ESSAY –"DOES WORK IN PROGRESS (WIP) - LIMIT IN AGILE SOFTWARE DEVELOPMENT MATTER?"

Introduction

In my master thesis I will analyze data from Software Innovation (SI); the data will consist of ten years of recorded data on software development. SI started off with Scrum but changed to Kanban in recent years, so I will try to see if the WIP matters on their productivity, based on their data, and see if I can find any factors taking in to consideration when finding the optimal WIP for a given interval.

Background

There's done some research on WIP limit vs. unlimited WIP with event-driven simulator (Giulio Concas and and Hongyu Zhang 2012) and when discussing the importance about agile and lean software development (Kniberg 2010).

But, how to find the best WIP in a given interval and context has not been much researched, but in manufacture business, some research has been done, Taho Yanga, Hsin-Pin Fub, Kuang-Yi Yanga stated that WIP could be defined: WIP = cycle time * throughput rate in manufacture business (Taho Yanga),

Kanban

"We can define Kanban software process as a WIP limited pull system visualized by the Kanban board" (David Anderson 2011)

Kanban systems focus on continuous flow of work, no fixed iterations, work is delivered when it's done, the team only work in few task at the time to limit WIP and make constant flow of released tasks(David Anderson).

Toyota production system introduced Kanban and WIP during late 1940s and early 1950s in order to catch up with the Americans (Ohno 1978) . In the last ten years software Development Company have started to implement agile methods and Kanban is one of them(Coboy 2009). Kanban splits the problem into many small pieces of problems and then, the problem is put up on the Kanban-board to visualize the problems and see potential bottlenecks. When people started to understand Kanban, they easily saw where the bottlenecks where, and started to help where the bottlenecks where (Shinkle 2009).

One of the most important people in software development-Kanban, David Anderson referred to as "father of Kanban in the software development industry"(InfoQ), author of "Kanban: Successful Evolutionary Change for Your Technology Business" once 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."(InfoQ). More and more software projects adapt to Kanban, and this is one of the reasons why I choose to focus my thesis on Kanban and WIP.

Kanban is one of the agile method in the wind these days, and is used with Lean Software development which is one of the fastest growing approaches in software development (David Anderson 2011).

One of the main difference between Scrum and Kanban is estimation, in simulation of software maintenance process, with and without a work-in-process limit (Giulio Concas and and Hongyu Zhang) estimation was defined to be the main source of waste. In their research, they find out, if they let the developers work with small tasks at time and not be interrupted, they will be more effective. The developers in this case was interrupted when they was assigned to estimate tasks. The research groups decided to implement Lean-Kanban, which includes throwing away waste, which meant estimation for this case. After implementing Lean-Kanban the team's increased the ability to perform work, lower the lead time and meet production dates.

Scrum

Scrum has three main roles, the Product Owner, the ScrumMaster and the members of the development team. The Product owner decides what work to be done. The ScrumMaster act like a team leader and helps the team and organization to take best advantages of Scrum. The development team works on tasks specific for the iteration there in. (Alliance)

Iteration is a sprint over a given time, usually from one to four weeks. Before each sprint, a sprint planning meeting is conducted, with all the team members attending. From the backlog, the Product Owner and the team members discuss each task to get a better understanding, shared understanding and what is required to complete the task. There's also given a shared understanding of what to be done in the sprint which tasks to be completed. After each sprint a visible part of the product should be done and ready for the end users. (Alliance)

Both Kanban and Scrum have a backlog which often is visible on a board, but Scrum also has a Sprint backlog which is a minimal backlog, unique develop to each sprint. (Alliance)

Lean-Kanban

The Lean approach was introduced between 1948 and 1975 in manufacturing in Japan. It was designed to find and eliminate waste, so the manufacturing could deliver value to the costumer more efficiently. In 2003, Mary and Tom Poppendieck first introduced Lean thinking to software development, they published "Principles of Lean Thinking". An important tool to manage work flow is the concept of pull-systems, which means tasks are put in the backlog when a costumer asks for it.

In the last three to four years, Kanban has been introduced more and more to software development, and is becoming one of the keys to Lean practice in the field. (David Anderson).

Kanban board

"The Kanban board make it clear to all the team members the exact status of progress, blockages, bottlenecks and they also signal possible future issues to prepare for"(Joyce 2012)

The Kanban board is one of the most important tools in Kanban. It’s used to minimize WIP and increase the information flow, with visualization (Giulio Concas and and Hongyu Zhang 2012)

The Kanban-board uses named columns to illustrate where each item is in the workflow. Every column has a WIP, to specify how many works in progress there are allowed (Kniberg 2010)

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” (Joyce 2012)

Lead time is an essential ingredient when you look for the optimal WIP for your project. Often in 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's and see the different lead-time and then measure which WIP that suits this project the best. Usually the lead time starts when a developer picks the task and starts working on it and stops when the task is finished.

WIP

“There’s stated when using short-cycle times and Kanban board to limit WIP, the software development team's learning is increased” (Joyce 2012)

Work in progress (WIP) is used in Kanban to visualize bottlenecks and limit task-switching since each developer is limit to a specific WIP.

When first implementing Kanban, Shinkle explains that the users don't care about the WIP, but rather the visibility of Kanban. When the user gets more experience with Kanban, they start to attempt limit WIP (Shinkle 2009) Shinkle defined novice and more experience users with an analogy.

“Think about a typical person wanting to bake a cake. They go to the store, purchase a boxed cake mix, and follow the directions as described on the back of the box. They have little to no knowledge about how to alter the recipe nor do they have a desire to do so. Their goal is simply to bake a cake."

“An advanced beginner understands how to apply some context to the instructions or rules on the back of the box. They can make minor adjustments for things like altitude, pan size, oven conditions, etc. They are still following

the basic recipe, but can make minor adjustments likely based on previous experiences."

A developer stated his opinion on WIP:

“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)

Setting a limit on WIP has a number of benefits according to Hopp, Spearman and Suri. They stated that it reduces flow times, reduces variation and improves quality. However Srinivasan, Ebbing and Swearing said that setting the WIP is not easy. They suggest that WIP are just set, and then observe throughput, and adjust after that.(Mandyam M. Srinivasan 2003)

A lot of people have their opinion on how to best measure the WIP, but most of them agree upon that's there is no clear rule of how to measure WIP in software development. Lukasz proposes to use The Effectiveness as an indicator measured at the end of each cycle. The effectiveness is a formula that takes into account the number of bugs found (ai) and the number of bugs found by the external people (e.g. lawyers, accountants, coaches, consultants, translators, internal and external service providers ,etc.)(ei).(Sienkiewicz 2012)

$$Ei = \frac{ai - ei}{ai} * 100\%$$

On the other hand, Kanban says you should limit WIP. So what should the limit be? – Kanban; Don't know, experiment (Kniberg 2010).

Lean Software Management suggest that WIP should be minimized, to keep high quality (Ikonen 2011) and to create continuous flow and bring problems to the surface (Joyce 2012) In Lean Software Management: BBC Worldwide Case Study(Joyce 2012) when the team realized the where bottlenecks and to high WIP the team started to determine WIP by their constraints. The team quickly realized that they had fewer quality assurance/testing staff and business analysts than software developers. This reflected the bottlenecks, so the team adjusted

WIP to how much work they could handle; this gave the team more experience in dealing with WIP (Joyce 2012)

Alwardt suggest that task should be prioritized from high to low, in order to keep the best WIP. Lean thinking in software development: Impacts of Kanban on projects(Ikonen 2011) stated in the conclusion that high WIP will keep people task switching and not be able to fully concentrate on each work in progress. (Ikonen 2011)

Limit WIP vs. Unlimited WIP

Simulation of software maintenance process, with and without a work-in-process limit(Giulio Concas and and Hongyu Zhang 2012) has done research on limit WIP vs. unlimited WIP, one of the result from this paper was at the end of a simulation, the average of closed issued was 4145 when the WIP was limited and 3853 when the limit was not limited (about 7% less). The paper concludes their finds; developers are more focused on fixing few issues, because the number of issues they can work on is limited. So, because of the limit WIP, the developers are more likely to continue on the issue from the day before, rather than starting on another issue. When, developers starts on a new issue, they need to use time to familiarize themselves with the code and the problem, and that creates unnecessary overhead if a developer already has done it, but that developer is now working on a another issue.

Research Questions

In my thesis I would like to find out if WIP matters in software development, if there exist an optimal WIP-limit for a given context, and if there exist such thing, how to best find the optimal WIP-limit, which context parameters shall be taken into consideration.

Research Method

From the data received from SI, I will make a program that will analyze the data. To this date, I have measured WIP, items in backlog at a given date and how long a given task is in the backlog before pulled out. I also measured number of tasks in backlog per quarter and WIP.

I have done this for year 2010, 2011 and 2012. Since theses measurements are independent of teams, teams size, developers skills and developers method, there is not much to get out of it - yet!

Hopefully, I will measure data per team in a given interval and I will also measure data from every team in a given interval to see if WIP matter.

Software Innovation (SI)

Software Innovation is a Scandinavian software company. SI develops and delivers market-leading Enterprise Content Management applications that helps organizations improve and increase efficiency in document management, case handling and technical document control.

Software Innovation has approximately 300 employees, and has offices in Oslo, Copenhagen and Stockholm and a development center in Bangalore(Innovation)

Analyzed Data

My plan is to measure the data in one big step, where I measure all the teams and data in some given interval. I would also measure the data in small steps, when IS used Scrum, when IS was in the transforming phase and when they had changed to Kanban. Also I will measure WiP values for a given day, week, month, quarter and year. This will give me data to compare between teams in a specific interval.

When I have measured WiP, I will reuse some data from the paper from *Quantifying the Effect of Using Kanban vs. Scrum: A Case Study*(Dag I.K Sjøberg 2012). Mainly I will look at churn, lead-time and throughput measured in the paper and compare them with WiP and between teams. Hopefully the data measured will give me some sense of what WiP limit does for SI and which factors helping determine if WIP-limit matters in agile software development

The topic of my thesis could also help me find out if there's a WIP-limit interval for a given context and which factors who can give and impact on how to measure WIP-limit.

Hopefully I would gather or get some other underlying data, so I will be able to compare my work, on the set from SI to the other data.

How I have analyzed

.

Result

Reference:

Alliance, S. (2012). "Scrum a description."

Coboy, K. (2009). "Agility from First Principles: Reconstructing the Concept of Agility in Information Systems Development."

Dag I.K Sjøberg, A. J. a. J. S. (2012). "Quantifying the Effect of Using Kanban vs. Scrum: A Case Study."

David Anderson, G. C., Maria Ilaria Lunesu and Michele Marchesi (2011). "Studying Lean-Kanban Approach Using Software Process Simulation."

Giulio Concas, M. L. L., Michele Marchesi and and Hongyu Zhang (2012). "Simulation of software maintenance process, with and without a workin-process limit."

Ikonen, M. (2011). "Lean thinking in software development: Impacts of kanban on projects."

InfoQ (2013). "InfoQ Interviews David J. Anderson at Lean Kanban Conference."

InfoQ (2013). "Kanban pioneer: Interview with David Anderson."

Innovation, S. "Software Innovation homepage."

Joyce, P. M. a. D. (2012). "Lean Software Management: BBC Worldwide Case Study."

Kniberg, M. S. H. (2010). "Kanban and Scrum - making the most of both."

Mandyam M. Srinivasan, S. J. E. a. A. T. S. (2003). "Woodward Aircraft Engine Systems Sets Work-in-Process Levels for High-Variety, Low-Volume Products."

Ohno, T. (1978). "Toyota Production System: Beyond Large scale production."

Shinkle, C. M. (2009). "Applying the Dreyfus Model of Skill Acquisition to the Adoption of Kanban Systems at Software Engineering (SEP)."

Sienkiewicz, L. (2012). "Scrumban the Kanban as an addition to Scrum software development method in a Network Organization."

Taho Yanga, H.-P. F., Kuang-Yi Yanga (2012). "An evolutionary-simulation approach for the optimization of multi constant work-in-process strategy a case study."