ISSN 2278-3091

Volume 9, No.5, September - October 2020

International Journal of Advanced Trends in Computer Science and Engineering

Available Online at http://www.warse.org/IJATCSE/static/pdf/file/ijatcse52952020.pdf

https://doi.org/10.30534/ijatcse/2020/52952020

Combination Of Scrum Methodology And IBM Design Thinking For Dashboard And Report System Development

Ardian Aditama[1], Nilo Legowo[2]
[1]Information System Management, Master of Information System Management
BINUS Graduate Program, Bina Nusantara University, 11530, Jakarta, Ardian.aditama001@binus.ac.id
Information System Management, Master of Information System Management
BINUS Graduate Program, Bina Nusantara University, 11530, Jakarta, nlegowo@binus.edu

ABSTRACT

Choosing correct methodology can bring many positive impacts for each team and user that involved. One of the most common methodology used now is Scrum. The Scrum methodology is known for developing more dynamic, adaptive and flexible systems. This methodology can be combined with the IBM Design Thinking framework in order to bring more understanding of user needs and offering more system value for users. The purpose of this research is to find out how combination of Scrum methodology and IBM Design Thinking can provide a better value for user in a system development. In order to combine these two methods, this research using common Scrum Methodologies and IBM Design Thinking from IBM. This research will combine Scrum and IBM Design Thinking and its deliverables to provide the best value for users. This case study shows that the combination of Scrum and IBM Design Thinking helped the development team in understanding user needs so that results are more effective and accepted by users. As both Scrum and IBM Design Thinking are sharing user-centric values, this study found that Scrum and IBM Design Thinking both complimenting each other in order to produce and provide the best value for users.

Key words*:* System Development, Scrum, IBM Design Thinking, User-Centric Methodology Framework, Improving System Development

1. INTRODUCTION

Today the use of applications in companies has become an obligation and normal things to do in order to achieve the goals and targets of the company. Applications that are made must have added value and innovation so that users can feel their usefulness. Lots of companies are competing to make applications that can help their business users do their daily work. Software engineering is a systematic, disciplined, measurable approach to the development, operation, and maintenance of software or applications [1]. The study began based on the background of the problem that PT. IBM Jasa Teknologi Informasi requires Dashboard and Reporting that currently usually being done manually. Thus, a new Dashboard and Report system need to be developed to be enabling all information will be enabled 24/7 without manual process anymore. They picked Scrum as the system development methodologies in this new system development.

In system development, SDLC is a conceptual framework or structural process of a phase that is in the development of an application from feasibility studies to launch and maintenance in the field [2]. In developing applications, certain methodologies are used, for example the waterfall methodology and the agile methodology. The adoption of methodology in developing this application is always considered useful and is a framework that can help the development team to plan, process, control its products and services in application development [3]. However, not all methodologies are suitable for one company or another as well as the methodology for developing one application with another. Disadvantages of waterfall model are as follows: the testing phase is reached then it becomes very hard to go back to requirement and modify them, high of risk level and uncertainties, is not suitable for complex projects, long development to deliver the result, is not suit for the projects that the requirements are always changing [4].

Scrum is the most well-known and most widely used form of agile methodology, which is about 56% compared to other agile methods [5]. Patton’s User Story Mapping is one approach that can be used in Scrum. Then comes the important question, “Who are our users and what do we build for them?” and implements a constructive but tiring process of contracting them into “User mapped backlogs”. This is a day-to-day process with a lot of thought and hard team work, but it is a valuable investment of time and effort that will be recommended to every Scrum Product Owner [6]. This is where Design thinking can be used.

On the other hand, IBM has a Design Thinking concept that can be used for system development. IBM expands the

Design Thinking method by providing an approach to writing requirements, managing teams, and tracking project progress, including End User feedback during all phases of project development. The purpose of IBM Design Thinking is to expand the principles of Design Thinking applied to develop software that captures user needs with the speed and scale needed for application development. IBM Design Thinking has several modifications, including three practices that are unique to the framework: User sponsorship, Playbacks and Hills [7]. The application of IBM Design Thinking can be used to improve the results of Scrum. Thus, IBM Design Thinking will also be used in this system development in alignment with Scrum.

Both Scrum and IBM Design Thinking are user-centric methodologies and framework. Combining Scrum and IBM Design Thinking may result better system that has more user-centric that will benefit both user and system development in time, cost, flexibility and user-developers alignment. This study will discuss about combining Scrum and IBM Design Thinking in Dashboard and Report system development.

2. LITERATURE REVIEW

2.1. Software Engineering

Software Engineering (1) A systematic, disciplined, measurable approach to software development, operation and maintenance; that is, the application of engineering to software. (2) Study the approach as in (1). However, the "systematic, disciplined, and measurable" approach adopted by one software team might be another burden [1].
We need discipline, but we also need the ability to adapt and agility. Software engineering is a multi-layered technology. The foundation for software engineering is the process layer.

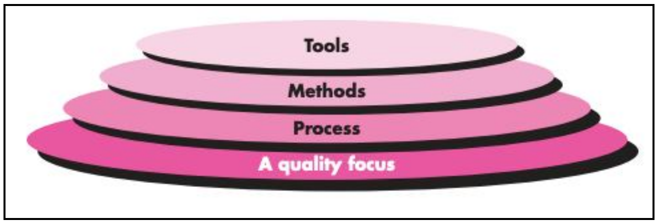


Figure 1: Sofware Engineering Layer
(Source: Pressman [1])

Figure 1 explains that as one of layer of software engineering, a process is not a rigid recipe for how to build computer software. Instead, it is an adaptable approach that allows people doing work to choose and choose the appropriate set of actions and work tasks. The aim is always to provide software in a timely and adequate quality to satisfy those who have sponsored the creation and those who will use it [1].

The software engineering process framework builds the foundation for the software engineering process by identifying a small number of activities that apply to all software projects, regardless of their size or complexity. In addition, this framework also includes a series of activities that apply throughout the software process. The generic process framework for software engineering includes five activities:

- Communication.
  Before technical work can begin, it is important to communicate and collaborate with customers (and other stakeholders, the intention is to understand stakeholder objectives for the project and to collect requirements that help define software features and functions.
- Planning
  Every complicated trip can be simplified if there is a map. A software project is a complicated journey, and planning activities create "maps" that help guide the team when traveling. "Maps" define software engineering work by explaining the technical tasks that must be performed, the risks that might occur, the resources that will be needed, the work products that will be produced, and the work schedule.
- Modeling
  Whether you are a landscaper, bridge builder, aeronautical engineer, carpenter, or architect, you work with models every day. You "sketch" the object so you will understand the big picture - what it looks like architecturally, how the constituent parts fit together, and many other characteristics. If needed, you refine the sketch into greater and greater detail in an effort to better understand the problem and how you will solve it. A software engineer does the same thing by creating a model to better understand the software and design requirements that will achieve these requirements.
- Construction
  This activity combines the generation of code (either manual or automatic) and the testing needed to uncover errors in the code.
- Launching
  Software (as a complete entity or as a partially completed improvement) is sent to customers who evaluate the product sent and provide feedback based on evaluation.

These five activities can be used during the development of small, simple programs, creating large Web applications, and for engineering large and complex computer-based systems. The details of the software process will be very different in each case, but the framework for the activities remains the same.

For many software projects, activities are applied iteratively as project progress. Communication, planning, modeling, construction, and dissemination are applied repeatedly through a number of project iterations.

2.2. Software Development Life Cycle

SDLC is a method by which software can be developed systematically and allows the completion of software projects within a specified deadline and maintain product

quality according to standards. The SDLC framework provides a series of activities for system designers and developers to follow software development [8].
SDLC is a conceptual framework or structural process of a phase that is in the development of an application from a feasibility study to its launch and maintenance in the field.

There are several models that illustrate various approaches to the SDLC process. The SDLC model is generally used to describe the steps that are followed in the life cycle framework. It should be remembered that the model differs from methodology in the sense that the first describes what must be done while the second, in addition, explains how to do it. So, the model is descriptive while the methodology is prescriptive.

The phases that are generally present in each and every SDLC model are: Understand the problem (through gathering requirements); Deciding on a plan for a solution (Designing); Coding the planned solution; Test the actual program; Product placement & maintenance. For large systems, each activity can be very complex, and methodologies and procedures are needed to do it efficiently and correctly. In addition, each basic activity itself may be so large that it cannot be handled in one step and must be broken down into smaller steps. For example, the design of large software systems is always broken down into a number of different design phases, from very high-level designs that only determine the components in the system to the detailed design in which the component logic is determined.

2.3. Scrum

Scrum was first introduced back in 1997 even though in 1995 there was initiation of scrum [9]. Scrum, as one of the most widely used agile methods, has been adopted in countless organizations [10]. It is one of the concepts of Agile Methodology. Scrum completes the project in minimum time and cost. The difference between Scrum and waterfall such as: Role, Meetings, Characteristics, Artifact, Language. The process inside Scrum is as follows:

a) Determination of Product Backlog

Product backlog is a list contains work items that will be done in system development (example: user stories, bugs, various chores) used by development team to coordinate work to be done [11] . Usually in the form of user stories and are user centric. Product backlog criteria are having added value for the customer, prioritized as needed, the details of the product backlog depend on the priority position, there are estimates, live and always developing, there are no items that can be ignored and not very technical.

b) Sprint Planning Meeting

Sprint planning is a plan for delivering the product backlog in a sprint and realizing it is as the Sprint goal, development team make the decision of choosing backlog from the Product Backlog which they can complete within a Sprint cycle [12]. Including discussing what they have to do in order to complete the backlog.

c) Sprint Backlog

The sprint backlog is the set of backlog product selected to work on during the sprint. The sprint backlog can be changed during the sprint and only the development team can change that.

d) Sprint

Sprint is an iteration in a certain period of time in the development of the system. In a sprint, the amount of work that must be done by the team has been arranged and is ready to be done. Usually 1 sprint ranges from 2-4 weeks.

e) Daily Scrum

In many projects, each sprint starts with a daily Scrum meeting [13]. This meeting takes no more than 15 minutes and is conducted every day by the Scrum Master and Scrum Team. At this meeting, what was discussed as follows: What has the team done or accomplished since the last Scrum? What will be achieved until the next Scrum? Are there obstacles that interfere with progress?

f) Scrum Review and Retrospectives

Sprint retrospectives are opportunities for teams to reflect on their sprints so they can be better in the next sprint. The aim of the Sprint Retrospective are as follows: evaluate how the Sprint went the results are related to people, relationships with the team, processes and also tools, to asses items that has chance of enhancement [14]. After the sprint retrospective, the process returns to sprint planning. The usual roles in Scrum such:

1) Product Owner
   Responsibilities such as follows: customer representative, work with everyone in the team, ,maintain product backlog, bridging teams and businesses, decide on a product feature, decide on product release date and content, responsible for profit, prioritize and adjust features, responsible for the final approval, accept or reject team work.
2) Scrum Master
   Responsibilities such as follows: assist the team and be responsible for overcoming team difficulties, have the role of leader over team members, responsible for the process, perform daily scrum and sprint planning, protect the team from outside interference.
3) Scrum/Development Team
   Responsibilities such as follows: develop products, each team member is responsible for their own work, maximum contribution in every sprint, range of 5-10 people, there are 2 types of teams: self-organizing and cross-functional teams, responsible for the success and failure of the team, usually consists of programmers, testers, UX designers, and others.

Product owner, Scrum Master and development team are classic team that existed in scum team. In scrum, these team are expected to be self-organized, mature and cross-functional without being dependent on others outside the team [15].

2.4. IBM Design Thinking

IBM Design Thinking adds a method to the original Design Thinking. Additions in the form of writing requirements, team settings and project track progress including user feedback.

The IBM Design Thinking Process recognizes that most problem solving is based on teamwork [16]. The collaborative team principle of all teams, stakeholders, and users in a fast response aimed at achieving the needs and expectations of users. IBM Design Thinking with its iterative process development, test, playbacks, and continuous synchronization has allowed for faster process and delivery on a tight schedule [17].

2.4.1. IBM Design Thinking Keys

IBM Design Thinking holds a few keys that can be conducted during project development.

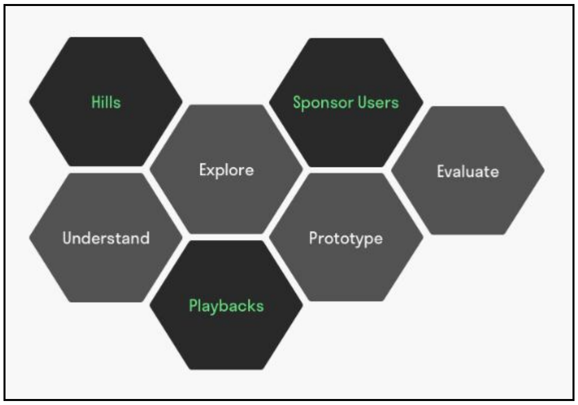


Figure 2: IBM Design Thinking Elements
Source: Robert Everett [16]

Figure 2 explains most critical IBM Design Thinking keys that makes IBM Design Thinking is different than common design thinking. The keys are as follows:

a) Hills

Hills is designed to gather teams around shared goals that are embedded in the needs and desires of the user. Based on a unified understanding of success, the team is given autonomy to produce solutions.

b) Playback

Different teams are often involved in different projects. The Playback technique is designed to bring these groups in the same direction that is directed to achieve the target project which is ultimately centered on user needs. In short, Playback is a rating and feedback platform that ensures everyone is harmonized

c) User Sponsor

This technique involves real users who are targeted in the IBM Design Thinking process. Working with people actually underlies design experience on the reality of user needs, unlike an imaginative approach. This joint creation with the sponsor user increases the value of the product in the Design Thinking process

2.4.2. IBM Design Thinking - The Loop

Loop is an activity that is repeated, a rational process in which the process ends, becoming the beginning for other processes to begin.

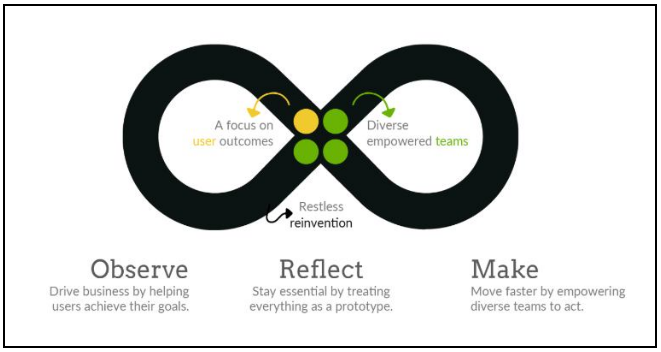

A focus on user outcomes
Diverse empowered teams
Restless reinvention
Observe
Drive business by helping users achieve their goals.
Reflect
Stay essential by treating everything as a prototype.
Make
Move faster by empowering diverse teams to act.

Figure 3: IBM Design Thinking The Loop

As seen in figure 3, the IBM Design Thinking Loop represents a continuous process of innovation carried out by the company and forms the Design Thinking Company. The purpose of this Loop is to look for user needs and solutions that are reflected in the prototype continuously.

a) Observation

Look deeper to realize that there are forgotten user needs and verify their expectations of the ideas that have been made.

b) Reflection

Reflection aims to consolidate the knowledge gained, build a solid view and build a solid plan.

c) Make

Transform abstract ideas into reality.

3. RESEARCH METHOD

This research framework is developed based on the needs of the company in developing Dashboards and Reports using IBM's Scrum method and Design Thinking.

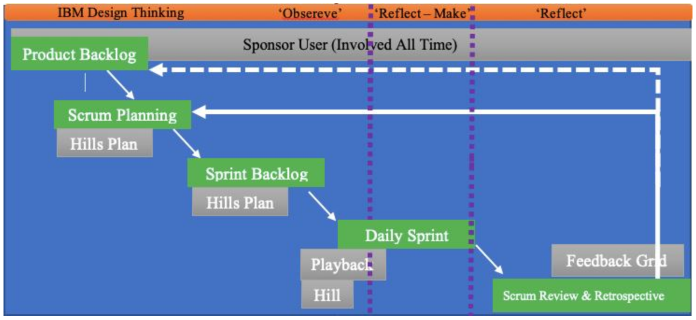


Figure 4: Framework Scrum and IBM Design Thinking

Figure 4 is the combined framework of Scrum and IBM Design Thinking processes that being used in this study. This framework will still be based on Scrum stages and processes. IBM Design Thinking will emphasize and will be implemented in certain stages and process of Scrum stages themselves.

As stated above, backbone of this framework is Scrum methodology's processes. Author will combine Scrum processes with IBM Design Thinking 'The Loop' phases. As stated in Literature Review, IBM Design Thinking 'The Loop' has three looping phases: 'Observe', 'Reflect' and 'Make'. These IBM Design Thinking 'The Loop' phases will be infused into Scrum processes as picture above.

- Hills.
  Hills are statements of the achievable goals made by the team in certain target period. Goals must be explicit and team will evaluate them regularly, hills help aligning the team around a single shared mission. Hills (Hills planning) will be set every time a Sprint backlog and Scrum planning are being created. Hills will also follow the nature of scrum itself: adaptive. Hills can be changed or be rearranged according to user needs during the sprint period.
- Playback.
  Playbacks are regular check point that bring users, stakeholders and teams together to tell stories and exchange feedback [18]. Playbacks allows us to measure progress on a system prototype that developers currently working on and in the same time playbacks will help to uncovering and addressing any misalignment that may exist.
- User Sponsor
  During the sprint the team will look for direct involvement with the user. Users will be involved from the beginning of the sprint to the end of the sprint. Before finally being able to use the system in their daily work.

Following table is the framework combination of Scrum and IBM Design Thinking.

Table 1: Combined Scrum and IBM Design Thinking Framework and its Deliverables

Scrum Process	**IBM Design Thinking - The Loop**	**Deliverables (Scrum + IBM Design Thinking)**
Product Backlog	Observe	Need Statements matrix, User story, User Sponsor Map.
Scrum Planning, Sprint Backlog	Observe	Sprint Backlog, Hills
Daily Sprint	Observe, Reflect, Make	Hills update (if any), Playback Plan, Team or Sponsor User Feedback (if any), Deployment (if any)
Sprint Review and Retrospective	Observe	Feedback grid, User Story Updates (if any). Backlog updates (if any), Hills update (if any), Next Sprint Planning

In table 1, both Scrum process and IBM Design Thinking Phase deliverables are being used in this study. Product Backlog, Scrum Planning, Sprint Backlog and Daily Sprint share the same main idea as 'Observe' in IBM Design Thinking that is: To extract meaningful outcomes for users and other involved stakeholders, first we need to gain a deep the understanding of the challenges they face and also their priorities. By doing this and set our mind in the worlds of our users, we ensure that our solutions always address real-world needs [16].

In Scrum's Product Backlog, Scrum Planning, Sprint Backlog and Daily Sprint all discussion is all about aligning and observing user's interest, needs and issues (if any). Thus, the aim is to know who users are and to have empathy to the users. With empathy build in these processes of scrum, it will be easier to set the user stories and backlogs that will be done in each sprint. Deliverables of this processes are 'Need user statements', 'User story', 'User Sponsor Map', 'Sprint Backlog', 'Hills'. With these deliverables, development team is expected to grow empathy and understanding towards their user and have sufficient knowledge towards their user an also their user sponsor. 'Hills' will assist developer team to set target and backlog that need to be achieved in certain period of time during the scrum. This process is crucial to maintain the very first step of user-developer alignment.

Daily sprint shares all the IBM Design Thinking Loop's all phases. In Daily sprint, development team are usually aligning their selves and users, re-prioritize their works, updating their works one to another and certainly developing the system. Thus, Daily sprint shares all the IBM Design Thinking's loop phases. After 'Observing', the development team will emerge to 'reflect' phase. In 'Reflect' phase, many input and ideas towards the system and its development emerged. To reflect on previous observations helps development team to synthesize and analyze findings, building a better understanding of our users across the team [16]. And also, in daily sprint, developers will continue the IBM Design Thinking 'Make' phase. In this phase, developer team is making one prototype to test the idea and understanding —even if the idea isn't fully baked yet. Quick, low-fidelity prototyping allows team to simulate ideas and test their understanding and or hypotheses cheaply, rapidly and easily. The expected result after the prototype is made is the solutions that are robust, effective, and battle-tested that will give the development team solid ground fundamentals to continue the system development [16]. In daily sprint, deliverables are 'Hills' update (if any), 'Playback Plan', 'Team, Sponsor User Feedback' (if any). 'Playback' is a review of the system development towards all the team involved, thus all teams will be able to share ideas, feedback and the developer team will be able to absorb and understand the feedback. 'Playback' is one of the events that will allow all team to collaborate after certain system prototype has been created.

After each sprint is over, there will be sprint review and retrospective. In this process, all feedbacks will be discussed, and developer team will sort them. The ideas and feedback that might bring better value for the team will be conducted in next sprint. In this process, the IBM Design Thinking Loop phase is looping back to 'observe'. It is good to understand the evaluation of previous sprint including its strength and weakness and also things need to be improved on the next sprint. This process's deliverables are 'Feedback Grid' to review all the feedback taken during the sprint from all team.

The following are the Dashboards and Reports needed and will be developed along with the business units and PIC users of each of the Dashboards and Reports:

1) Asset Management Details

This dashboard contains data assets from company include laptops, cameras, camcorders, portable DVD players, speakers, hard drives and laptop bags. The assets are held by each JTI employee. This dashboard is made so that the Asset IT team can find out who is using a particular asset, the age of the asset, the price of the asset in support of the Asset IT team to maintain the assets of company. Every asset that enters and enters the IBM JTI inventory will be recorded in the internal IT Asset system which will eventually be pulled by IBM Cognos data to be presented in the Dashboard and Report system pages.

2) Asset Summary

This report is a derivative report from Asset Management above. Some employees who did not extend the contract or resign did not rule out the possibility that they were still in possession of several items belonging to company. This report is projected to assist the Asset IT team in validating the items returned by employees who have resigned to the data in the system regarding asset ownership. Consist of following information: Deployed assets, Aging Report, Stock assets.

3) Employee Overtime Dashboard

Many employees from various divisions and projects at JTI often work overtime in meeting the needs of the project and making sure that deadlines can be met. Of the many projects and divisions, the management team and the BoD want to have a picture of what projects have the most amount of melting, personnel who most often overtime and the amount of overtime in certain periods, for example per week, per month or even during the project carried out. Knowing this will make it easier for the Management and BoD teams to validate the overtime process proposed by employees and can help the management team and BoD in estimating the level of health of the project for employees. But on this system

4) Employee Attendance Dashboard

The TSS team is a team that has high mobility which they must meet in answering the needs of employees, but that does not mean their presence in the office can be underestimated. The presence of employees from the TSS unit in the office must be noted because it will affect their incentives, leave and permits must be recorded daily so that the team leader from TSS can validate the incentives and discipline of their employees.

5) Preventive Maintenance Dashboard

Some hardware units that have been installed by JTI require routine maintenance in certain periods according to what is stated in the work contract. In order to facilitate the TSS team to track which hardware has completed maintenance and which hardware has not done maintenance.

6) Whereabouts Dashboard

The TSS unit has the obligation to maintain and check the machines with customers according to their needs. Often a TSS employee must move more than once a day to meet customer needs. The TSS Management Team must monitor the movement of TSS employees to be verified to the relevant customer. Tracking the whereabouts of TSS employees is important so that the TSS Team can collect the right bills to the customer and maintain the obligations of the TSS team in fulfilling their responsibilities in accordance with the contract.

Basically, the development will be carried out with the Scrum methodology which is the backbone of the activities carried out. IBM Design Thinking will be included in the Scrum methodology scheme which will also produce IBM Design Thinking deliverables themselves.

4. DISCUSSION

To begin the development process, after the developer team has been chosen, they determined the product backlog for this project. To find a good product backlog, the development team met the users to get an overview and better understanding and perception towards what the users need. Thus, developer team interviewed before the first sprint begin.

At this point, the development team had entered the early stages of IBM Design thinking, 'Observe'. To help understand the user's needs in accordance with the information obtained, the development team created a "Need Statements Matrix" as is one of the deliverable in early IBM

Design Thinking 'Observe' phase. Following are the steps for developing the Need Statement matrix:

a. Write down all the information currently known by all team members.

b. Write down activities that will later be the basis of user stories. **Writing the activity follows the following format [User] needs a way to [need] so they [get the benefit in this way].**

c. The development team discussed around the needs and benefits that they believe they will complete to meet the needs of users. There is no 'feature' can be written at this early stage, due to limited information. At this stage, only limited activities that users might need based on limited information.

d. The team will group similar ideas and try to write several statements of needs that represent the needs of the user. The subject must be focused on users not on the technology. Matrix statement does not need to focus on technical details that are too detailed but to keep us exploring what the user needs.

Following is one example of Need statement matrix developed:

Table 2: Example of Need Statement Matrix

Dashboard/Report	Activity (Need Statement)
Asset Management Detail	- IT Asset Admin requires information about the asset details held by employees to validate crosschecks in the field. - IT Asset admin requires information about IT assets, Non IT and leasehold to calculate the amount of IBM-JTI inventory - IT Asset admin requires information that is always updated automatically if there is a change in inventory data so there is no data redundancy

Table 2 is need statement collected from observing and interview activity with the users. From this table development team collect what are user need from the system. After interview is done, team will discuss all the information gathered and doing a 'mini' playback among themselves to discuss their findings. Then, developer team will develop user story as follows:

Table 3: Example of User Stories Collected

User Story Code	Dashboard / Report	User Story	Severity	Priority
AS-AMT-001	Asset Management Detail Report	As an IT Asset Admin, I want to have a Total Asset dashboard to see the availability of all Asset in JTI today when I open the dashboard system with information based on asset type. The data loaded is Name of asset, Asset Type, Quantity.	High	1
AS-AMT-002	Asset Management Detail Report	As an IT Asset Admin, I would like to have a report to see all IBM-JTI assets currently in ownership with information based on employee ID, employee name, manager name, asset name, serial number, tagging number, type, model, description.	High	1

a) Scrum Planning

After the User Stories are collected and arranged, then the development team must discuss the first sprint plan. At this stage, the developer compiled all user stories gathered and calculated the point on each user story. In each Scrum planning, all user stories must be calculated as story points.

Table 4: Example of User Story Points

User Story Code	**Dashboard / Report**	**User Story**	**Pts**
AS-AMT-001	Asset Management Detail Report	As an IT Asset Admin, I want to have a Total Asset dashboard to see the availability of all Asset in JTI today when I open the dashboard system with information based on asset type. The data loaded is Name of asset, Asset Type, Quantity.	4
AS-AMT-002	Asset Management Detail Report	As an IT Asset Admin, I would like to have a report to see all IBM-JTI assets currently in ownership with information based on employee ID, employee	4

User Story Code	Dashboard / Report	User Story	Pts
		name, manager name, asset name, serial number, tagging number, type, model, description.	

Table 4 is the example of user story that will be developed by the development team. This user story must be confirmed within all the team with user requirement-based function. Each user story will have certain point that decided by the development team.

b) Sprint Backlog

The sprint backlog only contains user stories that will be carried out in the next sprint. Each user story has points that have been calculated before in Scrum Planning. In this study, one sprint will have 40 user story points, thus user story points for each sprint is not allowed to exceed 40. The table will look alike the previous Scrum Planning but only contained one sprint backlogs.

During this process, development team also created a Hills plan. Target or Hills Plan will be a guidance for the development team to achieve targeted activities during the sprint. As stated above, hills are the list of developer's team targets of that to be achieved in certain period of time. Hills plan created in accordance to Sprint backlog created. In Hills plan, playback plan must also be included to make sure developer team aware of playback plan. Playback activities and methods that may be different from each other, for example playback can be conducted by presenting the prototype and allowing users to try or it can be done as simple with presenting the process and/or understanding that being understood by the development team and ask for clarification. Playback can also be conducted internally within developer team themselves. Usually internal playbacks were meant to keep them aligned one to another and can be held on any time in a sprint.

Table 5: Example of Hills Planning

Target Day	Hills
4	The development team has finished analyzing and cleaning the data that has been obtained.
	Playback of data analysis results to the internal Users and IT Developers
8	Playback to the PIC user Asset Management Detailed report and Asset Dashboard Summary in this first sprint and accommodating feedback.
	Playback is done by displaying and allowing the user to try the mirror directly.
10	Asset Management Detailed reports and Asset Dashboard Summary can be deployed

Table 5 is the example of hills statement that used in the development. Hills will help team to stay align yet focused on what achievement they need to reach during the sprint.

c) Sprint and Daily Sprint

In this study, one sprint will be held for 2 weeks. Thus, for 2 weeks, the team will work on user stories in accordance with sprint planning that has been made above. Development team also followed the hills that have been created so that the team stays inline while developing this system and user sponsors are still involved in developing this system. During the sprint, every day the development team conducts a Sprint daily meeting at 10:00 AM. After each sprint daily meeting, developer team were able to conduct internal playback to update findings, technical issues and also complex ideas. While sprint was on going, every day works were tracked, reported and calculated on Burned Down chart below. Here is an example of burned down chart from Sprint 1:

Table 6: Example Sprint 1 Burned Down Chart

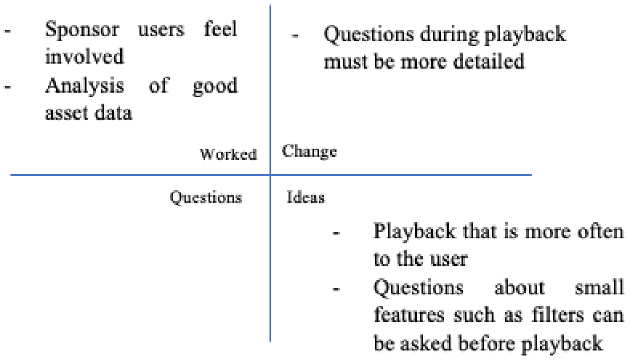
	Burned down		Balance		Daily Completed
Day	Planned	Actual	Planned	Actual	
0			40	40	#N/A
1	3	2	37	38	2
2	3	2	34	36	2
3	4	4	30	32	4
4	4	5	26	27	5
5	3	4	23	23	4
6	5	5	18	18	5
7	6	4	12	14	4
8	4	6	8	8	6
9	6	6	2	2	6
10	2	2	0	0	

In table 6 above, developer team were able to know whether they were on track to finish their work on targeted date or not.

d) Scrum Review and Retrospective

Sprint were closed in every 2 weeks, all the work that have not been done by the time will be accumulated into the next sprint and the current sprint will be marked as ‘Red’ or ‘Not Complete’. If all the work had been done in 2 weeks’ time, then the sprint will be marked ‘Green’ or ‘Complete’, developer team can start new fresh backlogs on next sprint.

Each sprint will be closed by a sprint review and retrospective. The sprint review and retrospective contain review on what has been done during previous sprint and team will look for improvements that can be done in the next sprint, including user understanding and feedback on previous sprint. Feedback Grid is one of IBM Design Thinking’s tools that is able to help developer understands and reflect what have been worked, what need to be changed, Questions and ideas within the team. Following is the example of feedback grid in sprint 1



- Sponsor users feel involved - Analysis of good asset data Worked	- Questions during playback must be more detailed Change
Questions	Ideas - Playback that is more often to the user - Questions about small features such as filters can be asked before playback

Figure 5 Example of Feedback Grid

Figure 5 is a feedback grid contains feedback from the user and the development team members themselves. The 'Worked' section is what has worked well during the sprint, 'Change' is something that can be changed for better in the next sprint, 'Questions' are unclear feedback and need further clarification, 'Ideas' are ideas that can be done in the next sprint based on feedback that has been obtained.

After the Scrum Review and Retrospective are done, the Scrum process will return to sprint planning. If there were any requirements that need to be change in the matter of its priority then, the team are able to return to the product backlog list to readjust the backlogs. Then the iterative scrum process will spin again.

In this study, all backlogs were cleared on 5th sprint. It took the team 10 weeks to finish the system. System has been implemented with all user acceptance.

5. CONCLUSION AND RECOMMENDATION

This study discussed Dashboard and Report System development process using Scrum Methodology and IBM Design Thinking. As we knew, system development nowadays should have been more and more user-centric, as system acceptance will rely so much on user’s opinion. Thus, the combination of Design Thinking and scrum are able to improve the system development more user-centric and to improve development system’s collaboration, ideas and innovation between system developers and user. In this study, we learned that Scrum and IBM Design Thinking have similarities that results a good outcome for system development. In fundamental basics, both Scrum and IBM Design thinking are user-centric methodology and framework, thus Scrum and IBM Design thinking complement each other in system development.

Scrum can emphasize, complement and support IBM Design Thinking discoveries better. Since IBM Design Thinking needs maximum collaboration within all team involved including users, development team and user sponsors, this collaboration is strengthened with scrum methodologies itself. For example, Scrum user story detail and user story points are results of collaboration of all team involved and also including Scrum planning, Scrum backlog planning, Scrum daily meeting and Scrum reflection will allow important information to be better understood and clear within teams including the priorities and all important matters. This will result more effective system development and time used in the development itself.

IBM Design Thinking also develops user sponsor profile to elaborate PIC’s of each matter and each department if needed. Also, in IBM Design thinking we need to record each sponsor user’s identity, relationships and interests. This will help the development team to understand better their users during system development and to enabling them wo proved better ideas in scrum meeting. IBM Design thinking enabling all team to think whether or not the new system design and usability will serve their interest or not. With IBM Design Thinking combined with Scrum, it is easier to align all involved team into one solid purpose and varies ideas. IBM Design Thinking is very effective in helping Scrum in understanding and meeting user needs. With the IBM Design Thinking, the development team can produce things that are more effective in answering the needs of the user. On the other hand, when IBM Design Thinking is not done, the development team loses direction and empathy for users who suffer from the waste of effort and time that has been done during system development.

From the study above, we also learned that Scrum methodology and IBM Design Thinking has extraordinary results in absorbing an addition or change in the development of a system (which is often the case). Scrum methodology and IBM Design Thinking helps the system development team and its users stay in line and remain open to each other. Empathy and sympathy are things that can be generated in the practice of IBM Design thinking. With empathy and sympathy, the development team can feel, understand and analyze user needs.

Limitation of this study is the Dashboard and Report system can be extended to bigger system that include more business processes. The use of Scrum and IBM Design Thinking can be extended to more complex and more complex applications to see other results from the combined methodology. In this study, the system developed is a dashboard and report system, but in other system's complexity level, perhaps different results can be extracted with more user stories and far more cases available, maybe the use of Scrum and IBM Design Thinking combination can be utilized more.

REFERENCES

[1] R. S. Pressman, Software Engineering A Practitioner 's Approach, New York: McGraw Hill, 2010.

[2] N. B. Ruparelia, "Software Development Lifecycle Models," 2010.

[3] C. W. L. K. Y. T. Whee Yen Wong, "The Importance of a Software Development Methodology in IT Project Management: An Innovative Six Sigma Approach - A Case Study of a Malaysian SME Organization," *The Importance of a Software Development Methodology in IT Project Management: An Innovative Six Sigma Approach - A Case Study of a Malaysian SME Organization,* 2013.

[4] Er. Sheilly Padda, Er. Apoorva Arora , Er. Sonali Gupta2 , Er. Priya Sharma, "Review of Software Development Methodologies Used in Software Design," *International Journal of Advanced Trends in Computer Science and Engineering,* vol. 3, no. 5, pp. 88-93, 2014.

[5] C. Kulbacki, "Scrumstar: What's more popular than scrum," 23 Jun 2018. [Online]. Available: https://scrumstar.com/articles/the-most-popular-agile-methodologies. [Accessed 2020 Apr 4].

[6] T. Yoshida, "Medium Try Design Thinking + Scrum," 19 Sept 2018 . [Online]. Available: https://medium.com/@takeshi.yoshida/design-thinking-plus-scrum-d671a1a8e67a. [Accessed 2020 Apr 4].

[7] Lucena, Percival; Braz, Alan; Chicoria, Adilson; Tizzei, Leonardo , "IBM Design Thinking Software Development Framework," pp. 98-109, 2017.

[8] Apoorva Mishra, Deepty Dubey, "A Comparative Study of Different Software Development Life Cycle Models in Different Scenarios," *International Journal of Advance Research in Computer Science and Management Studies,* 2013.

[9] K. Schwaber, "SCRUM Development Process in Business Object Design and Implementation," in *SCRUM Development Process in Business Object Design and Implementation,* 1997, pp. 117-134.

[10] Michal Hron, Nikolaus Obwegeser, "Scrum in practice: an overview of Scrum adaptations," *51st Hawaii International Conference on System Sciences* , pp. 5445-5454, 2018.

[11] Todd Sedano, Paul Ralph, Cécile Péraire, "The Product Backlog," *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE), Montreal, QC, Canada,* pp. 200-2011, 2019.

[12] A. Alhazmi and S. Huang, "A Decision Support System for Sprint Planning in Scrum Practice," *SoutheastCon 2018,* pp. 1-9, 2018.

[13] H. F. Cervone, "Understanding agile project management methods using Scrum," 2010.

[14] Sudarsono, Bernadus & Fransiskus, Fransiskus & Hartono, Henny & Bernanda, Devi & Andry, Johanes., "Adopting SCRUM Framework in a Software Development of Payrol Information System," *International Journal of Advanced Trends in Computer Science and Engineering,* vol. 9, no. 3, p. 2604 – 2611, 2020.

[15] L. Gonçalves, "Scrum. Controlling & Management Review," vol. 62, no. 4, pp. 40-42, 2018.

[16] Everett, "Tools Hero - IBM Design Thinking," [Online]. Available: www.toolshero.com/creativity/ibm-design-thinking/. [Accessed 20 Jan 2020].

[17] Eickhoff, F. L., McGrath, M. L., Mayer, C., Bieswanger, A., & Wojciak, P. A., "Large-scale application of IBM Design Thinking and Agile development for IBM," *IBM Journal of Research and Development*, 2018).

[18] IBM, "https://www.ibm.com/design/approach/design-thinking/," IBM, 20 May 2020. [Online]. Available: https://www.ibm.com/design/approach/design-thinking/. [Accessed 2 July 2020].