STATE OF MICHIGAN

SUITE Agile Process Guide

Another Companion to the Systems Engineering Methodology (SEM) of the State Unified Information Technology Environment (SUITE)

Michigan Department of Technology, Management & Budget

www.michigan.gov/SUITE

This guide is under revision

January 2015

Version 2.0

PREFACE

Agile is about collaboration, communication, and continuous improvement; if your team (including the customer) is not committed to these three items at the onset, Agile will not succeed. Many of the best practices associated with Agile have been successful in the public and private sectors for years. The State of Michigan has benefited from successful Agile projects, and will benefit from the continued growth in the use of Agile.

Agile is not a methodology. It is simply a list of four statements (The Agile Manifesto) and twelve supporting sentences (Principles of Agile Software). Users may apply best practices associated with Agile (e.g., sprint cycles or paired programming) to improve their team's existing methodology. This may sound like a nuance, but it is actually an important distinction.

The purpose of this process guide is to explain how to move forward with our approach to using Agile within the State of Michigan.

This process guide has been developed as part of a continuing effort to improve the quality, performance, and productivity of State of Michigan information systems.

REVISION HISTORY

The following information is used to control and track modifications to this document.

Revision Date	Section(s)	Summary
July 2012	All	Initial document creation
January 2015	All	Revisions for clarity and content

ACKNOWLEDGEMENTS

Numerous individuals and organizations have made significant contributions to the continuous improvement, standardization, and implementation of Agile at the State of Michigan. The following team of dedicated practitioners made this version of the Agile Process Guide possible.

Agile Process Guide, second release, January 2015	
Ward Beauchamp, PMP, CSM Director, Enterprise Portfolio Management Office DTMB	Mark Kinnamon, PMP, CSM Director, Project Management Division Enterprise Portfolio Management Office DTMB

John Carey Development and Operations Manager serving the Center for Educational Performance and Information (CEPI) DTMB	Krisanne McConnell PMO Manager serving the Center for Shared Solutions Enterprise Portfolio Management Office DTMB
Virginia Hambric, PMP, CSM Senior Project Manager, Enterprise Portfolio Management Office DTMB	Suzanne Pauley Director, eMichigan Center for Shared Solutions DTMB

Table of Contents

TABLE OF CONTENTS

CHAPTER 1.0 - INTRODUCTION AND BACKGROUND

The purpose of this guidebook is to provide background information, examples, and support for using Agile software development best practices within the context of the Department of Technology and Budget (DTMB) Standard Unified Information Technology Environment (SUITE) methodology.

Why Agile, and Why Now?

Agile best practices have been a part of mainstream software development for years, and continue to evolve and gain acceptance. Agile adoptees expect to gain the following:

- More frequent, smaller product releases
- Shorter maintenance streams (inherent Agile Quality Assurance (QA) reduces post-production support)
- Quicker (more Agile) response to changing customer demands
- More consistent coding practices leading to uniform support across a development team (focus is on teambased support, not on a particular individual)

The call for Agile adoption has come from many sectors – large and small, public, private and government alike. Of particular interest to the DTMB, Agile best practices are equally well suited for application production support and for new development efforts. Agile fits within the existing SUITE framework and this document focuses on the Agile process as adopted by the State of Michigan.

Considerations for using Agile

Agile can be used for any project (for new development, as well as for break-fixes and application enhancements), but is much more likely to be successful when you have:

- The full support of management and the development team
- Product Owner commitment, as they are an active part of the team
- A team that understands and promotes self-organizing teams, collaboration and individual responsibility
- If the team is new to Agile, an Agile coach can increase the likelihood of a successful project outcome
- A stable, well defined source code repository and check-in process

When getting started with Agile, smaller projects are probably better than large projects as this gives the team a chance to get used to the new processes. Troubled projects, where team members are looking for a new approach to stalled development and delivery efforts, may actually be good Agile candidates as well. Conversely, using Agile will be much more challenging when:

- The team is geographically dispersed (although there is technology available to help with this challenge)

- The team consists of more than 8 to 10 persons
- The Product Owner does not have the authority to make decisions, is not readily available on a daily basis or simply doesn't understand their role
- The project requirements are absolutely rigid and must be completely defined up front

Training Considerations

A well thought out Agile training plan (prior to using Agile) is essential for everyone involved. It should be tailored to fit the team's needs, but usually includes the following:

- Definition of team member
- Scheduled visits to existing DTMB Agile Team Scrums ('You can't do Agile if you don't see Agile')
- Scheduled visits to existing DTMB Agile Team planning sessions
- Formal training in Agile best practices

CHAPTER 2.0 - AGILE WITH SUITE AND CMMI

The State Unified Information Technology Environment (SUITE) provides methodologies, procedures, training, and tools for project management and systems development lifecycle management for all State of Michigan agencies. SUITE provides the framework for implementing repeatable processes in an effort to conduct systems development activities according to Capability Maturity Model Integrated (CMMI) Level 3 requirements. It is intended to ensure reliable, cost-effective, high quality Information Technology solutions that promote customer success.

CMMI is a collection of best practices for software maintenance and development. The focus of SUITE / CMMI is on the ability to manage the development, acquisition, and maintenance of technology products and services. CMMI is a model for process improvement that provides a common, integrated vision of improvement for all elements of a technology-based organization. It has been widely adopted by the private sector and is the most endorsed benchmark for delivery of reliable, high quality technology products and services.

SUITE primarily consists of the Project Management Methodology (PMM) and the Systems Engineering Methodology (SEM). The PMM aligns with industry standard best practices to promote on time, on budget delivery of products and systems that meet customer expectations. The SEM provides guidance for information systems development related activities and software quality assurance practices. The primary purpose of the SEM is to promote the development of reliable, cost-effective, computer-based solutions while making efficient use of resources. By policy, all State of Michigan development projects must utilize the PMM and SEM.

CHAPTER 3.0 - AGILE SUMMARY

Manifesto for Agile Development

In February 2001, a group of well-known software engineers met at Utah's Snowbird Ski Resort. They brainstormed ways to improve software development, based on their collective experience, and came up with the following manifesto:

Manifesto for Agile Software Development

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

That is, while there is value in the items on the right, we value the items on the left more.

Original Agile Manifesto Authors:

Kent Beck	Mike Beedle	Arie van Bennekum
Alistair Cockburn	Ward Cunningham	Martin Fowler
James Grenning	Jim Highsmith	Andrew Hunt
Ron Jeffries	Jon Kern	Brian Marick
Robert C. Martin	Steve Mellor	Ken Schwaber
Jeff Sutherland	Dave Thomas	

As a follow-up to the Manifesto, they developed the following principles:

Twelve Principles of Agile Software

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
4. Business people and developers must work together daily throughout the project.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity - the art of maximizing the amount of work not done - is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

What Agile is and isn't

- Agile is a framework around which to build and continuously improve a team approach toward software development. • Agile is often associated with a non-exclusive set of best practices, such as:
    - Paired programming ○ Swarming ○ Time Boxing ○ Scrum
    - Sprints (or iterations) ○ Test Driven Development ○ Continuous Integration ○ Continuous Improvement ○ Note that many of these practices are used outside of the Agile world.
- When done right, Agile is structured.
- Agile is **not** a methodology. It is simply a four point manifesto and twelve guiding principles.

CHAPTER 4.0 - AGILE AT THE STATE OF MICHIGAN

The State of Michigan has chosen Scrum as its primary Agile framework. Other Agile best practices (e.g., paired programming, swarming) can be incorporated based on the needs of the project.

The Scrum Team

The Agile Scrum Team consists of three fundamental roles: the Product Owner, the Scrum Master and the team. The responsibilities of each role are listed below:

Scrum Master:

- Single point of contact for the project
- Coaches the team on Scrum values and practices
- Facilitates the Daily Stand-Up Meeting
- Ensures that the team is fully functional and productive
- Enables close cooperation across all roles and functions
- Removes impediments
- Shields the team from external interferences

In addition to the above project management responsibilities, the SOM model includes the responsibilities for following the SUITE PMM framework and maintaining the project in the State of Michigan Project Portfolio Management (PPM) tool.

Therefore, for most projects the SOM model combines the Project Manager and Scrum Master roles into a single role. For large complex programs (multiple projects) there may be a requirement for a Project Manager with multiple Scrum Masters. The project management approach is determined by the PMO.

Product Owner:

- Empowered by the customer to convey the product vision to the team
- Outlines work in the product backlog (user stories and acceptance criteria)
- Prioritizes user stories based on business value (responsible for determine what work gets done)
- Communicates with other stakeholders (to make sure their interests are taken into account)
- Responsible for authorizing the product budget

- Must be available to the team to answer questions and deliver direction
- Required to attend sprint reviews and planning sessions

The Product Owner is the person that is ultimately responsible for bridging the gap between the business and the development team. This person should be a subject matter expert on the business needs and priorities and have the authority to make decisions regarding the project. The Product Owner must be readily available to the team to answer Chapter 4.0 – Agile at the State of Michigan

questions and clarify requirements.

Team:

- Participates in the daily stand-up
- Responsible for sprint review (demo)
- Participates in sprint planning by estimating user stories and tasks to complete the user stories
- Self-organizing (responsible for deciding how the work gets done)
- Commits to the work that will be in the sprint
- Responsible for inspecting and adapting the process (retrospectives)

The team is the group of people who do the work of creating the end product. The team can consist of programmers, testers, designers and anyone else who has a hands-on role in the creation of the product. The ideal size for the team is 5-9 people. Team members are dedicated to the project at effort and duration levels necessary to meet agreed upon commitments.

The Scrum Process

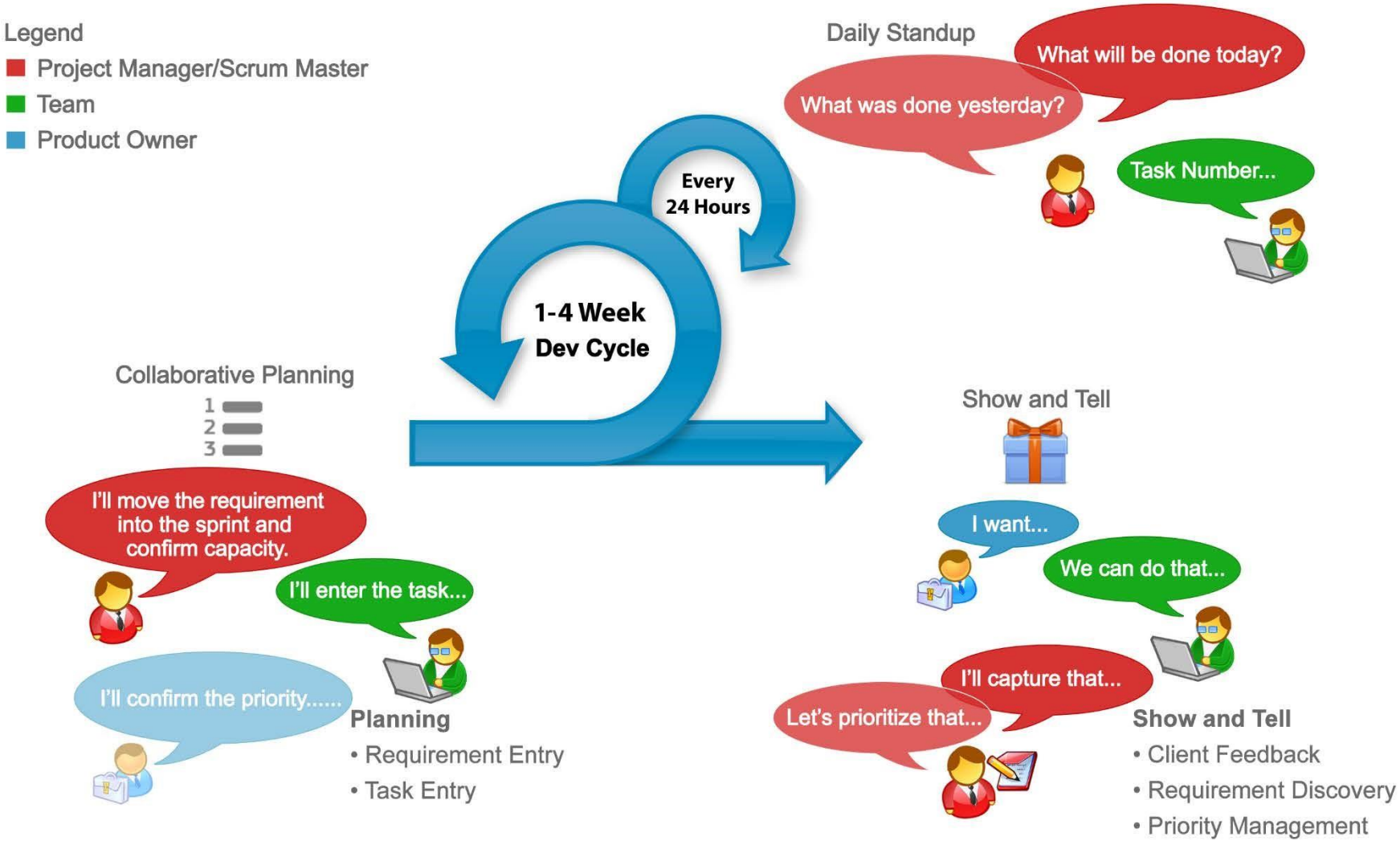
In Scrum, each iteration begins with a sprint planning meeting. At this meeting, the Product Owner and the team review the outstanding user stories and determine which ones will be tackled during that iteration/sprint. Time-boxed to four hours, this meeting is a conversation between the Product Owner and the team. It is the Product Owner's responsibility to decide which user stories are considered the highest priority to the release and which will generate the highest business value, but the team has the responsibility to voice concerns or identify impediments to moving forward.

When the team commits to the work, the corresponding user stories are put into the sprint backlog. If an electronic tool is not being used, this might be physically represented by moving a Post-It note or index card with a story written on it from the product backlog into the sprint backlog.

At this point, the development team decomposes the sprint backlog items into tasks. If the Product Owner is not present, he or she is still expected to be "on call" to answer questions, clarify acceptance criteria, or renegotiate. This

meeting, which was previously called the sprint planning meeting, is also time-boxed to four hours to limit all sprint planning activities to a single day's time.

Diagram of Major Sprint Components


Legend
Project Manager/Scrum Master
Team
Product Owner
Daily Standup
What will be done today?
What was done yesterday?
Task Number...
Every 24 Hours
1-4 Week Dev Cycle
Collaborative Planning
1
2
3
I'll move the requirement into the sprint and confirm capacity.
I'll enter the task...
I'll confirm the priority.....
Planning
• Requirement Entry
• Task Entry
Show and Tell
I want...
We can do that...
I'll capture that...
Let's prioritize that...
Show and Tell
• Client Feedback
• Requirement Discovery
• Priority Management

Scrum and Its Components

User Story

A user story is a short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or customer of the system. They typically follow a simple template:

As a <type of user>, I want <some goal> so that <some reason>.

A user story should contain acceptance criteria, provided by the Product Owner, for the team to use in determining when a story is complete. Acceptance criteria typically follows the pattern below:

When I do this <take some action>, this happens <the system does something>.

User stories are often written on index cards, sticky notes or stored in a tool, and arranged on walls or tables to facilitate planning and discussion. As such, they strongly shift the focus from writing about features to discussing them. In fact, these discussions are just as important as whatever text that is written.

Product Backlog

The product backlog is a prioritized features list, containing short descriptions of all functionality desired in the product. When using Scrum (an iterative and incremental Agile software development process for managing software projects and product or application development), it is not necessary to start a project with a lengthy, upfront effort to document all requirements. Typically, a Scrum Team and its Product Owner begin by writing down everything they can think of easily. This is almost always more than enough for a first sprint. The product backlog is then allowed to grow and change as more is learned about the product and its customers.

A typical product backlog comprises the following different types of items:

- Features
- Bugs
- Technical work
- Knowledge acquisition

Sprints

Sprints (also called iterations) are a core component of Agile. The team defines a time period (typically 1 – 4 weeks) during which they will develop and deliver a working product. Following are the major sprint elements:

The sprint backlog is the list of work the team must address during the next sprint. The list is derived by selecting stories/features from the top of the product backlog until the team feels they have enough work to fill the sprint. This is done by the team asking, "Can we also do this?" and adding stories/features to the sprint backlog. The team

should keep in mind their velocity (total story points from the previous sprints' stories) when selecting stories/features for the new sprint and use this number as a guideline of how much "effort" they can complete.

The stories/features are broken down into tasks by the team, which, as a best practice, should normally be between four and sixteen hours of work. With this level of detail the whole team understands exactly what to do, and potentially, anyone can pick a task from the list. Tasks on the sprint backlog are generally not assigned; rather, tasks are selected by the team members during the Daily Scrum Meeting, according to the set priority and the team member skills. This promotes self-organization of the team, and developer buy-in.

The sprint backlog is the property of the team, and all included estimates are provided by the team. A task board is used to visually display and update the state of the tasks in the current sprint. The state of tasks are often described as "to do," "in progress" and "done."

Stand-up

The daily stand-up is a brief meeting that occurs each day during the sprint. This meeting has specific guidelines:

- The meeting starts precisely on time
- All are welcome, but normally only the core roles speak (Scrum Team)
- The meeting length should be no longer than 15 minutes.
- The meeting should happen at the same location and same time every day

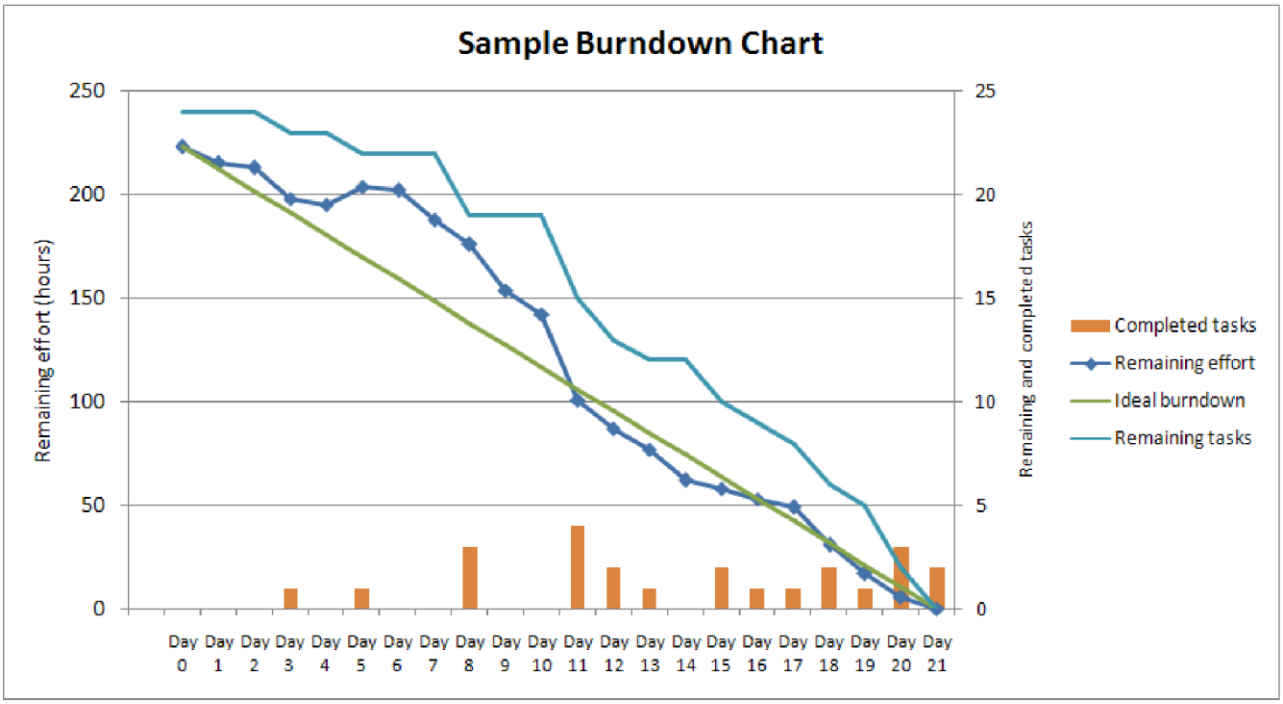
During the meeting, each team member answers three questions:

- What have you completed since yesterday?
- What are you planning to complete today?
- Any impediments/stumbling blocks?

It is the role of the Scrum Master to facilitate resolution of these impediments, although the resolution should occur outside the Daily Scrum Meeting itself to keep it under 15 minutes.

Burn Rate

The burn rate is viewed on a sprint burn down chart showing remaining work in the sprint backlog. Updated every day, it gives a high level view of the sprint progress. There are also other types of burn down. For example the release burn down chart that shows the amount of work left to complete the target commitment for a product release (normally spanning through multiple iterations):



Sprint Review

The sprint review serves the following purposes:

- Review the work that was completed and not completed
- Present the completed work to the stakeholders (a.k.a. “the demo”)
- Meeting length should be limited to one to four hours and can be combined with the retrospective.

Sprint Retrospectives

The sprint retrospectives have the following characteristics:

- All team members reflect on the past sprint
- Team is tasked with making continuous process improvements
- Two main questions are asked in the sprint retrospective: What went well during the sprint? What could be improved in the next sprint? Many times, the team will prioritize items to be improved, and only select one or two, so that it can hold itself accountable for real progress at the next retrospective

- Meeting length should be limited to one to four hours and can be combined with the review meeting

CHAPTER 5.0 – TAILORING SUITE PMM AND SEM FOR AGILE

The objective of this chapter is to provide guidelines and instructions on applying SUITE when using Agile best practices, such as Scrum and sprint iterations. To fully benefit, the users should have a good working knowledge of both SUITE and Agile best practices. For links and reference materials, see the References section of this document.

Differences between Agile and Waterfall Development

Agile is a different approach to system development than the traditional waterfall method. Under the waterfall approach, each SEM stage is completed and approved before the next stage can begin. For example, all Requirements are defined and approved prior to completing the Functional and System Design. They in turn are completed and approved prior to Construction and Testing. Under Agile, the system is defined, constructed and implemented over time, using a series of smaller efforts known as sprints. In effect, each sprint becomes an iteration of all the SEM stages, from Initiation and Planning through implementation.

Agile Projects and the PMM

SUITE may be tailored to meet the business needs of Agile projects, just as it is permitted for non-Agile projects. The requested information must still be provided, in whatever manner makes the most sense. As an example, information related to functional requirements may be housed in Team Foundation Server (TFS) story and task descriptions, with a link to that information being provided on the SUITE form.

Agile development should be conducted within the confines of a normal project, using the State of Michigan's Project Management Methodology (PMM). Projects must be defined by a formal Project Charter, have a defined scope, and utilize all the usual project templates.

In Agile development projects, requirements changes are expected and welcome, even late in the project. Most changes will be adequately handled within the Sprint planning process. However, the Scrum Master must review each requested change against the defined project scope. Changes that impact the major milestones of the schedule and/or budget of the project must be documented as change requests in the enterprise Project Portfolio Management (PPM) tool.

Since its first release in 2000, the State's PMM has continuously improved its processes and approach as the environment demanded. In May of 2013, PMM reduced the number of forms from 23 to 4. In March of 2014 the State released a significantly updated PMM manual. The four forms and the manual provide flexibility for agile projects.

The following four forms serve an important purpose for all projects:

- The Project Charter is important because it is the first step in the State's project management methodology and formally initiates project activities through authorization by the project sponsor. The

project charter provides a high level description of the project and initial project planning estimates. The charter also includes approvals of the sponsors to begin work on the project as well as the commitment of necessary resources such as budget and personnel. This document may serve as the Product Vision Statement, which is traditionally used in Scrum Projects, and elements of the Vision Statement may be incorporated into the Project Charter.

- Planning is important and the PMM Project Management Plan provides more detail than the project charter. However sections of the project management plan should be tailored for Agile projects.

- The purpose of the Lessons Learned document is to help the project team share knowledge gained from experiences that may benefit the entire organization in their future project work. This knowledge includes both positive and negative findings, and identifies practices the organization wants to repeat as well as avoid in the future.

- The purpose of the Project Closure Report is to provide a summary of the products delivered, comparison of baseline plans and actual performance, project metrics, lessons learned, and feedback from stakeholders. This document also includes a list of outstanding issues and defects, if any. This report serves as the official closure of the project and provides a permanent record for reference for future project teams.

The following is a sample PMM tailoring matrix applicable for most agile projects.

Document	**Document Section**	**Sample Agile Project Tailored Equivalent**
Project Charter	All	No Tailoring (Project Charter will be completed)
Project Management Plan	Project Summary	No Tailoring (Project Management Plan will be completed)
Project Management Plan	Project Schedule	Can be tailored to contain information on sprints and or releases (See sample in appendix A)
Project Management Plan	Human Resource Management Plan	Tailor as needed to include Agile team roles such as Product Owner, Development Team, etc.
Project Management Plan	Project Budget Estimate	No Tailoring (Project Management Plan budget will be completed)
Project Management Plan	Communication Management Plan	Tailor as needed to include the Agile ceremonies such as daily standups, sprint planning meetings and sprint reviews

Project Management Plan	Change Management Plan	Change management is a strength of agile projects. The Product Owner manages changes in the course of sprint planning and backlog grooming. However, changes to the release schedule and cost must not only be approved by the Product Owner but tracked in the enterprise PPM tool. This should be reflected in this section of the document.
Project Management Plan	Quality Management Plan	The sprint review process serves the
		same purpose as the structured walkthrough and stage exit process when done with the participation of the entire Scrum Team including Product Owners. Depending on the needs of the project, a separate Review and Approval form (SEM0185) can be created for each sprint performed during the life of the project. At a minimum, the form must be completed and signed for each release as it serves as the stage exit for production implementation.
Project Management Plan	Risk Management Plan	Tailoring is probably not necessary as risk management is a critical aspect of every successful project.
Project Management Plan	Issue Management Plan	Tailoring is probably not necessary as issue management is a critical aspect of every successful project.
Project Management Plan	Approval Information	Acceptance of the document is still required. Tailoring is not needed in this section.
Lessons Learned	N/A	Retrospective information can be utilized to update the Lessons Learned document on regular basis, for example, after every release.
Project Closeout	N/A	Tailoring of the project closeout report should reflect the tailoring in the project management plan.

Agile Projects and the SEM

While SEM tailoring for projects is encouraged, some key components must be included for all projects regardless of methodology such as the Enterprise Architecture Solution Assessment (EASA) and Security Assessment and Plan (DTMB-0170).

The following is a sample SEM tailoring matrix applicable for most agile projects.

SUITE Process/Document	Sample Agile Project Tailored Equivalent
Requirements Specification	Product Backlog consisting of user stories with acceptance criteria.
Requirement Traceability Matrix	Establish the linkage between product backlog item, tasks, test cases using an automated tool or spreadsheet such as the Requirements Traceability Matrix.
Functional Design Document	Prototype(s)/Wireframes can serve as documentation of functional design. The individual sprint reviews will serve as the approval for the design.
System Design Document	Tailoring may be considered based on team decision. System Design can be documented in a number of ways including a system design document or commented code. Some documentation is preferred as it helps orient new team members to the technical architecture of the system.

Stage Exits and Structured Walkthroughs	As stated in the Quality Management Plan, the Scrum process provides sprint demos or points in time during the project when the customer and stakeholders (as documented in the Roles and Responsibilities List) will review the deliverables in detail and accept or reject the work (or accept with noted revisions). Every effort will be made to identify all stakeholders and plan for their participation in the acceptance process. Each deliverable will be reviewed and approved if required before proceeding to the next sprint. Depending on the needs of the project, a separate Review and Approval form (SEM-0185) can be created for each sprint performed during the life of the project. At a minimum, the form must be completed and signed for each release as it serves as the stage exit for production implementation. These agile processes replace the stage exit and structured walkthrough forms.
Test Plan	Test planning including resource planning, environment preparation, any automated scripting and process for defect resolution should be included in either the quality section of the project management plan or in an SEM test plan document.
Test Cases	Should include steps to execute against the user story acceptance criteria, including expected results. Can be created using an automated tool, spreadsheet or an SEM Test Case document.
Maintenance Plan and Installation Plan	Depending on project needs, tailoring may combine (or keep separate) these two documents.
Review and Approval Form	Must be created or updated as part of the release. This serves as the stage exit for production implementation.
Configuration Management Plan	Note: An umbrella configuration management plan may be used for multiple systems or applications.
Security Plan and Assessment	Must be created or updated as part of the project.
EASA	Must be created or updated as part of the project.

SUITE Touch Points

Project Managers must take care to properly utilize the SEM Touch Points and ensure deliverables dependent on other areas of DTMB are available when needed (i.e., Procurement, Infrastructure Services, Enterprise Security, Enterprise Architecture, etc.). Failure to do so could result in significant delays to individual sprints and the project as a whole.

Review and Approval Template (SEM-0185)

The Review and Approval Form (SEM-0185) is the SEM form used for documenting the planning and implementation of each new sprint and/or release. Depending on the needs of the project, a separate form can be created for each sprint performed during the life of the project. At a minimum, the form must be completed and signed for each release as it serves as the stage exit for production implementation.

The Review and Approval Form provides two important functions. First, it documents the planned functionality to be implemented by the sprint or release and the planned implementation date. Secondly, it provides formal approval for the new functionality to be implemented.

The specific fields on the Review and Approval Template include:

1. System or Project Name
2. Sprint/Release Number and Description
3. Deliverables Subject to Approval 4. Scrum Master Name 5. Metrics:
    a. Sprint/Release Planning Begin Date
    b. Planned Sprint/Release Implementation Date
    c. Actual Sprint/Release Implementation Date
6. Sprint/Release Overview (narrative of the included features and functionality)
7. Sprint/Release Planning Approval (Name / Title, Signature, and Date):
    a. Client Agency Project Sponsor or Representative
    b. DTMB Project Sponsor or Representative
    c. Scrum Master
8. Sprint/Release Implementation Approval (Name / Title, Signature, and Date):
    a. Client Agency Project Sponsor or Representative
    b. DTMB Project Sponsor or Representative
    c. Scrum Master

REFERENCES

1. State of Michigan Project Management Methodology
http://www.michigan.gov/suite 2. CMMI Institute
http://whatis.cmmiinstitute.com/ 3. Manifesto for Agile
Software Development http://www.Agilemanifesto.org/

4. Federal Agile Blog http://www.federalAgilesolutions.com/

5. 25 Point Plan to Reform Federal Information Technology Management
https://www.dhs.gov/sites/default/files/publications/digital-strategy/25-point-implementationplan-to-reform-federal-it.pdf

6. Wikipedia Scrum (Development) http://en.wikipedia.org/wiki/Scrum_(development)

7. Martin Fowler – Continuous Integration
http://www.martinfowler.com/articles/continuousIntegration.html

8. Wikipedia – Pair Programming http://en.Wikipedia.org/wiki/Pair_programming

9. Wikipedia – Agile Software Development
http://en.wikipedia.org/wiki/Agile_software_development

10. Find White Papers – An Introduction to Agile Software Development
http://www.findwhitepapers.com/content11433

11. TechLife Columbus – The Promise and Reality of Agile after 10 Years
http://www.meetup.com/techlifecolumbus/events/28186891/?eventId=28186891&action=detail

12. Broadsword http://www.broadswordsolutions.com/resources/

13. Top 5 Software Development Process Challenges (white paper)
http://informationweek.com/whitepaper/development/tools/top-5-software-development-process- challenges-wp1318601421?articleID=191703607&cid=wallstreetandtech_fture_wp_default&itc=wallstreetandtech_fture_wp_default

14. ScrumWorks Pro® http://www.open.collab.net/products/scrumworks/capabilities.html

15. Pro-Tech Professional Technical Services, Inc. – Scrum Cheat Sheet
http://www.protechtraining.com/pdf/ScrumCheatSheet.pdf

GLOSSARY

Acceptance Criteria – The criteria that have to be met for a story to be assessed as complete.
Link: http://scrummethodology.com/scrum-acceptance-criteria/

Backlog – A comprehensive to-do list, expressed in priority order based on the business value each piece of work will generate.
Link: http://scrummethodology.com/the-scrum- backlog/

Baseline – An agreed-to description of the attributes of a product, at a point in time, which serves as a basis for defining change. (2) An approved and released document, or a set of documents, each of a specific revision; the purpose of which is to provide a defined basis for managing change. (3) The currently approved and released configuration documentation. (4) A released set of files comprising a software version and associated configuration documentation.

Burn Down Chart – A graphical representation of the work left to do versus time. Link: http://en.wikipedia.org/wiki/Burn_down_chart

Burn Rate – The rate at which hours allocated to a project are being used.
Link: http://en.wikipedia.org/wiki/Burn_rate

Configuration Audit – Configuration Audits determine to what extent the as- designed/as-tested/as-built product reflects the required physical and functional characteristics specified in the requirements. Functional Configuration Audit (FCA) and Physical Configuration Audit (PCA) are done once, to establish the Product Baseline.

Development Baseline – The baseline comprising the software and associated technical documentation that define the evolving configuration of the system during development. This baseline includes the software design, and implemented code, database schema, and COTS products, and evolves into the product baseline.

Fix number – An indicator of small updates that are to be built into a regular modification or release at a later time. The version, release, modification, and fix levels together comprise the program level (or version) of a program.

Functional Baseline – The baseline comprising documentation and possibly models that specify system functional, data, and technical requirements.

Functional Configuration Audit (FCA) – An inspection to determine whether the (software) configuration item satisfies the functions defined in specifications. Consists of someone acknowledging having inspected or listed each item to determine it satisfies the functions defined in specifications.

Impediment – In Scrum, an impediment is anything that keeps a team from being productive.

Link: http://scrummethodology.com/scrum-impediments/

Iteration – A distinct sequence of activities with a baselined plan and valuation criteria resulting in a release.

Modification number – The modification level of a program which is an indicator of changes that do not affect the external interface of the program. The version, release, modification, and fix levels together comprise the program level (version) of a program.

Physical Configuration Audit (PCA) – The formal examination of the "as-built" configuration of a configuration item against its technical documentation to establish or verify the configuration item's product baseline.

Product Owner – The primary person responsible for a project's success. The Product Owner leads the development effort by conveying his or her vision to the team, outlining work in the scrum backlog, and prioritizing it based on business value.
Link: http://scrummethodology.com/scrum-product-owner/

Program level – The version, release, modification, and fix levels of a program.

Regression Testing – The process of running a composite of comprehensive test cases that are always run after system modifications to detect faults introduced by modification.

Release number – An indicator of changes to the external programming interface of the program. The version, release, modification, and fix levels together comprise the program level (version) of a program.

Scrum – An iterative and incremental Agile software development method for managing software projects and product or application development.
Link: http://en.wikipedia.org/wiki/Scrum_(development)

Scrum Master – Team member who ensures the process is followed, removed impediments, and protects the development team from disruption.

Specification – A document that explicitly states essential technical attributes/requirements for a product and procedures to determine that the product's performance meets its requirements/attributes.

Version – (1) One of several sequentially created configurations of a data/document product. (2) A supplementary identifier used to distinguish a changed body or set of computer-based data (software) from the previous

configuration with the same primary identifier. Version identifiers are usually associated with data (such as files, databases and software) used by, or maintained in, computers.

Version Description Document (VDD) – A document associated with a product release that describes the version released and describes the versions of the items included in the product.

Version Number – An indicator of the hardware and basic operating system upon which the program operates. The version, release, modification, and fix levels together comprise the program level (version) of a program.