Comparative Study on the Software Methodologies for Effective Software Development

Sabbir M Saleh, M Ashikur Rahman, K Ali Asgor

Abstract— The software development models, which are also known as Software Development Life Cycle. There are too many software development models now, mainly waterfall, spiral, Rapid Action Development etc.; now a days Scrum, Kanban and Extreme Programming are widely used, which are in Agile Family; by the software industries. These prototypes have advantages and disadvantages as well. In this paper we will discuss about waterfall, Scrum, Kanban and Extreme Programming; the main objective of this investigation is to symbolize different models of software developments and make a judgement amongst them to show the functionalities of each model separately and thoroughly. And we hope that this study will help the software industries to take the correct decisions about software development models before running a development for new software.

Index Terms— Software Management Methods, Software Development, SDLC, Comparison between four models of Software Engineering, Scrum, Kanban, XP.

1 Introduction

Software development methods and life cycles are termed in the software engineering, which defines the shapes through which software develops. The development attentions on the product. It's defining the stage through which a software authorization after it launches. It's to be made to when software arrives into processes and finally deployed [1]. A software process model is an abstract demonstration of the design, or characterization of the software procedure [2]. In software improvement, process models are applied to accomplish various concerns associated with price, time period, and quality and fluctuating requirements of customers' etc. The reasons of project failure could be project development team, dealers, clients and other stakeholders; but the most common causes for project failure are embedded in the project management method itself and the aligning of Information Technology with structural philosophies [3].A literature review of developments in software growth show that most software tasks are deliberated as partial failures due to difficulties ascending from the software development methods [4][5]and schemes were providing for successful the software process regardless of the actual process models used. We will research the process models based on some parameters to develop an effective software.

2 Background Study

Several number of popular software development methodologies have been presented in the last couple of years such as -

Agile development is appealed to be an innovative and receptive effort to address users' needs concentrated on the prerequisite to distribute applicable working business applications faster and inexpensive. The software is usually provided in incremental or iterative fashion. The agile expansion approaches are typically concerned with continuing user involvement through the application of design teams and special workspaces. The provided increases are likely to be minor and inadequate to little supply stages to confirm quick end. The organization strategy consumed relies on the imposition of *time boxing*, the stringent supply to goal which edicts the possibilities, the collection of performances to be provided and the alterations to encounter the targets. Agile development is mostly convenient in situations that change gradually and execute difficulties of limited results. Agile approaches support the conception of parallel increase and distribution within an overall intentional framework.

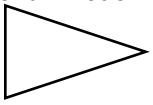
It is method to increase, based on the progress and provision of very insignificant augmentations of performance. It trusts on continual code enhancement, user contribution in the development team and pair sensible programming. It can be problematic to keep the interest of consumers who are involved in the process. Team members may be incompatible to the penetrating participation that describes agile approaches. Highlighting fluctuations can be problematic where there are several stakeholders. Agreements may be a difficult as with other methods to constant development. [6]
An enormous number of journalists release and technical journals treat the Scrum as the greatest method to software development. However, the original Scrum method is not acceptable for successively work in agile environment in a networking management. Due to that, researchers lengthy the Scrum-based model [19] and the Kanban methodology to make the most of both and find a more acceptable method for working software development in powerfully scattered agile background.

A Software process model is an abstract representation to de-

- *Sabbir M Saleh is currently a Lecturer at Department of Computer Science and Engineering at University of South Asia, Bangladesh, PH-01785547626. E-mail: sabbir@southasia-uni.org*
- *M Ashikur Rahman is currently a Lecturer at Department of Computer Science and Engineering at University of South Asia, Bangladesh, PH-01703259429. E-mail: ashik@southasia-uni.org*
- *K Ali Asgor is currently an Adjunct Lecturer at Department of Computer Science and Engineering at University of South Asia, Bangladesh, PH-01711873008. E-mail: aliasgorpavel@gmail.com*

fine the process from a certain standpoint. [2] There are various types of general models for software developments. This study will view the following four models:

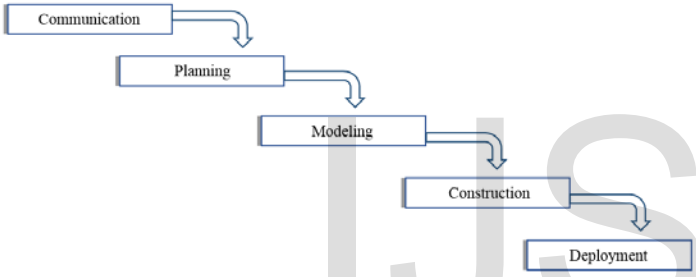
1. The Waterfall model
2. Scrum
3. Kanban
4. XP



Agile Process Models

Prescriptive software process models have been applied for many years in an effort to bring order and structure to software development. Each of these conservative prototypes suggests a slightly dissimilar procedure flow, but all accomplish the similar set of common framework actions –

- Communication
- Planning
- Modeling
- Construction
- Deployment [8]



3 Software Process Models

The Waterfall Model

The primogenital pattern for software engineering is the Waterfall model, sometimes called the classic life cycle, suggests unorganized, progressive method to software development. One phase initiates when another ends. Useful process model in circumstances where requirements are well-defined and constant, and work is to be proceeding to accomplishment in a direct mode.

Fig. 1. General Overview of the Waterfall Model

Communication: Project Initiation, Requirements Gathering and Analysis.
Planning: Estimating, Scheduling, and Following.
Modeling: System Analysis and Design.
Construction: Coding and Testing.
Deployment: Delivery, Support and regular Maintenance, Feedback.

Advantages of the Waterfall Model

- Identifies systems requirements long before programming begins
- Minimized changes to the requirements as the project proceeds.
- Easy to recognize and appliance.
- Generally cast-off and acknowledged.
- Categorizes deliverables and indicators.
- Mechanisms fit on advanced produces and feeble developer teams.

Disadvantage of the Waterfall Model

- It is often difficult for the customer to state all requirements explicitly
- The customer must be patient
- Idealized, doesn't match reality well.
- Problematic and exclusive to make variations to papers.
- Significant clerical upstairs, expensive for small teams and ventures. [9]

Agile Process Models

Contrasting the waterfall model, agile software development is iterative or cumulative. In agile software development (ASD) requirements and solutions are said to advance thru the development of the project. [10]

- An agile process must be flexible
- It must adapt cumulatively
- Involves customer feedback
- An effective substance for customer feedback is an working prototype or a portion of an functioning coordination
- Software augmentations must be provided in short time episodes
  - Assists the customer to assess the software augmentation frequently
  - Deliver basic feedback to the software team

This study will view the following Agile Process models:

- Scrum
- Kanban
- XP

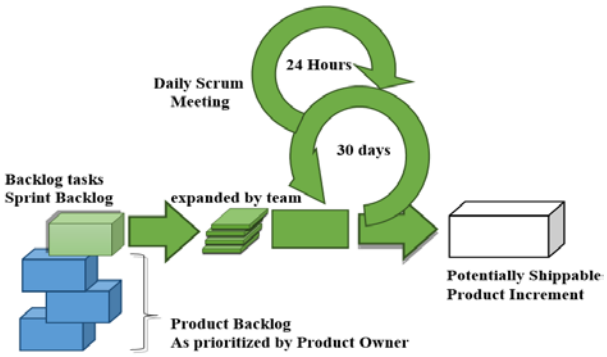
Scrum

Scrum was established by Jeff Sutherland in 1993 [11] and its objective is to be an improvement and organization methodology that follows the principles of the agile methodology. The Scrum team is collected by [12]:
Team: it's the development project team, poised by up to ten developers in which each member has a precise skill. However, members are not banned from performing task different from their skill. Thus, the team will become more combined and teams' members will know better the software, minimalizing the impact of another member's sacking.

• *Product owner*. He is the one with the responsibility on the

software functionality specification and to solve any doubts that might arise during development. He is the client's representative that must watch the project closely and help in the construction of a software that answers completely to the client's needs.

• *Scrum master.* He is the in charge to lead the team and to evade any dashes that might arise during the development.
An obstacle is something that might hamper a member from accomplishment his work. For illustration, requests to perform happenings not related to the project, problems in the test server, complications with the technology and unplanned prerequisites might be examples of sprints that might cause problems to the *sprint.*



Scrum is based on practices represented by –

- Daily meetings,
- Sprint planning meetings,
- Sprint review meeting,
- Backlog sorting and
- Release presentation [13]

Fig. 2. Sprint [14]

Advantages of Scrum Methodology

- It is easier to distribute quality software in a planned time.
- Due to tiny sprints and continual response, it suits easier to handle with the variations.
- Daily scrum meetings make it promising to measure specific production.
- Alike any other agile approach, this is also iterative in nature. It needs constant response from the customer.

Disadvantages of Scrum Methodology

- If the development team members are not dedicated, the project will either never complete or fail.
- It is good for small, fast moving developments as it works well only with trivial development team.
- This approach needs qualified team members merely. If the team contains of people who are beginners, the project cannot be accomplished in deadline.
- If any of the members leave during a development it can have a massive inverse consequence on the project development.

Kanban

Kanban accomplishes the Lean philosophy in practice [15], [16] and is one of the key operation management tools in Lean manufacturing [17]. It drives project teams to visualize the workflow, limit work in progress (WIP) at each workflow stage, and quantity the cycle time [18].

Advantages of Kanban Methodology

- Improve portfolio and decrease product undesirability.
- Diminishes surplus and clash
- Delivers tractability in manufacture
- Intensifications productivity
- Condenses total price
- Progresses stream
- Avoids failure

Disadvantages of Kanban Methodology

- It is less operative in shared-resource circumstances
- Flows in blend or mandate cause problems because Kanban accept sun changing monotonous manufacture policies.
- Kanban in itself doesn't reject inconsistency, so changeable and lifelong down times could dislocate the system; poor quality in terms of clash and rephrase also mark its good running. Kanban systems are not suitable for industrial locations with short manufacture runs.
- An interruption in the Kanban system cans consequence in the complete line end.

XP-Extreme Programming

Extreme Programming is one of the popular Agile Practices. It has already been established to be very effective at many organizations of all different sizes and productions global. Extreme Programming advances a software development in five essential ways-

- Communication
- Effortlessness

- Feedback
- Respect and
- Courage

Advantages of XP Methodology

- Client priority increase the chance that the software manufactured will actually encounter the requirements of the consumers.
- The focus on minor, incremental relief declines the risk on development.
- By presenting that a method works.
- By setting functionality in the hands of the customers.
- Incessant testing and combination assistances to raise the superiority of development.
- XP is friendly to programmers who usually are grudging accepting a software method.

Disadvantages of XP Methodology

- XP is geared to a solo development, established and sustained by a sole team.
- XP is incompetent in surroundings where a client or administrator claims on a complete requirement or strategy earlier they start programming.
- XP is incompetent in surroundings where programmers are disconnected geologically.

XP is incompetent to work with arrangements that have expandable matters.

4 Related Work

According to Holman (2002), job design, alongside with job control, has a optimistic alliance with worker well-being. Additionally, even though the author hypothesizes that a worker in a business can promote from job monitoring, a high level of monitoring has an unenthusiastic effect on interests (Chalykoff & Kochan, 1989; Holman, 2002). Human resources practices and team mentor support factors are designed to better reproduce supervisory aspects in administrative center. High-level support from administration with higher control of job design is a foremost issue in manipulating employee well-being (Holman, 2002; Kular, Gatenby, Rees, Soane, & Truss, 2008). Research on organizational behaviorism has also shown that the phases of employee well-being absolutely influence specific's job fulfillment (Wright & Bonett, 2007).

Workers and developers feel happy at workplaces when their job distinctiveness equivalent their own qualities (Warr, 2007). Precedent research has exposed that agile practices could effectively encourage developers and amplify their job fulfillment (Melnik & Maurer, 2006; Sharp & Robinson, 2008; Tessem & Maurer, 2007), as they are developed to ensemble people's requirements. For example, by using user story cards for the period of planning game activity, small dollops of functionality are discussed recurrently with consumer, allowing team members to preserve their sensitivity of passion (Syed-Abdullah et al., 2006). In addition, throughout pair programming, programmers are requisite to solve programming tribulations in pairs and habitually substitute associates, both of which endorse teamwork and a sense of project tenure surrounded by other team members, and consequently augment their well-being. Agile job distinctiveness, given that they place importance on the value of unremitting criticism and numerous liberate, are able to diminish hopelessness amongst workers and developers. This is as the practices endorse communal surroundings, which boosts members to have a clear track towards achieving development targets. Therefore, it is hypothesized that agile teams will practice a superior rank of well-being measured to non-agile teams. They have chosen some parameters to measure the effectiveness of these methodologies for software development. But unfortunately they did not consider the effect of required time for software development.

5 Problem Statement

There are couples of software methodologies which are widely used by many software development organizations. But sometimes, organizations especially new companies face problem to select the software methodology for a specific software project. Wrong selection of software development methodologies often makes those project failed.

As we see in section III (related work), there are several number of works have been presented for comparative study of software methodologies. But in our investigation we find several flaws or shortcomings for evaluating the software methodologies. For example, we find that many critical parameters for examples Communication, Requirement Specifications, Cost, Resource Control, Simplicity, Risk Analysis, Feedback from User, Customer Priority, Precondition, Elasticity, Practicality, Implementation, Usability etc. - are not considered in their study. So that sometimes their suggestion for choosing the software development methodologies might not effective in this regard. Besides, their way of measuring the evaluating parameters are not correct, since they did not define the relevant values of those parameters. Additionally, it is not mention clearly how these values of the relevant parameters come from.

Therefore, missing parameters are required to be considered for a proper comparative study of the software development methodologies.

6 Proposal For Comparing Software Development Methodologies

In our research, we are going to deliberate about which process to choose for best methodology to deliver the quality software to the customer.
Before determining the process to be used, we should get solution for some queries.

1. How steady are the requirements?

2. Who is the ultimate consumer for the software?
3. What is the opportunity of the product?
4. Where are the Development teams placed?

Our Development is to discuss for the prerequisite given by the customer which methodology to be used. Let us have a comparative study which process will be active in the below processes and the Advantages & Disadvantages of choosing the model.

For this research and comparative studies between different Software Methodologies we have chosen four types of methodologies, which are The Waterfall Model, Scrum, Kanban and XP.

We have selected these methodologies, because these four models are regularly followed by several software development industries. In these process models Waterfall is much known to all, and now Agile is very popular globally, and Scrum, Kanban and XP are the family members of Agile Software Development (ASD).

Waterfall Model is a further name for the more long-established approach to software development. It's called 'waterfall' as this category of development is often planned using a Gantt chart – developers complete one segment before touching on to the next segment. In Waterfall, it's rarely aim to return to a segment once it's finished. As corresponding, better gets whatever doing right the first time. This process is exceedingly precarious, often more expensive and usually fewer professional than further Agile methods.

Scrum process carries far less risk than Waterfall processes. It focuses on providing fully-tested, autonomous, precious, small features. As corresponding, it's diversifying the risk – if one feature goes erroneous, it should not get in touch with a new characteristic. With that said, developers still plan work in iterations and they will still release at the end of iteration.

Kanban refined as a sub constituent of the Toyota Production System. In Kanban the system is visualized: work is wrecked down into small, disconnected items and written on a card which is spellbound to a section; the board has singular columns and as the work progresses through different phases the card is moved consequently. In Kanban the quantity of objects that can be in progress at any one time is exactingly imperfect. The usual time it takes to absolute an item is tracked and optimized so that the process becomes as efficient and predictable as possible. The eradication of misuse is overriding.

In conjunction with shorter iterations, some other important things which distinguish XP from Scrum are: XP teams work on items in an authoritarian precedence classify while a Scrum might not unavoidably tackle each item in precedence order once in sprint. XP teams can fetch new objects of effort into iteration and switch out objects of corresponding size if the consumer chooses on a new precedence. In conditions of relationships, the responsibility of the client in XP is very parallel to that of the Product Owner in Scrum – in that they assist to write user stories, priorities them and are always accessible to developers – despite the fact that less well distinct. Mutually Scrum and XP maintain a daily stand up meeting.

The parameters we have identified for distinguishing these methodologies are:

- Communication
- Requirement Specifications
- Cost
- Resource Control
- Simplicity
- Risk Analysis
- Feedback from User
- Customer Priority
- Precondition
- Elasticity
- Practicality Implementation
- Usability

We have selected these parameter sets to optimize a model solution to a target solution.

By these parameters we can evaluate a better methodology of software development for industries and software development firms.

7 Discussion And Analysis of The Study

ABC is a trivial organization that required to transition their complete development team to Agile. It was an informal alteration to the applications people, tougher to the maintenance people (until they know about Kanban). The ones in the intermediate were the mainframe developers. This organization is in insurance, a production that has lots of legacy backend systems.

Our team has met mainframe developers in our Agile explorations before who were impervious to an Agile method. The details are many but habitually based on the limitation that legacy systems take lengthier to modification and mainframe systems in individual are not adaptable to Agile environment.

In this specific case, our team were enjoyably astonished to see attention, eagerness, at beginning a project using the agile method even though the systems complicated were COBOL, CICS and RPG. Here we want to abstract the knowledge by relating what Agile practices we executed, what experiments we faced and what accomplishments we saw.

Standard Agile Performs

- The development team was small: 3 programmers and a QA also playing the Scrum Master character. There was an enthusiastic Product Owner who was a

Business Analyst acquainted with the problem to be solved. The sponsor, a Product Manager, was also exceedingly convoluted.

- The development team choses a 3-weeks Sprint span based on unusualness with Agile in overall, the remark that tasks and stories might be bigger than in other developments and the understanding that development team members could not protection for separately well due to concentration.
- Complete system policy was incremental.
- Functionalities were prioritized and done in priority mandate.
- The Product Owner and end-users were convoluted on a daily basis.

Challenges to Standard Agile Performs

- The programmers were extremely focused in their technologies.
- Cross-training was inadequate by a vertical learning curve.
- Automated testing was measured to be unbearable. Our team thought this at inordinate length. At the unit phase, well – what is unit in COBOL? It is a structured programming language. Programs are enormous, massive. Accumulates yield an extended period. It is dangerous to run just portion of the code. Testing is done by moving through a debugger and trifling with capricious principles in recollection. This does not loan itself to computerization. Our team observed into screen footage and repetition, which is conceivable, but the subsequent writings are tremendously friable. In the termination, the competent cost of enquiry was too high for the little project timeline.
- Consuming a small development team did reason some postponements when individuals were absentminded.

What We Got

- Throughout backlog estimate, explanation of a story was short-circuited when it developed deceptive that the business was asking for something that was technically unbearable within the downstream host organism. The story, primarily estimated to be moderately large, was merely released. The Product Owner was not troubled at the loss of the features given the fundamentally unlimited cost of the story even though it had initially been protuberant in the inventive project budget. Cost investments were estimated at approximately 100000 BDT.
- Though planning Sprint 2, another great, high priority functionality vaporized when the Product Owner understood that the work done in Sprint 1 caused in an adequately useful explanation to the box of the floor. The functionality was a real-time update of data from one backend system to a front end on additional system. User testing of the first sprint product presented that batch explains had satisfactory price and were much trusting to gadget. Fairy another 100K BDT saved.
- The development team stroked that these investments would not have been comprehended in an old-style project method. The requirements would have been overheated into a design document and applied as well as likely without any conversation with the Professional.
- Variations in other requirements resulted from nonstop review by the Product Owner and were not a subject for the programmers.
- The project was small, but still accomplished so rapidly compared to pre-Agile potentials that the team stayed together to implement another set of functionalities out of the possibility of the original budget.
- The development team, some of who were incredulous, had pleasurable and appreciated the attainment. The Professional was very satisfied as fine.
- The pre-Agile approximation of work was 2 years gone time in this extremely multi-tasked situation. The Agile project was completed in less than 3 months thanks to an absorbed determination. Time worth of currency, which one?
- Cheers to administrative story trimming by the Product Owner and Sponsor, the product conveyed formerly than even the original agile estimated release date.
- Management had a superior familiarity of status without having status intelligences modestly by attending the daily stand-up meetings.

Our Statements

So, did Agile work out well for this all-mainframe project? Indeed, it did. Even though it was not full Extreme Programming in methodological performs, using a Scrum approach and agile principles resulted in both previous Return on Investment and minor cost. Oh, and everybody had great doing it – always a respectable signal.

A software process model is a streamlined demonstration of a software process, presented from a specific perception. A software process model is an intellectual illustration of a software process. Problems solving in software contain of these activities:

- Finding the problem
- Determining a plot for solution
- Code generating the intended solution
- Testing the authentic program

On behalf of big structures, each movement can be enormously composite and procedures and procedures are required to complete it professionally and appropriately. Moreover, each of the basic undertakings itself may be so big that it cannot be controlled in particular phase and must be fragmented into smaller phases. For example, design of a big software structure is always fragmented into several, discrete design segments, initially from a very high level design identifying only the mechanisms in the system to a thorough project where the logic of the mechanisms is identified [7]. The basic undertakings or segments to be performed for developing a software structure are-

- Purpose of System's Requirements
- System Analysis & Design
- Developing or coding of the software
- System Testing as an end user
- System deployment and regular maintenance

Table: Comparison table on Various Process Models

Parameters / Process Models →	Waterfall	Scrum	Kanban	XP
Communication	Initial level	Frequently	Frequently	Initial level
Requirement Specifications	Initial level	Frequently change	Frequently change	Initial level
Cost	Low	Much Expensive	Much Expensive	High
Resource Control	Yes	No	No	Yes
Simplicity	Simple	Complex	Complex	Intermediate
Risk Analysis	Only at beginning	Yes	Yes	Yes
Feedback from User	No	No	No	Yes
Customer Priority	Nil	High	High	Intermediate
Precondition	Requirement clearly defined	No	No	No
Elasticity	No	Very High	Very High	Medium
Practicality Implementation	No	High	High	High
Usability	Basic	Most use now a days	Most use now a days	Medium

8 Future Work

1. To identify a suitable knowledge sharing procedure while accomplishment user centered design in a startup.
2. To test other available software development processes in a startup such as V-Model, RUP etc. and see how well it can be applied in a solo developer startup environment.

9 Conclusion

This paper discussed what software process model is and various process models, also compare them with different parameter and highlight the factors for choosing them. This paper presents the chart based on usage. However, the existing model still can be improving and modified based on less cost, time and high efficient. The developer should find out following aspects-

1. Find out market analysis that why Agile Models [Scrum, Kanban & XP] are Popular now a day.
2. How can improve efficiency of given model?

As we discussed on Waterfall, Scrum, Kanban & XP's advantages and disadvantages, it depends upon the organization which model to choose.

- If requirement changes frequently and smaller projects, deliver product in short period time with skilled resources then we can choose "Agile model [Scrum, Kanban & XP]".
- If requirement is clear, larger project then we choose "Waterfall model".

References

[1] Walt Scacchi, "Process Models in Software Engineering", Institute for Software Research, University of California, Irvine, October, 2001.

[2] Watts S. Humphrey and Marc I. Kellner, "Software Process Modeling: Principles of Entity Process Models", Technical Report, New York: ACM Press, pp. 331-342, February 1989.
[3] B. Boehm and D. Port, "Escaping the software tar pit: model clashes and how to avoid them", Software Engineering Note, Volume 24, Issue 1, pp. 36-48, 1999.
[4] Humphrey, W.S. (1990). Managing The Software Process. Addison –Wesley. U.S.A.
[5] Paulk M, Weber C, Garcia S., Chrissis M. & Bush M.(1993a). Capability Maturity'ModelFor Software, Version 1.1. SEI-93-TR-024. Software Engineering Institute, Pittburgh.
[6] Nabil Mohammed Ali Munassar1 and A. Govardhan, A Comparison Between Five Models Of Software Engineering, *IJCSI International Journal of Computer Science Issues, Vol. 7, Issue 5, September 2010*
[7] Sanjana Taya, Shaveta Gupta, "Comparative Analysis of Software Development Life Cycle Models", IJCST Vo l. 2, Issue 4, Oct-Dec. 2011.
[8] Roger Pressman, *Software Engineering: APractitioner's Approach*, Sixth Edition, McGraw-Hill Publication
[9] Karlm, "Software Lifecycle Models', KTH, 2006.
[10] JENNIFER DORETTE J., Comparing Agile XP and Waterfall Software Development Processes in two Start-up Companies, 2011.
[11] Sutherland, J.; Schwaber, K. The Scrum Papers: Nut, Bolts, and Origins of an Agile Framework. 224p. 2011.
[12] Schwaber, K.; Sutherland, J. The Scrum Guide. 2010.
[13] Bona, C. Avaliação de Processos de Software: Um Estudode Casoem XP e ICONEX. Dissertação (Mestradoem Engenharia de Produção) – Universidade Federal de Santa Catarina, Florianópolis – 2002.
[14] Murphy, C. Adaptive Project Management Using Scrum. In: Methods & Tools - Software Development Magazine Programming, Software Testing, Project Management, Jobs. 2004.
[15] M. Becker and H. Szczerbicka, "Modeling and optimization of Kanban controlled manufacturing systems with GSPN including QN," in *International Conference on Systems, Man, and Cybernetics '98*, vol.1. IEEE, October 1998, pp. 570–575 vol.1.
[16] L. Chai, "E-based inter-enterprise supply chain Kanban for demand and order fulfilment management," in *International Conference on Emerging Technologies and Factory Automation EFTA '08*.IEEE, September 2008, pp. 33–35.
[17] J. Liker, *The Toyota Way*. New York, NY, USA: McGraw-Hill, 2004.
[18] H. Kniberg, "Kanban vs. Scrum: How to make the most of both," 2009, http://www.crisp.se/henrik.kniberg/Kanban-vs-Scrum.pdf [18-Jan-2010].
[19] Sienkiewicz, Maciaszek 2011