	15	11.7	46	1
2012-1	59	22.6	18	16.6	76	1
2012-2	39	19.5	16	15.2	54	1
2012-3	40	17.1	12.5	17.3	72	1
2012-4	66	12.5	8	12.5	58	1
2013-1	44	12.1	6.5	12.7	60	1
2013-2	20	11	10	9.7	34	1
2013-3	28	7.9	4.5	8.1	29	1
2013-4	12	18	14	18.7	75	1
Total	576	14.6	11	12.9	76	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	21	120.1	58	138.4	383	3
2010-4	59	60.2	28	67.2	295	2
2011-1	27	73.4	65	62.6	320	2
2011-2	51	79.7	36	104.8	423	1
2011-3	48	67.5	31.5	91	407	0
2011-4	62	37.4	18	66.3	343	0
2012-1	59	47.3	27	55.3	286	0
2012-2	39	38.2	20	66.4	365	0
2012-3	40	66.7	23.5	99.3	406	0
2012-4	66	79.7	28.5	114.5	494	0
2013-1	44	36.7	22	42.6	174	0
2013-2	20	59.5	40	60.9	216	0
2013-3	28	70.6	48.5	90.1	403	0
2013-4	12	79.2	49.5	80.5	237	0
Total	576	61.8	29	85.4	494	0

(b) DS - Churn

Figure A.13: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	7	197.4	141	136.1	383	16
2010-4	24	78.7	55.5	77	295	2
2011-1	15	89.9	75	73.8	320	14
2011-2	24	128.6	87.5	129.2	423	4
2011-3	24	88.4	40.5	108.8	407	0
2011-4	23	69.9	35	94.7	343	0
2012-1	17	80	59	76.2	286	0
2012-2	9	74.2	33	117	365	0
2012-3	15	111	66	126.1	406	0
2012-4	25	122.8	57	142.6	494	0
2013-1	11	53	65	52.7	174	0
2013-2	2	76.5	76.5	65.8	123	30
2013-3	5	146.6	120	149	403	24
2013-4	4	151.5	167.5	92	237	34
Total	205	98.9	62	109.2	494	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	14	81.5	20	126.8	383	3
2010-4	35	47.5	23	57.3	210	3
2011-1	12	52.8	41.5	38.5	121	2
2011-2	27	36.3	31	46.7	245	1
2011-3	24	46.5	13.5	64.5	222	0
2011-4	39	18.2	7	29	132	0
2012-1	42	34	21.5	37.9	157	0
2012-2	30	27.4	18	38.5	169	0
2012-3	25	40.1	15	69.2	302	0
2012-4	41	53.4	19	85	402	0
2013-1	33	31.3	18	38.1	146	0
2013-2	18	57.7	40	62.1	216	0
2013-3	23	54	18	65.8	223	0
2013-4	8	43.1	29.5	45.6	123	0
Total	371	41.4	20	59.8	402	0

(b) DS - Churn bug

Figure A.14: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	14	2.8	2	2.9	12	1
2010-4	39	2.1	1	1.5	8	1
2011-1	22	1.8	1	1.3	5	1
2011-2	28	3.1	2	3.3	16	1
2011-3	38	2.4	2	1.9	10	1
2011-4	35	2.9	1	5	30	1
2012-1	39	3.7	2	4	23	1
2012-2	31	1.8	1	1.4	7	1
2012-3	28	2.5	2	1.8	8	1
2012-4	42	2.5	2	1.7	6	1
2013-1	30	1.8	1.5	1	4	1
2013-2	19	1.5	1	1	5	1
2013-3	26	1.3	1	0.5	2	1
2013-4	8	1.9	1	1.7	6	1
Total	399	2.7	1	2.6	30	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	30	9	39	76.9	23.1
2010-4	75	6	81	92.6	7.4
2011-1	27	13	40	67.5	32.5
2011-2	79	7	86	91.9	8.1
2011-3	77	13	90	85.6	14.4
2011-4	88	13	101	87.1	12.9
2012-1	132	11	143	92.3	7.7
2012-2	44	12	56	78.6	21.4
2012-3	54	15	69	78.3	21.7
2012-4	97	10	107	90.7	9.3
2013-1	48	5	53	90.6	9.4
2013-2	21	7	28	75	25
2013-3	32	1	33	97	3
2013-4	15	0	15	100	0
Mean	58.5	8.7	67.2	86	14

(b) DS - Bugs per quarter

Figure A.15: Caption of Descriptive Statistic for Bugs and Finished bugs per quarter

A.4 Team 4 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	9	2.7	2	2.5	7	1
2010-4	92	4.5	4	2.9	14	0
2011-1	90	10.4	10	3.1	18	4
2011-2	91	13.8	13	4.5	31	5
2011-3	92	14.1	13	4.6	28	6
2011-4	92	16.4	16	4.8	30	6
2012-1	91	16	15	3.9	25	9
2012-2	91	11.7	12	3.5	20	5
2012-3	92	14	14	4.2	26	7
2012-4	92	20.6	19.5	5.6	33	10
2013-1	90	19.5	19	7.2	37	5
2013-2	91	16	16	4.8	29	6
2013-3	92	15.5	15	5.9	29	6
2013-4	91	10.5	11	4	19	1
Total	1196	14	14	6.2	37	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	4	3	1	4	9	1
2010-4	39	2.9	1	2.5	11	1
2011-1	48	3.8	3	2.8	15	1
2011-2	48	6.3	5	5.6	31	1
2011-3	54	6.3	5	5	31	1
2011-4	52	7.5	5	5.9	23	1
2012-1	61	6.8	5	4.4	17	1
2012-2	57	3.9	3	2.8	15	1
2012-3	33	6	5	4.4	15	1
2012-4	52	5.8	5	4.7	21	1
2013-1	61	8.6	7	6.9	34	1
2013-2	59	8.3	7	4.7	19	1
2013-3	60	8	7	5.1	26	1
2013-4	46	5	4	3.8	15	1
Total	674	6.2	5	5	34	1

(b) DS - Throughput

Figure A.16: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	4	3	1	4	9	1
2010-4	39	2.9	1	2.5	11	1
2011-1	48	3.8	3	2.8	15	1
2011-2	48	6.3	5	5.6	31	1
2011-3	54	6.3	5	5	31	1
2011-4	52	7.5	5	5.9	23	1
2012-1	60	6.7	5	4.4	17	1
2012-2	57	3.9	3	2.8	15	1
2012-3	31	6.2	5	4.4	15	1
2012-4	48	5.8	5	4.8	21	1
2013-1	51	9.2	8	7.3	34	1
2013-2	50	8.5	7	4.8	19	1
2013-3	58	8.1	7.5	5.1	26	1
2013-4	44	5.1	5	3.8	15	1
Total	644	6.2	5	5.1	34	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2012-1	1	11	11	-	11	11
2012-3	2	3.5	3.5	3.5	6	1
2012-4	4	4.8	5	3	8	1
2013-1	10	5.5	6	2.5	9	1
2013-2	9	7.2	7	4.7	15	2
2013-3	2	4.5	4.5	2.1	6	3
2013-4	2	1	1	0	1	1
Total	30	5.6	5.5	3.7	15	1

(b) DS - Throughput bug

Figure A.17: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-2	34	13.2	10.5	10.7	50	1
2011-3	54	13.5	12.5	8.7	34	1
2011-4	49	17.9	14	14.4	61	1
2012-1	65	13.4	13	9.2	46	1
2012-2	56	9.4	8	7.2	33	1
2012-3	32	15.3	11	11.5	43	2
2012-4	63	10.2	8	10	66	1
2013-1	97	8.9	7	7.8	40	1
2013-2	80	9.4	8	8.6	48	1
2013-3	44	10.1	5.5	11.1	53	1
Total	574	11.6	9	10	66	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-2	34	9.8	8	10.1	43	0
2011-3	54	8.4	7	6.9	26	0
2011-4	49	13.2	10	12.8	53	0
2012-1	65	8.4	6	8.7	41	0
2012-2	56	4.2	2	6.1	27	0
2012-3	32	9.2	5	10.7	37	0
2012-4	63	5.1	2	9.4	59	0
2013-1	97	5.5	3	7.3	33	0
2013-2	80	6.4	3.5	8.3	47	0
2013-3	44	7.9	3	11	52	0
Total	574	7.4	5	9.2	59	0

(b) DS - Churn

Figure A.18: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-2	33	10.2	8	10.1	43	0
2011-3	53	8.6	7	6.9	26	0
2011-4	49	13.2	10	12.8	53	0
2012-1	63	8.7	7	8.8	41	0
2012-2	55	4.2	2	6.2	27	0
2012-3	29	10.1	6	10.9	37	0
2012-4	50	6.1	3	10.3	59	0
2013-1	65	7.8	7	7.9	33	0
2013-2	62	7.9	5.5	8.9	47	0
2013-3	37	9.3	6	11.4	52	0
Total	496	8.4	6	9.5	59	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-2	1	0	0	-	0	0
2011-3	1	0	0	-	0	0
2012-1	2	0	0	0	0	0
2012-2	1	0	0	-	0	0
2012-3	3	0.7	0	1.2	2	0
2012-4	13	1.5	0	2.5	7	0
2013-1	32	0.9	0	1.6	5	0
2013-2	18	1.2	0	2.3	9	0
2013-3	7	0.4	0	1.1	3	0
Total	78	1	0	1.8	9	0

(b) DS - Churn feature

Figure A.19: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-1	1	1	1	-	1	1
2011-2	2	1	1	0	1	1
2011-3	1	1	1	-	1	1
2012-1	2	1	1	0	1	1
2012-2	1	1	1	-	1	1
2012-3	4	1	1	0	1	1
2012-4	12	1.8	1.5	0.9	3	1
2013-1	32	1.6	1	0.9	4	1
2013-2	19	1.5	1	0.6	3	1
2013-3	12	1.2	1	0.4	2	1
2013-4	2	1	1	0	1	1
Total	88	1.4	1	0.8	4	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2011-1	1	0	1	100	0
2011-2	2	0	2	100	0
2011-3	1	0	1	100	0
2012-1	2	0	2	100	0
2012-2	1	0	1	100	0
2012-3	4	0	4	100	0
2012-4	21	1	22	95.5	4.5
2013-1	49	2	51	96.1	3.9
2013-2	27	1	28	96.4	3.6
2013-3	14	0	14	100	0
2013-4	2	0	2	100	0
Mean	11.3	.4	11.6	98.9	1.1

(b) DS - Bugs per quarter

Figure A.20: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.5 Team 5 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	24	8.4	8	3.3	15	2
2010-4	92	18.7	18	6.2	40	8
2011-1	90	7.8	8.5	6.2	20	0
2011-2	91	21.3	18	12.9	58	0
2011-3	92	26.8	27	9.3	45	8
2011-4	92	27.8	27	9.9	46	10
2012-1	91	44.5	47	9.5	65	24
2012-2	91	51.3	51	7.5	74	38
2012-3	92	19.6	19	11.2	50	4
2012-4	92	20	19	9	38	7
2013-1	90	124.8	126	94.7	270	9
2013-2	91	231.1	266	85.9	286	12
2013-3	92	21.2	19	7.4	43	11
2013-4	51	9.6	10	4.3	19	1
Total	1171	48.4	24	70.5	286	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	12	3.6	3	2.2	7	1
2010-4	49	4.2	3	3.5	15	1
2011-1	34	3.6	3	2.7	12	1
2011-2	51	7.2	7	5.4	19	1
2011-3	63	5.6	4	5	24	1
2011-4	58	5	5	3.7	17	1
2012-1	59	6.2	5	4.5	17	1
2012-2	59	5.3	4	3.8	15	1
2012-3	49	6.4	5	5	27	1
2012-4	50	4.7	3	4.1	17	1
2013-1	60	15.8	9.5	15.1	59	1
2013-2	58	6.7	7	4.6	22	1
2013-3	53	4.6	3	4.3	17	1
2013-4	19	2	1	1.7	7	1
Total	674	6.3	5	6.9	59	1

(b) DS - Throughput

Figure A.21: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	1	1	-	1	1
2010-4	8	2.9	1	3.2	10	1
2011-1	7	3.4	4	2.2	7	1
2011-2	19	9.1	7	5.1	17	1
2011-3	23	6.4	4	5.7	24	1
2011-4	11	3.7	3	2.7	8	1
2012-1	6	4.7	3	4.2	10	1
2012-2	8	4.4	3	4.6	15	1
2012-4	10	5.1	3.5	4.2	14	1
2013-1	4	16.8	16.5	11.9	30	4
2013-2	1	2	2	-	2	2
2013-3	4	2.2	2	1.3	4	1
2013-4	6	1.7	1	1	3	1
Total	108	5.7	4	5.5	30	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	11	3.8	3	2.1	7	1
2010-4	41	4.5	3	3.5	15	1
2011-1	27	3.6	3	2.8	12	1
2011-2	32	6.2	5.5	5.4	19	1
2011-3	40	5.2	4.5	4.6	22	1
2011-4	47	5.2	5	3.9	17	1
2012-1	53	6.4	6	4.6	17	1
2012-2	51	5.5	5	3.7	15	1
2012-3	49	6.4	5	5	27	1
2012-4	40	4.6	3	4.1	17	1
2013-1	56	15.7	9	15.4	59	1
2013-2	57	6.8	7	4.6	22	1
2013-3	49	4.8	3	4.5	17	1
2013-4	13	2.1	1	1.9	7	1
Total	566	6.4	5	7	59	1

(b) DS - Throughput bug

Figure A.22: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	9	26.9	22	30	91	1
2010-4	37	24.6	20	19.3	71	2
2011-1	21	10.1	8	7.8	29	1
2011-2	47	15.2	14	10.2	41	1
2011-3	84	16.1	10	17.9	105	1
2011-4	69	24.5	15	25.8	153	1
2012-1	68	30.7	22	27.9	148	1
2012-2	72	36.3	26	30.3	138	1
2012-3	53	18.6	16	15.5	80	1
2012-4	54	27.3	14.5	39.7	259	1
2013-1	71	31.4	24	29.9	161	1
2013-2	60	34.5	21.5	37.5	178	2
2013-3	44	27.6	19	27	118	1
2013-4	9	11.9	10	9	31	1
Total	698	25.7	17	27.5	259	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	9	63.2	70	51.5	168	6
2010-4	37	59.6	44	58.2	205	1
2011-1	21	41.1	17	60.2	201	1
2011-2	47	35	20	45.1	185	1
2011-3	84	24.1	8	37.2	151	0
2011-4	69	29	15	37.8	172	0
2012-1	68	27.4	14.5	37	170	0
2012-2	72	40	22	48.7	192	0
2012-3	53	20.8	17	24.8	110	0
2012-4	54	24.4	6.5	40.6	244	0
2013-1	71	41	27	45	206	0
2013-2	60	37.8	24	39.8	161	0
2013-3	44	30.6	13.5	41.6	164	0
2013-4	9	32	27	36.6	115	0
Total	698	33.4	17	43	244	0

(b) DS - Churn

Figure A.23: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	9	93.6	78	79.8	205	8
2011-1	4	88.5	74	88.3	201	5
2011-2	8	69	45.5	57.4	182	23
2011-3	30	29.4	8	47.5	151	0
2011-4	18	46.3	13	58.2	172	0
2012-1	10	26.3	0	52.9	170	0
2012-2	13	75.7	83	65.9	192	0
2012-3	2	27.5	27.5	9.2	34	21
2012-4	9	38.1	41	38	100	0
2013-1	8	94.2	71	75.1	206	7
2013-2	8	31.5	4	40.5	91	0
2013-3	4	73.5	65	86	164	0
Total	123	52.1	27	61.2	206	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	9	63.2	70	51.5	168	6
2010-4	28	48.7	38	46	157	1
2011-1	17	30	15	48.7	187	1
2011-2	39	28.1	17	39.6	185	1
2011-3	54	21.1	9	30.2	149	0
2011-4	51	22.9	15	25.4	107	0
2012-1	58	27.6	15	34.1	152	0
2012-2	59	32.2	21	40.7	187	0
2012-3	51	20.5	17	25.2	110	0
2012-4	45	21.6	6	41	244	0
2013-1	63	34.3	23	35.1	153	0
2013-2	52	38.8	25	40	161	0
2013-3	40	26.4	13.5	33.7	113	0
2013-4	9	32	27	36.6	115	0
Total	575	29.4	17	36.8	244	0

(b) DS - Churn bug

Figure A.24: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	19	1.9	1	1.8	7	1
2010-4	46	2.6	2	1.7	8	1
2011-1	36	1.9	1.5	1.3	7	1
2011-2	45	5.5	4	4.7	19	1
2011-3	51	3	2	2.6	15	1
2011-4	53	3.5	3	2.4	11	1
2012-1	52	3.7	3	2.6	10	1
2012-2	56	2.7	2	1.9	7	1
2012-3	49	3.2	2	2.7	13	1
2012-4	35	3	2	2.7	15	1
2013-1	56	9.6	7	8.3	38	1
2013-2	49	4.2	4	2.7	12	1
2013-3	41	3.1	2	2.4	10	1
2013-4	11	1.4	1	0.9	4	1
Total	604	3.8	3	4.1	38	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	24	13	37	64.9	35.1
2010-4	108	13	121	89.3	10.7
2011-1	57	12	69	82.6	17.4
2011-2	202	47	249	81.1	18.9
2011-3	119	33	152	78.3	21.7
2011-4	147	37	184	79.9	20.1
2012-1	149	45	194	76.8	23.2
2012-2	116	35	151	76.8	23.2
2012-3	133	25	158	84.2	15.8
2012-4	99	5	104	95.2	4.8
2013-1	502	37	539	93.1	6.9
2013-2	183	21	204	89.7	10.3
2013-3	123	5	128	96.1	3.9
2013-4	15	0	15	100	0
Mean	109.9	18.6	128.5	74.3	25.7

(b) DS - Bugs per quarter

Figure A.25: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.6 Team 6 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	24	9.5	9	3.6	16	4
2010-4	92	10.3	10	2.6	16	6
2011-1	90	9.8	10	2	17	7
2011-2	91	10.4	11	2.4	16	4
2011-3	92	19.5	20.5	7.3	34	6
2011-4	92	22.9	22	9.3	44	9
2012-1	91	15.6	16	3.7	27	6
2012-2	91	17.5	18	6.1	42	8
2012-3	92	15.2	15	4.5	26	6
2012-4	92	26.3	25.5	10.6	50	11
2013-1	90	32.6	31	8.4	51	15
2013-2	91	43.7	43	5	60	36
2013-3	92	30.6	29.5	8	61	17
2013-4	85	37.4	39	20.8	125	10
Total	1205	22.1	18	13.4	125	4

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	17	4.5	3	3.1	10	1
2010-4	51	3.3	3	2.6	10	1
2011-1	45	2.3	1	1.9	8	1
2011-2	37	2.8	3	1.9	8	1
2011-3	49	2.7	1	2.1	7	1
2011-4	40	3.2	3	2.3	9	1
2012-1	54	3.3	3	2.4	9	1
2012-2	51	5.2	3	5.8	37	1
2012-3	45	4	3	3.6	21	1
2012-4	63	6	5	4.5	23	1
2013-1	59	6.3	5	4.2	16	1
2013-2	61	4.4	3	3.7	15	1
2013-3	61	4.7	4	3.6	15	1
2013-4	58	9.1	5	23.8	181	1
Total	691	4.6	3	181	1	7.8

(b) DS - Throughput

Figure A.26: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	14	4.3	3	3.4	10	1
2010-4	47	3.5	3	2.7	10	1
2011-1	42	2.3	1	1.9	8	1
2011-2	33	2.7	3	1.9	8	1
2011-3	45	2.7	1	2.2	7	1
2011-4	38	3.2	3	2.4	9	1
2012-1	51	3.3	3	2.4	9	1
2012-2	51	5.2	3	5.8	37	1
2012-3	43	4	3	3.7	21	1
2012-4	55	6.4	5	4.5	23	1
2013-1	49	6.7	6	4.3	16	1
2013-2	47	5	3	3.8	15	1
2013-3	44	4.8	4	3.8	15	1
2013-4	50	10.1	5	25.5	181	1
Total	609	4.8	3	8.3	181	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	3	5.3	6	2.1	7	3
2010-4	4	1	1	0	1	1
2011-1	3	2.3	1	2.3	5	1
2011-3	4	3.5	3	2.5	7	1
2011-4	4	2.5	2	1.9	5	1
2012-1	2	3.5	3.5	0.7	4	3
2012-2	3	3.7	2	2.9	7	2
2012-3	2	3	3	1.4	4	2
2012-4	8	3.5	2	3.3	11	1
2013-1	10	4.4	4.5	2.7	9	1
2013-2	14	2.4	1.5	2.9	12	1
2013-3	17	4.2	4	3.2	13	1
2013-4	8	2.5	3	1.1	4	1
Total	82	3.3	2.5	2.6	13	1

(b) DS - Throughput bug

Figure A.27: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	19	4.7	2	7.9	34	1
2010-4	34	7.7	3.5	10.7	48	1
2011-1	35	10.9	8	9.1	33	1
2011-2	21	9.9	6	10.2	44	1
2011-3	20	15.9	15.5	10.7	46	3
2011-4	33	17.4	15	11.9	52	1
2012-1	59	16.1	14	13.2	70	1
2012-2	53	22.6	18	17.8	77	1
2012-3	55	15.4	13	12	53	1
2012-4	88	17.3	11	19.6	120	1
2013-1	109	12.9	8	11.8	54	1
2013-2	67	12	8	11.6	73	1
2013-3	84	13.5	10.5	13.4	94	1
2013-4	79	13.9	8	18.1	93	1
Total	756	14.3	10	14.5	120	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	19	139.4	31	224.3	812	2
2010-4	34	185.4	67.5	255.5	1030	1
2011-1	35	110.5	31	214.6	901	1
2011-2	21	266.6	140	321.4	1187	1
2011-3	20	175.4	146	159.7	496	8
2011-4	33	68.5	7	149.4	596	0
2012-1	59	72.2	9	213.6	1191	0
2012-2	53	59.9	16	149.5	769	0
2012-3	55	60	8	196.9	1207	0
2012-4	88	75.7	16	160.1	658	0
2013-1	109	91.1	19	202.4	937	0
2013-2	67	144.3	40	213.5	766	0
2013-3	84	69.4	19	128.6	739	0
2013-4	79	92.2	19	198.5	1127	0
Total	756	98.3	19	197.3	1207	0

(b) DS - Churn

Figure A.28: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	13	191.1	37	255.8	812	2
2010-4	33	189	67	258.6	1030	1
2011-1	32	119.4	32.5	222.5	901	1
2011-2	21	266.6	140	321.4	1187	1
2011-3	20	175.4	146	159.7	496	8
2011-4	31	72.9	7	153.2	596	0
2012-1	52	81.2	14.5	226.3	1191	0
2012-2	53	59.9	16	149.5	769	0
2012-3	46	70.7	12.5	214	1207	0
2012-4	64	85.2	16	167.8	655	0
2013-1	62	87.8	20	199.3	937	0
2013-2	44	154.7	28	232.9	766	0
2013-3	54	72.4	24.5	128.2	739	0
2013-4	51	94.8	30	187.7	994	0
Total	576	105.9	21	204.9	1207	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	6	27.5	9	45.3	119	2
2010-4	1	68	68	-	68	68
2011-1	3	15.7	11	13.6	31	5
2011-4	2	0.5	0.5	0.7	1	0
2012-1	7	5.6	0	12.7	34	0
2012-3	9	4.8	1	8	24	0
2012-4	24	50.4	11.5	137.7	658	0
2013-1	47	95.4	12	208.4	934	0
2013-2	23	124.3	54	173.4	694	0
2013-3	30	64	13	131.2	574	0
2013-4	28	87.5	10.5	220.3	1127	0
Total	180	73.8	12	169.3	1127	0

(b) DS - Churn bug

Figure A.29: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	10	1.5	1.5	0.5	2	1
2010-4	7	1	1	0	1	1
2011-1	5	1.4	1	0.9	3	1
2011-2	8	1.1	1	0.4	2	1
2011-3	4	1.2	1	0.5	2	1
2011-4	2	2	2	0	2	2
2012-1	7	1.1	1	0.4	2	1
2012-3	11	1.3	1	0.5	2	1
2012-4	24	1.8	1.5	1	4	1
2013-1	39	1.9	2	1.2	7	1
2013-2	33	1.5	1	0.7	3	1
2013-3	34	1.6	1	0.8	4	1
2013-4	27	1.8	2	0.9	4	1
Total	211	1.6	1	0.9	7	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	14	1	15	93.3	6.7
2010-4	6	1	7	85.7	14.3
2011-1	6	1	7	85.7	14.3
2011-2	7	2	9	77.8	22.2
2011-3	3	2	5	60	40
2011-4	3	1	4	75	25
2012-1	8	0	8	100	0
2012-3	12	2	14	85.7	14.3
2012-4	41	2	43	95.3	4.7
2013-1	66	9	75	88	12
2013-2	43	7	50	86	14
2013-3	52	1	53	98.1	1.9
2013-4	49	0	49	100	0
Mean	23.9	2.2	26.1	87.0	13.0

(b) DS - Bugs finished quarter

Figure A.30: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.7 Team 7 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	12	17.7	8.5	19.5	54	1
2010-4	64	13	8	11.9	50	1
2011-1	57	12.8	8	15.1	89	1
2011-2	37	14.3	9	13	51	1
2011-3	36	17.8	11.5	18.1	79	1
2011-4	51	15	9	14.9	63	1
2012-1	35	14.9	11	18.3	86	1
2012-2	23	18.8	9	27.8	124	1
2012-3	42	15.1	7	18.6	81	1
2012-4	2	1.5	1.5	0.7	2	1
Total	359	14.8	8	16.6	124	1

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	11	3.9	2	4	14	1
2010-4	53	3.7	3	2.5	13	1
2011-1	54	3.2	3	2.3	13	1
2011-2	33	2.3	2	1.2	5	1
2011-3	36	2	2	1.1	4	1
2011-4	44	2.2	2	1.5	6	1
2012-1	37	2	1	1.5	7	1
2012-2	25	2.2	2	1.4	6	1
2012-3	32	3.4	3	2.5	13	1
2012-4	3	1	1	0	1	1
Total	328	2.7	2	2.1	14	1

(b) DS - Throughput

Figure A.31: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	8	3.2	2.5	2.4	7	1
2010-4	37	4	3	2.7	13	1
2011-1	22	3.4	3	2.4	10	1
2011-2	10	2.7	2.5	1.7	5	1
2011-3	17	2.1	2	1.2	4	1
2011-4	26	2	1.5	1.3	5	1
2012-1	12	2	2	1.3	5	1
2012-2	9	2.2	2	0.8	3	1
2012-3	12	2.7	3	1.5	5	1
2012-4	3	1	1	0	1	1
Total	156	2.8	2	2.1	13	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	3	5.7	2	7.2	14	1
2010-4	16	2.9	2.5	2	9	1
2011-1	32	3	3	2.3	13	1
2011-2	23	2.1	2	0.9	4	1
2011-3	19	1.9	2	1	4	1
2011-4	18	2.4	2	1.7	6	1
2012-1	25	2.1	1	1.6	7	1
2012-2	16	2.2	2	1.6	6	1
2012-3	20	3.8	3	2.9	13	1
Total	172	2.6	2	2.1	14	1

(b) DS - Throughput bug

Figure A.32: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	12	17.7	8.5	19.5	54	1
2010-4	64	13	8	11.9	50	1
2011-1	57	12.8	8	15.1	89	1
2011-2	37	14.3	9	13	51	1
2011-3	36	17.8	11.5	18.1	79	1
2011-4	51	15	9	14.9	63	1
2012-1	35	14.9	11	18.3	86	1
2012-2	23	18.8	9	27.8	124	1
2012-3	42	15.1	7	18.6	81	1
2012-4	2	1.5	1.5	0.7	2	1
Total	359	14.8	8	16.6	124	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	12	154.2	49.5	189	647	4
2010-4	64	94.1	30	137.1	662	1
2011-1	57	74.3	26	108.4	479	1
2011-2	37	106.7	29	183.6	726	0
2011-3	36	85.1	21.5	143.6	577	0
2011-4	51	68.1	23	112.8	458	0
2012-1	35	43.4	15	70.1	367	0
2012-2	23	55.7	33	73.6	302	0
2012-3	42	53.8	28	82.5	424	3
2012-4	2	44	44	1.4	45	43
Total	359	77.3	26	124.8	726	0

(b) DS - Churn

Figure A.33: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	7	248.4	248	201	647	41
2010-4	38	117.8	77.5	130.8	585	6
2011-1	18	131.7	67.5	153.5	479	4
2011-2	10	173.3	84.5	231.5	726	0
2011-3	11	204.3	115	208.8	577	0
2011-4	25	114	55	141.8	458	0
2012-1	9	37.6	31	31.6	82	1
2012-2	7	40.6	34	47	140	0
2012-3	14	68.6	43	87.7	318	3
2012-4	2	44	44	1.4	45	43
Total	141	121.2	57	150.7	726	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	26	59.5	11.5	141.2	662	1
2011-1	39	47.9	20	67	276	1
2011-2	27	82	21	160.6	719	0
2011-3	25	32.6	17	50.1	226	1
2011-4	26	24	14.5	44.9	234	0
2012-1	26	45.4	11	79.7	367	0
2012-2	16	62.3	28	83.1	302	0
2012-3	28	46.4	22	80.4	424	4
Total	218	48.9	18	94.8	719	0

(b) DS - Churn bug

Figure A.34: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	17	1.8	2	0.8	3	1
2010-4	28	1.7	1.5	0.9	4	1
2011-1	39	3.1	2	2.5	12	1
2011-2	26	2	2	1.3	5	1
2011-3	26	1.6	1	1.1	5	1
2011-4	24	2.2	2	1.5	7	1
2012-1	29	1.7	1	1.4	8	1
2012-2	18	2.7	2	2.6	11	1
2012-3	29	2.3	2	1.3	5	1
Total	240	2.1	2	1.7	12	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	20	10	30	66.7	33.3
2010-4	47	1	48	97.9	2.1
2011-1	119	2	121	98.3	1.7
2011-2	45	8	53	84.9	15.1
2011-3	35	6	41	85.4	14.6
2011-4	45	7	52	86.5	13.5
2012-1	46	2	48	95.8	4.2
2012-2	36	12	48	75	25
2012-3	67	0	67	100	0
Mean	38.3	4.3	42.7	65.9	34.1

(b) DS - Bugs per quarter

Figure A.35: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.8 Team 8 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	19	0.4	0	0.5	1	0
2011-1	90	3.7	3	2.4	9	0
2011-2	91	5.9	6	1.9	11	1
2011-3	92	11.2	12	2.6	16	7
2011-4	92	7.9	7	3.3	14	3
2012-1	91	9.7	9	3.7	16	3
2012-2	91	4.3	2	4.6	12	1
2012-3	92	9.1	9	7.3	32	1
2012-4	92	5.6	7	4.6	18	1
2013-1	90	8.4	4	9.1	30	1
2013-2	91	19.7	18	11.2	55	2
2013-3	92	8.2	4	8.5	29	0
2013-4	77	4	5	2.8	11	0
Total	1100	8.1	6	7.3	55	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	2	1	1	0	1	1
2011-1	12	1.5	1	0.7	3	1
2011-2	21	1.2	1	0.5	3	1
2011-3	15	1.7	1	1.1	4	1
2011-4	19	1.3	1	0.6	3	1
2012-1	16	1.4	1	1	5	1
2012-2	3	1	1	0	1	1
2012-3	23	2.5	2	2.5	12	1
2012-4	10	1.7	2	0.7	3	1
2013-1	25	3.8	3	3.4	14	1
2013-2	20	3.5	2.5	2.9	9	1
2013-3	21	3.5	2	3.5	13	1
2013-4	4	3.5	3	2.6	7	1
Total	191	2.3	1	2.4	14	1

(b) DS - Throughput

Figure A.36: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-1	2	1	1	0	1	1
2011-2	5	1.4	1	0.9	3	1
2011-3	1	1	1	-	1	1
2011-4	6	1.5	1	0.8	3	1
2012-1	6	1.8	1	1.6	5	1
2012-3	20	2.7	2	2.7	12	1
2012-4	6	1.8	2	0.8	3	1
2013-1	18	3.6	3	2.6	9	1
2013-2	12	5	5	2.8	9	1
2013-3	13	4.8	4	3.9	13	1
2013-4	3	4.3	4	2.5	7	2
Total	92	3.2	2	2.8	13	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	2	1	1	0	1	1
2011-1	10	1.6	1.5	0.7	3	1
2011-2	16	1.2	1	0.4	2	1
2011-3	14	1.8	1	1.1	4	1
2011-4	13	1.1	1	0.4	2	1
2012-1	10	1.1	1	0.3	2	1
2012-2	3	1	1	0	1	1
2012-3	3	1	1	0	1	1
2012-4	4	1.5	1.5	0.6	2	1
2013-1	7	4.1	1	5	14	1
2013-2	8	1.4	1	0.7	3	1
2013-3	8	1.2	1	0.5	2	1
2013-4	1	1	1	-	1	1
Total	99	1.5	1	1.6	14	1

(b) DS - Throughput bug

Figure A.37: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	1	1	1	-	1	1
2011-1	3	3	3	2	5	1
2011-2	8	19.6	9.5	25.8	71	1
2011-3	13	20.5	15	18.9	69	1
2011-4	10	21.4	18.5	16	43	2
2012-1	9	27	8	61.3	190	1
2012-2	1	1	1	-	1	1
2012-3	20	28.6	28.5	26.3	89	1
2012-4	10	17	15	13.3	45	3
2013-1	22	14.6	9.5	17.2	75	1
2013-2	16	23.1	5	41.4	150	1
2013-3	20	24.9	13.5	39.2	161	1
2013-4	4	70.2	75.5	56.3	129	1
Total	137	22.6	11	32.2	190	1

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	1	3	3	-	3	3
2011-1	3	7.7	3	9	18	2
2011-2	8	9.5	1.5	12.3	26	0
2011-3	13	12.8	3	16.4	51	0
2011-4	10	6	1.5	10.2	32	0
2012-1	9	17.7	0	50.4	152	0
2012-2	1	8	8	-	8	8
2012-3	20	12.4	2	20.5	84	0
2012-4	10	3.7	0.5	5.2	13	0
2013-1	22	9.6	5	15.5	73	0
2013-2	16	19.6	1.5	40.6	149	0
2013-3	20	17.9	4.5	35.9	145	0
2013-4	4	45.8	29	57.2	125	0
Total	137	13.4	3	27.9	152	0

(b) DS - Churn

Figure A.38: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2011-1	1	2	2	-	2	2
2011-2	2	0	0	0	0	0
2011-3	1	0	0	-	0	0
2011-4	3	10.7	0	18.5	32	0
2012-1	3	50.7	0	87.8	152	0
2012-3	19	13.1	3	20.9	84	0
2012-4	7	3.6	1	4.9	13	0
2013-1	15	5.5	4	5.7	17	0
2013-2	10	31.2	6.5	48.4	149	0
2013-3	15	23.3	7	40.2	145	0
2013-4	3	58.3	50	62.9	125	0
Total	79	17.4	4	34.4	152	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	1	3	3	-	3	3
2011-1	2	10.5	10.5	10.6	18	3
2011-2	6	12.7	12.5	12.8	26	0
2011-3	12	13.8	6.5	16.7	51	0
2011-4	7	4	3	5.3	14	0
2012-1	6	1.2	0	2.9	7	0
2012-2	1	8	8	-	8	8
2012-3	1	0	0	-	0	0
2012-4	3	4	0	6.9	12	0
2013-1	7	18.3	12	25.1	73	0
2013-2	6	0.2	0	0.4	1	0
2013-3	5	1.6	0	3.6	8	0
2013-4	1	8	8	-	8	8
Total	58	8	1.5	13.6	73	0

(b) DS - Churn bug

Figure A.39: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	1	1	-	1	1
2010-4	7	1.1	1	0.4	2	1
2011-1	9	1.2	1	0.4	2	1
2011-2	16	1.6	1	1.5	7	1
2011-3	15	1.3	1	0.6	3	1
2011-4	13	1.2	1	0.6	3	1
2012-1	9	1.2	1	0.4	2	1
2012-2	2	1	1	0	1	1
2012-3	4	1.2	1	0.5	2	1
2012-4	2	1	1	0	1	1
2013-1	10	3.3	2.5	2.8	10	1
2013-2	4	1	1	0	1	1
2013-3	4	1.5	1.5	0.6	2	1
2013-4	1	1	1	-	1	1
Total	100	1.5	1	1.3	10	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	0	1	1	0	100
2010-4	2	6	8	25	75
2011-1	7	4	11	63.6	36.4
2011-2	16	10	26	61.5	38.5
2011-3	16	4	20	80	20
2011-4	15	1	16	93.8	6.3
2012-1	9	2	11	81.8	18.2
2012-2	2	0	2	100	0
2012-3	2	3	5	40	60
2012-4	1	1	2	50	50
2013-1	27	6	33	81.8	18.2
2013-2	3	1	4	75	25
2013-3	6	0	6	100	0
2013-4	1	0	1	100	0
Mean	6.7	2.6	9.3	59.5	40.5

(b) DS - Bugs per quarter

Figure A.40: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.9 Team 9 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	52	4.5	4.5	3.4	10	0
2011-1	90	11.8	12.5	3.8	19	5
2011-2	91	11.2	8	6.9	34	3
2011-3	92	12.8	12	4	24	6
2011-4	92	16	17	5.1	25	5
2012-1	91	16.2	15	4.7	30	8
2012-2	91	35.4	33	16.4	67	8
2012-3	92	32.6	33.5	7.9	51	15
2012-4	92	21.8	23.5	10.4	39	3
2013-1	90	21.4	20.5	8	38	7
2013-2	91	26.6	21	12.7	47	11
2013-3	92	15.9	14	6.3	35	6
2013-4	84	17.1	17	4.5	29	7
Total	1140	19.2	16	11.6	67	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	15.0	1.7	1.0	1.0	4.0	1.0
2011-1	30.0	1.8	1.0	1.0	4.0	1.0
2011-2	31.0	2.2	2.0	1.8	9.0	1.0
2011-3	27.0	1.6	1.0	0.9	5.0	1.0
2011-4	33.0	2.0	2.0	1.3	6.0	1.0
2012-1	41.0	2.4	2.0	1.4	5.0	1.0
2012-2	48.0	3.4	3.0	2.0	9.0	1.0
2012-3	53.0	3.3	3.0	2.2	9.0	1.0
2012-4	43.0	2.9	2.0	2.0	10.0	1.0
2013-1	51.0	2.9	2.0	1.8	9.0	1.0
2013-2	46.0	3.5	3.0	2.5	12.0	1.0
2013-3	50.0	2.6	2.0	1.8	9.0	1.0
2013-4	53.0	2.4	2.0	1.5	7.0	1.0
Total	521.0	2.6	2.0	1.9	12.0	1.0

(b) DS - Throughput

Figure A.41: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	12	1.7	1	1	4	1
2011-1	13	1.6	1	0.8	3	1
2011-2	5	1.4	1	0.5	2	1
2011-3	9	1.2	1	0.4	2	1
2011-4	17	1.7	1	1.4	6	1
2012-1	11	1.6	1	0.8	3	1
2012-2	23	2.9	3	1.6	6	1
2012-3	12	3.8	3.5	2.8	9	1
2012-4	20	2.7	2	1.7	6	1
2013-1	15	2.1	2	1	4	1
2013-2	24	3.6	3	2.8	12	1
2013-3	22	2.5	2	1.7	7	1
2013-4	31	2.4	2	1.5	7	1
Total	214	2.4	2	1.8	12	1

(a) DS - Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	3	2	2	1	3	1
2011-1	17	1.9	1	1.1	4	1
2011-2	26	2.4	2	1.9	9	1
2011-3	18	1.7	1.5	1	5	1
2011-4	16	2.3	2	1.1	5	1
2012-1	30	2.7	2.5	1.5	5	1
2012-2	25	3.8	3	2.3	9	1
2012-3	41	3.1	3	2	9	1
2012-4	23	3	3	2.3	10	1
2013-1	36	3.2	3	1.9	9	1
2013-2	22	3.4	3	2.2	8	1
2013-3	28	2.6	2	1.9	9	1
2013-4	22	2.5	2	1.6	7	1
Total	307	2.8	2	1.9	10	1

(b) DS - Throughput bug

Figure A.42: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	14	11.4	8	10.2	33	2
2011-1	23	20.3	14	15.2	56	3
2011-2	24	19.7	17	12.8	54	4
2011-3	18	10.8	9	7.6	32	2
2011-4	27	10.5	6	7.9	30	2
2012-1	44	12.1	10.5	9.3	46	2
2012-2	58	12.8	10.5	10.2	59	2
2012-3	62	17.6	14	14.2	62	2
2012-4	37	20.5	13	26.1	140	2
2013-1	63	16.8	16	11.6	48	2
2013-2	56	20.8	15	22	128	2
2013-3	60	15	12.5	12	62	2
2013-4	52	14.4	10.5	11.4	48	2
Total	538	15.9	12	14.7	140	2

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	14	59.4	39.5	62	212	4
2011-1	23	69	56	63.8	204	1
2011-2	24	48.3	24.5	50.4	171	1
2011-3	18	68.6	17	104.3	309	1
2011-4	27	119.9	70	126.5	401	2
2012-1	44	79.9	35	102.8	426	1
2012-2	58	58.3	25	85.2	423	1
2012-3	62	61.9	31	93.6	472	1
2012-4	37	53.6	21	86.5	367	0
2013-1	63	43.1	21	62	218	0
2013-2	56	88.5	35	115.5	445	0
2013-3	60	90.2	31.5	113.3	432	0
2013-4	52	95.8	40	114.9	382	0
Total	538	72.2	30	97.5	472	0

(b) DS - Churn

Figure A.43: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	10	60.9	35.5	68.4	212	5
2011-1	7	74.6	61	44	154	19
2011-2	3	127.7	162	67.4	171	50
2011-3	2	204.5	204.5	130.8	297	112
2011-4	12	210.1	182.5	128.7	401	70
2012-1	14	135.5	72.5	133	426	12
2012-2	22	115.6	76.5	113.4	423	13
2012-3	17	63.8	41	82.1	310	3
2012-4	15	98.1	53	119.6	367	0
2013-1	26	79.3	44	82.8	218	0
2013-2	25	133.7	91	149.5	445	0
2013-3	24	96.7	29	114.1	354	0
2013-4	24	172.4	178.5	130.9	382	0
Total	201	115.9	72	118.8	445	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	4	55.5	49	50.7	120	4
2011-1	16	66.6	35.5	71.9	204	1
2011-2	21	37	20	37.3	136	1
2011-3	16	51.6	10.5	91.7	309	1
2011-4	15	47.7	28	64.3	242	2
2012-1	30	54	24	74.6	312	1
2012-2	36	23.3	13.5	27.8	134	1
2012-3	45	61.2	30	98.4	472	1
2012-4	22	23.3	12	30	110	0
2013-1	37	17.7	16	15.6	52	0
2013-2	31	52	31	59.3	244	0
2013-3	36	85.9	35	114.2	432	0
2013-4	28	30.2	21.5	25.9	102	0
Total	337	46.1	23	70.4	472	0

(b) DS - Churn bug

Figure A.44: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	2	1	1	0	1	1
2010-4	13	1.9	2	1.1	4	1
2011-1	18	1.8	1	0.9	3	1
2011-2	20	2.2	1	2.6	12	1
2011-3	14	1.6	1.5	0.6	3	1
2011-4	23	1.8	1	1.2	5	1
2012-1	33	2.1	2	1.5	7	1
2012-2	43	2.2	2	1.5	7	1
2012-3	40	2.7	2	1.9	9	1
2012-4	33	2.2	2	1.7	8	1
2013-1	34	2.2	1	1.7	8	1
2013-2	39	2.1	2	1.6	6	1
2013-3	38	2.1	1.5	1.5	7	1
2013-4	42	1.5	1	0.8	4	1
Total	403	2.1	1	1.5	12	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	0	2	2	0	100
2010-4	8	17	25	32	68
2011-1	19	13	32	59.4	40.6
2011-2	42	3	45	93.3	6.7
2011-3	11	11	22	50	50
2011-4	22	19	41	53.7	46.3
2012-1	52	18	70	74.3	25.7
2012-2	73	22	95	76.8	23.2
2012-3	100	7	107	93.5	6.5
2012-4	58	16	74	78.4	21.6
2013-1	73	3	76	96.1	3.9
2013-2	80	4	84	95.2	4.8
2013-3	79	1	80	98.8	1.3
2013-4	63	0	63	100	0
Mean	37.9	8.3	46.17	58.4	41.6

(b) DS - Bugs per quarter

Figure A.45: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

A.10 Team 10 - Descriptive Statistics

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	24	2.7	3	0.9	5	1
2010-4	92	13.1	13.5	6	26	2
2011-1	90	8.1	8	6.2	22	0
2011-2	91	6	4	4.8	17	0
2011-3	92	0.9	1	0.8	3	0
2011-4	92	16.7	17.5	13.7	40	1
2012-1	91	24.6	24	3.8	36	17
2012-2	91	34.5	35	8.4	51	18
2012-3	92	12.7	10	8.7	44	4
2012-4	92	25.8	19.5	13.6	59	10
2013-1	90	16.3	6	14.5	49	5
2013-2	91	8.9	8	4.5	21	5
2013-3	92	12.5	12	5.8	29	3
2013-4	57	15.4	15	4.8	26	7
Total	1177	14.8	12	12.2	59	0

(a) DS - WIP

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	5	1.6	1	0.9	3	1
2010-4	44	2.2	2	1.6	7	1
2011-1	29	2.5	2	1.8	7	1
2011-2	21	1.9	1	1.4	6	1
2011-3	8	1	1	0	1	1
2011-4	34	2.6	2	1.7	7	1
2012-1	32	1.8	1	1.2	6	1
2012-2	52	2.7	2	1.6	7	1
2012-3	38	1.7	1	1.1	6	1
2012-4	47	2.7	2	3	16	1
2013-1	25	2.9	2	1.8	8	1
2013-2	10	1.9	1.5	1.1	4	1
2013-3	36	1.8	1.5	1.2	5	1
2013-4	23	1.8	1	1.2	5	1
Total	404	2.2	2	1.7	16	1

(b) DS - Throughput

Figure A.46: Caption of Descriptive Statistic for WIP and Throughput

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	5	2	1	2.2	6	1
2011-1	2	2	2	0	2	2
2011-2	1	6	6	-	6	6
2011-3	3	1	1	0	1	1
2011-4	7	3	2	2.2	7	1
2012-1	7	1.1	1	0.4	2	1
2012-2	15	2.8	2	1.8	7	1
2012-3	6	1	1	0	1	1
2012-4	11	3	1	4.4	16	1
2013-1	2	2.5	2.5	0.7	3	2
2013-2	1	1	1	-	1	1
2013-3	4	1.2	1	0.5	2	1
2013-4	5	2.4	2	1.7	5	1
Total	69	2.3	1	2.3	16	1

(a) DS Throughput feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	5	1.6	1	0.9	3	1
2010-4	39	2.2	2	1.5	7	1
2011-1	27	2.6	2	1.9	7	1
2011-2	20	1.7	1	1.1	5	1
2011-3	5	1	1	0	1	1
2011-4	27	2.5	2	1.5	6	1
2012-1	25	1.9	1	1.3	6	1
2012-2	37	2.7	3	1.5	6	1
2012-3	32	1.8	1	1.2	6	1
2012-4	36	2.6	2	2.6	13	1
2013-1	23	2.9	2	1.9	8	1
2013-2	9	2	2	1.1	4	1
2013-3	32	1.9	2	1.2	5	1
2013-4	18	1.7	1	1	4	1
Total	335	2.2	2	1.6	13	1

(b) DS Throughput bug

Figure A.47: Caption of Descriptive Statistic for Throughput feature and Throughput bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	18	18	-	18	18
2010-4	30	18	13.5	12.9	45	2
2011-1	26	11.6	8	9.6	41	3
2011-2	18	16.2	6	18.2	60	2
2011-3	7	9	6	7.2	21	3
2011-4	37	21.1	13	17.9	56	2
2012-1	20	27.8	27	21.6	78	2
2012-2	69	27	22	21.3	106	2
2012-3	27	22.7	17	23.2	97	2
2012-4	46	29.8	17.5	48.3	313	3
2013-1	26	19.5	10.5	19.1	67	2
2013-2	13	28.1	31	14.1	52	11
2013-3	24	24.4	19.5	18	62	2
2013-4	17	19.7	15	21.6	96	2
Total	361	22.7	15	24.8	313	2

(a) DS - Lead time

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	5	5	-	5	5
2010-4	30	58	15	106.2	469	1
2011-1	26	41	17.5	56.8	266	0
2011-2	18	14	3.5	19.1	59	0
2011-3	7	70	1	131.8	358	0
2011-4	37	24.2	7	53.9	309	0
2012-1	20	43.6	13.5	103	441	0
2012-2	69	39.3	11	85	438	0
2012-3	27	38.3	12	60.7	267	0
2012-4	46	52.2	17	88.1	373	0
2013-1	26	71.7	26	110.7	406	0
2013-2	13	37.1	24	45.9	123	0
2013-3	24	64.2	16.5	113.9	469	0
2013-4	17	61.6	10	112.8	321	0
Total	361	45.4	14	86.2	469	0

(b) DS - Churn

Figure A.48: Caption of Descriptive Statistic for Lead time and Churn

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-4	1	219	219	-	219	219
2011-1	4	77	21	126.5	266	0
2011-2	3	29.3	25	27.8	59	4
2011-3	2	47.5	47.5	67.2	95	0
2011-4	10	35.3	0	97	309	0
2012-1	1	441	441	-	441	441
2012-2	25	66	5	131.9	438	0
2012-3	4	79	24.5	127.4	267	0
2012-4	8	95.5	16	133.9	310	0
2013-1	3	45.7	13	68.1	124	0
2013-2	2	48	48	67.9	96	0
2013-3	3	168.3	219	149.6	286	0
2013-4	3	78.3	0	135.7	235	0
Total	69	75.5	5	123.6	441	0

(a) DS - Churn feature

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	1	5	5	-	5	5
2010-4	29	52.5	15	103.6	469	1
2011-1	22	34.4	17.5	35.6	125	0
2011-2	15	10.9	1	16.5	54	0
2011-3	5	79	1	156.7	358	0
2011-4	27	20.1	14	26.4	128	0
2012-1	19	22.7	13	44.3	195	0
2012-2	44	24.1	14.5	32.5	141	0
2012-3	23	31.2	12	42.3	151	0
2012-4	38	43.1	17	74.6	373	0
2013-1	23	75.1	27	115.8	406	0
2013-2	11	35.1	24	45.2	123	0
2013-3	21	49.3	14	104	469	0
2013-4	14	58	11	113	321	0
Total	292	38.3	15	73.2	469	0

(b) DS - Churn bug

Figure A.49: Caption of Descriptive Statistic for Churn feature and Churn bug

Quarter	N	Mean	Median	Std.Dev	Max	Min
2010-3	11	2.3	1	3	11	1
2010-4	32	2.6	2	1.7	8	1
2011-1	29	2.1	2	1.3	6	1
2011-2	24	1.3	1	0.7	4	1
2011-3	15	1.5	1	0.6	3	1
2011-4	37	2.5	2	2.1	9	1
2012-1	26	1.6	1	0.9	4	1
2012-2	34	2	2	1.5	8	1
2012-3	29	1.6	1	0.9	4	1
2012-4	35	2	1	1.5	7	1
2013-1	29	2.3	1	2.7	13	1
2013-2	16	1.5	1	0.6	3	1
2013-3	22	2.3	2	1.8	7	1
2013-4	19	1.4	1	0.6	3	1
Total	370	1.9	1	1.6	13	1

(a) DS - Bugs

Quarter	Finished	Not finished	Total	Finished	Not finished
2010-3	8	17	25	32	68
2010-4	65	17	82	79.3	20.7
2011-1	49	11	60	81.7	18.3
2011-2	29	2	31	93.5	6.5
2011-3	9	13	22	40.9	59.1
2011-4	72	22	94	76.6	23.4
2012-1	23	19	42	54.8	45.2
2012-2	53	16	69	76.8	23.2
2012-3	30	15	45	66.7	33.3
2012-4	65	6	71	91.5	8.5
2013-1	62	6	68	91.2	8.8
2013-2	16	8	24	66.7	33.3
2013-3	45	5	50	90	10
2013-4	26	0	26	100	0
Mean	30.8	9.3	40.13	67.0	33.0

(b) DS - Bugs per quarter

Figure A.50: Caption of Descriptive Statistic for Bugs and Bugs finished within quarter

Bibliography

Adams, M. and B. Smoak (1990). 'Managing manufacturing improvement using computer integrated manufacturing methods'. In: *Semiconductor Manufacturing Science Symposium, 1990. ISMSS 1990., IEEE/SEMI International*, pp. 9–13. DOI: 10.1109/ISMSS.1990.66111.

Alliance, Scrum (2012). 'Scrum, A description'. In:

Anderson, David J. (2010). *Kanban: Successful Evolutionary Change for Your Technology Business*. Blue Hole Press. ISBN: 0984521402.

Anderson, David et al. (2011). 'Studying Lean-Kanban Approach Using Software Process Simulation'. In: *Agile Processes in Software Engineering and Extreme Programming*. Ed. by Alberto Sillitti et al. Vol. 77. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, pp. 12–26. ISBN: 978-3-642-20676-4. DOI: 10.1007/978-3-642-20677-1_2.

Beedle, Mike et al. (1999). 'SCRUM: An extension pattern language for hyperproductive software development'. In: *Pattern Languages of Program Design* 4, pp. 637–651.

Birkeland, JørnOla (2010). 'From a Timebox Tangle to a More Flexible Flow'. In: *Agile Processes in Software Engineering and Extreme Programming*. Ed. by Alberto Sillitti et al. Vol. 48. Lecture Notes in Business Information Processing. Springer Berlin Heidelberg, pp. 325–334. ISBN: 978-3-642-13053-3. DOI: 10.1007/978-3-642-13054-0_35.

Brekkan, Elin and Eystein Mathisen (2010). 'Introducing Scrum in Companies in Norway: A Case Study'. In:

Cocco, Luisanna et al. (2011). 'Simulating Kanban and Scrum vs. Waterfall with System Dynamics'. In: *Agile Processes in Software Engineering and Extreme Programming*. Springer, pp. 117–131.

Conboy, Kieran (2009). 'Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development'. In: *Information Systems Research* 20.3, pp. 329–354. DOI: 10.1287/isre.1090.0236.

Concas, Giulio et al. (2013). 'Simulation of software maintenance process, with and without a work-in-process limit'. In: *Journal of Software: Evolution and Process* 25.12, pp. 1225–1248. ISSN: 2047-7481. DOI: 10.1002/smr.1599.

D. Sjøberg Yamashita, A, B Anda, A Mockus et al. (2012). ‘Quantifying the effect of code smells on maintenance effort’. In:

El-Emam, Khaled (2000). ‘A methodology for validating software product metrics’. In:

Gandomani, TAGHI JAVDANI et al. (2013). ‘Important considerations for agile software development methods governance.’ In: *Journal of Theoretical & Applied Information Technology* 55.3.

Gerring, John (2006). *Case Study Research: Principles and Practices*. Cambridge University Press. ISBN: 0521676568.

Gupta, Vikram (May 2013). *InfoQ Interviews David J. Anderson at Lean Kanban 2013 Conference*. URL: http://www.infoq.com/articles/David_Anderson_Lean_Kanban_2013_Conference_Intervew (visited on 01/10/2013).

IBM (Jan. 2014). *IBM SPSS Statistics 21 Core System User's Guide*. URL: http://www.sussex.ac.uk/its/pdfs/SPSS_Core_System_Users_Guide_21.pdf (visited on 30/01/2014).

Ikonen, Marko et al. (2010). ‘Exploring the sources of waste in Kanban software development projects’. In: *Software Engineering and Advanced Applications (SEAA), 2010 36th EUROMICRO Conference on*. IEEE, pp. 376–381.

Ikonen, M. et al. (2011). ‘On the Impact of Kanban on Software Project Work: An Empirical Case Study Investigation’. In: *Engineering of Complex Computer Systems (ICECCS), 2011 16th IEEE International Conference on*, pp. 305–314. DOI: 10.1109/ICECCS.2011.37.

Investopedia (30th Nov. 2013). *Quarter - Q1, Q2, Q3, Q4*. URL: http://www.investopedia.com/terms/q/quarter.asp (visited on 30/11/2013).

Javadian Kootanaee, Akbar, K Babu and Hamid Talari (2013). ‘Just-in-Time Manufacturing System: From Introduction to Implement’. In: *Available at SSRN 2253243*.

Kniberg, Henrik (2010). *Kanban and Scrum - making the most of both*. lulu.com. ISBN: 0557138329.

Lai, C.L., W.B. Lee and W.H. Ip (2003). ‘A study of system dynamics in just-in-time logistics’. In: 138, pp. 265–269.

Leonardo Campos Rafael Buzon, Eric Fer (Mar. 2013). *Kanban Pioneer: Interview with David J. Anderson*. URL: http://www.infoq.com/articles/David-Anderson-Kanban/ (visited on 30/09/2013).

Manning, James (2013). ‘Lean Software Development’. In:

Microsoft (2013). *Team Foundation Server*. URL: http://msdn.microsoft.com/en-us/library/ff637362.aspx (visited on 20/04/2013).

Middleton, P. and D. Joyce (2012). ‘Lean Software Management: BBC Worldwide Case Study’. In: *Engineering Management, IEEE Transactions on* 59.1, pp. 20–32. ISSN: 0018-9391. DOI: 10.1109/TEM.2010.2081675.

Munassar, Nabil Mohammed Ali and A Govardhan (2010). ‘A Comparison Between Five Models Of Software Engineering.’ In: *International Journal of Computer Science Issues (IJCSI)* 7.5.

Ohno, Taiichi (2001). *Toyota Production System on Compact Disc: Beyond Large-Scale Production*. Productivity Press. ISBN: 1563272679.

Oracle (2013). ‘Arraylist’. In:

Poppendieck, Mary (2003). 'Lean Development and the Predictability Paradox'. In:

Poppendieck, Mary and Tom Poppendieck (2003). *Lean Software Development: An Agile Toolkit*. Addison-Wesley Professional. ISBN: 0321150783.

– (2006). *Implementing Lean Software Development: From Concept to Cash*. Addison-Wesley Professional. ISBN: 0321437381.

– (2009). *Leading Lean Software Development: Results Are not the Point*. Addison-Wesley Professional. ISBN: 0321620704.

Raman, S. (Oct. 1998). 'Lean software development: is it feasible?' In: *Digital Avionics Systems Conference, 1998. Proceedings., 17th DASC. The AIAA/IEEE/SAE*. Vol. 1, C13/1–C13/8 vol.1. DOI: 10.1109/DASC.1998.741480.

Rouse, Margaret (2005). *Throughput*. URL: http://searchnetworking.techtarget.com/definition/throughput (visited on 04/03/2014).

Seikola, M., H. Loisa and A. Jagos (Aug. 2011). 'Kanban Implementation in a Telecom Product Maintenance'. In: *Software Engineering and Advanced Applications (SEAA), 2011 37th EUROMICRO Conference on*, pp. 321–329. DOI: 10.1109/SEAA.2011.56.

Shepard, Jon M. and Robert W. Greene (2002). *Sociology and You*. Glencoe/McGraw-Hill. ISBN: 0078285763.

Shinkle, C.M. (2009). 'Applying the Dreyfus Model of Skill Acquisition to the Adoption of Kanban Systems at Software Engineering Professionals (SEP)'. In: *Agile Conference, 2009. AGILE '09.* Pp. 186–191. DOI: 10.1109/AGILE.2009.25.

Sienkiewicz, Lukasz (2012). 'Scrumban - the Kanban as an addition to Scrum software development method in a Network Organization'. In:

Sjøberg, D.I.K., A. Johnsen and J. Solberg (2012). 'Quantifying the Effect of Using Kanban versus Scrum: A Case Study'. In: *Software, IEEE* 29.5, pp. 47–53. ISSN: 0740-7459. DOI: 10.1109/MS.2012.110.

Software Innovation (Dec. 2013). URL: http://www.software-innovation.com/EN/COMPANY/pages/default_.aspx (visited on 12/12/2013).

The-Kanban-Way (2011). *Importance of Kanban work-in-progress (WIP) limits*. URL: http://www.kanbanway.com/importance-of-kanban-work-in-progress-wip-limits#.U0Pmzq2Swkc (visited on 14/10/2013).

Vesaliusstraat, Andreas (n.d.). 'The Non-Transitivity of Pearson's Correlation Coefficient: An Educational Perspective'. In:

Yin, Liangfang, Yuanqing Xiao and Zhenyu Xu (June 2013). 'A new principle based on Pearson correlation coefficient to avoid mal-operation of the restricted earth fault protection'. In: *Electricity Distribution (CIRED 2013), 22nd International Conference and Exhibition on*, pp. 1–4. DOI: 10.1049/cp.2013.0799.

Yin, Robert K. (2008). *Case Study Research: Design and Methods (Applied Social Research Methods)*. SAGE Publications, Inc. ISBN: 1412960991.