Assignment 3

Team Members

Gray Li
Michael Huang
Xiaoyang Wei

Date

November 11th, 2022

Course title

IT Governance & Risk Management

Contents

Executive Summary

XYZ Corporation is a supplier of laboratory automation, lab equipment, and chemical supplies to various universities, hospitals, and industrial and clinical labs. It has a centralized IT organization and already supported sales, marketing, manufacturing, and supply chain. The corporation is willing to move towards the adoption of an agile approach to sell its MTSTPro system to its current customers to be used for Covid-19 testing. It has good support for CRM, ERP, MES, and SCM systems. It mostly relies on in-house expertise for those systems. It has an internal data center and thus far has made limited use of Cloud capabilities. It relies on the Waterfall delivery cycle. It also has an IT Project Management Office that centralizes the project management skills within the organization. It has decided to implement several systems, including a Data Management System, a Product Lifecycle Management (PLM) system, a Quality Control System, etc. It uses two steps to achieve its goal: start with an experiment with a small team and then move initiatives to a larger agile effort. Part 1 will help the CRM team of XYZ Corporation to adopt an agile approach, and Part 2 will help other teams from the MTSTPro system to adopt an agile approach.

For Part 1, our teams review the Scrum and Kanban methodologies, and our recommendation is Scrum methodology because of its shorter delivery cycle and Scrum's ability to help the CIO and other teams to understand the agile approach better. By using Scrum methodology, a couple of technical tools must be evaluated: project management tools, communication tools, and source version control tools. For Part 2, our team reviews Scrum of Scrums, Spotify Model, and SAFe methodologies. Our team highly recommends the Scrum of Scrums methodology to achieve the goal of transferring to agile methodology while developing the MTSTPro system quickly.

Part I

Introduction

In this part, we will help the CRM team of XYZ corporation to adopt an agile approach. The team is currently composed of 9 team members, mostly CRM analysts and developers. It also includes one business analyst that helps capture the requirements for new development. It wants to start with a simple adoption of an agile approach (either Kanban or Scrum) to implement the new requirements to support MTSTPro customers. The same team also supports the current production CRM system and will also be handling business requirements from legacy users.

It's an experiment of the whole company's adoption of the agile approach, so we won't provide a more straight and more perceptible methodology that can help the development process. Thus we recommend they use Scrum, which has a clear process of events and transparent artifacts. In addition, we also recommend the role of the team when based on CRM team members, and provide several useful tools for project management, communication, and sources version.

Methodology Overview

Kanban and Scrum are project management methodologies that complete project tasks in small increments and emphasize continuous improvement. Both Kanban and Scrum borrow from Agile and Lean approaches, which means Kanban and Scrum are both adaptive, transparent, and able to reduce inefficiencies in the project process. But the processes they use to achieve those ends are different.

Methodology	Kanban	Scrum
Roles	No defined roles	Scrum master, product owner, and development team
Cadence	Continuous	Regular fixed length sprints (ie, 2 weeks)
Change policy	Can be incorporated any time	Generally not made during sprint
Artifacts	Kanban board	Product backlog, sprint backlog, product increments
Practices	Visualize the flow of work, limit work-in-progress, manage flow, incorporate feedback loops	Sprint planning, sprint, daily scrum, sprint review, sprint retrospective
Ideology	Use visuals to improve work-in-progress	Learn through experiences, self-organize and prioritize, and reflect on wins and losses to continuously improve.

Scrum has **three clearly defined roles**:

- The product owner advocates for the customer, manages the product backlog and helps prioritize the work done by the development team.
- The scrum master helps the team stay grounded in the scrum principles.

- The development team chooses the work to be done, delivers increments, and demonstrates collective accountability.

Scrum moves fast, with **sprints** that usually last between one to four weeks, and have clear start and end dates. During the sprint retrospective, scrum teams should discuss how to **limit change** in the future, as changes put the potentially shippable increment at risk. It has **three main artifacts**:

- Product Backlog is the primary list of work that needs to get done and maintained by the product owner or product manager.
- Sprint Backlog is the list of items, user stories, or bug fixes, selected by the development team for implementation in the current sprint cycle.
- Increment (or Sprint Goal) is the usable end-product from a sprint.

It also has several **key practices**:

- Organize the backlog: Sometimes known as backlog grooming, this event is the responsibility of the product owner.
- Sprint planning: The work to be performed (scope) during the current sprint is planned during this meeting by the entire development team.
- Sprint: A sprint is the actual time period when the scrum team works together to finish an increment.
- Daily scrum or stand up: This is a daily super-short meeting that happens at the same time (usually mornings) and place to keep it simple.
- Sprint review: At the end of the sprint, the team gets together for an informal session to view a demo of, or inspect, the increment.
- Sprint retrospective: The retrospective is where the team comes together to document and discuss what worked and what didn't work in a sprint, a project, people or relationships, tools, or even for certain ceremonies.

Unlike Scrum, there are **no roles** like "kanban master" who keeps everything running smoothly. It's the collective responsibility of the entire team to collaborate on and deliver the tasks on the board. Kanban is based on a **continuous workflow** structure that keeps teams nimble and ready to adapt to changing priorities. A kanban workflow can **change at any time**.

Analysis & Recommendation

Here is the pros & cons of ***Scrum***:

Pros	Cons
Scrum provides achieves transparency in all stages of the project development process	Scrum is not related to the project's deadline. Although adopting the Scrum methodology involves many smaller deadlines for everyone involved, it doesn't offer any support for the project meeting its overall deadline. Although this approach increases the odds of everyone involved working to the best of their abilities and meeting expectations, the project manager and stakeholders also need to make sure that the project is on track to be completed on time.
Scrum improves customer satisfaction. Having everyone on the team working to the best of their abilities and constantly adjusting based on internal and external feedback can result in popular products and solutions with customers.	Scrum requires extensive training. Although using the Scrum framework can potentially deliver quick and high-quality results, it requires a well-trained and skillful team to implement it properly. Before committing to Scrum, everyone within the team needs to understand the benefits and particularity of this approach for the project to be a successful one.
Scrum provides continuous feedback. Because this methodology requires daily check-ins for progress reports, there is always feedback offered at the team and individual level. This helps to make the project better in the long run.	Scrum requires experienced personnel. Adopting the Scrum methodology involves extended periods of intense work and everyone involved needs to have experience and skills to quickly and successfully perform their own tasks. Everyone on the team needs to be able to execute and provide educated feedback on the results and overall process.

It encourages creative approaches. With Scrum teams working together and analyzing ideas from all its members, creativity is encouraged and new ideas are likely to appear.	It requires the use of small teams. The Scum methodology usually works best with teams of at least three people but no more than 10. Although this can promote collaboration and teamwork, some organizations may find it difficult to rearrange their workforce into teams.

Here are the pros & cons of ***Kanban***:

Pros	Cons
Kanban is easy to use. Kanban is a very simple and easy-to-understand approach, which makes it practical for the management of a company to apply effectively. You do not have to be an expert to work with the Kanban approach.	Supervision of the use of a Kanban board, cards, and analysis of output is easier as compared to most methods/approaches to project management.
Kanban advances collaboration and makes the whole team work together to convey the ideal outcomes.	Building software in iterations is a foundation for most development processes, which is not integral to Kanban at a ticket level. You can build iteration on top of Kanban, but it often ends up being its own separate process.
Supervision of the use of a Kanban board, cards, and analysis of output is easier as compared to most methods/approaches to project management.	There are no timeframes associated with each phase, which can be disadvantageous.

Recommendation:

Considering the differences and pros&cons of Scrum and Kanban above, we will recommend XYZ Corporation mainly adopt Scrum methodology based on XYZ's situation and goal. The reasons are listed below:

- XYZ wants to start with an experiment with a small CRM team. Besides the CRM system, more systems are needed to implement, so XYZ would be more likely to develop a CRM system quickly and provide a good template for other systems' development. In this case, Scrum would be more suitable. The delivery cycle of Scrum is shorter than Kanban. In addition, Scrum has more artifacts than Kanban like product backlog, sprint backlog, and product increments, and Scrum also has daily meetings which Kanban doesn't need. Some practices of Scrum like sprint review and sprint retrospective also can help the team get feedback and find what worked and what didn't work in a sprint. Thus, using Scrum can provide XYZ with more files or recordings and more experience, which can help CIO and other teams understand the agile process better.
- The component of the CRM team also fits Scrum requirements. Business analysts in this team can play the role of product owner, and BA can transfer customers' needs to Scrum epic and stories. Then CRM analysts and developers have enough experience and skills to quickly and successfully perform their own tasks, which can stand intense development during the Scrum sprint.
- After developing the CRM system of the MTSTPro system, the team will also be handling business requirements from legacy users. Scrum is still workable, however, if this task is continuous or may change frequently and XYZ wants to explore more about agile methodology, Kanban is also a choice.

Roles Needed

- Product Owner (1) is responsible for maximizing the value delivered by the team and ensuring that the team is aligned with customer and stakeholder needs. The product owner is also accountable for effective product backlog management, including developing and explicitly communicating the product goals, creating and clearly communicating Product Backlog items, ordering Product Backlog items, and ensuring that the product Backlog is transparent, visible, and understood.
- Project Manager (1) is in charge of the entire project and handles everything involved. He or she is responsible for the planning, procurement, execution, and completion of a project. In this project, we need a project manager to help the team plan for the product and make sure that the team follows the strategic plan and turns every milestone on time.
- Scrum Master (1) is responsible for helping to facilitate large teams by ensuring the scrum framework is followed. Scrum has a clearly defined set of rules and rituals that should be followed and the scrum master works with each member of the scrum team to guide and coach the team through the scrum framework. He or she is committed to the scrum values and practices, but should also remain flexible and open to opportunities for the team to improve their workflow. For this project, we need a scrum master to decide the methodology to use and create rules and rituals for the strategic plan.
- Development Team:
    - Business Analyst (1) is responsible for assessing how organizations are performing and helping them improve their processes and systems. Their primary responsibilities include liaising between IT and the executive branch, improving the quality of IT services, and analyzing business needs. They conduct research and analysis in order to come up with solutions to business problems to help to introduce these solutions to businesses' problems and help to introduce these solutions to businesses and their clients. They are experts in both business administration and information technology. For this project, the business analyst will help capture the requirements for new development and BA can play the role of **product owner or Scrum master.**
    - CRM & Developers (8) are programmers who specialize in systems that collect users, consumers, and subscriber data. They are not directly responsible for using those systems to improve customer satisfaction or boost values.

ADLM (Application Development Lifecycle) Tools

Project management Tool:

monday.com can help with project management with features like reporting, Calendar, time tracking, planning, etc. It is suitable for any business size. It can improve business management by easily monitoring everything from budget planning to sales, getting a clear overview of your progress, and making informed data-driven decisions.

The features monday.com have are: Project development can be tracked through Kanban, Timeline, or Charts. It has functionalities for planning sprints, creating user stories, and assigning to team members. It has explicit reporting.

Communication tools:

The manner that development teams collaborate has evolved as a result of COVID. Development teams had to find a way to interact when they couldn't physically meet because they were all working from home at once. **Slack** and **Zoom** working together became the standard. It is still.

Slack has a big library of plugins and integrations thanks to its widespread usage, which enables you to perform a lot of things that you might not have thought of before you started playing around. The standard videoconferencing software for groups and meetings of all kinds is now Zoom. It's simple and effective to combine Zoom and Slack. The widespread use of both technologies, which makes it much simpler to connect with staff members, clients, and vendors throughout your entire organization rather than just the development team, is why I advise using both together.

A close runner-up here is Microsoft Teams, which has the advantage of combining chat and videoconferencing in a single app.

Source version control:

For a while, Mercurial was a competitor there, but **Git** has unquestionably prevailed in the race for source control programs. Git is the standard now, so if the company does not utilize it, it will fall behind.

Although **GitHub** is the undisputed leader in this field, Git is a distributed version control system that must be used with a centralized repository for proper operation. I chose GitHub because of its widespread use and characteristics that go beyond those of a coding repository. The most

popular host for open-source projects is by far GitHub. Every business ought to have a presence on GitHub in some capacity.

Issue tracking, code review, and hooks into the build and deploy process are a few of GitHub's extra features. With its Codespaces product, which is based on Visual Studio Code, GitHub is even leading the drive towards IDE-in-a-browser technology.

GitLab, which has an open-source version and is closely following GitHub in almost every regard, comes in a close second for source control.

Part II

Introduction

In this part, we will help the other teams from the MTSTPro work to adopt an agile approach. These teams will develop or enhance the PLM System, Data Management System, Analysis Enhancements, and QC Capability. They do not need to consider legacy requirements or requirements coming from the design and manufacturing of existing services or products. They are not involved in any production support either.

Because these teams only focus on the MTSTPro system and should have some experience from the ERM team's work in part I, we recommend they use Scrum of Scrums during MTSTPro development. Also, we strongly recommend they adopt SAFe after the development, so that they could build a detailed and up-to-down agile progress for the future.

In addition, we also recommend the role of the team when based on whole team members, and explain the delivery approach. Finally, we try to ease the CIO and answer questions. We want to explain how agile methodologies do involve some level of planning and have a way of demonstrating progress to a plan. The solution is technically and architecturally sound based on the use of our recommendation.

Scrum of Scrums:

An approach for coordinating among multiple teams in Scrum is the Scrum of Scrums (also called SOS). A Scrum of Scrums is a technique to scale Scrum up to large groups of over Twelve people, where the groups are divided into Agile teams of 5-10. Scrum includes a designated member as an "**ambassador**" daily in order to participate in a daily meeting with ambassadors from other teams, called Scrum of Scrums.

This involves various teams coordinating their inter-team work. The team engaged with SOS is composed of individual members of the various development teams. The development team decides which team member should be sent to the Scrum of Scrums based on who can be a better speaker in order to solve the inter-team dependency issues. The person who is representing a team during the SOS can change over time and can be replaced by another who is best able to represent the team and be able to solve the issue at that point in time.

Spotify model:

The Spotify model is centered around simplicity. When Spotify began organizing around their work, they identified a handful of important elements on how people and teams should be structured.

Elements of Spotify model:

Squads	Similar to a scrum team, Squads are cross-functional, autonomous teams (typically 6-12 individuals) that focus on one feature area. Each Squad has a unique mission that guides the work they do, an agile coach for support, and a product owner for guidance. Squads determine which agile methodology/framework will be used.

Tribes	When multiple Squads coordinate with each other on the same feature area, they form a Tribe. Tribes help build alignment across Squads and typically consist of 40 - 150 people in order to maintain alignment (leveraging what we call Dunbar's Number). Each Tribe has a Tribe Lead who is responsible for helping coordinate across Squads and for encouraging collaboration.
Chapter	Even though Squads are autonomous, it's important that specialists (e.g. Javascript Developer, DBAs) align on best practices. Chapters are the family that each specialist has, helping to keep engineering standards in place across a discipline. Chapters are typically led by a senior technology leader, who may also be the manager of the team members in that Chapter.
Guild	Team members who are passionate about a topic can form a Guild, which essentially is a community of interest. Anyone can join a Guild and they are completely voluntary. Whereas Chapters belong to a Tribe, Guilds can cross different Tribes. There is no formal leader of a Guild. Rather, someone raises their hand to be the Guild Coordinator and help bring people together.

SAFe:

The Scaled Agile Framework (SAFe) is a set of organizational and workflow patterns for implementing agile practices at an enterprise scale. The framework is a body of knowledge that includes structured guidance on roles and responsibilities, how to plan and manage the work, and values to uphold.

SAFe promotes alignment, collaboration, and delivery across large numbers of agile teams. It was formed around three primary bodies of knowledge: agile software development, lean product development, and systems thinking.

As businesses grow in size, SAFe provides a structured approach for scaling agile. There are four configurations in SAFe to accommodate various levels of scale: Essential SAFe, Large Solution SAFe, Portfolio SAFe, and Full SAFe.

Analysis & Recommendation

The Spotify model is simple, but the environment it's implemented in is complex. Many organizations try to copy the Spotify model. To some, it may seem like a simple matrix organizational structure where people report to a functional area (Chapter) but work with a cross-functional team (Squad). However, it's more complex than that. Although it may look like a matrix organization, the key cultural elements of the model need to be in place to allow the structure to thrive, such as trust and autonomy. If an organization doesn't shift its behaviors (and ultimately its culture), the benefits of the Spotify model will never be realized.

Considering the situation of XYZ Corporation, the IT organization is structured as a matrix organization. Project Managers and Architects have their own managers within Strategy and Governance. The Application Developers and product specialists report to their own managers within the Application & Delivery (A&D) organization. The Testing Engineers and Business Analysts have their own managers within the A&D organization. Infrastructure & Operations personnel report to their own vertical. Almost every part only reports vertically, which is not a good company culture for the Spotify model. Besides, XYZ didn't have scale agile development experience before but Spotify's model doesn't have a clear process for corporations to adopt. Thus, we believe the Spotify model would be the worst choice since it's hard for XYZ to adopt Spotify and utilize the benefits of this model.

Now, we want to compare the Scrum of Scrums and SAFe to see which is more suitable for XYZ.

Scrum of Scrums is an extension of Scrum, so it has both pros and cons of Scrum.

Here are the pros & cons of ***SAFe***:

Pros	Cons

Adopting SAFe is the opportunity to tap into a relatively lightweight framework that creates efficiency in software development while maintaining the centralized decision-making necessary at the enterprise level.	The SAFe framework is incredibly complex and detailed, with much use of complicated terminology and jargon.
SAFe is particularly beneficial for organizations that need to work across teams, as its centralization makes multi-team coordination possible. In this scenario, it allows for standardized processes across teams and helps avoid obstacles and delays that may pop up when different teams need to work together.	Many teams find that SAFe takes too much of a top-down approach. So SAFe removes front-line players, including developers, testers, and product owners, from the decision-making processes and even from one another. This distance can limit the understanding of the entire software development lifecycle and hinder their ability to conduct well-informed and collaborative planning sessions a result.
SAFe helps teams maintain alignment with business goals. This alignment can often get lost in agile environments that take more of a bottom-up approach, as developers and testers can sometimes lose sight of bigger-picture business objectives. In contrast, SAFe's top-down alignment and centralized decision-making help ensure strategic objectives remain top of mind and that all decisions get made in support of those objectives.	SAFe emphasizes the big picture can often lead to longer planning cycles and more fixed roles within development cycles — two points that go against the ideas of delivering in short sprints to get new software to market faster and creating a continuous loop to ensure quality at every step of the way.

Recommendation:

The goal of XYZ is to transfer to an agile methodology while developing the MTSTPro system. SAFeThe framework is a great guide for a company to transfer to an agile team from traditional processes like the Waterfall delivery cycle. However, it's also really complex and detailed which needs much time for all employees and the board to be familiar with SAFe. Thus, it might not be friendly to XYZ now since we suppose XYZ wants to implement the MTSTPro system as soon as possible.

On the other hand, since Scrum of Scrums, is derived from Scrum, So it's not so hard to get started with teams that are now making scrum successfully in a small team. Because we recommend XYZ adopt Scrum in step 1, we believe XYZ will have more experience with Scrum after implementing the CRM system. Furthermore, the team members of CRM development can instruct other teams' adoption as a coach. In addition, compared with other agile methodologies, Scrum focuses on fast development during a sprint, which will help XYZ implement the MTSTPro system more quickly.

Besides, From the roadmap, we can see that many of these systems are being developed in parallel but they will need to integrate with each other at various points in time. Using Scrum of Scrums can help each system development team communicate with each other so that they can know if they can integrate with each other team next sprint or need to adjust the stories.

As a result, for now, we will recommend XYZ adopt Scrum of Scrums to develop their MTSTPro system. In the future, if XYZ wants to adopt agile methodology further, we will recommend XYZ learn the framework of SAFe and adopt agile methodology from up to down.

Roles and Approach

Roles Needed

- Scrum of Scrum Master (1) is responsible for focusing on progress and impediment backlogs visible to other teams, facilitating prioritization or removal of impediments, and continuously improving the effectiveness of Scrums of Scrums.
- Chief Product Owner (1) is responsible for having an overall vision of the entire product or product suite. This role helps maximize customer and market value.
- Product Owner (6) is responsible for maximizing the value delivered by the team and ensuring that the team is aligned with customer and stakeholder needs.
- Quality Assurance Manager (1) is responsible for overseeing the activity of the quality assurance department and staff, developing, and maintaining a system of quality and reliability testing for the organizations and products and/or development processes.
- Scrum Masters (6) are responsible for the goals of each specific system and their system requirements. There will be more than one scrum master for this larger group, and the scrum of scrum master will be the leader of all these scrum masters of each team.
- **Project Managers** (6) is in charge of the entire project and handle everything involved. He or she is responsible for the planning, procurement, execution, and completion of a project. For this larger group, we need more than one project manager to manage each system and sub-project. If needed they can play the roles of **Scrum masters or product owners.**

Based on systems requirements, other roles are listed below with each system:

- Technical Professionals (70) are responsible for designing and implementing all software to assist in the increase of all productivity for various engineering applications.
- R&D Subject Matter Experts (3) provide the knowledge and expertise in a specific subject, business area, or technical area for the projects
- Data Scientists (3) are responsible for analyzing data for actionable insights, including identifying the data analytics problems that offer the greatest opportunities to the organization and determining the correct data sets and variables.
- Systems Administrators (1) are IT professionals who make sure the systems are functioning and meeting the needs of the whole business.
- Server & Storage Engineers (3) are responsible for evaluating, implementing, and maintaining storage hardware and software, including storage subsystems, switches, and cabling.

- Database Administrators (3) are responsible for ensuring databases run efficiently. They create and organize systems to store and secure data.

Our team believes that project managers can also be scrum masters or product owners. They are responsible for the goals of each specific system and their system requirements. They can set up their own meetings for each team or system.

Delivery Approach

Our team believes that these projects should be project teams instead of long-standing product teams. Although the product team often uses the same group of people on the project throughout its duration, which may help to increase communication and idea-sharing, and is good for agile methodology, our team believes that we can start with the project teams and transfer to product teams later.

The reason is that we want to ensure XYZ uses an agile approach to implement the MTSTPro system. From the IT organization structure, we can see that XYZ is more familiar with the project team and waterfall development process. Since we change the Waterfall model to agile methodologies, we decides to maintain the project team with reserve. Adopting the agile process and product team simultaneously may confuse the employees, which may affect the delivery of the MTSTPro system. Thus, in this case, we believe developers can implement quickly and successfully working as project teams rather than product teams

We hope through the development of MTSTPro successfully, CIO and board can realize the benefits of agile and move towards the adoption of an agile approach totally. So same as SAFe, we recommend XYZ think about the product team for future work if they want to go further.

Agile Planning

Take our recommendation -Scrum- for example, for Agile planning, the steps are pre-planning - Planning - Release Planning - Iteration Planning - Managing Product Backlog.

Preplanning

Before the start of an Agile project, the organization identifies a project vision, product roadmap, and business case. The Agile project life cycle starts with a pre-planning step. This includes collecting and prioritizing business and technical requirements (the product backlog), team formation, and high-level time and cost estimation. The product backlog contains all requirements needed to deliver the final product, system, or service, and it is collected from various sources and then prioritized. The team and experts meet and provide high-level estimates for each feature, including the time required to perform all requisite architecture, development, and release activities. Based on these estimates, a first rough estimate for the entire project duration (time), as well as for the overall cost for the realization of the backlog (scope), can be determined. Estimating is an iterative process (Schwaber, 2003). In this step, the team also chooses the duration of the iterations if the organization upfront has not decided it.

Planning

The next step is the planning of releases and iterations (Exhibit 1). The organization can decide to release each iteration to the customer when it becomes available or release a collection of iterations at once in one defined release. Release planning can be done either right after the creation of the product backlog (Sliger, 2008) or after the first iterations have been completed (Schwaber, 2003).

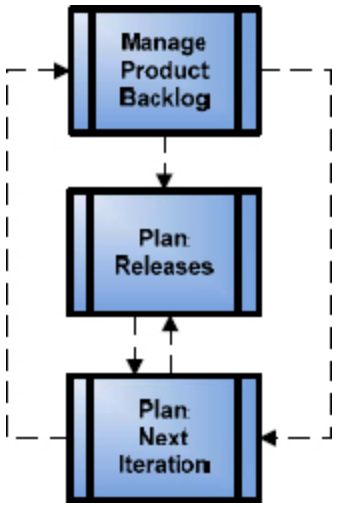


Exhibit 1: Processes for the planning of an Agile project (based on Sliger, 2008)

Release Planning

During release planning, the project manager, product owner, project team, and other stakeholders break the functionality in the product backlog into a number of iterations, ensuring that each iteration can be completed within the duration of an iteration. They then assign iterations to releases (release backlog). Every release delivers a working product increment. Based on this release plan, the dates for the release milestones and the final product release can be identified (time). The overall project cost can be determined from the labor cost of the Agile team and the number of identified iterations (Sliger, 2008). Overhead and material costs have to be added to this number. Exhibit 2 shows the inputs, tools and techniques, and outputs for the "Plan Releases" process.

Inputs	Tools & Techniques	Outputs
Product Vision Product Roadmap Prioritized Product Backlog Estimated or measured Team Velocity	Planning Meeting Brainstorming Dependency Determination Speculate Expert Judgment Analogous estimating	Release Plan - Assumptions - Risks - Action Items - Dependencies - Release Backlog Top-Down Cost estimate & updates

Exhibit 2: Plan Releases: Inputs, Tools and Techniques, and Outputs for an Agile Project (Sliger, 2008; Schwaber, 2002; PMI, 2004; Highsmith, 2004)

Iteration Planning

Agile project planning focuses on the planning of the next iteration. After each iteration, the project team meets to identify the content (scope) of the next iteration (iteration backlog). Exhibit 3 shows the activities during the iteration planning meeting. At the beginning of the meeting, the team verified that there is agreement on the product backlog and the priority of the listed features. Reprioritization will happen if needed, for example, when the team needs clarification about the current order of features listed in the product backlog (step 1 in Exhibit 3). After getting more detail on the features from the customer (if needed), the development team chooses features from the product backlog they think can be accomplished within the next iteration (step 2). In step 3 the team identifies the tasks (user stories) required to implement the chosen features, identifies initial task sequencing, assigns task owners, and estimates tasks (time). During step 3 the team might discover that the effort to complete the chosen features is more or less than the time allotted for the iteration. As a result, features might be removed or added to the iteration backlog (step 4). When functionality is moved back to the product backlog, it can lead to a reprioritization of the product backlog.

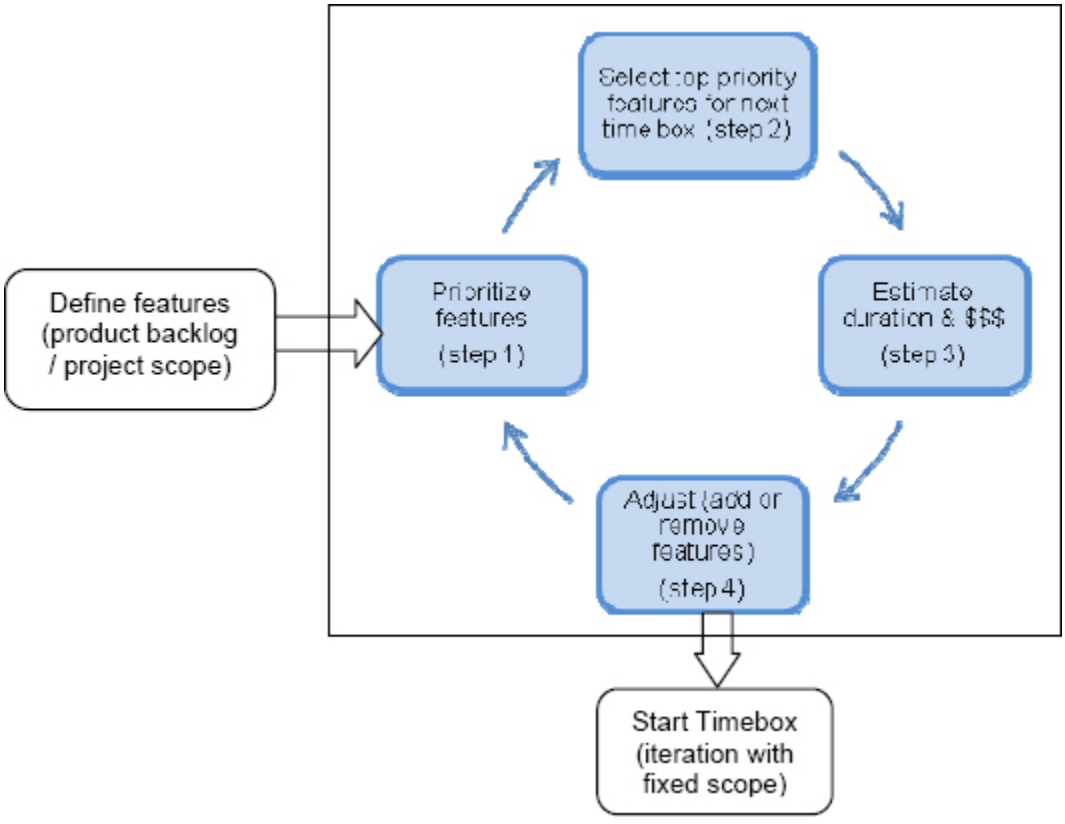


Exhibit 3: Iteration's planning process

The planning process is iterative. The team might go through the full planning circle several times. At the same time, planning meetings are limited to four hours. The guideline is that the team should spend at most two meetings to identify the scope and tasks of the next iteration (Schwaber, 2003). The idea is that the team learns from its actions and becomes better at estimating over time. After a number of completed iterations, the team members can measure their team velocity, which is the average time they spend on implementing one feature. When they understand and know their team's velocity, they can come up faster with their estimates during their planning meetings. Since iterations are short (one to six weeks), uncertainty is limited, and estimates are, in general, quite good.

The cost for an iteration can be determined by the labor cost of the Agile team for the duration of an iteration. As the product backlog and the team velocity change, the cost for the project changes too. Exhibit 4 shows the inputs, tools and techniques, and outputs for the "Plan Next Iteration" process.

Exhibit 4: Plan Next Iteration: Inputs, Tools and Techniques, and Outputs for an Agile Project (Sliger, 2008; Schwaber, 2002; PMI, 2004; Highsmith, 2004)

Manage Product Backlog

After each iteration, the time to complete the full project can be recalculated based on the results of the previous iteration and any changes the customer(s) has made to the product backlog, including priority changes or adding or deleting features on the product backlog. The scope is only locked in for the duration of the ongoing iteration.

As the product backlog can be modified throughout the duration of the project, the number of iterations and releases are subject to change as well. The outcome of iteration and release planning can influence each other (refer to Exhibit 1). In case fewer (or more) features would be selected during the planning of the next iteration, the content and number of future iterations could change. As a result, the collection of iterations assigned to one release could change as well. The activities of changing the scope, priority, and estimates of the product backlog are summarized in the “Manage

Product Backlog" process. See Exhibit 5 for the inputs, tools and techniques, and outputs for this process.

Exhibit 5: Manage Product Backlog: Inputs, Tools and Techniques, and Outputs for an Agile Project (PMI, 2004; Sliger, 2008; Schwaber, 2003)

Time management in Agile projects start in the pre-planning step, where the then-known project scope (product backlog) is estimated by feature and, on a high level, planned out by iteration and release. Since iterations have a fixed duration (timebox), the project timeline is based on the number of iterations necessary to deliver the product or service needed. In traditional planning approaches, the project timeline is calculated by identifying every task, sequencing the tasks, and estimating the individual tasks. Agile planning processes only look at task-level estimates to make sure a feature can be completed in an iteration.

Tracking Progress

Knowing the progress of the work at any time is important for the Scrum team, the senior management, and the stakeholders. As with traditional projects, comprehensive project control and weekly or monthly status reports do not exist in Scrum. However, this does not mean that the progress of the project, releases, or sprints is not monitored and reported.

In Scrum, work, progress and results are monitored in each Daily Scrum and in each Sprint Review. In case of deviations from the plan, one can quickly initiate actions, reschedule, possibly adapt the procedure and learn from it (inspect, adapt and learn).

At the sprint level, the Scrum team has very effective tools to monitor project progress with the Daily Scrum, Taskboard, and Burndown Charts. This allows the team to identify problems early, take action quickly and solve problems while it's still easy. This is a great advantage over non-agile projects.

The reports that the developers create a show which user stories have been implemented and whether work is progressing as planned. However, they do not show how much work has been done. In Scrum, it is not important who has worked how much and for how long. What is relevant is what functionality has already been delivered and where you stand in product development. They are actually not reports but Scrum artifacts. However, they are very useful when senior management or a sponsor wants to have periodic reports that show the progress of the project.

Openness and maximum transparency in the Scrum team and towards stakeholders is one of the five Scrum Values. Therefore, the Scrum team should make the information about the sprint or release progress as easily accessible as possible to all stakeholders. This creates trust, interest, and commitment. A picture of the taskboard, burndown, or parking lot charts is a good tool for this.

The following "reports" are often used in Scrum. Here in order of importance:

1. Taskboard
    - The taskboard represents the sprint backlog and gives an overview of all user stories that will be implemented in the sprint, and the associated tasks and shows the work progress.
2. Sprint Burndown Chart
    - In the Sprint Burndown Chart, the Developers visualize the remaining work in relation to the remaining time in the Sprint.
3. Product Burnup Chart

4. Release Burnup-Chart
5. Parking-Lot-Chart
    - The Parking Lot Diagram is a report originally taken from Feature Driven Development (FDD). For the Product Owner, it is important to know the degree of completion of functionality clusters and, thus, their release capability.
6. Velocity Chart
    - The velocity of development is not constant; rather, it varies. If the velocity has leveled out after a while, you can easily see whether the team is in a high or low phase.

Architectural Direction

Agile Architecture is a set of values, practices, and collaborations that support the active, evolutionary design and architecture of a system. This approach embraces the DevOps mindset, allowing the architecture of a system to evolve continuously over time, while simultaneously supporting the needs of current users. It avoids the overhead and delays associated with the start-stop-start nature and large-scale redesign inherent in phase-gate processes and Big Up Front Design (BUFD).

The agile architecture supports Agile development practices through collaboration, emergent design, intentional architecture, and design simplicity. Like Agile development practices, Agile architecture also enables designing for testability, deployability, and release ability. It is further supported by rapid prototyping, domain modeling, and decentralized innovation.

The architecture of a system can either accelerate or impede the ability to provide frequent, independent releases for the business to meet its objectives. Agile architects support business alignment by optimizing the architecture to support the end-to-end value stream. This enables the business to achieve its goal of continually delivering 'value in the shortest sustainable lead time'. Agile architects lead this process by supporting a "just enough" architectural runway to support evolving business needs. They continually invest in legacy modernization initiatives and know where to refactor to eliminate bottlenecks. They communicate the need for these ongoing technical objectives in clear business terms.

To support the continuous flow of value through the Continuous Delivery Pipeline, Agile architecture:

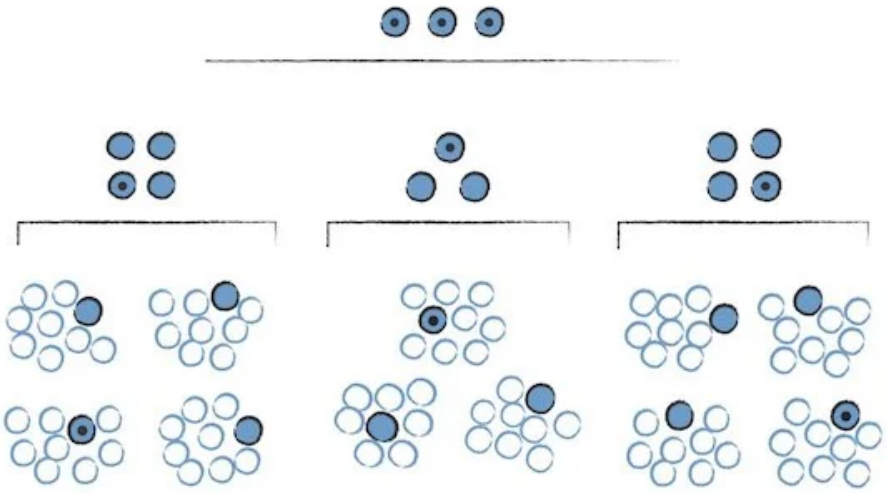
- Evolves over time while supporting the needs of current users
- Avoids overhead and delays associated with phase-gate and BUFD- methods
- Ensures the 'system always runs'
- Balances emergent design and intentionality
- Takes a systems view across the full value stream

We are applying the scrum of scrums in the second stage. An agile team in a Scrum environment often still includes people with traditional software engineering titles such as programmer, designer, tester, or architect. But on a Scrum team, everyone on the project works together to complete the set of work they have collectively committed to complete within a sprint, regardless of their official title or preferred job tasks. Because of this, Scrum teams develop a deep

form of camaraderie and a feeling that "we're all in this together." When becoming a Scrum team member, those who in the past fulfilled specific traditional roles tend to retain some of the aspects of their prior role but also add new traits and skills as well. New roles in a Scrum team are the ScrumMaster or product owner.

A typical Scrum team is three to nine people. Rather than scaling by having a large team, Scrum projects scale through having teams of teams. Scrum has been used on projects with over 1,000 people. A natural consideration should, of course, be whether you can get by with fewer people. Although it's not the only thing necessary to scale Scrum, one well-known technique is the use of a "Scrum of Scrums" meeting. With this approach, each Scrum team proceeds as normal, but each team identifies one person who attends the Scrum of Scrums meeting to coordinate the work of multiple Scrum teams. These meetings are analogous to the daily Scrum meeting but do not necessarily happen every day. In many organizations, having a Scrum of Scrums meeting twice a week is sufficient.

The illustration below shows how a Scrum of Scrums facilitates cross-team coordination. Each circle represents one person on a Scrum team. The bottom row of this illustration shows teams with eight or nine members each. One person from each team (the shaded circle) also participates in a Scrum of Scrum to coordinate work above that team. Those teams further coordinate their work with a Scrum of Scrums.



Contributors

Identify the team members and each individual's contribution to the paper:

Guorui Li: Executive Summary(corporate with Xiaoyang), Part I introduction, Part I Methodology Overview, Part I Analysis & Recommendation, Part II introduction, Part II Methodology Overview, Part II Analysis & Recommendation,

Xiaoyang Wei: Executive Summary (corporate with Gray), Part 1 Roles Needed, Part 2 Roles Needed, Delivery Approach.

Michael Huang: What tools or products could be considered for evaluating for managing their work? CIO's biggest concern, is the three questions, how agile do planning, what is their way to attract plan, and what architectural direction?