Zahoor Ahmad Khan

Scrumban - Adaptive Agile Development Process

Using scrumban to improve software development process

Helsinki Metropolia University of Applied Sciences

Degree: Masters in Engineering

Degree Programme: Masters in Business Informatics

Thesis: Scrumban – Adaptive Agile Development Process

Date: May 2014

Author(s) Title Number of Pages Date	Zahoor Ahmad Khan Scrumban – Adaptive Agile Development Process 73 pages + 3 appendices 23 May, 2014
Degree	Masters in Engineering
Degree Programme	Masters in Business Informatics
Specialisation option	
Instructor(s)	Thomas Rhowder, DSc (Econ) / Principal Lecturer

This thesis concentrates on improving the existing agile software development process used in the case company in order to overcome the various problems faced during the product development based on the existing Scrum model. The case company, consisting of sixty five employees, turns any online content, images, videos and applications into interactive and viral storefronts by means of non-intrusive and content relevant advertisement products.

After describing the research objective, researcher started with the current state analysis of the agile scrum process within the Applications Development team of the case company. Qualitative research methodology was utilized in this study. The data for current analysis was collected based on interview discussions, analysing past sprint retrospectives, and sprint velocity data. The participants used for interviews and discussions were from various backgrounds and departments of the case company and included experts such as, Scrum Master, Product Owner, Product Manager, Quality Assurance Team, Team Manager, Operation Manager and Software Developers. Overall eight (8) persons were interviewed. Based on current state analysis and literature review, a new process model was defined by the researcher. This research model was further improved after gathering qualitative research data from different stake holders and experts mentioned earlier. The new proposed process model was further pilot tested in the case company for four iterations. The results and final conclusion from the piloted iterations were further documented in this study.

The results from the pilot testing indicated that the new process model has provided the needed flexibility to the team and product owners, work flow across different stages of process has improved, team collaboration has improved and in addition team is constantly improving their work policies.

Keywords	Agile Scrum Methodology, Scrumban, Kanban, Product Development Process

Contents

ABBREVIATIONS/ACRONYMS

PBIs	Product Backlog Items
PO	Product Owner
UI	User’s Interface
QA	Quality Assurance
WIPs	Work In Progress Items
XP	Extreme Programming

1 Introduction

1.1 Company Context

This research is done in the context of a Finnish case company. The case company was founded in the year 2010. It is a fairly new start-up that has won several awards since its inception. It provides a platform enabling smart content and turns any online content, images, videos and applications into interactive and viral storefronts. With the faster growth of internet, nowadays almost everyone in the retail industry is turning online. At the same time, media is becoming decentralized; there are several bloggers, publishers and social media clouds. Old and intrusive way of online advertisement is not anymore the answer to current marketing needs. As a result, the concept of smart content is becoming the answer to current online marketing needs. Smart content turns advertising into a service for consumers. It is a non-intrusive service and by simply hovering over or placing a mouse over the smart content on internet (which can be any media, images, videos, content or anything you see online) displays high relevant information about the product or service in the content. This enables users, publishers and brands to monetize the content and that is always relevant to the context of the online content displayed, For example, brands can display products inside an image or video on some online publisher's site and those products will be relevant to the context of that particular image or video.

The case company enables brands to turn their content into storefronts and engage directly with their fans everywhere. Publishers can monetize the impulses they generate through the content where the case company added non-intrusive and relevant online products called Kiosks. Through kiosks anyone can buy, want and get more information whenever user does any interaction with the content. Further, the platform provided by the case company makes it rewarding for the online users to discover and share any online content.

1.2 Business problem and Objective

The current business problem is related to the Software product development process of applications team within the case company context. In recent years, many organizations have started using agile methodologies for the implementation of software development projects. Few years back, the company in context also started using agile methodology called Scrum. However, after implementing the scrum methodology for a certain period of time, the software development teams, especially applications team, faced problems while applying the same. Some of the problems faced were:

- **Difficult time completing the sprint planning:** Many times team members could not agree on the time estimates because of incomplete stories, so the planning meetings dragged on or sometimes effort estimation was cancelled altogether for many tasks.
- **Missing collaboration and problems in self-organization:** One of the biggest obstacles that the team members faced was missing collaboration and self-organization. The team members tried to concentrate only on their tasks and not the interdependencies within and outside the team. They assumed that the other team members would handle those interdependencies as they only focused on their own tasks.
- **Incomplete tasks within a sprint**. Often, the team could not complete the sprint tasks as the nature of the tasks was very complex. There were various tasks which included prototype work or research work and it was quite difficult to assess the effort complexity of such tasks. Also, sometimes tasks were getting held in some stage of the work flow. For instance, in some sprints, many tasks were moved to testing stage simultaneously, and as a result team's Quality Assurance person could not complete the tasks by the end of sprint iteration.
- **Product Owners change priority within Sprint**: Product owners were changing the priority of tasks in the middle of a sprint. Further, additional tasks were added to the sprints which were not estimated in sprint planning.
- **Insufficient tasks for some team members:** Team members used to commit to X number of story points based on the average velocity of last 5 sprints. As the scrum methodology focuses on filling sprint backlog based on velocity rather than the number of tasks, members can actually perform, some team members who were specialized in particular technologies were left with insufficient tasks in the middle of the sprint. Further, the team could not pull new tasks from the product backlog as the scrum does not allow that. Sometimes, the team added some new tasks into the sprint which were left incomplete and changed the overall scope of the sprint.

- **Abrupt switch to Kanban mode in some sprints:** Sometimes, the team was shifting from Scrum to Kanban mode because of frequent changes in requirements or the pressure from product management. The management expected the team to get the new tasks done within the same sprint instead of the next one which often lead to the shift from Scrum to Kanban as the team could not simply follow sprint commitments. Further, tasks were getting held in testing or review stage as there were no work policies or maximum limits followed in different workflow stages.

Looking at the issues mentioned above, these lead to delays in the project delivery, increase in the project costs, poor quality of deliverables, and sometimes even failure of sprints. Given this, the objective of this study is to find out a suitable proposal for improved agile software product development process which can handle the situations described above and use the best practices from other agile methodologies such as Kanban or Scrumban.

The output of this study will be improved agile software development process suitable for the applications development team within the context of the case company. Further, this improved process will be pilot tested by the same team for certain iterations. This thesis output could help the team as well as organisation to avoid the issues discussed above and handle those situations in a better way.

1.3 Limitations & Scope

The scope of this thesis is limited to the case company described in the introduction section 1.1. Within the case company, the scope is further valid only for the applications development team. The aim of this research is not create any new agile methodology but rather adapt good ideas from the existing methodologies which are suitable for the team under discussion within the context of the case company.

2 Research Approach

This section describes the research approach used in this thesis which includes overall research design and data collection process.

2.1 Research Process

The research starts with describing the business problem and defining the research objective. This will be followed by the current state analysis of the Agile Scrum Process used within the Applications Development team of the case company. This team consists of five developers and one dedicated quality assurance member. Further, two developers are having dual roles, one is scrum master and the other is the team leader. The main purpose of the current state analysis is to establish reasons for the needed change in the current software development process of the case company. The current state analysis will be compiled based on analysing the past sprint retrospectives of the team, analysing the data collected from discussions and interviews of various stakeholders.

The current state analysis is followed by the literature review in order to understand how similar business problems have been handled using the existing literature, and understanding what are the best industry practices in Agile Software Development which can handle the business problem described in this thesis. Accordingly, both the current state analysis and literature review are used together to create the new process model, which is further developed based on data collected from qualitative interviews with all the concerned stakeholders. Finally, new process model will be pilot tested within the development team of the case company for four sprints, so that process can be analysed in the day to day working environment and accordingly final conclusions will be drawn.

The research design Implemented in this thesis is illustrated in Figure 1.

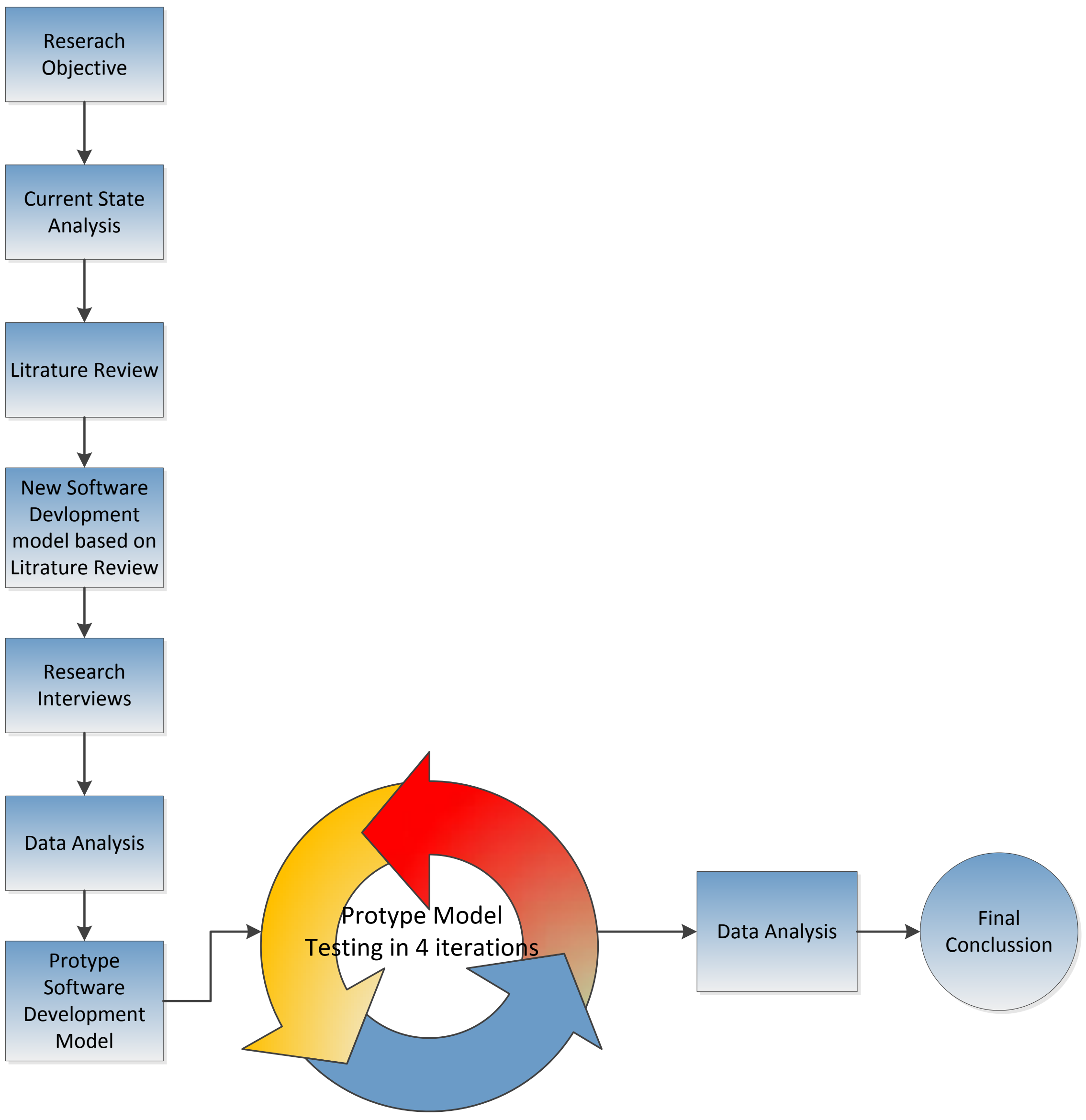

Reserach Objective
Current State Analysis
Litrature Review
New Software Devlopment model based on Litrature Review
Research Interviews
Data Analysis
Protype Software Development Model
Protype Model Testing in 4 iterations
Data Analysis
Final Conclussion

Figure 1. Research design of the study

2.2 Data collection and data analysis

The data collection will start with the current state analysis of the agile scrum software development process within the case company. This will include

1. Analysing all the possible sprint retrospective data of past twenty three (23) sprints, because it is mainly the sprint retrospectives in a scrum process where teams highlight and discuss three main questions namely "What went wrong in

the last Sprint", "What went right in the last Sprint" and "What can be improved in the next Sprint".

From the data collected from all the sprint retrospectives, each problem which the team was trying to improve in the upcoming sprint was given a score point of one (1). And in case a similar problem was repeated in some other sprint, the score for that problem was incremented by 1. Finally, only those problems were presented in the final report as shown in Table 4 which occurred more than once.

In addition to scrum retrospective data, velocity related data of the past twenty three (23) sprints was also collected. This information is crucial in determining the output of the team in terms of story points completed within a sprint (Velocity) or average story points completed (Average Velocity), and for calculating standard deviation of the velocity. The data was collected from different intranet resources available within the company, and the details of the data collection are presented in the Table 1.

.

Data Source	**Data Collected**	**Number Of Sprints**	**Data**
Company Intranet tools like Trac, Jira, Google Drive	Sprint Retrospectives	Data was collected from last 23 Sprints	Table 4
Company Intranet tools like Trac, Jira, Google Drive	Sprint Velocity	Data was collected from last 23 Sprints	Table 5
Interview Discussions	Notes, Audio Recordings	Not Applicable	Appendix 3

Table 1. Details of Data Collection in the Current State Analysis

2. Interviews and discussions with all the necessary stake holders within and outside the development team, within the context of case company. The participants for the interviews were selected from a wider range of expertise areas within the case company in order to get a broader perspective on the current development process and its pros and cons. The participants were selected from within the Applications Development Team, Product Management, Quality Assurance, Ex-Scrum Master of the team and Operations Team. These participants are the main stakeholders who are impacted by the development process used by the applications development teams within the case company. The details of the data collection are presented in the Table 2

Data Source	Participants	Duration	Topic/ Discussion	Data
Interview/Discussions	• Team Leader/Product Owner • 2 x Team Member • Head Quality Assurance • Team QA • Head of the Software Development • Product Manager • Ex Scrum Master of the team	8 x 30 minutes sessions	Current state analysis related questions (Appendix 1)	Appendix 3

Table 2. Details of Data Collection in the Current State Analysis

The outcome of the current state analysis along with the literature study and review was used to formulate the main research interviews with different stake holders in order to define the prototype software development process. The participants selected in this round were from within the Applications Development Team, Product Management, Quality Assurance, Operations Team, and ex-Scrum Master of the team.

From the applications development team, two senior developers and the team leader were selected. They have more knowledge on how the team has been functioning from the developer’s perspective, and they know what challenges and problems team has faced in the long run. The dedicated quality assurance (QA) person of the team was also interviewed in order to know how the team has been performing from the QA perspective, and what challenges and problems need to be addressed for ensuring better quality.

In addition, product owner/product manager of the team was also interviewed. Product owner observes how a team is performing when it comes to commitments, customer delivery, planning, prioritization and customer feedback. Also, head of overall software development was interviewed, since he is the main owner of processes being followed in different software development teams within the case company. The other responsibilities of head of the software development include looking after team’s resource availability, skill set, work culture, and overall delivery of different software products. So his feedback is very important in the current state analysis of this thesis. Further, the quality assurance (QA) head of the case company was also interviewed in order to get his feedback and opinions on current development process and how it impacts the overall quality of software being shipped.

Finally, ex-scrum master of the team was also interviewed in order to get some feedback on how well the current process has worked for the team in past and what where the challenges and issues team was facing during his tenure of team’s scrum master. The other reason was interviewing the ex-scrum master is to bring in new ideas on how well the scrum is working in his current team, know about the challenges and what changes have helped his new team. The details of the data collection used for defining the prototype process model are described in Table 3

Data Source	**Participants**	**Duration**	**Topic/ Discussion**	**Data**
Inter-view/Discu	• Team Leader • 2 x Team Mem-	8 x (45 to 60	Prototype Process Related	

ssions	ber • Head Quality Assurance (QA) • Team QA • Head of Software Development • Product Owner/Manager • Ex Scrum Master of the team	minutes) sessions	Questions (Appendix 1)	Appendix 2

Table 3. Details of Data Collection in the Prototype Model Creation

Both in current state analysis and prototype model creation phase, qualitative interviews were conducted and all the interviews were documented. After the new software development model was developed, a power point presentation was presented to all the stake holders which included product management, development team, team leaders, operations team and quality assurance team. After the presentation, a “Go Ahead” decision was taken by the head of the “Software Development & Services” and Product Management in order to pilot test this process model for four sprints within the applications development team. The result of the piloted sprints was analyzed and is documented in section 6.

3 Current State Analysis

3.1 Analysis of the Sprint Retrospectives

In this analysis, all the retrospective related data of the last 23 sprints was collected from the company’s various intranet resources such as Jira, Google Drive, Trac Online tool, and Scrum Master’s notes. In the Sprint Retrospectives, every team member answers these three important questions

1. What was good or positive in the last sprint?
2. What was bad or negative in the last sprint?

3. What could be improved in the next sprint?

Based on the above mentioned questions, Scrum Master would collect all the answers from different team members and accordingly create a list of action points on what could be improved in the upcoming sprint. This information is important for analysing what were the challenges faced by the team in different sprints. The analysis of this data would give very important information on how agile scrum model was used within a team, and whether it was working for the team. From the retrospective data, each negative point that was brought up in the sprint was given a score of 1 and in case the same problem repeated in some other sprint, the score of the problem was incremented by 1. The details of the sprint retrospective data analysis are described in the Table 4

Order	ISSUE or PROBLEM WITHIN A SPRINT	SCORE POINTS	TOTAL
1	Missing collaboration (with product owners, sales, inter-team, intra-team, with quality assurance team)/Miss Communication.	1+1+1+1+1+1+1 +1+1+1	10
2	Stressful sprint, loaded work for team members, tight schedules.	1+1+1+1+1+1+1+1+1	9
3	Unclear process, unclear software delivery process for prototypes and products.	1+1+1+1+1+1	6
4	Urgency from management, tasks coming from outside the product owners like designers, top management or client managers. Those tasks were not present in team's task management tool such as Jira.	1+1+1+1+1+1	6
5	Not adhering to process, (For example, tasks not present in Jira, Jira not used properly, overlapping tasks, missing acceptance criteria's for tasks)	1+1+1+1+1	5
6	By the end of a sprint, many tasks are stuck in testing phase.	1+1+1+1+1	5

7	Improve code review process/workflow as tasks get stuck in the review stage and developers continue with other tasks.	1+1+1+1+1	5
8	Unclear roles and responsibilities within team (For example, product owners role was unclear in some sprints, documentation responsibilities were unclear at times)	1+1+1+1	4
9	Missing or poor specifications for some user stories within a sprint	1+1+1	3
10	Over commitment of story points. The number of story points committed within a sprint was more that the amount of story points delivered by the end of a sprint.	1+1+1	3
11	Too big stories were taken inside a sprint, which usually last for more than a sprint. It could have been better to split those tasks into smaller ones	1+1+1	3

Table 4. Analysis of the Sprint Retrospectives

From the sprint retrospective data analysis as described in the Table 4, it can be seen that the team was facing many challenges:

Missing Collaborations or Problems in self-organizing and decision making: Team members only concentrate on their own tasks; in a self-organizing team they should look at interdependences within a team or outside a team and handle those issues as needed. Team members shall organize themselves and not simply concentrate on a task at hand, as is clear from point number 1 in the Table 4.

Some Sprints are simply constant stress: As indicated by points 2, 4, 5 and 10 from

Table 4, things keep on changing within a sprint, managements adds new tasks within a sprint or modify existing tasks within a sprint. Team members have to complete tasks from the sprint backlog as well as the tasks coming from others. As a result team members are constantly under workload which is not good for the motivation of developers.

Tasks came from every direction: As indicated by point number 4 from Table 4 , in addition to sprint backlog tasks other tasks were coming from top management, product owners, designers, and client managers. All these tasks were putting additional workload on the team and as a result team was unable to complete tasks on time or unable to focus on sprint goals. Sometimes incomplete user stories were added to the sprints that were not planned as part of the sprint planning. This could lead to poor quality of deliverables.

Unclear process or weak process: Some aspects of the process were not clear, for instance how to deliver product prototypes? Do tasks related to product prototypes have to follow the same process as for the normal product related tasks? Sometimes workflow process was skipped and tasks were taken from outside the tasks management tool. This is evident from points 3, 4 and 5 in Table 4.

Tasks getting stuck in a particular development stage: By the end of a sprint, many tasks were stuck in some stage of the workflow. As indicated by points 6 and 7 from the Table 4, it occurred in many sprints and some stage of the work flow process often became a bottleneck. Reasons were many, for instance when team overcommits to the amount of tasks within a sprint and then some stage or other became the bottleneck. Sometimes testers had many tasks in their tasks lists or the developers could not review tasks on time.

A team tends to have too big tasks: From point number 11 in Table 4, it is clear that team was not splitting big tasks in certain sprints and as a result such tasks were spanning across multiple sprints. This often led to belief that team’s estimation accuracy is low, or their visibility and transparency is low**.**

Poor quality of specifications: As indicated by points 5 and 9 from Table 4, sometimes scope of the task was changed within a sprint, or incomplete user stories were planned for the sprint. On one hand, team cannot stop working on tasks because certain aspects of it were not fully specified yet, being as startup company requirements often change as managements wants to try and test the results, but on the other it may lead to poor quality of deliverables.

Conflicting Roles & Responsibilities: As indicated by point number 8 in Table 4, there were certain sprints where non-technical persons were team's product owners and they did not have enough competencies to specify requirements clearly. In some other sprints, product owners were having more than one role at a time; also there were no clear owners for the internal or external documentation.

Over commitment of Story Points within a Sprint: As indicated by the point number 10 in Table 4, there were some sprints were team was overcommitting to the story points compared to what they should have committed based on average velocity of past sprints. This also led to the belief that team's estimation accuracy was low, even though the actual reasons could be many such as pressure from the top management or product owners.

3.2 Analysis of Velocity Data

According to whatis.techtarget.com (2013), "Velocity is a metric that predicts how much work an agile software development team can successfully complete with a time boxed iteration called sprint". Velocity of a Sprint is the number of story points completed within a sprint. The data regarding the team velocity was collected from different intranet sources of the case company such as Jira, Google Drive and Trac. The different sprint metrics, such as, velocity per sprint, velocity per team member, average team velocity, average velocity per team member and standard deviation of velocity data are described in Table 5. Velocity per team member is obtained dividing sprint velocity with the total number of team members present in the team. Total number of team members in the team under discussion is six (6).

Sprint No.	Velocity (V)	Velocity per team member= V/6	Average Velocity (AV)	Average Velocity per team member= AV/6	Standard Deviation of Velocity (SDV)	Standard Deviation of Velocity per team member= SDV/6

1	56	9,333333	40,65217	6,775362	10,56413	1,760688
2	23	3,833333				
3	41	6,833333				
4	49	8,166667				
5	35	5,833333				
6	32	5,333333				
7	34	5,666667				
8	44	7,333333				
9	29	4,833333				
10	61	10,16667				
11	25	4,166667				
12	44	7,333333				
13	43	7,166667				
14	45	7,5				
15	42	7				
16	40	6,666667				
17	58	9,666667				
18	29	4,833333				
19	39	6,5				
20	32	5,333333				
21	51	8,5				
22	31	5,166667				
23	52	8,666667				

Table 5. Velocity data collected from various sprints

Based on the velocity data described in Table 5, a bar chart representation of the sprint velocity for last 23 sprints is depicted by the Figure 2

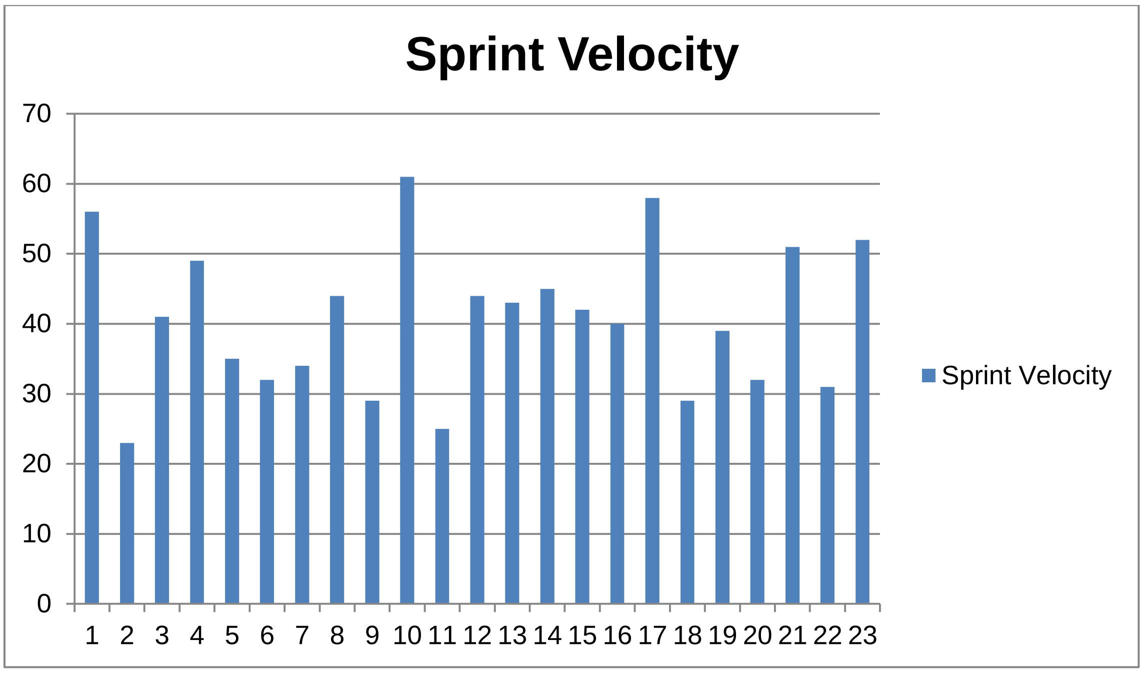


Figure 2. Bar chart representation of the team velocity related data

As seen from the data in Table 5, the average team velocity for the applications development team using the scrum methodology is 40.65 with a standard deviation of 10.56. The data also indicates that for the majority of sprints, sprint velocity is within the range of one standard deviation from the mean value. In addition, the average sprint velocity per team member is 6.77 with a standard deviation value of 1.76. This data will help the researcher in comparing the current team velocity with the team velocity once new process model is pilot tested.

3.3 Analysis of Product Development Process

The current product development process being followed in the case company is agile scrum methodology. In this section, researcher will be describing step by step process, followed by the applications development team within the case company context. At the end of each stage, researcher has done the analysis of that particular stage. In the scrum methodology, each software development cycle called sprint, starts with the sprint planning meeting.

Sprint Planning: Usually last for 2 hours. In this meeting team normally take the tasks from a prioritized set of product backlog, try to roughly estimate those using a method called scrum poker. Using scrum poker, each team member roughly estimates tasks based on story points, and then the whole team tries to decide on the most appropriate story points for tasks based on what majority votes for. Similarly, team tries to estimate the rest of the prioritized stories and then roughly selects a subset of tasks into the sprint. The number of story points taken into the sprint is roughly based on average velocity of last 5 sprints.

Strength:

- Sprint Planning is so far very helpful; this is the place where team plans and estimate different stories and tasks.
- Task estimation by the team helps product owners to know the complexity of a task

Weakness:

- Sometimes sprint planning is not good enough in case product backlog stories are incomplete or product owner has not properly prioritized the stories, and therefore in certain sprints team end up prioritizing tasks in planning sessions rather than selecting and estimating tasks. In these situations, sprint planning meetings starts to drag and often lose focus.
- Estimating stories using story points is very much misused by the management who usually compare it to number of days or man days. On the contrary story points shall indicate nothing but the complexity of a task.
- Mostly team tends to focus on new stories rather than bugs, so often anything else that blocker bugs keeps on pending in the backlog. Team tends to ignore bugs in planning because they are not assigned any story points, and also developers prefer to work on new tasks.
- Another drawback in sprint planning is team doesn’t discuss about design issues related to tasks, they just estimate the complexity.
- Some tasks are so technology centric that only one or two team members can roughly estimate the related stories.

Sprint: After planning is done scrum master starts the sprint. Every sprint in the context of case company lasts for two weeks. Within a sprint team, team members try to focus on completing tasks from the sprint backlog. One by one, team members select tasks from the backlog and each task within a sprint goes through different stages like “To Do (Sprint Backlog)”,“In Progress”, ”Review”, “Testing”, and finally “Done”. Each team member pulls a task from the “To Do” list and put that in the “Progress” queue. And when the task is completed, it is moved to the “Review” stage where some other team member will review the completed task. In case the reviewer is satisfied with the task he’ll move it forward towards the testing stage, otherwise in case the reviewer has some comments related the particular task then the task is not moved forward unless those comments are taken care of by the developer. Once the task is reviewed, it is moved forward to “Testing” stage where quality assurance team members test it on their development setup. In case the task passes the acceptance criteria, it will be moved to “Done” state, otherwise task will be moved back to “Progress” state.

.

In addition, every day, team will have daily scrum (morning stand-up meeting) meeting which usually last for 15 minutes. In this meeting each team member describes what he did yesterday? What he is planning to do today? And is there any impediment that needs immediate attention?

*Strength***:**

- Sticking to two weeks iteration is good because it helps a team to roughly predict how much work they can complete within a sprint, usually based on work completed in last five sprints.
- Daily scrum meeting is very good to keep team focused because it ensures that each team member describes what he did yesterday and what he is planning to do today. In order to answer these questions, team members ensure that they stay focused. Also this meeting helps in resolving work stopper impediments immediately.
- Sprint keeps the team focused all the time. Since the team has to fulfil its sprint commitments, there are no dull phases of development which usually may happen in traditional waterfall models.
- Reviewing each task within a sprint is very helpful in finding problems at the early stage of development. It also it ensures the quality of the code because someone else has reviewed it.

Weakness:

- Within a sprint, sometimes scope of the story is changed by the product owner. This leads to delay in completing tasks and at the same time may result in unfinished tasks by the end of a sprint.

- Also many times there is pressure from the product management or the top management to take new tasks in the middle of the sprint in addition to the ones committed by the team during planning. What happens is this usually starts a chain reaction; the flood of additional tasks and changes suddenly break the sprint flow, and for the next couple of sprints process changes to Kanban mode.
- In Kanban mode, team simply processes tasks one by one either from the product backlog or the re-prioritized sprint backlog tasks. There are no policies followed like for instance, rules related to maximum limit on tasks in a particular stage. As a result, often tasks are still stuck in some stage of the workflow when the sprint is about to end.
- Switching between the Scrum and Kanban process modes also reduce team's motivation, and eventually slow the development process.
- Sometimes tasks also get stuck in the review stage of the workflow when many tasks are simultaneously moved to this particular stage. Often these tasks may stay there when team members are too busy with other tasks, and no one has time to review tasks.
- Sometimes product owners add new requirements (in addition to sprint tasks), or new user interface (UI) design changes in the middle of the sprint because of customer or management pressure. Even if the developers try to put in extra efforts, this usually leads to many tasks in the "Testing" stage simultaneously. Some of these tasks may remain incomplete by the end of a sprint, even the ones where there was some possibility of getting completed before.
- After interviewing the quality assurance (QA) team, it seems sometimes there is disconnect between the development team and QA. They simply do not tune in or match each other's expectations, QA and others tend to think that team does not produce quality code in terms of maintainability and documentation. Development team and QA will need to bridge these gaps, trust and work as single team. Further, they need to have common goals, shared understanding of how work needs to be done, and also agree on common QA standards.

Sprint Demo: After the sprint ends, team prepares for the demonstration of the sprint tasks to the scrum product owners, product management, designers (Optional), and sales (Optional) team. In this meeting, the main highlights are the stories or tasks which team has completed in the sprint that ended. The team starts with what was the sprint goal, what they managed to achieve (committed vs completed stories/tasks), metrics like sprint velocity, and followed by demonstration of stories/tasks.

*Strength***:**

- Sprint Demonstration is good way of getting the feedback from the product management, designers or other teams.
- Product Management knows about the progress of development team and can accordingly plan or prioritize future product requirements

Weakness:

- Sometimes management seems to compare team's performance based on story points completed within a sprint (i.e. team velocity). Rather than using sprint velocity as a performance indicator of a team**,** velocity shall only be used for planning the capacity of sprints.

Sprint Retrospective: During sprint retrospective, team retrospect on sprint performance and lessons learned, every team member discusses about negative and positive things related to the sprint. Based on negative points, team tries to create action plan that would help them in improving the next sprint. Every team member writes what went right and what went wrong, then they discuss on how to improve upon in the next sprint and what are the points where they need management's intervention. Some action points which cannot be handled by the team are discussed in retrospective of retrospectives, where all the scrum masters and product owners of different teams discuss what can be improved at the company level.

*Strength***:**

- Sprint retrospective is very important tool that helps in achieving constant improvement.

Weakness:

- Sometime when team members pin point too many problem areas it is very difficult to take action points related to all. One improvement could be restricting the number of issues team members can raise, so that concrete actions can be taken related to those rather than trying to improve everything at once.
- Many times some action points are hardly handled because managers or team members are too busy with sprint tasks.

- Sometimes there is no right owner who can handle the action points from management side.
- Product owners are not part of sprint retrospectives even if they are important team members. They can handle many actions points from the sprint retrospective which are related to top management or applicable company wide.

Product Backlog Grooming: During the sprint, once before the start of next sprint team members along with scrum product owners do the backlog grooming in order to prioritize and foresee what stories/tasks are coming in the next sprint. Optionally, product managers are also part of the meeting in case they want to bring in some additional stories or tasks. Team's product owner adds new stories or tasks and discusses those with team members. If possible team members can also give some rough estimation in terms of story points. This meeting usually lasts for an hour.

Strength:

- Product backlog grooming helps the team to foresee future tasks & stories.
- It helps to keep product backlog in order and prioritized.
- In case product backlog is in order, it simplifies sprint planning sessions.

Weakness:

- This meeting doesn't solve its purpose in case product owner has not done its homework, sometimes product owners are not sure about the scope of tasks or the tasks itself.
- Many times product owners are not sure of what new user stories are coming from the management, customers or designers, and as a result this meeting becomes fruitless. In this case, team starts to shift to Kanban mode when they do not have enough tasks to select from in the sprint planning meeting.
- Sometimes the product owners and product managers are not synching the product backlog effectively well in advance of the sprint planning. Many times sprint planning happened before the product council meetings (product council meetings within the case company are used for defining sprint milestones and product milestones).Some cycle or structure is needed to ensure product backlog is in order and prioritized well ahead of sprint planning or team backlog grooming meetings.

Product Release Testing: Once the sprint ends, quality assurance team creates a new product release based on the work completed by many teams, and henceforth start testing the product release on staging servers. In case some features or user stories are not working as expected, bugs are created for the development team, which they will need to fix in the release branch provided they are blocker/critical bugs. Rest of the bugs are added to team’s product backlog.

Strength:

- Release testing is very important because it is for the first time where sprint related work is tested on the staging servers, a setup which is similar to production release environment.
- Release testing helps to keep a constant check on product quality.

.

Weakness:

- Sometimes bugs detected during release testing are not prioritized properly as far as their criticality is concerned. This means if lots of bugs are treated as critical, team will need to work on those in the current sprint in addition to tasks which were planned as part of sprint planning. This usually may lead to incomplete tasks by the end of a sprint.
- Bugs detected in a particular release branch usually need to be fixed in that release branch, and in case there are more than one active release branches (e.g. release in production is also getting patches and a different release in the staging environment needs bug fixes as well). This often leads to software merging issues when team merges software components from two different release branches to the main trunk.

3.4 Current Product Development Process Chart

The current product development process being followed in the Applications Development Team within the case company is depicted by the Figure 3

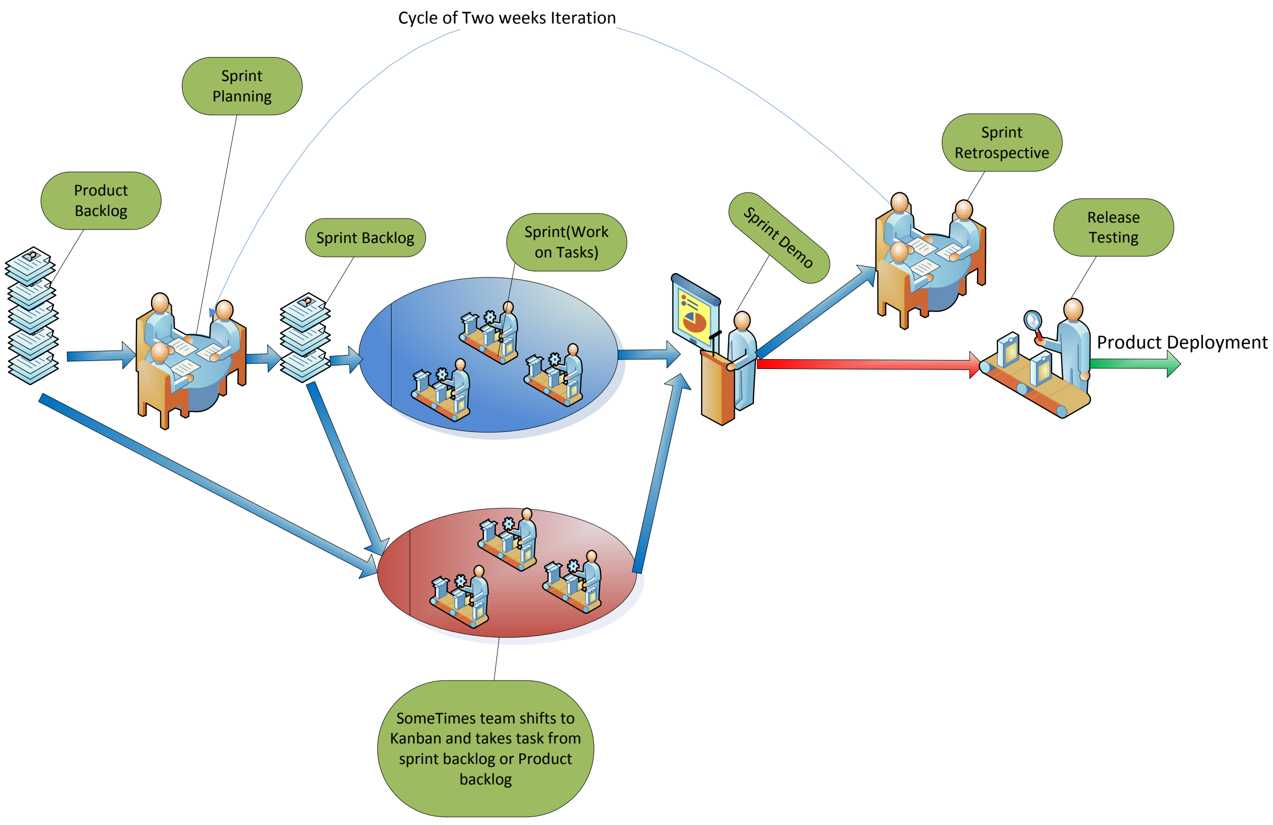


Figure 3. Software Product Development Process in case company context

As a summary team is facing many challenges while using scrum as the product development methodology, and in certain situations team is switching back and forth to use pseudo Kanban approach which is not completely Kanban either. The good and the bad practices in the current process are further summarized in the next chapter.

3.5 Summary

Overall, the good and the bad practises of the current product development process within the case company are summarized in Table 6.

Stage	**Good Practices**	**Bad Practises**
Sprint Planning	Sprint planning in case product backlog is in order and complete.	Not prioritized stories/tasks in product backlog, incomplete stories in the product backlog.

	Using whole team to estimate the complexity of the task in the sprint planning	More focus on new features than backlog bugs in the sprint planning
		Implementation/design of tasks is not considered in sprint planning.
		Grooming of product backlog between the product owners and product management is missing in the whole process, and this can lead to incomplete stories/tasks or wrongly prioritized product backlog.
		Misusing story points by comparing it to number of days or man days.
Sprint	Sticking to well defined iteration duration helps better prediction of Sprint Estimations in future based on past data	Changing scope of a story or task in the middle of a sprint may lead to delays or unfinished tasks.
	A smooth sprint keeps team motivated all the time.	Adding more tasks in the middle of a sprint (In addition to tasks committed by the team during sprint planning) breaks the whole scrum and teams starts to work in Kanban mode.
	Reviewing tasks improves quality of code	Whenever team shits to Kanban mode, they hardly follow any rules related to limit of tasks in progress, or review or testing stages.
	Daily scrum stand-up meet-	Switching between scrum

	ing ensures team stay focused every day. This meeting also helps in resolving any work stopper impediments.	and Kanban modes also reduces team's motivation and eventually slows the development process.
		Too many tasks stuck in particular stage of development leads to unfinished tasks at the end of a sprint.
		No rules are followed while adding some new tasks in the middle of a sprint by product owners.
		Disconnect between the development team and quality assurance team as far as code maintainability and documentation is concerned.
Sprint Demo	Team gets feedback from the product management, user interface designers and others.	Using sprint velocity as the indicator of team's performance.
	Product management gets the first hand details of team's progress. .	-
Sprint Retrospective	Helps team in achieving constant improvement.	Too many action points without right owners to handle those.
		Management/team too busy to handle action points.
		Not involving product owners in the sprint retrospectives. They are the first class team members and can provide

		quality inputs on teams sprint performance and outputs
Product backlog grooming	Helps to foresee future tasks & stories.	Doesn't help if product owner has not done its homework about scope of tasks, requirements, etc.
	Helps to keep product backlog in order and prioritized.	Incomplete stories or tasks.
	Simplifies sprint planning in case Product backlog is in order.	Not effectively synching the grooming meeting ahead of sprint planning.
Release Testing	Tasks are tested on a staging setup which is similar to production release setup	Prioritization of bugs shall be improved.
	Helps to keep a constant check on product quality	Not to have more than one active release branch.

Table 6. Summary of best and bad practices in Scrum with respect to case company context

4 Best Practices of Agile Product Development Processes

This section discusses the main features of the product development processes based on different agile methods. According to Wikipedia (2014), "agile methods are focused on different aspects of the Software development life cycle. Some focus on the practices (For example, XP, Pragmatic Programming, Agile Modeling), while others focus on managing the software projects (For example, Scrum)". Even though a lot of information is available on different agile methodologies, the researcher in this study only concentrated on agile methods which focus on managing the software projects. Accordingly, three main methodologies Scrum, Kanban and Scrumban were studied by the researcher. Further, this section also tries to compare these three agile methods based on the literature study and existing best practices.

4.1 Scrum

Scrum is one of the most commonly used agile software development approach since last 10 years. “Scrum is founded on empirical process control theory, or empiricism. Empiricism asserts that knowledge comes from experience and making decisions based on what is known. Scrum employs an iterative, incremental approach to optimize predictability and control risk.” (Schwaber and Sutherland, 2013). In Scrum model, an organisation is divided into small self-organizing teams with sizes ranging from 4 to 10 people. According to the scrum practice, a scrum team should be self-organized and cross functional, and it should have all the needed competencies to accomplish the project without the need for external competencies. The Scrum framework consists of scrum teams and their associated roles, events, artefacts, and rules. Scrum prescribes four formal events Sprint Planning, Daily Scrum, Sprint Review and Sprint Retrospective. Further, the Scrum Team consists of a Product Owner, the Development Team, and a Scrum Master (Schwaber and Sutherland, 2013).

Every team has a product owner who is responsible for creating user stories based on high level customer/business requirements in a queue called product backlog. Product backlog is a dynamic list of requirements and is the only source for storing requirements and is maintained throughout the life cycle of product development. The product owner is the sole person responsible for managing the product backlog, and the responsibilities include clearly specifying product backlog items, prioritizing the items, ensure the backlog is clear to development team, and direct the scrum team on what to work next (Schwaber and Sutherland, 2013).

In the Scrum model, product development is done in small iterations called Sprints, facilitated by a person called Scrum Master. Scrum master is responsible for making sure that the scrum team lives by the values and practices of scrum. Sprint iteration duration usually ranges from 2 to 4 weeks; however the duration is fixed within a particular team or an organisation. At the beginning of each sprint, teams have a sprint planning meeting where they usually commit to a sub set of prioritized tasks from the product backlog. This subset of tasks is called Sprint Backlog and is managed during the lifetime of sprint iteration. In the sprint planning session, team members try to estimate the effort needed for stories in the backlog. Scrum does not prescribe a single way for teams to estimate their work. However, it does ask that teams not estimate user-stories/requirements in terms of time, but, instead, use a more abstracted metric to quantify effort. Common estimating methods include numeric sizing (1 through 10), t-shirt sizes (XS, S, M, L, XL, XXL, XXXL), the Fibonacci sequence (1, 2, 3, 5, 8, 13, 21, 34, etc.). The product owner needs these estimates, so that he or she is empowered to effectively prioritize items in the backlog and, as a result, forecast releases based on a sprint velocity. Velocity of a sprint is the total number of story points completed within the sprint iteration. This means the product owner needs an honest appraisal of how difficult work will be. Even when the team estimates amongst itself, actions should be taken to reduce influencing how a team estimates. As such, it is recommended that all team members disclose their estimates simultaneously. Because individuals “show their hands” at once, this process is like a game of poker (Scrum Methodology, 2014).

During the scrum sprint, team members pull the tasks from sprint backlog and start working on those. Every day team members will have a short stand-up scrum meeting called Daily Scrum, which is facilitated by scrum master where every team member answers three main questions: What he has done since last meeting? What he is planning to do today? Are there any impediments stopping him to do his work? Throughout the sprint, team members need to be self-organised and ensure that no impediments are stopping their work. In scrum, whole team is responsible for completing the committed stories by the end of sprint.

At the end of each sprint, a Sprint Review meeting is held to inspect the sprint targets and accordingly adapt the product backlog. In this meeting, attendees include scrum team, product owner and the key stake owners invited by the product owner, For instance, the key stakeholders may include top management, sales team, or customers. Team members demonstrate the completed stories to product owner as well as key stake holders, and accordingly answer question related to the sprint increment. The entire group collaborates on how sprint can be improved and what can be done next so that valuable inputs are provided to the subsequent Sprint Planning. Usually, the result of the Sprint Review is a revised Product Backlog that defines the probable Product Backlog items for the next Sprint (Schwaber and Sutherland, 2013).

Sprint Review Meeting is intentionally kept informal so that it doesn’t become a burden on team members, rather it shall be a natural result of a sprint. During the sprint review, the project is assessed against the sprint goal determined during the sprint planning meeting. Ideally, the team has completed each product backlog item brought into the sprint, but it's more important that they achieve the overall goal of the sprint (Cohn, 2012). There are circumstances when sprints can be cancelled all together and in such a case all incomplete Product Backlog Items are re-estimated and put back on the Product Backlog and the ones which are completed already are reviewed and marked as done. According to Schwaber and Sutherland (2013),

> A Sprint can be cancelled before the Sprint time-box is over. Only the Product Owner has the authority to cancel the Sprint, although he or she may do so under influence from the stakeholders, the Development Team, or the Scrum Master. A Sprint would be cancelled if the Sprint Goal becomes obsolete.

After the end of a sprint, the last thing team members do is reflect on how things went inside the last sprint so that they can improve upon shortcomings and continue to do good things in upcoming sprint. This is usually done in a meeting called Sprint Retrospective Meeting. The Sprint Retrospective occurs after the Sprint Review and prior to the next Sprint Planning. The main purpose of this meeting is to constantly look for improvement opportunities and retrospect on how team can do things in a better way in future. According to Schwaber and Sutherland (2013), the purpose of sprint retrospective meeting is,

> To inspect how the last Sprint went with regards to people, relationships, process, and tools; Identify and order the major items that went well and potential improvements; and, Create a plan for implementing improvements to the way the Scrum Team does its work.

In summary, Scrum model provides a framework and tools for software development and believes in empowering the team to take its own decisions during small iterations called Sprints. The whole team works in collaboration and commits to tasks taken inside a sprint which they finally demonstrate at the end of a sprint, and at the same time team constantly try to improve by taking on learnings from sprint to sprint. Further, this model defines certain roles to facilitate and manage the work in a team.

4.2 Kanban

Kanban model is known to have originated from Toyota production system. Kanban is a Japanese word and literally means “signboard”. When used in manufacturing, it is a production control system, aimed to have just-in-time production and making full use of workers capabilities (Sugimori et al., 1977). It is claimed that Kanban is one of the important models which execute lean thinking in practice (Chai, 2008). Kanban system drives project teams to visualize the workflow, limit work in progress (WIP) at each workflow stage, and measure the cycle time (that is, average time to complete one task) (Kniberg, 2009). Kanban has been used in manufacturing since decades however; it is relatively a new concept in the area of software development. Using Kanban model, the workflow in software development project is visualized using a board called Kanban board. Kanban board usually is a white board however recently many software/electronics tools are also used to represent Kanban board. On the Kanban board, work items (usually user stories) are represented using Kanban cards, the most commonly used Kanban cards are sticky notes. This board consists of several columns with each column representing a work flow stage of the development process. The number of work items in each column is limited in order to manage the workflow. As a result, developers concentrate on the work items in progress and try to complete those before starting working on new work items. When a work item is completed in a particular stage, it is moved to the next column and a result some other work item can be pulled from the previous column. A simple representation of the Kanban board is depicted by the Figure 4

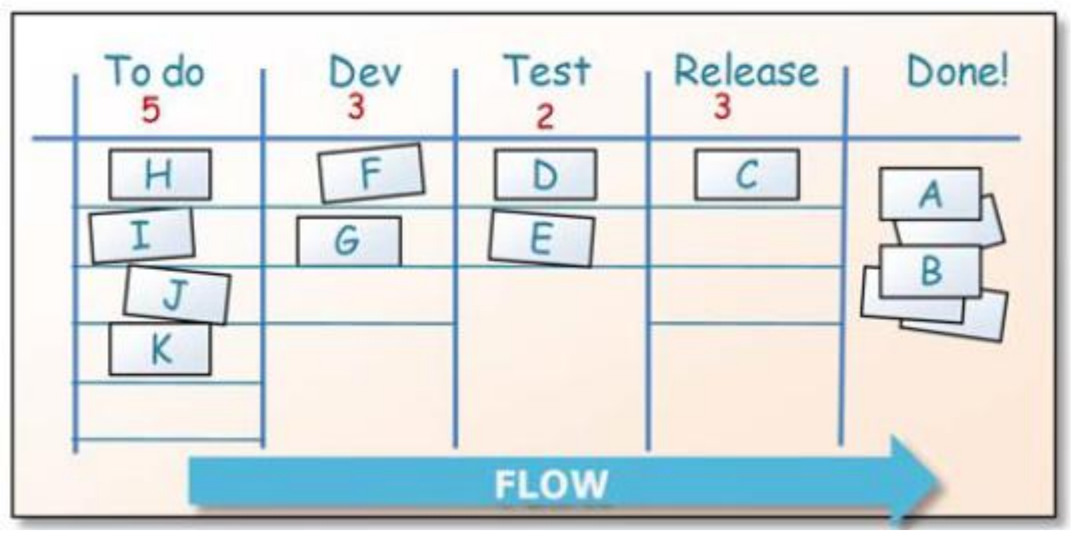


Figure 4. A simple Kanban board (Kniberg and Skarin, 2010)

Kanban methodology is based on principles of visualizing the process workflow, limiting work in progress and controlling the lead time.

Process Visualization: means representing workflow stages on a board called Kanban board. It serves as a visual control mechanism, indicating how work flows though different stages of the software development process. This visualization is done by dividing the board into different columns indicating different work flow stages. Further, tasks or user stories are usually represented by sticky notes, which are moved from one column to another as the work progresses from one stage to another. Visualization helps team to know the progress of each task. In addition, if some stage of the work flow becomes a bottleneck, it prompts the development team to handle it immediately (Mahnic, 2013).

Limiting Work in Progress: Kanban puts a maximum limit on the Work in Progress (WIP) items of every stage within the process workflow. Its ideology is based on the fact that something new shall not be started unless an existing piece of work is delivered or pulled by a downstream function. New work is only pulled into the system when there is a capacity to handle it rather than being pushed into the system from outside. This mechanism is called “pull” mechanism. WIP limit defines the capacity of each stage of workflow in terms of number of work items that can be in progress in a particular stage of the software development process. This pull based system ensures sustainable pace without overloading different stages of the workflow. Further, it significantly reduces lead time, which is used as a major measure of development team’s throughput and productivity (Mahnic, 2013).

Measure Lead Time: Lead Time is one of the important metrics in Kanban and measures average time to complete one work item. In Kanban, the idea is to optimize the process to make the lead time as small and predictable as possible (Kniberg, 2009). Development team focuses on improving the lead time so that they can easily predict how many tasks can be completed within a particular period of time, and further improve the overall process so that lead time is reduced over the time, which ultimately improves the team’s output performance.

In Summary, Kanban focuses on visualizing the work flow, pulling the work when capacity allows it, improving the workflow process by limiting the amount of Work in Progress (WIP) items in different stages, and finally improving the process further in order to have small and predictable lead times.

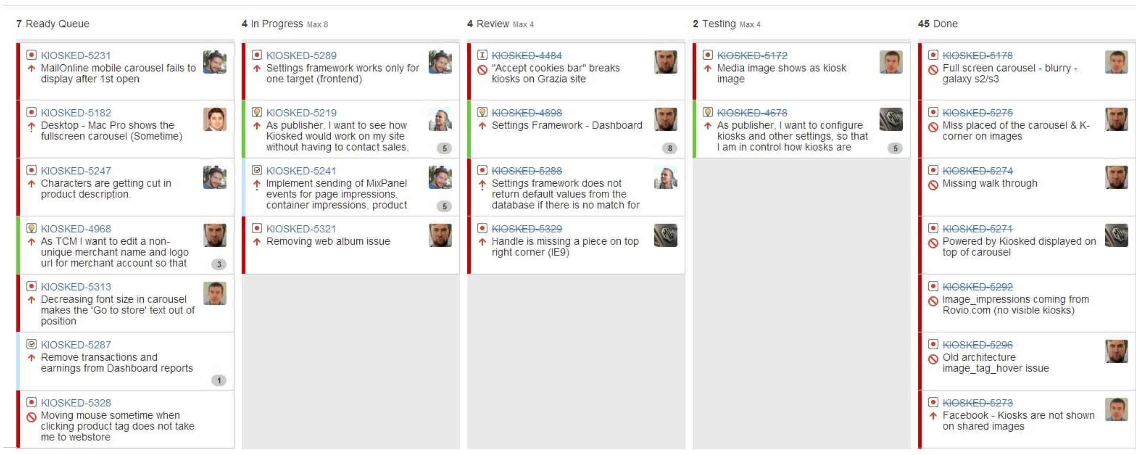
4.3 Scrum-ban

Scrum-ban is the combination of Scrum and Kanban and tries to use features from both the software development models. On one hand, it uses the prescriptive nature of

Scrum to be agile; on the other it encourages the process improvement of Kanban to allow teams to continually improve their process (Pahuja, 2012). The term Scrumban was first used by Ladas (2008) in his whitepaper on ‘Scrumban-Essays on Kanban Systems for Lean Software Development’. On one hand Scrum model has helped the software development teams to self-organise, collaborate, improve efficiency constantly, work in small iterations, and avoid management overhead, applying lean methods like Kanban can extend these benefits. A lot of literature is available on the web related to Scrumban, and there are two thoughts of people; some apply Scrum to Kanban where process is more inclined towards Kanban and others apply Kanban to scrum where process is more inclined towards Scrum. Irrespective of this, both versions seems to take certain principles from Scrum and Kanban and accordingly adjust those to their organisation/team needs and requirements. Following are the core principles of Scrumban:

Visualize the workflow: This is one of the most important tools taken from Kanban and applied to Scrumban. Visualizing workflow literally means team visualizes different phases their Product backlog Items (PBIs) or stories go through starting from the sprint backlog and ending in the done phase, on a white board. Even though there are lots of digital and online tools that can be used to visualize the work flow, many Scrumban users still prefer white boards because they are easy to manage and easy to change. In a normal Scrum, team usually starts from the sprint backlog and works on those items and finally moves them to the done stage. However, in Scrumban idea is to visualize the flow of work into and out of the sprint (Yuval, 2012). Once the visualizing of workflow is achieved, it helps team including product owners to know the bottleneck areas within the workflow. Further, visualizing helps in knowing who is working on what tasks, and what is progress state of different stories at a particular point of time.

Visualizing on white board or any digital tool is done by dividing the whole board into different stages represented by columns. And then digital PBI’s/stories or sticky stickers representing the PBI’s/stories are moved around those columns starting from the backlog stage on the left side of the board (or digital tool) to the done stage on right side of the board (or digital tool). A typical visual workflow representation in Scrumban using some software tool like Jira is shown in Figure 5.


7 Ready Queue

KIOSKED-5231
MailOnline mobile carousel fails to display after 1st open

KIOSKED-5182
Desktop - Mac Pro shows the fullscreen carousel (Sometime)

KIOSKED-5247
Characters are getting cut in product description.

KIOSKED-4968
As TCM I want to edit a non-unique merchant name and logo url for merchant account so that
3

KIOSKED-5313
Decreasing font size in carousel makes the 'Go to store' text out of position

KIOSKED-5287
Remove transactions and earnings from Dashboard reports
1

KIOSKED-5328
Moving mouse sometime when clicking product tag does not take me to webstore

4 In Progress Max 8

KIOSKED-5289
Settings framework works only for one target (frontend)

KIOSKED-5219
As publisher, I want to see how Kiosked would work on my site without having to contact sales,
5

KIOSKED-5241
Implement sending of MixPanel events for page impressions, container impressions, product
5

KIOSKED-5321
Removing web album issue

4 Review Max 4

~~KIOSKED-4484~~
"Accept cookies bar" breaks kiosks on Grazia site

~~KIOSKED-4898~~
Settings Framework - Dashboard
8

~~KIOSKED-5288~~
Settings framework does not return default values from the database if there is no match for

~~KIOSKED-5329~~
Handle is missing a piece on top right corner (IE9)

2 Testing Max 4

~~KIOSKED-5172~~
Media image shows as kiosk image

~~KIOSKED-4678~~
As publisher, I want to configure kiosks and other settings, so that I am in control how kiosks are
5

45 Done

~~KIOSKED-5178~~
Full screen carousel - blurry - galaxy s2/s3

~~KIOSKED-5275~~
Miss placed of the carousel & K-corner on images

~~KIOSKED-5274~~
Missing walk through

~~KIOSKED-5271~~
Powered by Kiosked displayed on top of carousel

~~KIOSKED-5292~~
Image_impressions coming from Rovio.com (no visible kiosks)

~~KIOSKED-5296~~
Old architecture image_tag_hover issue

~~KIOSKED-5273~~
Facebook - Kiosks are not shown on shared images

Figure 5. Visual workflow representations in Scrumban using some software tool

An example visual workflow representation in Scrumban using whiteboard is shown in Figure 6.

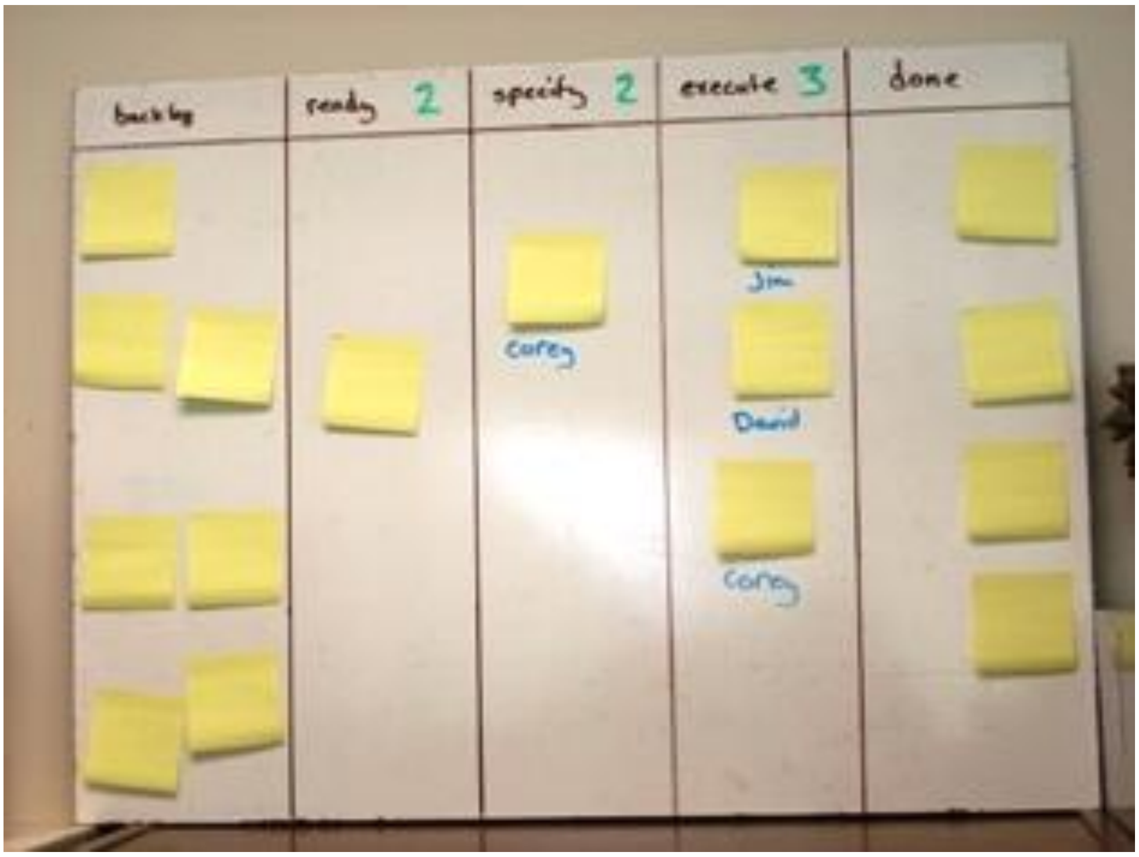


Figure 6. An example visual workflow representation in scrum-ban (Ladas, 2008)

Pull Work: In Scrumban, the work is pulled as and when needed into a queue unlike the approach used in a traditional Scrum where all the work to be completed within a sprint is assigned in the beginning of the sprint to the sprint backlog. This backlog queue is different than normal sprint backlog because it can be updated many times within an iteration as and when required. According to Ladas (2008), one enhancement in Scrumban that can enable teams to decouple the process of assigning work from the process of prioritizing work is to add a ready queue in between the backlog and work in progress queue. This ready queue contains items that are pending from the backlog, but have high priority. In this queue no tasks are bound to any individuals, but as soon as somebody becomes available, they should take one of these tasks instead of picking something out of the general backlog (Ladas, 2008). Ideally ready queue shall not contain too many tasks which will ensure that team focus on just next prioritized PBIs. Also from one stage to another work is pulled to the next step once there is a capacity to do so. This is different from the scrum where work item is pushed to the next step in the process once some stage is completed (Strange, 2013).

Limit Work-In-Progress (WIP) Items: One of the important aspects of Scrumban is to apply limits to the work in progress items at every stage based on team capacity. In scrum context, it means limiting the Product Backlog Items (PBIs) that are in progress at any point of time, including the sprint backlog. The idea is to keep team focused on completing work at hand rather than starting a new task. This means once a limit is reached within a particular stage of the workflow, rather than starting working on something new, it is time to help someone else within the team. This will ensure that the team’s work flow becomes smoother and no stage becomes a bottleneck. Further, this is one of the important aspects of scrumban that helps team to achieve real collaboration and a smoother workflow. Once a limit is reached in some stage of the workflow, team members must prefer helping others rather than starting a new task (For example, developers help developers, developers help testers, testers help product owners, etc.). Down the road once team observes improvements in their capability and collaboration they can further tighten the limits to catalyse even more improvement in capabilities. Limiting WIP is a great way to drive real collaboration at the team level (Yuval, 2012). Setting limits also include setting multitasking limits for individuals; team can have rules to like,

> Prefer completing work to starting new work, or you might express that as a rule that says: try to work on only one item at a time, but if you are blocked, then you can work on a second item, but no more (Ladas, 2008).

Just that people can work on more than one task at a time does not mean everybody shall work on more than one task at a time. So it makes sense to sets limits to "Progress" queue in such a way that it allows for some team members to work on more than one task but not all. For instance, if there are five members in a team, a limit of 8 to the Progress queue will disallow everyone to work on more than 2 tasks.

Make Team Rules Explicit: In tradition scrum, the idea is that teams are self-organized and they will work and co-ordinate themselves, however in practice there are always gaps between how a team shall organize themselves and how things are working out. "Self-organised teams cannot work if they don't have shared understanding of how work is done" (Yuval, 2012). According to Business Dictionary (2014), the definition of policy is,

> A set of policies are principles, rules, and guidelines formulated or adopted by an organization to reach its long-term goals and typically published in a booklet or other form that is widely accessible.

However, as per Heuvel (2011), policies in Scrum-ban/Kanban are;

> Mechanisms, rules or processes that govern how a system works. By making these policies explicit, it becomes easy for others observing the system to understand it, and for those inside it to continually evaluate and improve the current mechanisms where necessary.

In Scrumban, team rules or in simple terms process is made explicit so that everyone in the team is empowered how to manage flow, self-organize and coordinate in order to achieve smoother workflow. Making team policies explicit will help the team members manage themselves, make quicker decisions without putting much effort into thinking, and even reduce the likelihood of giving in to special requests under stress. These policies related to process are something what team decides upon, some policies can even be organisation related. And mainly policies are trying to address recurrent situations, where someone needs to make a decision on how to proceed or what to do if such a situation arises.

Planning Meetings: Unlike Scrum, Scrumban has shorter planning meetings in order to update the backlog queue as and when required. According to Ladas (2008), "The planning can still happen at regular intervals, synchronized with review and retrospective, but the goal of planning is to fill the slots available, not fill all of the slots, and certainly not determine the number of slots. This greatly reduces the overhead and ceremony of iteration planning".

Team shall always plan for shorter period ahead. Having longer planning meetings does not make sense in case the priorities often change. And also, since the team pulls work into a small ready queue before pulling it into work in progress, then from the product owner's as well as team's perspective iteration backlog shall just contain prioritized stories which the team shall work on next. Therefore, "the ideal work planning process should always provide the development team with best thing to work on next, no more and no less*"* (Ladas, 2008).

In Scrumban, WIP limit is set to the sprint backlog queue as well. Therefore, assuming that in the planning meeting, team pulls fixed number of tasks to work on into the backlog queue. And in case the backlog queue is about to empty, team can decide to have a next planning session, in order to fill the queue with the next prioritized list of tasks. This way team is not filling the whole iteration capacity with tasks and planning meetings will be shorter. At the same time product managers will be able to respond to changing requirements (or event driven requirements) quickly compared to normal scrum where tasks are locked for the whole sprint duration. Further, according to Ladas (2008)

> Scrum styled time-boxed planning usually provides a much bigger backlog than what is strictly necessary to pick the next work item, and as such, it is unnecessary inventory and therefore unnecessary waste. Once you've broken up the time- box, you can start to get leaner about the construction of the backlog. Agility implies an ability to respond to demand. The backlog should reflect the current understanding of business circumstances as often as possible. This is to say, the backlog should be event-driven. Time-boxed backlog planning is just that, where the event is a timer, but once we see it that way, we can imagine other sorts of events that allow us to respond more quickly to emerging priorities.

Therefore event driven, time-boxed backlog planning, will make process leaner and introduce ability to respond to demands faster. This is one of the major improvements in Scrumban compared to Scrum model.

Review, Retrospectives and Daily Stand-up meetings: These are the very important ceremonies Scrumban retains from Scrum. Review provides the team with the direct feedback from product owners and the team's key stakeholders such as product managers and customers. Usually customers or product managers prefer to have this meeting at regular intervals like in Scrum.

Scrumban Retrospective is the place where; team can improve upon their team rules, improve overall process, constantly look for improvement opportunities and retrospect on how team can do things in a better way in future, and define ideas to experiment within upcoming iterations.

Daily stand-up meetings keep team members up to date on who is working on what, coordinate activities, know and manage impediments in order to keep the work flow smoother. Again, the main idea is to manage the workflow and use the stand-up meetings as a daily platform to remove impediments.

Metrics and optional estimations in scrum-ban: In Scrum, PBI's are estimated using metrics like story points and number of tasks taken into the sprint is done based on average team velocity of last few sprints. Even though velocity gives idea to the product owners how much team can complete within a sprint, and according he/she can prioritize and make release plans, the problem with this approach as mentioned by Gambell (2013) is

> Often metrics are abused by managers and business stakeholders who want to unnaturally simplify a complex process into a one-dimensional number. Velocity, the amount of story points a Scrum team completes in a single Sprint, is such a metric that incentivizes lower quality at the end of a Sprint as a team scrambles to finish every last story they committed to. When the number fluctuates, as is common with a newer team, the stakeholders begin to question the outputs of the team, and even the effectiveness of Agile itself.

Therefore rather than estimating each and every story within an iteration, why not divide the stories into similar size items. And accordingly team can decide on selecting the fixed number of prioritized tasks into the backlog based on statistical analysis of the work items completed in past. Hence, Scrumban prefers metrics like cycle time and lead time over velocity calculation. According to Ladas (2009),

> Lead time clock starts when the request is made and ends at delivery. Cycle time clock starts when work begins on the request and ends when the item is ready for delivery. Cycle time is a more mechanical measure of process capability. Lead time is what the customer sees. Lead time depends on cycle time, but also depends on your willingness to keep a backlog, the customer's patience, and the customer's readiness for delivery. Another way to think about it is: cycle time measures the completion rate; lead time measures the arrival rate. A producer has limited strategies to influence lead time. One is pricing (managing the arrival rate); another is managing cycle time (completing work faster/slower than the arrival rate).

A team shall focus on average size of items in the backlog. A statistical analysis of all tasks in the project can yield a mean cycle time and standard deviation, which can be very useful planning tool (add up the number of stories and multiply by mean cycle time.) for how many stories can be completed in certain iteration. And if the cycle time is under control, based on average cycle time team's capacity can be balanced against the demand, which in other words will control the lead time as well.

In nutshell, scrum-ban is a methodology which makes scrum leaner and flow oriented. It empowers team, help them to collaborate and organize by utilizing Kanban tools like visual workflow board, WIP limits at every stage of development, team rules, focusing on improving cycle times rather than estimations, etc. Further, it makes scrum flexible towards change by my having shorter planning sessions, avoiding planning for whole iteration, avoiding unnecessary estimations, late binding of tasks, pulling work than pushing, and all this is synchronized within important scrum ceremonies like sprint planning, sprint review, sprint retrospectives and daily stand-up meetings.

4.4 Comparison of different agile methodologies

The comparison of three main agile methodologies Scrum, Kanban and Scrum-ban is described in Table 7. Further the data in this table is also composed from other references marked with a, b, c and d in case some other author has specified a different behaviour for that particular point. These references a, b, c and d are defined at the bottom of the Table 7.

	Scrum	**Kanban**	**Scrum-ban**
Workflow Visualization	Partial Workflow visualization(Backlog, Progress and Done) (As per a)	Full Visualization(Granular Process) (As per a)	Full Visualization (As per a)
Backlogs	Sprint Backlog, Product Backlog	Backlog with limits	Backlog with limits
Limit WIP's	No limit on different stages within a sprint- it is a black	WIPs limited directly per workflow state (As per c)	WIPs limited directly per workflow state

	box process inside a sprint. WIPs are limited indirectly per sprint (As per c)		
Changes to work scope	Should wait until the next sprint	Added as needed (As per b) Can add items whenever capacity is available (As per c)	Added as needed (As per b)
Roles	Product Owners, Scrum Masters, Team	as needed roles (As per b) Not prescribed usually (As per d)	Team plus as needed roles (As per b and d)
Teams	Recommended cross functional (As per b and c) Must be cross functional (As per d)	Can be cross functional or specialized (As per b) Cross functional teams optional. Specialized team allowed (As per c)	Can be cross functional or specialized
Estimations	Yes (In story points or days)	No, Similar sized work items	No, Similar sized work items (As per b and d) Optional(As per c)
Board	Simple board burn-down chart	Board mapped on the process (As per b)	Board mapped on the process (As per b)

			Board only (As per d)
Iterations	Yes (Sprints)	No, continuous flow) (As per b) Iterations Optional (As per c)	No, Continuous flow (As per d) Not mandatory (continuous flow); could have sprints (As per b)
Teamwork	Collaborative as needed by task	Based on pull approach swarming to achieve team goals (As per b)	Based on pull approach swarming to achieve team goals (As per b) Swarming to achieve team goals (As per d)
Impediments	Addresses immediately	Addresses immediately and team shall swarm to solve the impediment (As per b)	Addresses immediately and team shall swarm to solve the impediment (As per b) Avoided (As per d)
Prioritization	Backlog grooming done by Product Owner	Out of Process. There shall be a prioritized product backlog (As per b) Prioritization is Optional (As per c)	Out of Process. There shall be a prioritized product backlog (As per b)
Ceremonies	Daily Scrum,	None required (As	Daily Scrum other

	Sprint planning, Sprint review, Sprint retrospective	per b) Dynamic planning (As per a)	Scrum related ceremonies if needed (As per b) Depends on iteration decision (As per a) Daily scrum (Planning, retrospective and review as needed) (As per d)
When does it fit?	Product development, Small value adding increments development possible, Requirements are in good shape	Support/maintenance work (operational level)	Product development (unclear vision), Evolving requirements (no clear roadmap), Need to include support/maintenance (event driven) work in the process
Metrics	Use velocity as default metrics for planning and process improvement (As per c)	Use lead time as default metrics for planning and process improvement (As per c)	Velocity is optional, Use lead time as default metrics for planning and process improvement
Scrum board	Reset after every sprint (As per a and c)	Persistent board (As per a and c)	Depends on iteration decision (As per a)

Table 7. Comparison of Scrum, Kanban, and Scrum-ban done by Radics, S (2013)

a) Cagley, T. (2013)
b) Radics, S. (2013)
c) Kniberg, H & Skarin, M. (2010: 50)
d) Pahuja, S. (2012)

4.5 Conclusion and summary of framework

After analysing the literature review, it seems that many problems faced by the Applications Development team within the case company may be fixed by choosing process based on Scrumban methodology. Researcher's choice is based on following reasons:

- Scrumban framework is the Combination of Scrum and Kanban methodologies, and uses good features from both the software development models. On one hand, it uses the prescriptive nature of Scrum to be agile (self-organized teams, self-improvement, constant information flow, etc.) and on the other hand it encourages process improvement using Kanban.
- Scrum ideology is based on spend less time analyzing and estimating work items that may end up as low priority on the backlog. This case is very much valid within the context of case company. The current scrum process is very rigid and Scrumban would provide flexibility to handle event driven priority changes or handle important requirements which need immediate attention rather than waiting for the next sprint.
- Even though Kanban alone may help the team to manage workflow by setting Work-In-Progress limits, and further improve team collaboration by setting the explicit work policies. But these are not the only problems team is facing. Team needs a process which is iteration driven so that sprint releases are in synchronization with other teams. And in addition, scrum ceremonies brings in many additional benefits as evident from the current state analysis, so using scrumban is the most suitable process.
- Scrumban is a framework where it provides the development team with best thing to work on next, no more and no less. As a result, team can focus on what is important next rather than the whole iteration duration.
- Scrum-ban also creates a good structure process wise, Scrum is too strict approach and Scrum-ban gives you added flexibility of Kanban.

Accordingly, the summary of the possible improved product development process based on Scrumban methodology is described by Figure 7

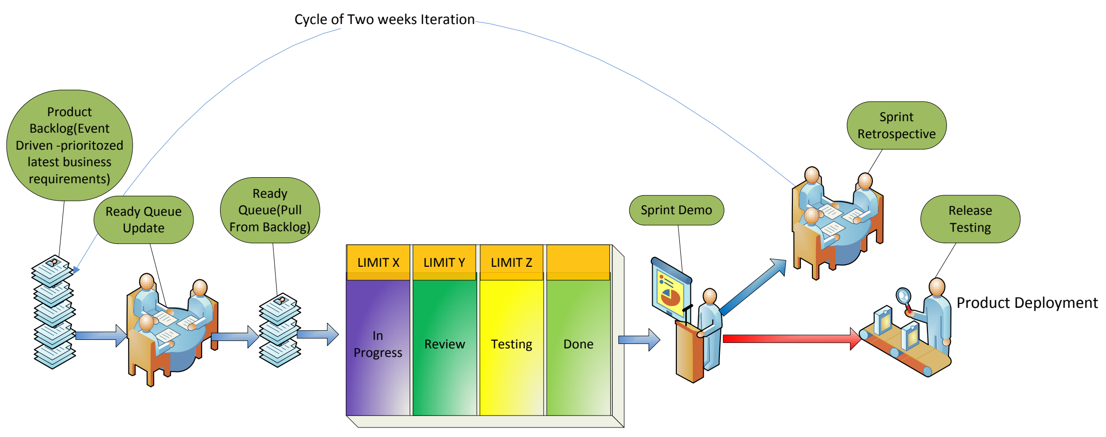


Figure 7. Summary of possibly improved product development process based on Scrum-ban model

Further, the crux of new improved product development process, based on ideas and concepts used in Scrum-ban methodology is summarized below in 7 main points.

.

1. **Pull work and Shorter Planning's:** As per Ladas (2008), "The ideal work planning process in Scrum-ban should always provide the development team with best thing to work on next, no more and no less", therefore avoid planning for the whole sprint. Team shall have shorter planning session in order to fill the slots available in the backlog. These planning sessions will be time-boxed and event driven whenever team capacity is available. For example, whenever there are few tasks in the backlog queue or the queue is about to get empty- team can have next planning session to pull the tasks from the top of the product backlog.

2. **Fixed sized backlog and new Ready Queue:** Instead of Sprint backlog, team will have a fixed sized backlog; basically a backlog with a maximum story limits set. Further, in order to decoupling the process of prioritization and work assignment a new queue called "Ready Queue" can be placed between the backlog and the progress queue. Ready queue shall contain the most prioritized set of requirements/stories which management wants the team to work next as soon as someone completes a task. Ideally this queue shall not contain many tasks; this will ensure that team simply focuses on next prioritized tasks. And

hence, new requirements or priority changes can be done easily to the backlog for ensure leaner process.

3. **WIP Limits:** As seen from the scrumban literature and literature review, Scrumban follows Kanban approach to maintain a flow around backlog by putting limits on Work In Progress Items (WIP's), and accordingly manage the flow. Further, team shall prefer completing tasks in hand rather than starting the new work. This means, every stage of the development process will have a limit in terms of number of work items that can be put to the BACKLOG/TO-DO state, READY state, PROGRESS state, REVIEW state and even TESTING state. By enforcing WIP limits, process will ensure that bottleneck stages are cleared immediately and tasks do not get stuck in a particular state for long.

4. **Explicit Work Policies:** Team need to ensure that work policies or rules are explicit in order to improve collaboration. Team shall have clear definitions of done, how to collaborate team work when some stage becomes a bottleneck, rules followed in different stages of work flow, team rules related to intra-team or inter-team communication that will help to manage the work flow, how to improve communication and further work policies related improving quality of work.

5. **Continue Daily Scrum, Sprint Review, Sprint Retrospective:** Team shall continue to use scrum practices like daily morning scrums, sprint retrospective, sprint review meetings as they constantly help to manage impediments, improve development process and help to get direct feedback from product owners and other key stakeholders.

6. **Avoid estimations:** Team shall have PBI's as small as possible. The best is to have similar sized items which can be completed within the similar if not same amount of time, such as 1 day or 2 days or 3 days. Once team starts getting better in creating tasks based on average size of a backlog item, estimations won't be needed because product owners can guess release times by simply multiplying number of tasks taken into the iteration with the average lead time.

7. **Focus on controlling Lead Time rather than Velocity:** Velocity incentivizes lower quality. In case team cannot complete tasks on time, they may hurry up things in order to maintain the velocity score. Estimations are good, as long as they mean something to the product owners; most often management uses velocity (which is calculated based on story point estimations) to judge team's performance. Better is taking the concept of velocity out, focus on ensuring smoother workflow and controlling the average lead time of a story. Not only will this reduce the stress levels of the team, it'll improve the motivation of the team as well. Further, in normal Scrum, when a team focuses on velocity, they mostly prefer selecting new tasks into the sprint rather than bugs, the main reason being no story points are usually assigned to bugs. When a team will not use velocity for filling the sprint capacity, they will start looking into the bugs as normal tasks and hence quality of code will also improve by fixing bugs.

5 Building new product development process proposal

5.1 New product development process proposal overview

As discussed in the current state analysis, current process which is loosely based on scrum is not good enough to solve problems faced by the team under discussion within the case company context. This team needs a process which can handle event driven requirements as they often are changing within a sprint. Further product management in this team has more pressure to handle urgent requirements from customers and sometimes within the same sprint in order to reduce "go to market" time for certain customer features. Not only that, there are other issues this team is facing related to collaboration, communication and process flow, etc. which were evident from the current state analysis. Therefore, as discussed in the literature review, product development process based on scrumban can help the team to overcome many such issues. Based on the "summary of possibly improved product development process" as discussed in Literature review, section 4.5, qualitative interviews were conducted with many stake holders to get their feedback and fine tune the new process model. The details of the data collection are described in Table 3. The theme of the semi structured qualitative interview is presented in Appendix 1, and the data collected based on those theme based interviews is presented in Appendix 2. Not all the interview questions described in Appendix 1 were relevant for all the interviewees. Based on the expertise and skills of interviewees, the researcher asked different set of questions from different interviewees. After a few interviews, as the knowledge of the researcher increased with respect to the new process and also responses from interviewees started becoming familiar, the researcher added more questions to the theme interview template in order to further understand the problem deeper. The qualitative data was analysed in various stages: reading and listening to interview transcripts, spending time with data and summarizing it, extracting the ideas and concepts and finally looking for some patterns in the data and how they fit in with respect to the theories (Mayor & Blackmon 2005, 348-49). Based on the data collected from the interviews and the best practices from the scrumban methodology, the new product development process was developed.

5.2 New product development process proposal

This chapter will focus on creating the new product development process based on the literature review and feedback received during interviews with different stakeholders and experts within the case company. The details on how the data was collected are described in Table 3, and the summaries of the data collected are described in Appendix 2. The new software development model will be based on Scrumban methodology. In the new process model, some good practices were retained from the current process, and the new process model is further described in different stages below:

Pull process and planning meetings: In the Scrumban model, team will start using shorter planning sessions in order to fill the slots available in the backlog. Unlike a sprint backlog, Scrumban backlog may be updated more than once within a sprint. In Scrumban, as described in literature review, the idea of having iterations is optional. However, after having discussions with different key stakeholders, it was clear that team will continue to use iteration duration of two weeks so that the Sprint demonstrations and Sprint Retrospectives can be held in synchronisation with other development teams. In addition, using the iteration model will also simplify the creation of main software release by the end of every sprint. Further, most of the interviewees were in favour of sprint iteration. Lauri Oherd mentioned on 21 February, 2014 that “sprint cycles helps team to focus; otherwise developers would lose sense of time.”

One of the interviewees, Product Owner of the team, Mika Mannermaa mentioned on 17 February, 2014 that, “It is better to have shorter pull sessions than mammoth sprint planning sessions”. At the beginning of each sprint, team will have a first planning session to fill the slots available in the backlog queue. The size of the backlog queue will be limited to twelve (12) and once the team is about to run out of tasks, it will have another planning session to pull the tasks from the top of the product backlog and fill the available capacity.

Sprint Backlog and Ready Queue: As mentioned in the literature review, one of the ideas in Scrumban is to put a new queue called ready queue in between the backlog and process queue. After having interview and team discussions, a simpler idea came out to have smaller fixed backlog act as a ready queue itself rather than adding a separate queue in between the backlog and progress queue. Quality Assurance team member, Ashish Mahindroo mentioned on 19 February, 2014 that “working on smaller list of prioritized tasks will definitely give flexibility to include changes in priorities within a sprint. So why not have our backlog act as ready queue”. In this way, the team will be only managing one queue and whatever tasks are put in the ready queue (aka backlog) are the ones that the product owner wants team to work on next as soon as someone completes a task. At the same time, the product owner can change the priorities of tasks in the product backlog. The size of the ready queue (aka backlog) was preferred to be limited to twelve (12), which is just double the size of team members, in order to ensure that not too many tasks are put to this queue; this will ensure that product owners can plan anything urgent or important requirements from customers or top management to the next ready queue update within the sprint. In the scrum model, this was not possible as tasks could not be added in the middle of the sprint. As one of the interviewees, Edward Karvinen mentioned on 5 February, 2014 that “we cannot be customer oriented if we are having too strict process. So, our process shall be able to handle the customer driven requirements and their immediate need”. The leaner backlog will ensure that the team focuses on what is important and what needs to be completed next. For product management, this gives added flexibility in a controlled way and hence this new process will be able to handle immediate customer needs within the framework of this process rather than breaking it.

WIP Limits: In Scrumban, maintaining the work flow is important and that is why Scrumban applies Kanban limit to each and every development stage of the work flow. The different stages of workflow with respect to the team under discussion are “To-Do” (aka Ready queue), “Progress”, “Review”, “Testing” and finally “Done”.

For the “Progress” stage, most of the interviewees agreed that it does not make sense for a team member to work on more than one task at a time. Ex-Scrum Master of the team, Maksim Luzik mentioned on 14 February, 2014

> One task at a time is good, technically one can never work on more than one task at a time, but there are situations when someone needs to work on more than one task. Accordingly, we shall give flexibility on working more than one task but not to have that as a general practice. In case a developer started working on another task and he already has an existing task in progress, I would prefer moving the old task back to the TO-DO list.

Based on the interview discussion analysis, a rule of (2xN – 2) was selected to limit the number of tasks inside the Progress queue; here N implies the number of developers within a team provided N is greater than 1. This way if we have 5 developers in a team, task limit for Progress queue will be (2x5 – 2 = 8) eight (8).

For the “Review” stage, this is the stage when a team member completes a task, tests it himself and then sends the code changes for review. For reviewing the code (it may even be a documentation task), one of the rule that was implemented by the team is that every developer will make a patch for the code to be reviewed and attach it to the corresponding task in the task management tool, Jira. Once the task is put to the review stage, some other developer who is available will review his task. One of the interviewees mentioned, “It is good to have some limit for the review stage, though sometimes there might be exceptions”. Many interviewees agreed that setting some limit will definitely ensure the smoother workflow. Others were strict in their approach to solve the problem of tasks getting stuck in the review stage. One such interviewee, Ex-Scrum Master, Maksim Luzik mentioned on 14 February, 2014,

> Whenever a team member completes a task, he shall give first priority to the tasks in the review column, second priority goes to the blocker bugs and then third priority is to take tasks from TO-DO list. Main problem is, developers are not fond of doing something like reviewing a task or testing somebodies task, so as a last resort setting rules will help. If we need to go a bit strict, we shall not have the tasks in the review more than half the number of developers. So if we have 6 developers, limit shall be something like 3. The idea shall be that there is at least one developer to review a task.

Another interviewee, Sr. Software Developer of the team, Perry Mitchell mentioned on 21 February, 2014, “With review limit does not need to be too strict. If needed, team can put the limit equal to the number of developers”. After more data analysis, in the “Review” stage, a rule of (N – 1) was selected to limit the number of tasks inside this queue; here N implies the number of developers in a team provided N is greater than 1.

This way if we have 5 developers in a team, task limit for “Review” queue will be (5 – 1 = 4) four (4).

Once the task is reviewed, the developer will move the task forward to the “Testing” stage. At the same time, the reviewer will ask the developer to commit the task to the main code base called “Trunk”. In the “Testing” stage, Quality Assurance (QA) will test the story/task based on requirements and acceptance criteria, and accordingly move it further to “Done” stage if the task is working as per requirements or back to “Progress” stage in case some bugs or problems were found. One of the main problems the team has been facing as discussed in current state analysis was tasks getting stuck in the “Testing” phase at the end of the sprint. Because of the commitment pressures, the team members sometimes hurried up and moved the tasks to the “Done” stage without testing all the possible test scenarios. Sometimes, when the tester was unable to test all the tasks, they were left in the “Testing” stage when the sprint was over; indicating either the team had overcommitted or they had underperformed. So, managing the work flow around “Testing” stage is very important and these problems were discussed with all the interviewees. Many interviewees were of the opinion to set a maximum limit to the “Testing” stage as well. However, when it comes to handling the situations when testing becomes a bottle neck, opinions differed. One of the interviewees, Maksim Luzik mentioned,

> Usually testing is the responsibility of a tester. In case the tasks are small, it won’t be a problem for a tester, but in case they are big then too many tasks in the testing stage becomes a problem. Automation of tasks might help partially but I do not think we shall have a situation when the number of tasks in testing is equal to number of developers in a team, it shall be always less than that number.

Another interviewee Team Leader, Edward Karvinen mentioned:

> As long as QA ensures that tasks are tested in staging and production, the local testing could be supported by developers if it becomes a bottleneck within a sprint. However, work shall be done in coordination with testers and it shall be done in rare cases. Further, those tasks shall be marked as tested by developers so that the testers are aware that they might need to focus a bit more on those tasks in the release branch.

Therefore, more favourable opinion was to set a limit of less than number of developers within a team and in case maximum limit is reached. Further, in order to avoid bottlenecks in testing stage, many interviewees were of the opinion that other team members can help the Quality Assurance testers with testing someone else’s task under the guidelines of the Quality Assurance team. Accordingly a limit of (N – 1) was selected to limit the number of tasks inside the Testing queue; here N implies the number of developers in a team provided N is greater than 1. This way if we have 5 developers in a team, task limit for Testing queue will be (5 – 1 = 4) four (4).

Explicit Work Policies:

In order to improve collaboration within and outside the team, it is important for a team member to know what to do in certain recurring situations where he otherwise has no way to decide upon himself. In traditional Scrum, process assumes that team will be self-organising and will address their issues and impediments whenever they happen. However, often teams struggle to achieve this in real world and it leads to miscommunication, delay in delivery, weaker collaboration and decision making. Setting explicit work policies empowers the team members to handle decisions with ease and less stress. The team under discussion had some basic unwritten team rules, however, after interview discussions many more rules were made explicit in the task management tool used by the team and further those rules were improved in the sprint retrospectives.

Most of the interviewees were in favour of making the team rules explicit. In addition, some were favouring setting explicit rules on what to do if some stage becomes a bottleneck, such as one of the interviewees mentioned, “Whenever a team member completes a task, he shall give first priority to tasks in the review column. The problem is the developers are not fond of reviewing a task or testing someone else’s task”. Other interviewee had strong opinions on splitting a task that will help the team in solving over-commitment problems. He mentioned, “It makes more sense to split the tasks into smaller ones”.

After the interview data analysis and Definition of Done (DOD) review meeting headed by the Quality Assurance team, the work policies defined by the team under discussion are mentioned below:

- Limit for Tasks in PROGRESS stage: 2*N-2 (N is number of developers)
- Limit for Tasks in REVIEW stage: N-1 (N is number of developers)
- Limit for Tasks in TESTING stage (N is number of developers): N-1
- No new tasks can be added to any stage if it is already under its maximum limit. In these situations, team member shall first work on task from that particular queue rather than taking a new task from the backlog.

- If the “Testing” queue reached its maximum limit, after completing the existing task, the team member shall work on tasks related to the “Testing” queue. This shall happen under the supervision of Quality Assurance team who must provide some guidelines for testing different tasks.
- While planning, always split a task if work estimations indicate that task can span across sprints.
- Requirements/tasks shall only come from the Product Owner (PO) or agreed with PO.
- Stories are written and prioritized by the product owner
- Improve the work policies over time using sprint retrospectives as a place to experiment.
- If the feature in production is not working properly, a new bug should be created rather that reopening the corresponding task. A deployed feature must never be reopened.
- All the bugs found in staging and production must be reported to task management tool.
- Blocker bugs will be added directly to the ongoing sprint.
- Blocker bugs must be prioritized by the developer before taking a new task from the backlog.
- Any critical bugs found in the system shall be informed to the Product Owner (face-to-face).

In addition to work policies mentioned above, every developer within the team shall follow the common DOD for each and every story. Definition of done handles rules relating to test case coverage for tasks, review rules and who would approve the tasks. Most of those rules are companywide rules and are further improved for each and every team.

Scrum Ceremonies: In Scrumban, most of the ceremonies related to Daily Stand-up Meetings, Sprint Retrospectives and Sprint Demonstrations are retained from Scrum. This also became evident from the interview discussions where almost all the participants favoured those.
One of the interviewee, Product Owner (PO) of the team, Mika Mannermaa, mentioned on 17 February, 2014 that “Sprint demonstration is very important aspect of the current process as far as PO’s are concerned. This is where PO’s know how team has performed when it comes to sprint goals”. Another interviewee, Head of the Software De-

velopment, Timo Valtonen, mentioned on 31 January 2014, “Reviews and other Scrum ceremonies like daily stand-up meetings, sprint retrospectives are working nicely in the current process”

Therefore, the team will continue to use daily stand-up meetings, which normally lasts for fifteen minutes, and every member in the team must answer three important questions such as, what they accomplished yesterday, what are their plans for today and are there any impediments related to their work. One of the interviewees, Lauri Oherd on 21 February, 2014 mentioned, “daily stand-up meetings keep me focused as everyday morning I have to answer what I was doing yesterday and it feels good in case I managed to complete my task”.

Regarding Sprint Reviews, they will also continue as usual, every two weeks by the end of the sprint, team will demonstrate what stories and tasks they were able to achieve in the last sprint. One interviewee, leader of the team mentioned, “Reviews are very good because Product Owners, sales guys and even other teams get to know what we accomplished within a sprint”. About sprint Reviews, Ex-Scrum Master mentioned, “Sprint demonstrations are quiet OK in our team, and it acts as what is coming next to sales and Product management, and it also acts as feedback for the team from the key stakeholders, whether things are done as per expectations”.

For Sprint Demonstration, Product Owners and the key stakeholders invited by them such as Technical Architects, Product Management as well as Sales teams join the demonstrations. This helps team to get direct feedback on how the sprint performed and at the same time Product Management can plan the main release features based on sprint demonstrations.

After Sprint Demonstrations, the team will have sprint retrospective discussions as it used to happen during old scrum process. However, there were some good suggestions during the interview discussions on how to improve the retrospectives. Ex-Scrum Master of the team mentioned,

> Retrospectives shall be used to improve only your own team’s process rather than outside. They are just like therapy sessions to release frustrations about what went wrong and what can be improved but you can rarely change something in sales sides or anywhere else except your team. Grouping the action points and voting will help which action point’s team shall concentrate on and rest action points can wait until next retrospective.

Quality Assurance leader, Erwann Cleudic on 12 February, 2014 mentioned,

> Teams shall invite Product Owners to the retrospective discussion; they are the first class team members and they are the ones who know how a team performed based on sprint expectations, so their feedback is vital.

Based on different suggestions and ideas from interviews, team will only try to handle most important action points from retrospectives, rather than focusing on all. The reality is no matter how many action points you come up with, you cannot concentrate on all of those, so team will have to make a choice what action points to implement next. For this purpose, once every member suggests what needs to be improved in next sprint, those action points will be prioritized based on team votes and only most important action points will be addressed in the next upcoming sprint. Further, Product Owners shall also be part of the retrospective discussions, they are part of the team and it will further try to bridge the gap between team and the management. In addition, Product Owners have valuable information on how team performs from a business perspective and even they can handle any action points where management comes into picture.

Estimations and Metrics: As discussed in the literature review, in the Scrumban, the team shall have smaller and perhaps similar sized backlog items. And once the team becomes better is creating stories based on average size (Average Lead Time), story estimations may not be needed at all. In practice, it is often difficult to split a story or even roughly estimate the story. One of the interviewees, Team Leader, mentioned, “It makes more sense to split the tasks into smaller ones. Sometimes, people are lazy to do that and things keep on dragging for many sprints, however splitting a task doesn’t affect productivity because there are instances when it doesn’t make sense to create too many tasks if only one guy is working on that, it even takes time to create tasks in tools”. Another interviewee, Ex- Scrum Master mentioned, “Splitting bigger tasks will always help, but it is difficult to split a task. Further, there are task dependencies which act are blockers while splitting a task. One task cannot be started before completing the previous one. So, I would say it is often tricky to have similar size of tasks”.

So in situations where there is a big story which is split into smaller stories and in addition tasks are interdependent, only one person can work on those tasks. Therefore, it may not make sense to divide this story into smaller ones because no matter what only that person can work on the next task. Another interviewee, Head of Software Development, Timo Valtonen mentioned on 31 January, 2014,

> It is very important to split a story which is estimated more than 8 story points. From the past sprint data, often stories with more than eight (8) story points continue in the following sprint which is not good when it comes to sprint commitments. Further, this gives incorrect visibility to Product Owners

So it may be difficult to split a story into smaller ones but at the same time it is also important to split a bigger story for providing better visibility and achieving sprint commitments. Often the process of estimating the size of a task is a very difficult job, even though people may estimate tasks, but at the end of the day those are just estimation and may or may not be correct. That is one of the reasons why many Scrum practitioners suggest using abstract methods to quantify effort such as estimate the size of tasks in story points based on Fibonacci series or sizes like XS, S, M, L, and XL. Estimation is a difficult task and it may only get better with years and years of experience in that particular technology (Scrum Methodology, 2014). One of the of the interviewee Ex-Scrum Master mentioned,

> The problem is estimation is a very difficult thing to do, so often team overestimates or underestimates tasks. Under pressure teams often underestimate. However rough estimation is still good, it still gives some ideas to outsiders (Sales and Product Owners) how big a task is and how long it may take

Most of the interviewees were in favour of estimating rough size of the tasks. However, many agreed that it might be difficult to split tasks into similar sizes. At the same time many agreed that it is good to split bigger tasks for more visibility rather than dragging a task around many sprints. Therefore in Scrumban process team will continue to do estimation of tasks based on story points as was done in earlier process. In addition, team shall try to split the stories which are estimated more than 8 story points.

As described in literature review, measuring velocity is optional in Scrumban because some practitioners prefer story estimations and others do not, and often it depends upon whether they follow Scrumban which is more inclined towards Scrum or Kanban respectively. One of the interviewees, Team Leader mentioned,

> Velocity does give some perspective on what can be done in next sprint but often estimations are not accurate as often research work related to tasks is difficult to estimate.

Another interviewee, Perry Mitchell mentioned on 21 February, 2014 that “Velocity is often misused and team may even inflate the estimations to get better velocity. Secondly, there is too much interest from the management side regarding velocity.”
Most of the interviewees preferred story estimations and agreed that velocity helps product owners in understanding how much a team can accomplish within a sprint. Many of the interviewees also agreed that velocity shall not be used by management to determine team’s performance. Ex-Scrum Master of the team mentioned that “People can inflate story points just to get more velocity, so it may not be a good metrics if management is using velocity for determining performance of the team”. There can be sprints where team has to fix many important bugs that might impact their velocity, but it does not mean they performed badly. Average Team Velocity shall only be used by Product Owners to roughly estimate what team may accomplish within a sprint and certainly not what team must accomplish. Since story points are associated with new tasks only that is why team was focusing on new tasks rather than bugs during the sprint planning sessions. As a result, based on average team velocity, team had a tendency to fill the sprint backlog slots with new tasks. This was also mentioned by Ex-Scrum Master of the team, “Partially velocity is a reason that bugs get deprioritized”. Therefore in the new process team will continue to measure Velocity but it would not be used as a direct means to fill the sprint backlog capacity. Accordingly, during planning meetings new tasks would not be preferred over bugs because they have story points associated. Team shall choose based upon what is important that needs to be completed next, and it can be either a new story or an existing bug.

Although Scrumban prefers measuring Average Lead Time (or Average Cycle time), this approach will only work in case all the stories in the backlog queue are roughly similar sized. It is difficult to have similar sized stories in new product development especially with respect to the team under discussion because many tasks need some research work as well. Also many of the interviewees mentioned that it will be difficult to create similar sized tasks when requirements are changing and scope of the task is not clear. Accordingly, in the new scrumban process, measuring Average Cycle time will be optional but it may give some information in the long run about average time to complete tasks.

Product Backlog Grooming: means constantly improving the quality and priorities of stories in the product backlog. In Scrum, product backlog grooming is done by the Product Owner and in scrumban as described in literature review, product backlog grooming is optional and it assumes that product backlog is prioritized. However, the problem is if the product backlog grooming is out of process, pulling stories from the product backlog into the sprint backlog queue will become difficult. Further, as described in current state analysis in case Product Backlog is not prioritized, choosing the right stories for sprint backlog becomes difficult and sprint planning might end up planning priorities rather than filling available backlog slots. Also, if stories are not well defined in the product backlog, it makes it difficult for team to estimate stories. Often the scope of incomplete stories changes inside a sprint and this leads to delays or incompletion of other stories. One of the interviewees, team leader mentioned that “we shall not stop working in case stories are incomplete and rather we shall deliver such story in increments but at the same time management shall understand such tasks will take time”. However most of the interviewees preferred to have well defined product backlog grooming process. Head of Software Development teams, Timo Valtonen mentioned, “Majority of the problems within the team will be solved in case Product Backlog is prioritized and the quality of stories is improved”. Another interviewee Ex-Scrum Master of the team was also having the similar opinion. He mentioned, “Product Owners and product managers shall groom the backlog in order to ensure prioritization, in this way when team will be pulling the tasks from product backlog they shall be prioritized already”. Product Owner of the team, Mika Mannermaa, mentioned on 17 February, 2014 that

> Product Owners shall ensure that the product backlog is in order; we have been lacking this lately. We need to break big epics into smaller stories and therefore we must have weekly grooming sessions with Product Management as well as sales team. Further, we need to synchronize backlog grooming sessions ahead of sprint planning meetings.

After analysing current state and all the ideas from interview discussions, it became evident that product backlog grooming has to be part of Scrumban process, especially with respect to the product development process of the team under discussion. We cannot exclude Product Backlog grooming or assume that our product backlog will be in order without making this activity as part of the process. One of the good ideas was to have the weekly product backlog grooming which involves the Product Owner, Team Leader or any needed team member based on technical expertise required for different stories. This group will do the backlog grooming which include improving quality of stories, rough estimation of stories and prioritization of stories so that top of the backlog is always in order and ready to be pulled into the Ready Queue. Further, Product backlog grooming will be done ahead of the beginning of next sprint planning meeting (i.e. before backlog/ready queue update meeting, and the first update meeting always happens just after the sprint demonstration is over). The process of Backlog Grooming which will be part of the new development process is described in Figure 8:

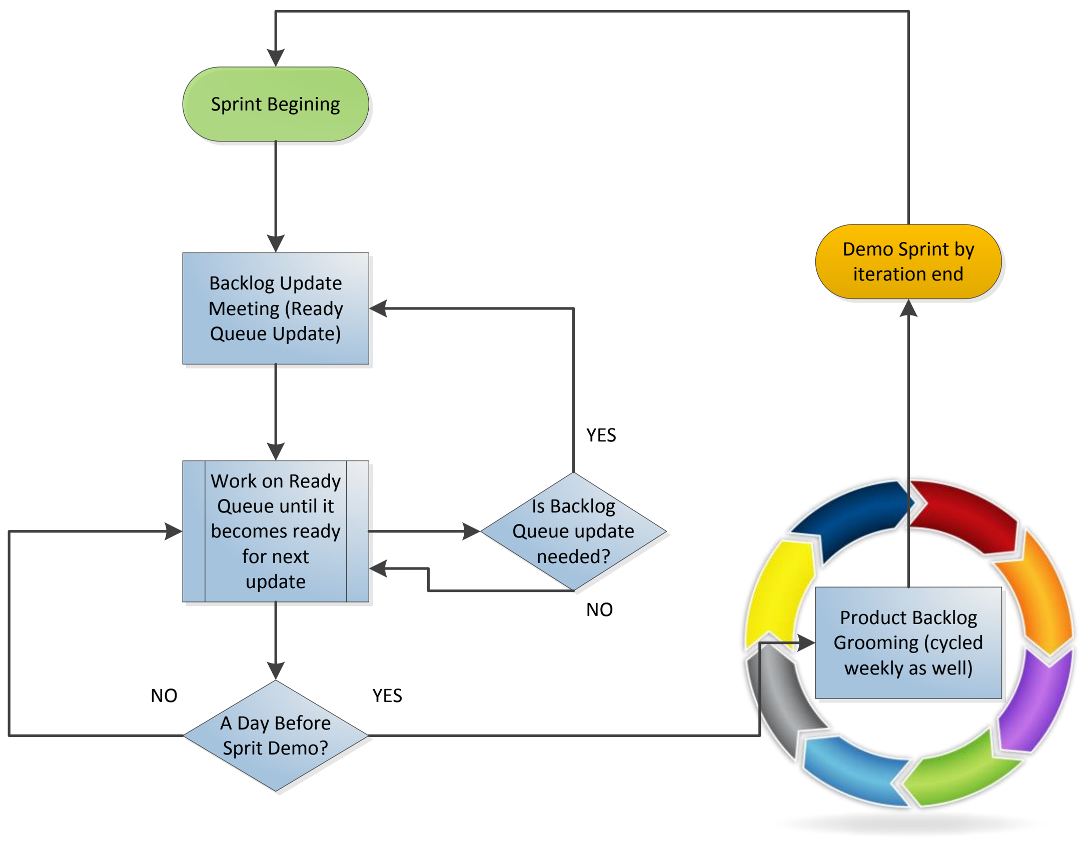

Sprint Begining
Backlog Update Meeting (Ready Queue Update)
Work on Ready Queue until it becomes ready for next update
Is Backlog Queue update needed?
YES
NO
A Day Before Sprit Demo?
NO
YES
Product Backlog Grooming (cycled weekly as well)
Demo Sprint by iteration end

Figure 8. Product Backlog Grooming added to the Scrumban process

5.3 Prototype Proposal for Product Development Process

The new software development process proposal (prototype) is depicted in Figure 9, and further briefly summarized in 8 main points.

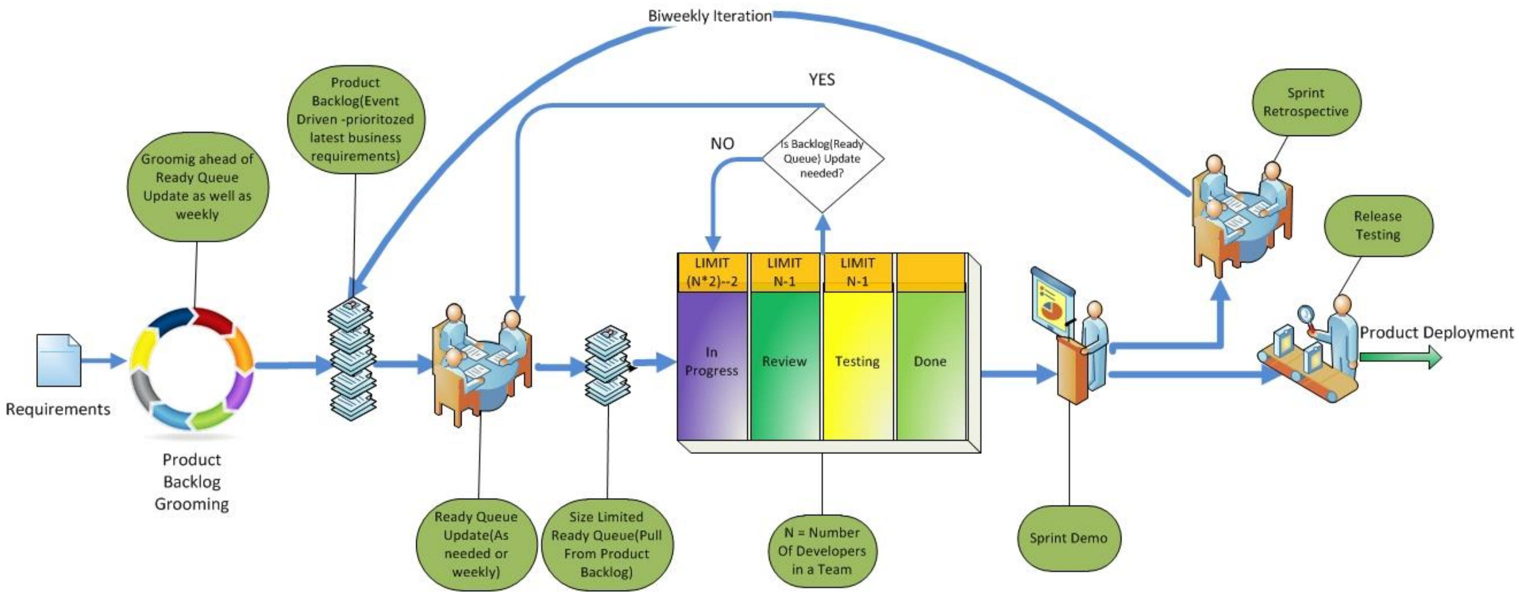

Biweekly Iteration
YES
NO
Is Backlog(Ready Queue) Update needed?
Product Backlog(Event Driven -prioritozed latest business requirements)
Groomig ahead of Ready Queue Update as well as weekly
Requirements
Product Backlog Grooming
LIMIT (N*2)--2
LIMIT N-1
LIMIT N-1
In Progress
Review
Testing
Done
Sprint Retrospective
Release Testing
Product Deployment
Ready Queue Update(As needed or weekly)
Size Limited Ready Queue(Pull From Product Backlog)
N = Number Of Developers in a Team
Sprint Demo

Figure 9. New Software Development Process based on Scrumban

1. Planning Meetings: In Scrumban model, the team will start using shorter planning sessions in order to fill the slots available in the backlog. The ideal work planning process should always provide the development team with best tasks to work on next, no more and no less. Within a sprint, once the team is about to run out of tasks they will have another planning session to pull the tasks from the top of the product backlog and fill the available capacity.

2. Backlog and Ready Queue: Unlike Scrum, the team’s new backlog will be fixed sized backlog and may be updated many times during a sprint iteration based on available capacity. This backlog will also act our Ready Queue in terms of Scrumban Process as it will only contain the prioritized list of work items that team members must work next. The team has enough development stories to work on in the Ready Queue, not too many not too less. This will help the team to reduce long planning meeting and focus on prioritised tasks. At the same time, new requirements can be brought in making it more agile than Scrum. For product management, this gives added flexibility in a controlled way and hence this new process will be able to handle immediate customer needs within the framework of process rather than breaking it.

3. Work In Progress (WIP) Limits: The team needs to maintain a flow around backlog and different stages; therefore, they will limit Work In Progress Items

(WIP's) for each and every stage and manage flow. The team shall prefer completing tasks in hand rather than starting new work. This means every stage of the team's development process has a limit in terms of number of work items that can be in TO-DO state, PROGRESS state, REVIEW state and in TESTING state. This will ensure that our tasks do not get stuck in a particular state for long or there are not too many tasks in a particular state as team will need to enforce limit WIP policies

4. Explicit Work Policies: Making the work policies explicit will help the team members to collaborate better within the team as well as outside the team. In addition, it will help in managing the workflow around different stages. The team will have explicit work policies in terms of WIP Limits, maintaining quality, how to collaborate when some stage becomes bottleneck, written and clear definition of done, how bugs and priorities will be handled within team, who can create stories, communication policies, and who will handle external communication. All those policies were described in chapter 5.2

5. Scrum Ceremonies: In Scrumban, most of the ceremonies related to Daily Stand-up Meetings, Sprint Retrospectives and Sprint Demonstrations are retained from Scrum. All these ceremonies constantly help to manage impediments, improve development process and help to get first hand feedback from product owners, managers and customers. In order to improve the sprint retrospective further, the team will only choose most important action points based on voting rather than focusing on all. Further, the Product Owners will also be part of the retrospective discussions. They are part of the team and they have valuable information on how a team performs from a business perspective. Product owners can even handle any action points where management comes into picture. This will further try to bridge the gap between team and management.

6. Estimations: In Scrumban process, the team will continue to do estimation of tasks based on story points as was done in earlier process. Unlike Scrumban, the team may not be able to split tasks into similar sizes; however, the team shall try to split the stories which are estimated more than 8 story points.

7. Metrics: In the new process, the team would continue to measure Velocity but it will not be used as a direct means to calculate number of stories needed in sprint backlog. Accordingly, during planning meetings new tasks will not be preferred over bugs because they have story points associated, team shall choose based upon what is important that needs to be completed next and it can be either a new story or an existing bug. Also, measuring Average Cycle time will be optional but it may give the team some information in the long run in case it measures the average cycle time of stories and bugs within different sprints.

8. Product Backlog Grooming: Even though optional and out of scope in Scrumban, product backlog, grooming will be part of the new software development process. Product Management including team leader or the needed team members will do a weekly product backlog grooming, usually ahead of Ready Queue update meetings. This will ensure that correct priorities are set to the backlog, quality of stories is improved and most importantly work items are at the top of priority list. As a result, team can pull stories from the top of the Product backlog to Ready Queue as soon as there are not enough stories in the ready queue. The flow of product backlog grooming within the new process is also depicted in Figure 8.

6 Pilot Testing of the proposed New Process

The main purpose of the pilot testing of is to ensure the feasibility of the New Software Development Process. The efficiency of the new process can be verified by checking the impact of new process on the team's velocity and further analysing the team's retrospective discussions whether it has improved the teams day to day functioning in any way.

6.1 Pilot Testing Overview

The new product development process described in Chapter 5 was presented to Applications Development Team as well as other important stake holders like Product Management, Quality Assurance Team and Head of Software Development, using power point presentation on February 25th, 2014. The whole session lasted for around two hours and included presentation on findings from current stage analysis, describing new process proposal and why process based on scrumban might address various problems described in current state analysis. After the presentation, the Head of the Software Development and Product Management agreed to test this prototype process in the Applications Development team for four sprints, starting from March 4th, 2014 until April 29th, 2014. After every pilot sprint, process could be improved depending upon the discussions in team's sprint retrospectives. Once the results of this new prototype process will be concluded on April 29th, 2014, next decision will be taken further whether to continue with this new process or not. The timeline indicating different milestones and stages in the piloting of prototype process is shown in Figure 10.

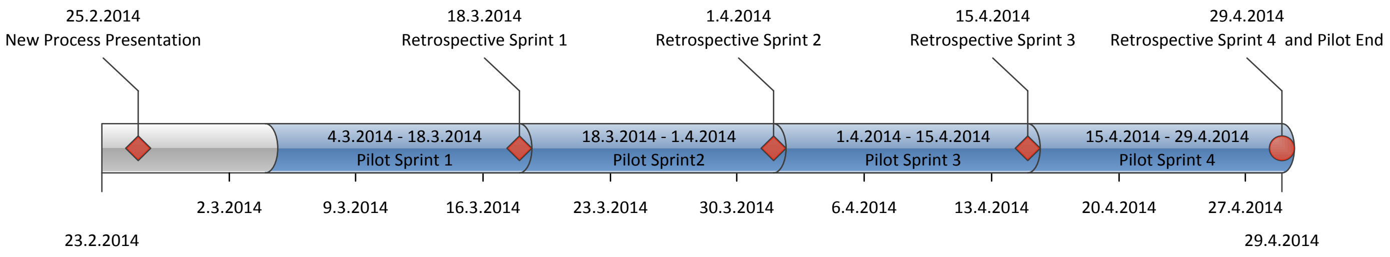


Figure 10. Pilot testing timeline of proposed new process

The applications development team which was going to pilot test this new process consists of six team members, including five developers and one Quality Assurance member. Two of the developers were having dual roles of Scrum Master and Team Leader. In addition, team had a dedicated Product Owner and User Interface Designer. In order to test the new process, there were no changes in the team roles; team's Scrum Master was driving the new process. Sprint duration was set to 2 weeks same as before, and the maximum limits for the different queues was set as described by the Table 8.

Queue	**Formula Used**	**Maximum Limit for WIP's**
Ready (Team's Backlog) Queue	None	12
Progress Queue	2N - 2 = 2x5 – 2 (where N is no of developers within team)	8
Review Queue	N-1 = 5-1 (where N is no of developers within team)	4
Testing Queue	N-1 = 5-1 (where N is no	4

	of developers within team)	

Table 8. Work In Progress Limits for different Queues in new prototype process

In addition, the new prototype process model was well documented and listed on team's intranet pages of the case company.

6.2 Pilot Testing Analysis

Impact on Team's Velocity: One of the keys metrics than can used to analyse whether the new process has improved the team's ability to deliver user stories is Team's Velocity. Team's velocity indicates the number of story points completed with a sprint and the average of those can be used to determine whether the overall velocity has increased or decreased. The data regarding the velocity of four Sprints (piloted) was collected from team's task management tool (Jira) and is described in Table 9. In this table, Velocity per team member is obtained by dividing the sprint velocity with total number of team members in the team.

Sprint Velocity per Team Member = Sprint Velocity/No. of team members

Pilot Sprint Number	Velocity	Velocity per team member	Average Velocity	Average Velocity per member	Standard Deviation of Velocity	Standard Dev of Velocity per team member
1	48	8	32.00	7	11.04	0,978945
2	27	6,75				
3	23	5,75				
4	30	7,5				

Table 9. Velocity data of different sprints during pilot testing

As seen from the Table 9, team has a better velocity in the first sprint of Scrumban compared to other sprints. The drop in sprint velocity from second piloted sprint onwards is mainly because the case company reduced the number of team members in the Applications Development Team. The team size was reduced because of reasons which researcher cannot explain in this thesis, as the information is company confidential. Initial size of the team was six members, however after first piloted sprint; the size of the team was reduced to 4 which included three developers and one Quality Assurance Team member. Further, the drop in the team velocity could also be attributed to the fact that team was no more filling the backlog capacity based on team velocity, selection was purely done based on Product Owner's priority and that included bugs as well as new stories. Accordingly in the four piloted sprints, there is a decrease in the Average Team Velocity from 40.65 to 32 compared to team velocity of the last twenty three sprints. However, if one looks into the Average Velocity per team member and compares it with the data from last 23 sprints, it has increased in the four piloted sprints from 6.77 to 7.0, which is 3.32% increase in average velocity per team member.

Sprint Retrospective Analysis:

Being the Scrum Master of the team, researcher also collected and analysed the sprint retrospectives of piloted sprints. From all those sprints retrospectives, the data collected related to the positive points and the things to be improved in every sprint, is listed in Table 10. In this table, however some team specific points, which were not related to the overall new process, have been omitted by the researcher.

Sprint Number	**Positive points from retrospective**	**Things to be Improved in next Sprint**
Pilot Sprint 1	• Ready Queue is good and promising • Task flow within the sprint has improved. There was hardly any workflow stage, which became a bottleneck or work stopper. • Planning meetings were short and to the point.	• Pressure from management to get things done in hurry. • Ready Queue got too big in second update. • Include another team rule, "Before any code check-in, use Js-Hint rules for improving code quality".
Pilot Sprint 2	• Js-Hint rules are being used by everyone. • Better and short planning	• Product backlog grooming meeting happened only once. It

	sessions • No bottlenecks. • Team members helping and collaborating better in case maximum limit is reached in some development stage. For instance, one developer helped testers to test some tasks when maximum limit was reached in testing stage.	shall be PO's responsibility to reserve specific time slots.
Pilot Sprint 3	• Now more bugs are getting fixed. It is good not to have focus on team velocity; otherwise mainly new stories were considered during sprint planning. • Our process is more flexible, ready queue updates allows us to change priorities within a sprint.	• Improve Code Review, some quick fixes were added without proper code review.
Pilot Sprint 4	• Code reviews were done properly. • Process is more stable and at the same time flexibility has improved. • Ready queue updates within sprint is helping avoiding pressure situations.	• Outside communication shall be improved. • Specs related to stories shall be improved. • Avoid hacks and quick fixes in the code.

Table 10. Sprint Retrospective Data from the Piloted Sprints

Analysis of First Piloted Sprint: From the retrospective data of first piloted sprint, it is clear that team appreciated having shorter planning sessions in order to update ready queue capacity, and all the meetings were short and to the point. Accordingly, the researcher can conclude that the new process has helped the team members to focus on what is important next, and provided added flexibility by planning for shorter periods rather than the whole sprint at once. At the same time, workflow across different stages has improved, and there was hardly any development stage that became a bottleneck. With the earlier process, team was often struggling with this issue. In the new process model, every stage has a maximum limit and team members ensure that they need to process the queue which has reached its maximum limit, before continuing with newer tasks.

In addition to positive points from the retrospective, team members discussed on improving handling pressure situations from the top management. Since management wanted some important customer requested changes to be completed in hurry, some additional tasks were added to the ready queue than the permitted capacity. Accordingly, it was agreed by team not to fill the ready queue beyond its maximum limits until its size is improved further. In addition, team agreed on adding on more rules to the explicit team policies. This new rules was related to improving code quality, and says “Before any code check-in, use Js-Hint rules for improving code quality”. This rule will further help the team to improve the quality of code, and every team member has to ensure that they will follow this rule before adding or modifying any new code to the existing code base. This is the best example of how team started to use scrumban retrospectives to improve the team policies.

Analysis of Second Piloted Sprint: The notes from the second sprint retrospective were also positive as far as new process model is concerned. There were no bottlenecks, planning meetings were short, and team members were helping each other in case maximum limit was reached in some development stage of the work flow. For instance, developers started helping quality assurance guys in reducing the testing queue size once the queue reached its maximum limit. This kind of team collaboration was missing in earlier process, as team members were only focusing on their individual tasks. Setting the limit on queues and making work policies explicit has helped the team to improve collaboration.

In addition, team members discussed that Product backlog grooming shall happen weekly and product owners shall take the responsibility of reserving time slots.

Analysis of Third Piloted Sprint: From the retrospective data of third piloted sprint, team realized the power of focusing on work flow rather than focusing on team velocity. Using the new process framework, team was no more having sprint planning sessions which filled the whole sprint backlog based upon the team velocity. Rather, team was simply filling the available backlog queue capacity with new stories as well as bugs; this is a significant change to include bugs into the sprint as well. In addition, ready queue updates within a sprint gave flexibility to the Product Owners in case they want to change priorities within a sprint.

In the retrospective, team members also agreed to improve code review rules, so that no matter how small a task is or no matter how urgent the bug fix is needed, code review shall always be done by the team members in order to maintain code quality.

Analysis of Fourth Piloted Sprint: In this sprint retrospective, team members especially the team leader appreciated the stability and flexibility of new prototype process, even though team would like improvements in other areas such as, improvements in quality of story specifications, improvements in communication from product and sales departments so that team is in synchronization with management expectations and sprint goals. Improvements in user story specifications can only happen with constant backlog grooming as described in the new process. Further, to improve communication, Product owners will need to ensure that sprint goals and expectations are in synchronization with Product Management and Sales team’s expectations.

Pilot Conclusion: Summing up, as a result of the new development process described in Section 5.1, sprint retrospectives indicates that team’s new process has provided the needed flexibility to the product owners and hence enhance the time to market of new features. And at the same time, team has the flexibility to handle urgent customer requirements within a sprint. Further, work flow across the different stages has improved, and team is managing the bottlenecks and collaborating better based on explicit rules and policies. Scrum ceremonies such as sprint retrospectives were used effectively to improve the scrumban process further, and outcome was often a small set of prioritized action points. Overall, the output of team has changed little as far as Average Velocity per team member is considered. Average Velocity per team member has increased from 6.77 to 7.0, which is just a marginal increase of 3.32%. Only future sprint velocity calculations can point out whether the output has increased or decreased over a longer term. Using the new process model, team was still trying to improvise some aspects of the new process in order to improve collaboration and handle different situations which are not addressed 100% by the new process model. There is a scope of improvement when it comes to product backlog grooming, review process and handling pressure situations. After the Pilot ended, team agreed to continue using the new process model in future sprints as well. Head of the Software Development Team of the case company, Timo Valtonen (2014) mentioned, “Eventually it is the responsibility of the team to follow the process, no matter how good a process is on the paper, if it is not followed in the right spirit, it is worthless”.

7 Conclusion

This section summarizes the research process from the initial stage of setting objective to the final proposal of improved software development process. In addition, the research methodologies used in this research are described, followed by the final conclusions and thesis validation.

7.1 Summary

The objective of this thesis was to improve the agile software development process within the case company which could help them in addressing problems described in the current state analysis of this study. One of the teams, namely Application Development Team, within the case company was following Agile Scrum model as a software development process; however the team was facing many problems that could not be addressed by Scrum. An analysis of this process based on interview discussions and sprint retrospectives found many issues related to: process flexibility, frequent priority changes and pressure situations, team collaboration and communication, workflow, incomplete tasks within sprints, shift in process and many other issues. If these issues were not addressed, they could lead to delays in software delivery, increased project costs and low team motivation. As a result a new or improved software development process was needed that could address many of these issues.

The proposed model was developed and verified in four iterations. Further, this study was mainly based on qualitative analysis of discussions with different stake holders which included: Product Manager/Owner, Team Leader, Scrum Master, Team members, Quality Assurance (QA) head, QA team member, and Head of Software Development Teams. The interview discussions were conducted with same stakeholders both in the current state analysis as well as the new process development.

The researcher started with defining the study objective which was followed by the current state analysis of the current process. Based on the findings from the current state analysis data, researcher focused his search for best practices in literature as well as literature review. In literature study and review, the researcher mainly focused on three agile models namely Scum, Kanban and Scrumban. From the literature study and current state analysis a new software development process was created. This model was further developed based on the data collected from qualitative analysis of interview discussions. Finally, the new proposed model was further improved and pilot tested in four sprints during the thesis study, which resulted in the final process proposal and conclusions.

Thus, the outcome of the thesis is the new software development model for the case company based on Agile Scrumban model. And the results from the pilot testing indicate that the output of team has slightly improved as far as the average velocity per team member is considered; there is just a marginal increase of 3.32%. The results also indicate that many process related things have improved such as team collaboration, improved work flow across different stages, flexibility to address important and urgent business requirements, some aspects of the team communication, process stability, etc. There are still some areas where process improvement is needed further, and many of those issues can be addressed easily within the framework of new software development process.

7.2 Next Steps

Even though the new software development model has addressed many of the issues faced by the development team, there is still some scope of improvement in certain areas such as, code review process, product backlog grooming and handling high pressure situations. For instance, in the new process backlog queue and the ready queue are same, which means that the new user stories or tasks can only be processed by the team on the next ready queue update meeting within a sprint. In order to add more flexibility and separate the process of task assignment from task prioritization, ready queue can be separated from the backlog queue. Further, ready queue shall only contain a fixed prioritized subset of backlog queue. As a result, developers can pull the new tasks from the ready queue and product managers will have the flexibility to update the backlog queue as and when needed. Product owners will not need to wait for the next Ready Queue update, as it is a separate queue from the backlog queue. Other issues like improving the quality of specifications can be addressed by properly implementing the product backlog grooming sessions, based on the new process model defined by this study. In addition, over the time team could also improve the team related work policies. Further, the pilot tests for few sprints were done with less number of team members; therefore it would be interesting to know the actual changes in Average Velocity once the team size is increased back to the original size of six members.

7.3 Evaluation

7.3.1 Outcome Vs Objective

The objective of this thesis was to improve the agile software development process of the Applications Development Team within the context of the case company. The study proposed a new model based on Agile Scrumban and it retained the vital elements from the current model based on data from current state analysis and qualitative analysis. Overall the thesis was successful based on two main decision points; first the proposed software development model was approved by different stake holders within the case company for pilot testing, and second, after the completion of pilot testing the team continued to use the new process in future. Further, the positive feedback of the new process based on retrospective data of piloted sprints indicates that new process has added flexibility, simplified the planning, improved team collaboration, made workflow smoother, and addressed many other issues found in the current state analysis of this study.

7.3.1 Reliability and validity

In order to ensure the trustworthiness of one’s thesis, researcher has evaluated it in two important aspects: reliability and validity.

Reliability is the tendency towards consistency found in repeated measurements of the same phenomenon. A research is said to be highly reliable if it consistently gives the same results if measured repeatedly and if the results are less consistent, the reliability of the study is low (Carmines & Zeller, 1979). Once the data is analyzed, it should be possible to be certain that if the study is conducted again, same results will be produced. If there is uncertainty in this aspect, then the research will not be considered reliable (Mayor & Blackmon, 2005).

In this Thesis, the proposed software development model was produced based on literature study, inputs from current state analysis, and data collection based on qualitative interview discussions. Further, the research was done in various stages, with each stage providing inputs to the next phase. As a result, knowledge of research problem as well as solution ideas improved from one stage to another. The reliability of the literature data was realized by using the data sources for the literature study and literature review from broader range of leading academic journal articles, reputed online sources and books published related to the application of agile software development processes. In the data collection and analysis phase, reliability of data sources for qualitative analysis was realized by choosing the participants from various backgrounds within and outside the team in order to get broader perspectives and viewpoints. Reliability of the data collection for the current state analysis was realized by using the case company's reliable intranet resources and online tools. Finally, the new process model creation was also done in iterations, where it was further refined and improved gradually. The consistency in the results of the iterations also indicates that the study is reliable.

Validity is affected by the researcher's perception of validity in the study and his/her choice of paradigm assumption (Creswell & Miller, 2000). As a result, many researchers have developed their own concepts of validity and have often generated or adopted what they consider to be more appropriate terms, such as, quality, rigor and trustworthiness (Davies & Dodd, 2002; Stenbacka, 2001). The research is valid if it captures the truth of the situation and is not influenced by outside influences or personal preferences (Mayor & Blackmon, 2005).

In this thesis, the validity of the research is accomplished since the researcher achieved the main objective he had established in the beginning of the research. The new proposed model addresses most of the issues that were presented in the current state analysis of this study. Validation of the new process model was further realized by pilot testing of the proposed model in several iterations. The data collection for this study at various stages was based on the data from reliable and valid sources, which accordingly reflected the true situations in current state analysis as well as in the results of pilot testing phase. Further qualitative interview discussions were handled one to one with different stakeholders so that the interviewee is not influenced by others opinions. In order to get the comparable data, same structure of interviews was followed to collect common and diverse opinions from different stakeholders.

References

Cagley, T. (2013) *What makes Scrumban Scrumban? ,* Available at: http://tcagley.wordpress.com/2013/09/24/what-makes-scrumban-scrumban-daily-process-thoughts/ (Accessed 18th April 2014).

Carmines, E. & Zeller, R. (1979) *Reliability and Validity Assessment,* California: Sage Publications Inc.

Chai, L. (2008). 'E-based inter-enterprise supply chain Kanban for demand and order fulfilment management', *International Conference on Emerging Technologies and Factory Automation,* EFTA '08. IEEE, September 2008, pp. 33–35.

Cohn, M. (2012) *Sprint Review Meeting,* Available at: http://www.mountaingoatsoftware.com/agile/scrum/sprint-review-meeting/ (Accessed 11th Feb 2014).

Creswell, J. W. & Miller, D. L. (2000) 'Determining validity in qualitative inquiry', *Theory into Practice,* Vol. 39*,* Issue 3, pp. 124-131.

Davies, D., & Dodd, J. (2002) 'Qualitative research and the question of rigor', *Qualitative Health research,* Vol. 12, Issue 2, pp. 279-289.

Heuvel, M V D. (2011) *Explicit Policies Make Life Simpler,* Available at: http://scrumfamily.wordpress.com/2011/10/10/explicit-policies-make-life-simpler/ (Accessed 29th April 2014).

Gambill, P. (2013). *Scrumban: A different way to be agile,* Available at: http://www.deloittedigital.com/us/blog/scrumban-a-different-way-to-be-agile (Accessed 10th April 2014).

Kiosked (2013) *What is Smart Content,* Available at: http://www.kiosked.com/smart-content/what-is-smart-content/ (Accessed 10th December 2013).

Kniberg, H. (2009) *Kanban vs. Scrum: How to make the most of both*, Available at: http://www.crisp.se/henrik.kniberg/Kanban-vs-Scrum.pdf (Accessed 8th May 2014).

Kniberg, H. & Skarin, M. (2010) 'Kanban and Scrum- making the most of both', C4 media -Publisher of InfoQ, Available at: http://www.infoq.com/minibooks/kanban-scrum-minibook (Accessed 8th May 2014).

Ladas, C. (2008) *Scrumban: Essays On Kanban Systems For Lean Software Development*, Seattle: Modus Cooperandi Press,

Ladas, C. (2008) *Scrum-ban. Lean Software Engineering: Essays on the Continuous Delivery of High Quality Information Systems,* Available at: http://leansoftwareengineering.com/ksse/scrum-ban (Accessed 2nd Feb 2014).

Ladas, C. (2009) *Lead Time Vs Cycle Time,* Available at: http://leanandkanban.wordpress.com/2009/04/18/lead-time-vs-cycle-time/ (Accessed 10th April 2014).

Mahnic, V. (2013) 'Improving Software Development through Combination of Scrum and Kanban', *Conference Proceedings,* 8th WSEAS International Conference on Communications and Information Technology, Tenerife, pp. 281 - 288.

Mayor, H., & Blackmon, K. (2005) *Researching Business and Management*, New York: Palgrave Macmillan

Pahuja, S. (2012) *What is Scrumban?,* Available at: http://www.solutionsiq.com/resources/agileiq-blog/bid/87799/What-is-Scrumban (Accessed 5th Feb 2014)

Radics, S. (2013) *Scrum, Kanban, Scrumban - a fast overview and rough categorization when to use what method,* Available at: http://www.ontheagilepath.net/2013/09/scrum-kanban-scrumban-fast-overview-and.html (Accessed 18th April 2014).

Schwaber, K. and Sutherland, J. (2013) 'The Scrum Guide, The definitive Guide to Scrum: The rules of the Game', Available at: https://www.scrum.org/Portals/0/Documents/Scrum%20Guides/2013/Scrum-Guide.pdf (Accessed 3rd December 2013).

Scrum Methodology (2014) *Scrum Effort Estimation and Story Points,* Available at: http://scrummethodology.com/scrum-effort-estimation-and-story-points/ (Accessed 7th February 2014).

Stenbacka, C. (2001) 'Qualitative research requires quality concepts of its own', *Management Decision*, Vol. 39, Issue 7, pp. 551-555.

Strange, M (2013) *Basics of Scrum-ban,* Available at: http://www.xsinc.com/basics-of-scrum-ban/ (Accessed 11th April 2014)

Sugimori, Y. et al. (1977) 'Toyota production system and Kanban system: Materialization of just-in-time and respect for human system', *International journal of production research*, vol. 15, no. 6, pp. 553-564.

Techtarget (2013) *Agile Velocity*, Available at: http://whatis.techtarget.com/definition/Agile-velocity (Accessed 8th May 2014).

Wikipedia (2014) *Agile Method*, Available at: http://en.wikipedia.org/wiki/Agile_software_development (Accessed 8th May 2014).

Yuval, Y (2012) *So what is scrum-ban?,* Available at: http://yuvalyeret.com/2012/04/28/so-what-is-scrumban/ (Accessed 10th January 2014).

Interview Questions

Questions for Current State Analysis

Topic of Interview Discussion	Questions
General	• Do you think we need changes in scrum process? • What has been working well in the current model which is loosely based on Scrum methodology? • What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team? • When do you think agile scrum doesn't work in a team or an organisation? • What are the main factors which can lead to the success of software development projects? What is successful project for you?

Questions for New Process Creation

Topic of Interview Discussion	Questions
General Questions	• What are the main requirements or expectations for the new process?
Sprint Planning	• What do you think are the problems with our sprint planning meetings? And where can we improve? • What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in

	the whole sprint?
Work In Progress Limits	• What is the best limit for WIP items in your opinion? (To-Do stage, Progress stage, Review stage, and Testing stage).
Metrics and Estimation	• Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? • What is your opinion on calculating average cycle time to complete a task? • What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? • Is it more effective to have similar smaller tasks rather than having stories with different sizes? • Why does team often focus on new tasks rather than bugs?
Workflow	• In order to keep flowing going what shall we do in case testing stage becomes a bottleneck? • Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For Example, stories get stuck in different columns)?
Explicit Work Policies and Quality	• Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? • What are the quality requirements for the new model?

	• What are the ways that can help the team to re-move disconnect between Quality Assurance and the development team?
Sprint Retrospectives and Sprint Reviews	• How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? • Within Sprint Retrospectives, many times it hap-pens that team has too many action points but no right owners to handle those? How do you think we can improve there? • Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?
Backlog, Ready Queue and Backlog Grooming	• What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling as-signment and prioritization)? • And how often you think we shall update them? Weekly or as soon as queue is about to empty? • Many times it happens that there is pressure from the team management to take on new tasks in-side the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not plan-ning for whole sprint but rather plan for shorter periods

	• What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? • Who can we improve the quality of stories, poor specifications and prioritization of stories?

New Process Creation Interview Discussions

1. Theme interview with Team Leader

Date: February 5, 2014 14:30-15:30

Participant: Edvard Karvinen (Team Leader)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	What are the main requirements for the new process	• Some framework that provides us better flexibility; and management needs to understand that if things are not clear, things will take time to get done. • We cannot be customer oriented if we are having too strict process. So it shall be able to handle customer driven requirements and their immediate needs as well even if e.g. some customer wants some stuff done within a week.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• In work in progress, because we are flexible and priorities might change and developer might move to another priority task, so he shouldn't put the first one back. So in progress limit of 2 is good per developer. • Review: It is good to have some limit, though sometimes there might be exceptions.

Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Velocity does give some perspective on what can be done in next sprint but often estimations are not accurate as there needs to be done some research and people cannot estimate things properly when there are many unknown things. So how can someone estimate it correctly if one doesn't know how to do it correctly? Personally we shall not think too much about velocity as long as everybody in team does work best and are productive.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size. More focus on bugs rather than tasks? new	• It makes more sense to split the tasks into smaller ones. Sometimes people are lazy to do and things keep on dragging for many sprints, however splitting a task doesn't affect productivity because there are instances when it doesn't make sense to create too many tasks if only one guy is working on that, it even takes time to create tasks in tools.
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• If things become bottleneck in some stage, one way of course is to hire more people but that is not economical. Often automated testing might help but when too many things are creating problems, and then there are many customer related tasks coming then things start becoming bottlenecks. • My opinion is developers shall only do their part and test their stuff well before moving tasks forward for reviewing. However a developer cannot take tester's job and testers have their own way of testing stuff and that is their responsibility. There

		are two stages in our testing, one is within the team i.e local testing and second is in the main release i.e. staging. As long as QA ensures that things are tested in staging and production, local testing could be supported by developers if it becomes a bottleneck within a sprint. However work shall be done in coordination with testers and it shall be done in rare cases. Further those tasks shall be marked as tested by developers so that testers are aware that they might need to focus a bit more on those tasks in the release branch.
Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	• Shorter planning is good because often important customer tasks are coming from product owners even within a sprint.
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team (new)?	• We already have some kind of team rules at place, e.g. while reviewing a task, so making them explicit will help the team to collaborate better
Sprint Retro-	How do you think sprint retro-	• So far retrospectives and reviews are

spectives and Sprint Reviews and Sprint Demos	spectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	pretty good. • Reviews are very good because Product Owners, sales guys and even other teams get to know what we accomplished within a sprint.
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty? Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones	• If there is a pressure from team management, they should understand that things will take time and we might need to drop some other tasks, so scrum in that sense is very tight and restrictive.

	committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	• The problem is we cannot stop working if a story or task is not complete, being a start-up we shall try things in increments and get feedback from the management.

2. Theme interview with Ex-Scrum Master

Date: February 14, 2014 14:30-15:30

Participant: Maksim Luzik (Ex Scrum Master)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	What are the main requirements for the new process	• If the company has a steady income scrum might be good for them, but for start-ups Scrum is often challenging. So we need a process which can handle abrupt or pressure situations.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• One task at a time is good, technically one can never work on more than one task at a time, but there are situations when someone needs to work on more than one task. Accordingly, we shall give flexibility on working more than one task

		but not to have that as a general practice. In case a developer started working on another task and he already has an existing task in progress, I would prefer moving the old task back to the TO-DO list. • In Review column, if we need to go a bit strict, we shall not have tasks in the review column which is more than half number of developers. So if we have 6 developers, limit shall be something like 3. The idea shall be that there is at least one developer to review the task. • Usually testing is the responsibility of a tester, in case tasks are small then it won't be a problem for a tester but if the tasks are big then too many tasks in testing becomes a problem. • Automation of tasks might help partially, however I don't thing we shall have a situation when number of tasks in testing is equal to number of developers in a team, it shall be always less than that number.
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• People can inflate story points just to get more velocity, so it may not be a good metrics if management is using velocity for determining performance of team. • Problem with cycle times is not all the tasks are of the same size, because sometimes it is difficult to split tasks.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying	• The problem is estimation is a very difficult thing to do, so often team overestimates or underestimates tasks. Under pressure teams often underestimates. However rough estimation is still good, it still gives some ideas to outsiders (Sales and PO's) how big a task is and how long it may take.

	size.	• Splitting bigger tasks will always help, but the difficulty is to split a task. Then there are task dependencies which act are blockers while splitting a task, one cannot be started before completing the previous task. So I would say it is often tricky to have similar size of tasks. • Partially velocity is a reason that bugs get deprioritized
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• I think it is important to maintain the workflow rather than pushing things in the last minute. When some tasks gets stuck in the review or testing stage, team might start hurrying up things by the end of the sprint to move those to testing or done phase without testing the tasks properly. • I would say rather than hurrying we shall keep the tasks there and flag the tasks out of the release if it cannot be completed by the end of sprint.
Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	• The problem with planning is our stories are not sometimes complete, or there is missing acceptance criteria, therefore often scope of those stories change within a sprint and hence planning meetings become ineffective. If our stories are in order and complete that will help a lot. Even if we have shorter term planning sessions, important is that stories are still complete
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within	• Actually we have similar kinds of rules in my team, e.g. whenever a team member completes a task, he shall give first priori-

	a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	ty to tasks in the review column, and second priority goes to the blocker bugs and then third priority is to take tasks from TO-DO list. • Main problem is people are not fond of doing something like reviewing a task or testing somebodies task, so as a last resort setting rules will help. E.g. in Review stage, it is good to have a rule to test the task as well, just to see how it works.
Sprint Retrospectives and Sprint Reviews	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	• Retrospectives shall be used to improve only your own team's process rather than outside. They are just like therapy sessions to release frustrations about what went wrong and what can be improved but you can rarely change something in sales sides or anywhere else except your team. • Grouping the action points and voting will help which action points team shall concentrate on rest action points can wait until next retrospective. • If PO's have time they shall join the retrospectives but usually they don't have time • About sprint demos they are quiet OK in our team, and it acts as what is coming next to sales and Product management, and it also acts as feedback whether the team management to get things done as per expectations.
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is	• Backlog/Ready Queue Updates: It is good to have the ready queue updates weekly or whenever needed. May be it is good to have weekly updates in order to maintain a rhythm.

	decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty?	
	Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods	• Well it depends on company, if the task is split well enough it will help but one week sprints might not work in some teams.
	What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	• Product Owners and product managers shall groom the backlog in order to ensure prioritization, in this way when team will be pulling the tasks from product backlog they shall be prioritized already.

3. Theme interview with Head of Software Development and Services

Date:	January 31, 2014 14:30-15:30
Participant:	Timo Valtonen (Head of Software Development)

Topic Of Interview	**Questions**	**Summary of Field Notes**

General Questions	What are the main requirements for the new process	• Improved planning and better product backlog grooming
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• In Progress, less than twice the no of developers is a good rule. • In Review, at least some developer shall be available to review the task. Developers must review a task before starting anything new. • In Testing, Similar limit as that of review stage.
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Velocity helps management in predictability and accuracy. • Customer promises are made based on velocity accuracy. • Cycle time depends on the complexity of task, as long as we split tasks it may get better. • We must split stories which are estimated 8 or more points, often they span across multiple sprints.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size.	• It is effective to 2 to 3 story point sized task and not the big ones.
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the	• Team members shall help each other. So testing on local environment can be helped by team members if testing becomes a bottleneck. • Smoother workflow is very important for any process.

	workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	
Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	• We end up prioritization and estimating in planning, so it becomes too big. • Having backlog act as ready queue will help in having shorter planning session. • It will give flexibility to Product owners.
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	• Yes, e.g. if someone doesn't have a task he shall first ensure if there is anything to be reviewed.
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action	• Sprint Reviews and other ceremonies are working nicely in the current process.

	points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty? Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to	• We shall have the sprint focus but need added flexibility of handling new requirements inside a sprint. • If ready queue gives us flexibility it is good to have, so we can take something out if we want to bring something in, having backlog act as ready queue will be good. • We can have it weekly because that brings in some cycle. • Scrum gives stability but we need mechanism if we bring in new tasks inside a sprint, we shall a similar sized task of the sprint. • If the spec is bad it is difficult to make a good quality feature, so regular, frequent grooming is very important. • If backlog is in good shape it will solve

	the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	most of the problems. • Frequent weekly backlog grooming used in other teams has helped, so we shall try that. PO's shall present new epics and we create stories and estimate them. Involve team in defining specification. • We must have PO's which are technically sound as well as good business knowledge.

4. Theme interview with Head of Quality Assurance Team

Date:	February 12, 2014 14:30-15:15
Participant:	Erwann Cleudic (Head of Quality Assurance Team)

Topic Of Interview	**Questions**	**Summary of Field Notes**
General Questions	What are the main requirements for the new process	• Teamwork-Team shall be more involved in the new process and shall own responsibility. • New process shall help us in reducing speed and not to get in panic state. • Smooth workflow, tasks shall not get stuck in some development stage.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• In Progress, no developer shall work on more than 2 tasks simultaneously. So twice the no of developers may be a good idea. • In Review, at least some developer shall

		be available to review the task. Developers must review a task before starting anything new. • In Testing, limit can be equal to no of developers.
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• In sprint planning we shall not worry about velocity and take in how much team can-based upon capacity in hours. • Our management is using velocity wrongly and are even adding velocity from different teams to know the overall performance.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size. Why more focus on tasks than bugs?	• I suggest we continue estimation and estimate in hours rather than story points.
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• As long as developers can help and test the tasks under the supervision of QA member, it will improve workflow. • Having smoothing workflow is important, and if we follow some rules workflow will be smoother.

Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	• We need to follow some code conventions, improve code review process within the team so that quality can be improved. • Only Product Owners shall bring in new stories, team shall not accept tasks from others. • Before starting new tasks, developers shall first review a task in case there is task to be reviewed.
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product	• In sprint retrospectives, Teams shall invite Product Owners to the retrospective discussion; they are the first class team members and they are the ones who know how a team performed based on sprint expectations, so their feedback is vital. Also PO’s drive Product Backlog grooming so they shall be involved.

	Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty?	• Pulling tasks from product backlog is a good approach
	Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods	• Both Product Owner and Scrum Master shall protect the team in pressure situations. • To reduce pressure if we bring in some tasks we shall take out similar sized tasks from the sprint.
	What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints?	• Product backlog shall be in shape and prioritized. Grooming shall happen before we take in new tasks into the sprint.
	How can we improve the quality of stories, poor specifications and prioritization of stories?	• Only Product Owners shall bring in or write user stories.

5. Theme interview with Product Owner of the Team

Date:	February 17, 2014 14:20-15:00
Participant:	Mika Mannermaa (Product Manager)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	What are the main requirements for the new process	• Our process does not need to be perfect, but it must offer a framework where we can constantly improve.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• Tasks shall be limited but finding the right limit shall be left to the team and can be further improved in retrospectives.
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Velocity is not that relevant as long as PO's know how long or how big a particular story is. Also important point is how much team can accomplish within a sprint. It shall not be used as a performance indicator.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size.	• Estimations helps PO's in prioritization. As long as there is some rough estimation, it is good no matter whether we use story points or some other means for estimation. • Me must split big stories for visibility purposes
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck?	• Workflow is important and tasks shall move from left to right without any bottlenecks. • We shall collaborate by helping others; it

	Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	can be for instance helping QA team with testing tasks or helping PO's estimate/improve stories.
Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	• Even in Scrum we shall have some kinds of team rules that can help the team to collaborate better.
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives,	• So far I have not been involved in retrospectives, but I would like to be involved to improve process at team level.

	many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	• Having smaller list of action points is always helpful, so ignore what is waste. • Sprint demonstration is very important aspect of the current process as far as PO's are concerned. This is where PO's know how team has performed when it comes to sprint goals. It further helps PO's in improving backlog based on overall sprint feedback.
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps in decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty? Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have	• It is better to have shorter pull sessions than mammoth sprint planning sessions. • Size of ready queue is not important as long as we often update the queue with prioritized set of tasks within a sprint. This will provide PO's with flexibility. • We need some flexibility in the process; scrum is very strict for us. • PO's shall ensure that product backlog is in order; we have been lacking this lately. We need to break big epics into smaller

	seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	stories and therefore we must have weekly grooming sessions with Product Management as well as sales guys. • We shall sync backlog grooming sessions ahead of sprint planning meetings.

6. Theme interview with Senior Team Member

Date:	February 21, 2014 11:20-12:00
Participant:	Lauri Oherd (Senior Software Engineer)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	What are the main requirements for the new process	• We need a process which is a bit flexible but at the same time manageable.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• Progress Queue: Maximum two tasks per developer • Review Queue: At least one guy shall be able to review, so one less that number of developers. • Testing Queue: Same limit as that for Review queue.
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Focusing on velocity creates hurry just to achieve sprint goals; it is good if we can take hurry out.

	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size.	• Story estimation can hardly be accurate, but at-least it gives a rough idea about task complexity based on developers knowledge • It is very hard to have similar sized task. Scope, nature of the task and possible re-search work needed make it harder to have similar sized backlog item.
Workflow	In order to keep flowing going what shall we do in case Test-ing stage becomes a bottle-neck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• It is important that we help others when something becomes a bottleneck, For ex-ample, reviewing others code will help de-velopers to learn more. And testing others tasks will help to broaden the overall knowledge of the system.
Sprint Plan-ning	What do you think are the prob-lems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	• Planning shorter periods might help to take hurry out.
Explicit Work Policies & Quality	Do you thing that setting explic-it work policies will help in maintaining the work flow within a sprint?	

	What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	• Sprint retrospectives constantly help the team to achieve improvements if they are effectively conducted. • We shall discuss ideas in retrospectives and then experiment those in real sprints.
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty?	• In general for developers, it is a very good idea to have prioritized list which indicates what to work on next. However the reality is there are still pressure situations from the business side to get everything done.

	Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	• It is very hard to work in hurry, developers can do it over a shorter period of time, but it cannot last for ever. So best is try to take hurry out. • In the long run, pushing workers to limits is very harmful for the companies. It is a management failure if workers overwork. • Product owners need to be active and take responsibility so that stories and backlog is in order.

7. Theme interview with Senior Team Member

Date:	February 21, 2014 15:15-16:00
Participant:	Perry Mitchell (Senior Software Engineer)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	What are the main requirements for the new process	• Primarily it shall ensure better quality of the code, Secondly it shall make sense and relatively be logical for the team to follow.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• Progress Queue Limit: A developer cannot do more than one thing at a time. However working on 2 tasks is within the comfortable zone but 3 tasks means kind

		of pushy situation. • Review Queue Limit: With review we do not need to be too strict. However, if needed we can put the limit to equal to the number of developers. • Testing: Probably same as review limit
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Velocity is often misused and team may even inflate the estimations to get better velocity. Secondly, there is too much interest from the management side regarding velocity.
	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size. Why do we focus on new tasks rather than bugs?	• It is difficult to estimate stories in case task complexity is very large. And even team can inflate story points. • In case team fixes many bugs within a sprint compared to number of new tasks, management feels that team's output is probably less because velocity is lower. However if you look from the quality perspective, team's output is better in this situation. There is problem how management looks at metrics and we shall change that as well.
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• It wouldn't bother developers in a short term basis to help testers with testing, because at the end of the day we must ensure smoother workflow. But if that is happening daily then there is some problem with the process and we need to reassess the situation.

Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint? What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	• We need to have some work policies for improving our team collaboration. For instance, policy not to handle tasks from any other source than PO’s.
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstra-	

	tions help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty? Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	• As long as we don't focus on filling our sprint backlog with tasks, it will be very fruitful and solve many of our team problems. • Using Ready queue might even improve our velocity because our focus will not be velocity but rather prioritized set of tasks. And we pull in the tasks whenever we run out of those.

8. Theme interview with Quality Assurance Team Member

Date: February 19, 2014 11:30-12:30

Participant: Ashish Mohindroo (Senior Software Tester)

Topic Of Interview	**Questions**	**Summary of Field Notes**
General Questions	What are the main requirements for the new process	• Improve quality of work and handle pressure situations in a better way.
Work In Progress Limits	What is the best limit for WIP items in your opinion? (In Progress, Review, Testing).	• Progress Queue Limit: A developer shall only prefer to work on single task at a time, exceptionally two tasks, so limit can be less than twice the number of developers. • Review Queue Limit: A developer shall always be available to review a task, so the limit of less than the number of developers in a team is good. E.g. N-1 is a good limit. • Testing: Probably same as review limit so that queue doesn't grow bigger
Metrics and Estimation	Do you think that metrics like calculating velocity is a good indicator of estimations or how well a team is performing? What is your opinion on average cycle time to complete a task?	• Velocity gives some idea of how many stories team can complete within a sprint provided team doesn't wrongly estimate stories • Our tasks are so different both technology and complexity wise so average cycle time might not provide exact estimate of a task.

	What is your opinion on story estimations within a sprint, is it wastage of time or do you thing management is using them wrongly? Is it more effective to have similar smaller tasks rather than having stories with varying size.	• Estimation is good provided the main purpose is to provide rough estimation to PO's. However, often tasks are too complex to estimate.
Workflow	In order to keep flowing going what shall we do in case Testing stage becomes a bottleneck? Is it important to maintain the workflow rather than having incomplete stories by the end of sprint (For example, stories get stuck in different columns)?	• In addition to sprint tasks, QA testers are also working on other tasks like automation testing and release testing. So if developers can help in certain tight situation, it will help to remove bottlenecks in testing stage and accordingly improve workflow.
Sprint Planning	What do you think are the problems with our sprint planning meetings? And where can we improve? What is your opinion on having short planning session and not to plan for the whole sprint? Do you think it will bring in value by concentrating on prioritizing what we want to do next rather than in the whole sprint?	• Even if we plan a sprint, often additional tasks are added to sprint because of tight customer commitments. So it might be a good idea not to plan for the whole sprint and only work on prioritized sprint backlog items.
Explicit Work Policies & Quality	Do you thing that setting explicit work policies will help in maintaining the work flow within a sprint?	• Having a set of rules at place will definitely improve team collaboration provided whole team adhere to those rules. When everyone knows what to do in certain situations, it reduces dependencies on oth-

	What are the quality requirements for the new model? What are the ways that can help the team to remove disconnect between Quality Assurance and the development team?	ers and improves time utilization. • We shall follow the Definition of Done at each and every stage of development to improve quality, which includes writing unit test cases, reviewing code according to certain guidelines, etc. • By improving quality of code, follow code conventions and finally by improving specifications of stories which include good acceptance criteria definitions.
Sprint Retrospectives and Sprint Reviews and Sprint Demos	How do you think sprint retrospectives work in your current team, any ideas on how they can be made effective? Within Sprint Retrospectives, many times it happens that team has too many action points but no right owners to handle those? How do you think we can improve there? Do you think sprint demonstrations help to keep Product Owners/Product Management up to date of what has happened in the sprint? Any opinions on improving sprint demonstrations?	• All these ceremonies are important and helpful so far.
Backlog, Ready Queue and Backlog Grooming	What if we have a ready queue between the main backlog and work-in-process queues (Ready queue is something which is prioritized set from the main backlog and helps is decoupling assignment and prioritization)? And how often you think we shall update them? Weekly or as soon as queue is about to empty?	• Working on smaller list of prioritized tasks will definitely give flexibility to include changes in priorities within a sprint. So why not have our backlog act as ready queue i.e. short and prioritized list of tasks.

	Many times it happens that there is pressure from the team management to take on new tasks inside the sprint rather than focusing on the ones committed during sprint planning? do you thing we shall be flexible in scrum, how about not planning for whole sprint but rather plan for shorter periods What do you think we can improve upon our product backlog grooming, as we have seen Product Owners often bring in incomplete stories to the sprints? How can we improve the quality of stories, poor specifications and prioritization of stories?	• The only way to improve quality of stories is to have Product backlog grooming ahead of sprint planning meetings. PO’s shall take the initiative rather than leaving it up to the team to have grooming sessions. Further, only PO’s shall write stories into the backlog.

Current State Analysis Interview Discussions

1. Interview Discussion with Team Leader

Date:	February 5, 2014 14:00-14:30
Participant:	Edvard Karvinen

Topic Of Interview	Questions	Summary of Field Notes
General Questions	Do you think we need changes in our process?	• Probably some changes are needed and we are very flexible team but focus shall not be too much on process and we need to get things done as well. SO I prefer a balanced approach in a process which is flexible and it helps us to get things done as well • Since out team is flexible enough, we shall not let others misuse us and get things done in hurry. • We shall not have black and white way of doing things rather we shall try to fit ourselves to the process and be flexible • We shall not do things in hurry, because that might lead to testing in hurry and we might miss out on many things. But on the other hand if some customer is facing important problems we shall not then thing too much about process but try to help in solving the customer problem. • Scrum is too strict approach for us, better is Kanban or scrum-ban that can give us flexibility. •
	What has been working well in the current model which is loosely based on Scrum? And what we can improve in different stages?	• Our quality is pretty good at the moment • We really do research before taking on important tasks, we speak as a team • Quality still needs improvement •

	What are the main challenges in your existing Scrum model?	• Specks are not clear or constantly changing but we cannot stop working because of that as we are a start-up and we are trying to build something new, so it's better to build on iterations in those cases. And as long as management doesn't want commitments in those case because specs are constantly improving it shall be ok • From the process wise we shall not need approval from top management for small trivial issues, product management or design shall be able to handle those issues. • We shall not have too much pressure from product management. • Sometimes tasks get stuck in some stage.
	When do you think agile scrum doesn't work in a team or an organisation?	
	What are the main factors which can lead to the success of software development projects? What is successful project for you?	

2. Interview Discussion with ex-Scrum Master

Date:	February 14, 2014 14:00-14:30
Participant:	Maksim Luzik

Topic Of Interview	**Questions**	**Summary of Field Notes**
General Questions	Do you think we need changes in our process?	• People start working on tasks and later requirements or acceptance criteria changes which changes the scope of tasks • Pressure from the team management to get other things done besides what was planned in sprint planning
	What has been working	• In scrum, team plans and knows what they are tak-

	well in the current model which is loosely based on Scrum? And what we can improve in different stages?	ing inside a sprint based on team velocity but the problem is plans are not complete in terms of requirements. • It seems that because of the tight schedules and changes in requirements, our scrum process in other teams is also moving to Kanban mode just like the team in discussion • One thing that we shall improve is acceptance criteria shall come from Product Owners directly when we take tasks inside a sprint rather than a developer specifying it later. This way less bugs happen as acceptance criteria is clearly specified.
	What are the main challenges in your existing Scrum model?	• We concentrate on stories only and we don't prioritize the bugs. • When the whole sprint is full of tasks based on team velocity there is hardly any margin for taking bugs inside a sprint. And if other critical or major bugs arrive within a sprint it leads to incomplete stories. • Often Time is fixed, when PO's change scope of the tasks, either you change the time or deprioritize something
	When do you think agile scrum doesn't work in a team or an organisation?	
	What are the main factors which can lead to the success of software development projects? What is successful project for you?	

3. Interview Discussion with Head Of Software Development and Services

Date:	January 31, 2014 14:0-14:30
Participant:	Timo Valtonen (Head of Software Development)

Topic Of	Questions	Summary of Field Notes

Interview		
General Questions	Do you think we need changes in scrum process?	• Yes we have a fragile process and do not follow scrum disciplines completely. • Estimation accuracy and quality is fluctuating. • Cutting corners in process. • Technical Debt.
	What has been working well in the current model which is loosely based on Scrum methodology?	• The Kanban angle sometimes provides us flexibility and team can change direction according to customer's orientation. • Scrum is providing some stability to the team as we lock ourselves for 2 weeks.
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• Product Grooming is not effective and process has been broken. • We had product owners inside the team and it was not working when team leader is the product owner as well. • We are bringing new tasks inside the sprint but not taking anything out from the sprint that changes the scope. • Poor estimations and bad quality of specs at times.
	What are the main factors which can lead to the success of software development projects? What is successful project for you?	• Completing work on time with better quality and workflow is smoother. • More transparency and visibility.

4. Interview with Head Of Quality Assurance Team

Date:	February 12, 2014 14:00-14:30
Participant:	Erwann Cleudic (Head of Quality Assurance Team)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	Do you think we need changes in scrum process?	• Yes I believe we need changes and there is no particular process followed in Applications Team. • Team needs to improve quality delivery.
	What has been working well in the current model which is loosely based on Scrum methodology?	• With the current process, team has been good at making prototypes but not products. • We need to improve the whole process; scrum is not working so well in this team.
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• We need to improve retrospectives. • We need to follow some code conventions, improve code review process within the team. • Whole team must follow Definition of Done; follow certain team rules and conventions related to quality. • Product Owner shall not be from within the development team.
	When do you think agile scrum doesn't work in a team or an organisation?	• Scrum doesn't work because team is accepting tasks from everyone such as management, designers etc. • Week Product owner always bring in pressure to the team. • Changing priority of sprint tasks all the time breaks scrum.
	What are the main factors which can lead to the success of software development projects? What is successful project for	• Team work results in success and whole team shall decide on how they want to work.

	you?	

5. Interview Discussion with Product Owner of the Team

Date:	February 17, 2014 14:00-14:20
Participant:	Mika Mannermaa (Product Manager)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	Do you think we need changes in scrum process?	• Our current process is not structured. It is partly tool problem and some problems are within our team. • We don't want to have too structured process as well. • For prototype creation, existing process is OK but it cannot handle product development.
	What has been working well in the current model which is loosely based on Scrum methodology?	• Our planning meeting are getting better but still they are longer.
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• Lot of work is done by the team but usually stories are so big so that they are not visible by the end of a sprint. We shall start splitting big tasks.
	What are the main factors which can lead to the	• A clear vision is important even though goals can be fuzzy.

	success of software development projects? What is successful project for you?	

6. Interview Discussion with Senior Team Member

Date:	February 21, 2014 11:00-11:20
Participant:	Lauri Oherd (Senior Software Engineer)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	Do you think we need changes in scrum process?	
	What has been working well in the current model which is loosely based on Scrum methodology?	• Having sprint cycles helps team to focus; otherwise developers would lose sense of time. • Morning daily stand-up meetings keep me focused as everyday morning I have to answer what I was doing yesterday and it feels good in case I managed to complete some task
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• Sometimes there are too many meetings and we shall reduce those. • Very often our sprints are stressful and work comes from many directions. We need to have some kind of filter which blocks unnecessary tasks. • We need to improve prioritization of tasks.
	When do you think agile scrum doesn't work in a team or an organisation?	

	What are the main factors which can lead to the success of software development projects? What is successful project for you?	

7. Interview discussion with Senior Team Member

Date:	February 21, 2014 15:00:15:15
Participant:	Perry Mitchell (Senior Software Engineer)

Topic Of Interview	**Questions**	**Summary of Field Notes**
General Questions	Do you think we need changes in scrum process?	• Our process is easy to follow but it is hard to treasure a process when nobody follows it. Everyone outside our team somehow has negative impact on our team's process and that damages the usefulness of our process. If we all adhere to our process it will be smoother to follow.
	What has been working well in the current model which is loosely based on Scrum methodology?	• Everything from the Scrum model is very useful and probably timed very well within our team.
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• Our team shall not be influenced by the outside management, which usually breaks our process. • Product backlog grooming from the Product management is missing, they wait for the development team to start grooming. • Management, sale or someone outside the team is directly pushing tasks through developers and not adhering to the process. • Trouble with estimation of stories, complexity of stories is so large that we have trouble estimating those. It is hard to adhere to the process when our

		estimations are wrong. • Partially Scrum is working for our team but then it shall be either everybody is working on Scrum or nobody is working on Scrum.
	When do you think agile scrum doesn't work in a team or an organisation?	• When everyone doesn't adhere to the same process it will fail.
	What are the main factors which can lead to the success of software development projects? What is successful project for you?	

8. Interview with Quality Assurance Team Member

Date:	February 19, 2014 11:00-11:30
Participant:	Ashish Mohindroo (Senior Software Tester)

Topic Of Interview	Questions	Summary of Field Notes
General Questions	Do you think we need changes in scrum process?	• Our team is doing both prototype work and Product development work, so current process is not suitable for both. • Our process is not clear when it comes to prototype development. • Often pressure from the management creates problems and team shifts to some other process mode like Kanban.
	What has been working well in the current model	• Daily stand-up meetings and Retrospectives are good to remove impediments and constantly im-

	which is loosely based on Scrum methodology?	prove our team's way of working but when it comes to how team works within a sprint and product backlog grooming, improvements are needed. • Effort Estimation by team is very helpful. • Sprint Reviews are very helpful and feedback from key stakeholders is very vital for future implementations.
	What are the main challenges in the existing process model being followed by the team under discussion? What are the key challenges while implementing agile methodology in your team?	• In planning meetings, implementation/design of stories is not considered and that often leads to inaccuracy of estimates. • Incomplete stories in backlog. • Changing the scope of stories also creates problems in testing, which further leads to delays • Adding more tasks in the middle of the sprint usually breaks our scrum to Kanban mode. • Our code is not well documented and sometimes developers don't follow definition of done, code is committed without review, and even code is very complex when it comes to maintainability. • Sometimes our planning meetings are fruitless because prioritization or scope of new requirements was unclear. • From the quality perspective, it is bad to fix bugs on more than one active release branch. This often leads to merging issues. •
	When do you think agile scrum doesn't work in a team or an organisation?	• When everyone doesn't adhere to the process or when there are too many customer requirements team needs to complete in the middle of the sprint.
	What are the main factors which can lead to the success of software development projects? What is successful project for you?	