PM.2204

Can Your Project's Organization Be Agile?

Joseph A. Lukas, PE CCP

Abstract—Agile is a group of project methodologies based on iterative development, where requirements and solutions evolve through collaboration by self-managing, cross-functional teams. Agile has been effectively used with custom software and product development projects. Unfortunately, based on this success, the management in some companies is advocating the use of agile on construction projects.

This paper will describe the Agile Manifesto and the guiding principles for agile use. The differences between agile and waterfall life cycles will be explained, along with recommendations on when to use each life cycle model. Planning and team organization on agile projects will be discussed, along with two of the more popular agile methodologies, which are Scrum and Kanban. Even though an agile life cycle model is not appropriate for construction projects, this paper will explain how to successfully utilize specific agile tools and techniques that can add value.

Table of Contents

List of Figures

List of Tables

Introduction

By definition, a project is a "temporary endeavor undertaken to create a unique product, service or result [1, p.553]." There are different types of projects done by project teams. Members of the cost management community typically deal with construction projects such as new manufacturing facilities, office and laboratory buildings, and infrastructure projects such as roads, bridges and airports. These projects include planning, engineering design, build, and start-up phases.

However, projects also include product development, and with the advent of a world economy and the electronics age, getting to market quicker with new products has become a priority. With the development of modern computing after World War II, information technology has also become another important and growing category of projects. The point is that the world of projects is much bigger than just construction projects. Other categories of projects include the following:

- Research, such as a new drug development
- Business process change, such as automating the manual invoicing done by an accounting department
- Company reorganization, such as merging two business units
- Political campaign, with the project objective of getting a candidate elected

Historically, the life cycle method used to complete projects has been the waterfall model. The Total Cost Management (TCM) Framework summarizes waterfall as four sequential phases as follows [2, p. 39]:

1. Ideation – overall requirements are given, and the project team assesses alternative concepts for performing the project and selects an optimal performance strategy.
2. Planning – the project team develops project plans that address the strategic requirements and selected performance strategy.
3. Execution – The plans are implemented through the execution of planned project activities.
4. Closure – the asset or deliverable is reviewed, tested, verified, validated, and turned over to the customer.

With the waterfall methodology, each project phase is typically completed before the next phase begins, even though projects can be done with some amount of phase overlap. In addition, the sequential phases can be recursive. For example, a project to construct a new sports stadium on an old and contaminated industrial site will include the four sequential phases for the environmental cleanup, along with the four phases for building the stadium.

While the TCM waterfall life cycle model has been effective for construction projects, it is not in alignment with the evolving expectations of clients. The Project Management Institute (PMI®) has done extensive market research with global companies to identify the project management

skill employers need now, and in the future, from project professionals to deliver better results. The outcome of this research is called the talent triangle with the three legs as follows [3]:

- Technical skills such as risk management and critical path method scheduling
- Leadership skills specific to motivating and guiding others
- Strategic and business management skills that enhances performance and better delivers business outcomes

The implication from this research is that clients want project personnel to have a business focus. This includes pre-project (also called genesis), which is where the business benefits and business case are established. It also includes post-project benefits realization, where the project benefits are brought to fruition. The measure of succeed for completed projects is no longer staying within budget and schedule, and delivering all scope functionality. The new project success measure is delivering the expected business outcome.

Based on this business management focus, a more complete waterfall life cycle model is shown in Figure 1.

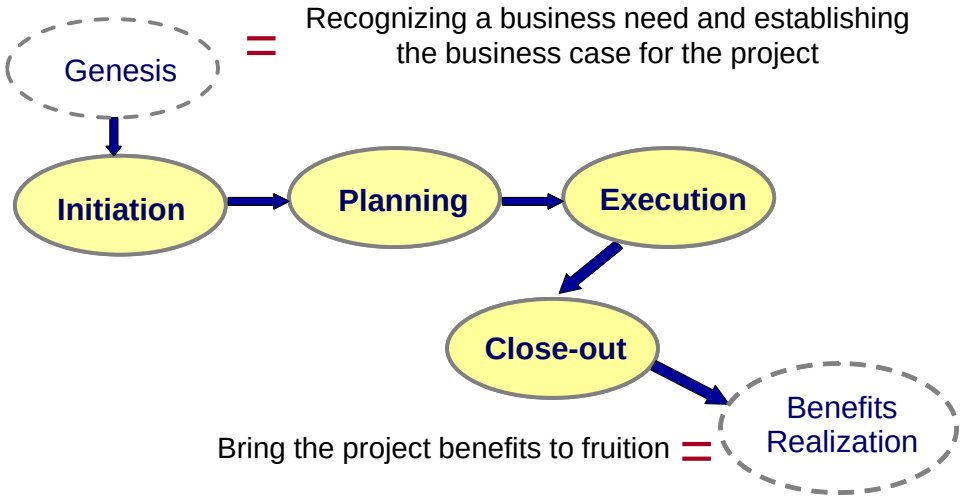

Figure 1 – The Complete Waterfall Project Life Cycle

While the waterfall life cycle model in Figure 1 is an improvement and can be used for most projects, it poses a problem when dealing with new product and large software development projects, where requirements are evolving, technology is rapidly changing and speed-to-market is critical. In 1970, Dr. Winston Royce, an American computer scientist at Lockheed Software Technology Center, presented a paper which criticized sequential development [4]. Royce stated there was inherent risk in a single-pass approach and suggested an iterative and incremental approach for large software development projects [5].

In 1995, the Standish Group published a report that detailed the alarming failure rate of software projects using traditional waterfall methods. According to the report [6]:

- 16% of the projects came in on time and on budget
- 31% of the projects would be cancelled, some just before or even after implementation
- 53% would run over 189% of their original budget

Many other studies were done over the years for both product and software development, with the same dismal results by projects using a waterfall life cycle approach. The times were ripe for change and project teams experimented with different iterative life cycle approaches. An iterative life cycle is also called incremental development, and the basic idea is to do "mini-projects", also called cycles, sprints, or iterations, lasting typically two or four weeks as shown in Figure 2. The goal for each iteration is to create some amount of useable product. With the iterative approach, requirements and scope evolve over the iterations.

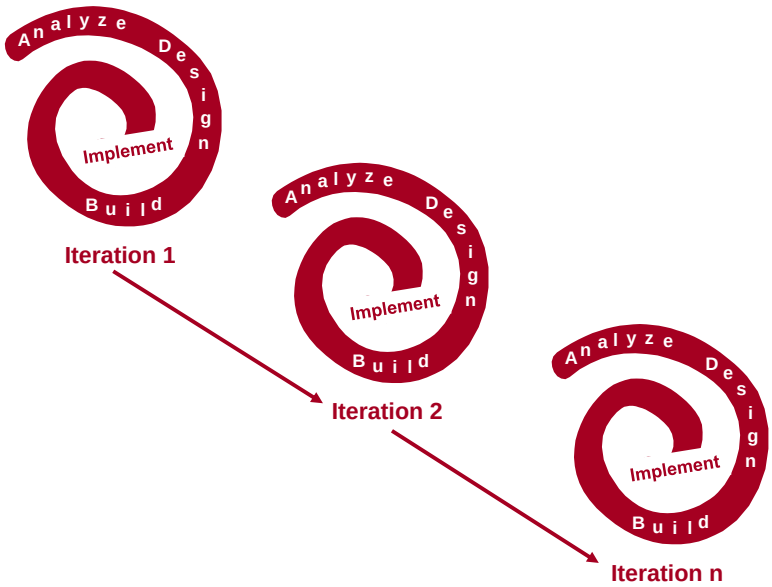

Figure 2 – Generic Iterative Life Cycle Model

In 2001, seventeen leading proponents of iterative life cycle approaches gathered at the Snowbird ski resort in Utah to discuss the future of software development. The attendees named themselves the Agile Alliance, and agile became the umbrella name for the various iterative methodologies [7, p.28].

This paper will review the Agile Manifesto and the guiding principles for agile use developed at the Snowbird ski resort in 2001. The differences between agile and waterfall life cycles will also be explained, along with recommendations on when to use each life cycle model. Planning and team organization on agile projects will be discussed, along with two of the more popular agile methodologies, which are Scrum and Kanban. Even though an agile life cycle model is not appropriate for construction projects, this paper will discuss how to successfully implement specific elements of agile that can add value.

The Agile Manifesto

One definition of agile is a "set of principles for software development in which requirements and solutions evolve through collaboration between self-organizing, cross functional teams [8]." Unfortunately, this definition focuses on software development only and in reality agile is frequently used for product development, research and business process change projects. Later in this paper, how agile can be used for portions of some constructions projects, and how agile tools can also be applied to construction projects will be explained.

A suggested more encompassing definition for agile is *iterative project methodologies where the requirements and solution evolve through close collaboration between the team and client*. The key points with agile are the iterative approach and the evolving requirements and scope. Waterfall uses a philosophy known as big design up front (BDUF), where requirements are completely defined, the project scope is then developed to meet the requirements, design is done, product deliverables are completed, testing is done and the product is then put into production [7, p. 24].

The Agile Alliance gathering in 2001 at the Snowbird ski resort in Utah resulted in the manifesto for agile software development, which includes the set of values shown in Figure 3 [9].

We are uncovering better ways of developing
software by doing it and helping other 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 items on the left more.

Figure 3 – Agile Software Development Values

It is difficult to argue that the items on the left have higher value. Interactions with individuals involved with the project will always be more important than processes and tools for project success. What good is comprehensive documentation if the product doesn't work? Collaboration is also always preferable than contract negotiation. Finally, if a change is needed due to business conditions or the discovery of new information, that is more important than following a plan that may not deliver the optimal business benefits. In reality, the manifesto values really applies to any project life cycle, so it does not differentiate the agile and waterfall life cycle models.

However, the manifesto also includes a set of principles for agile methodologies. The principles, also developed at the Agile Alliance gathering at the Snowbird ski resort, distinguishes agile from waterfall. The principles are listed below, along with comments on how each principle relates to the waterfall life cycle used for construction projects [9].

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software. **Comment:** *Finished product is delivered with each agile iteration while waterfall delivers finished product only at the end of the project. However, an incremental waterfall life cycle model will be discussed in the next section of this paper, and this model can also deliver finished product in iterations*.
2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage. **Comment:** *A late change in a waterfall project can be expensive and delay the project finish, so this differentiates agile and waterfall*.
3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale. **Comment:** *This seems to be redundant to principle #1*.
4. Business people and developers must work together daily throughout the project. **Comment:** *This applies to any project. Having the business people and project team work together will help ensure a more successful project. Agile does have a specific role called product owner that will be discussed in the Agile Project Team section of this paper*.
5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done. **Comment:** *This applies to any project.*
6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation. **Comment:** *This applies to any project. With any project team, face-to-face communication is always the most effective.*
7. Working software is the primary measure of progress. **Comment:** *Working product should be a primary measure of success for any project. Agile delivers working product with each iteration, so the client will see some working product sooner compared to a waterfall project.*
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely. **Comment:** *A sustainable work pace makes sense for any project team regardless of the life cycle model used. For agile, it means the team decides on how much work can be accomplished with each iteration, and the team only commits to that amount of work.*
9. Continuous attention to technical excellence and good design enhances agility. **Comment:** *This applies to any project.*
10. Simplicity--the art of maximizing the amount of work not done--is essential. **Comment:** *This applies to any project since non-value adding work should not be done on any project.*
11. The best architectures, requirements, and designs emerge from self-organizing teams. **Comment:** *Agile uses self-managing teams, so the role of the project manager does not exist. This is a difference with the agile life cycle model.*

12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly. **Comment:** *This applies to all projects, and it is called lessons learned, which can be done at regular intervals, not just at project completion.*

In summary, the Agile Alliance principles are really a list of best practices to achieve successful projects. Most of the twelve principles are just as pertinent for the waterfall life cycle model. What differentiates agile from waterfall is the delivery of some product with each iteration (principles 1 and 3), changing requirements (principle 2) and the use of self-organizing (and self-managing) teams (principle 10). The next section of this paper will explore the differences between the agile and waterfall life cycle models, along with recommendations on when to use each life cycle model.

Differences Between Agile and Waterfall Life Cycle

The waterfall life cycle model is plan-driven. Requirements are established and baselined early in the project, and a project plan is developed based on meeting the requirements. Each project phase is completed before the next phase begins, but on some projects there can be some phase overlap. A well-defined change control process is needed, and late changes can have a major impact on the project objectives. Waterfall also uses project teams with well-defined roles, and with a project manager in charge of the project. In summary, waterfall has been in use for a long time and is a disciplined approach for completing projects.

In contrast, agile methodologies are change-driven. Requirements and scope evolve over the life of the project, and increments of product are delivered in short time-frame boxes. Each iteration goes through a "mini-life cycle" of analyze, design, build and implement. There can be an iteration "0" where a high-level schedule, resource plan, risk register, communications plan and other project documents are prepared, but detailed plans are only developed for the next iteration. With agile, there is no change control process because changes are expected and welcomed over the life of the project. In addition, an agile team is self-managing and decides on the workload and plan for each iteration. The project manager role does not exist on agile teams. There is a team leader, often called a scrum master. However, the role is not leadership, but to provide process advice and to remove any impediments that may reduce team effectiveness, such as a client not providing timely feedback on system features.

Table 1 summarizes the differences between the waterfall and agile life cycle models.

Waterfall	Agile
Requirements established early in project	Requirements evolve over the project life
Waterfall phases are typically sequential	Mini-life cycles of 2-4 weeks typically used
Product delivered at the end of the project	Product delivered at the end of each iteration
Entire project is planned early	Limited early planned, detailed planning only done for the next iteration
Formally organized team lead by a project manager	Self-managed team with no project manager
Change management process used	No change management since changes are expected and welcomed

Table 1 – Summary of Waterfall and Agile Life Cycle Differences

The business case for agile is the faster delivery of useable product, resulting in an earlier return on the investment. With waterfall, product is not delivered until the end of the project. For a waterfall project with a one-year duration, there are project expenditures during the one-year period with no product income. With agile, product functionality increments are delivered every two or four weeks, at the end of each iteration. Product is released to the marketplace after multiple iterations, such as every three months for the one-year project. The result is that income is obtained by the early release of some product functionality to the marketplace.

An interesting point is that an incremental waterfall model, shown in Figure 4, can be used to closely mirror the advantages of faster delivery of useable product with the agile approach. The incremental waterfall model creates a series of mini-projects, with functionality delivered at the end of each mini-project. The example shown is using overlapping phases, which means mini-project #2 design starts once mini-project #1 design is finished and the build is being done.

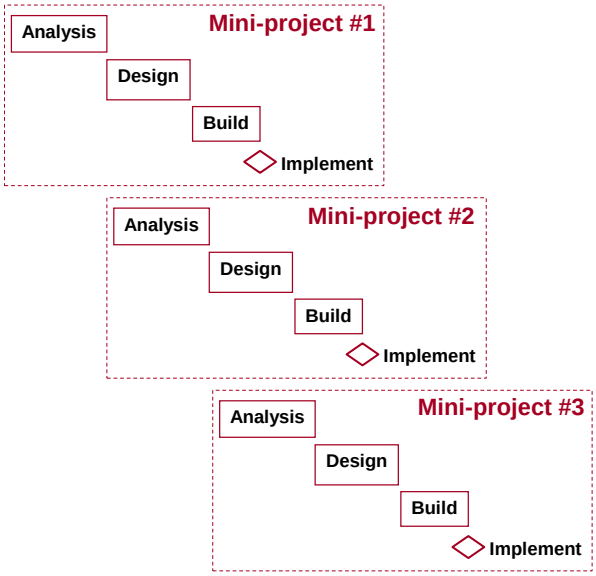

Figure 4 – Incremental Waterfall with Overlapping Phases

Waterfall	Agile
Well-established business processes	Exploring new business processes
Requirements can be defined early in the project	Requirements will evolve over the life of the project
Significant interfacing with legacy systems	Experimental approach is needed to find a solution
Client decisions involve numerous people	Client empowers a person to make project decisions
Team works best under a traditional team structure	Experienced team that can self-manage the project work

Table 2 – When to Use Waterfall or Agile

Table 2 provides guidelines on when to use a waterfall approach and when to use an agile approach. Agile works best for custom-built software and product development projects, since these types of projects deal with new business processes and evolving requirements. Construction projects should use a waterfall approach since complete requirements and design are needed before construction starts to avoid rework. However, there are cases when a modified approach is most effective such as:

- Environmental clean-up of an old industrial site can use an incremental waterfall approach. The first mini-project would be removal of the buildings on the site including asbestos removal. The second mini-project would be removal of the concrete building slabs and foundations including disposal of contaminated concrete and soil. The third mini-project would be remediation of contaminants in the groundwater.

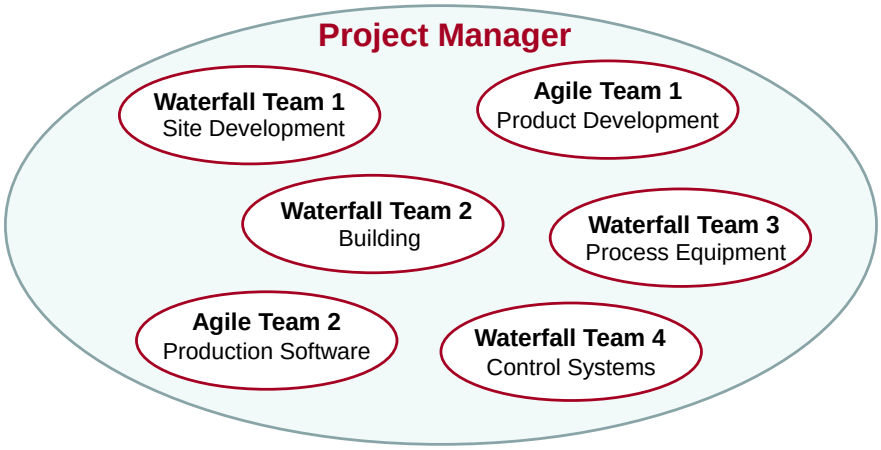

Figure 5 – Waterfall & Agile Blend for New Product Manufacturing

- Construction of a manufacturing facility for a new product can use both waterfall and agile as shown in Figure 5. The new product development is done using an agile team. While the development team is finalizing the product design, waterfall teams can start work on the site development and building. Once product design is done, a waterfall team can design, purchase and install the process equipment needed to make the new

product. An agile team is used for design and installation of the custom software needed to control production of the product. In this case, an iterative life cycle is more appropriate since requirements for new product control software are probably not well understood.

The Agile Project Team

The core agile project team is typically five up to twelve people. The three major roles are as follows:

1. Product Owner – this is the critical role on agile projects. The person must have the authority to make decisions on behalf of the client, and is expected to spend considerable time with the team to discuss the product vision and features, and answer questions. The product owner sets the product vision and roadmap, and manages the list of prioritized features and requirements.
2. Scrum Master – this person is the expert on the agile process, removes any impediments to the project, and creates an environment that fosters collaborative decision-making. The scrum master replaces the project manager role, and is expected to facilitate, coach and guide the agile project team.
3. Delivery Team – the team is self-managing and decides on how much work can be done for each iteration, using the prioritized requirements list. The team prepares the detailed plans for each iteration and gets the work done. Roles and responsibilities are not firmly established, and cross-functional work is typical. For example, a designer on the delivery team may have expertise in using scheduling software, and that person handles preparation of the schedule task list for each iteration.

The key for making the core agile team successful is having dedicated team members, spending all of their time on the project. There can be some part-time project personnel involved in the project, such as an information security officer who advises the team and reviews some work products, but part-time personnel are not part of the core agile team.

Some of the key characteristics of successful agile teams equally applies to waterfall teams. This includes having dedicated team members, co-location of team members, empowered and collaborative team members, and a good working relationship with the customer.

Planning on Agile Projects

One of the myths associated with agile projects is that no planning is done, and that's not the case. There are five levels of agile planning as shown in Figure 6, and the levels are as follows:

1. **Product Vision** – this is what a new product or the next product version should look like.

2. **Product Roadmap** – this is a high-level chronological depiction of how the product will be brought to market. This includes any construction projects needed to make, store and/or distribute the new product.
3. **Release Planning** – this is the agile planning document that most resembles the project plan. The release plan includes the preliminary list of features to be included with each release and a high-level schedule. The release plan can also include other project documents such as a risk register and communications plan.
4. **Iteration Planning** – this is done just before the start of each iteration. The team, working with the product owner, decides on the work that will be done during the next iteration, and the product functionality that will be delivered. The work that will be done during the iteration is planned at a detail level, including tasks, resources, hours and dates. Iteration planning is analogous to the two week look-ahead plan used on some construction projects.
5. **Daily Planning** – this is done each day during the iteration, and is also called the daily scrum. Daily planning meetings are used on some construction projects, such as a power plant shutdown for equipment and control upgrades. The core team assembles for a brief stand-up meeting to discuss three items:
    a. Work accomplished since yesterday's meeting
    b. Work that will be done by tomorrow's meeting
    c. Any obstacles and/or issues impeding progress

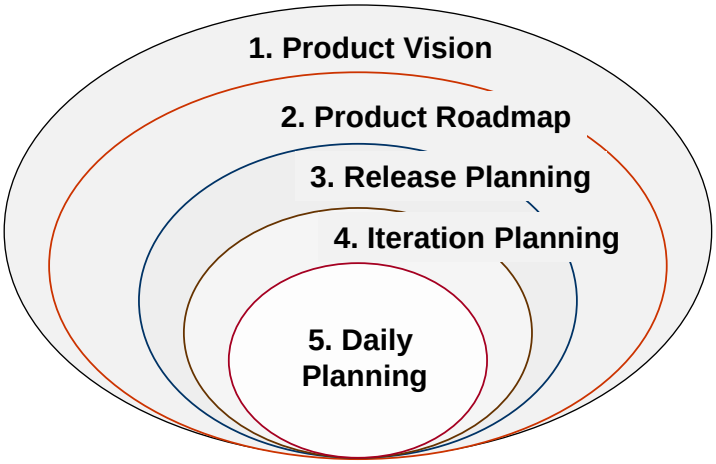

Figure 6 – The Five Levels of Agile Planning

In summary, planning on an agile project is not drastically different from a waterfall project. The product vision and product roadmap involve pre-project planning and are the same for agile or waterfall projects. The release plan is the agile planning document that somewhat resembles a project plan. The difference is that detailed scope, schedule and budgets are established on waterfall projects, while on agile projects scope, schedule and budget are done at a summary level. The agile iteration plan is very similar to the two week look-ahead plan used on some construction projects. Agile uses brief daily planning and status meetings, known as daily scrums, and construction projects can also use daily meetings.

Agile Requirements

As discussed earlier in this paper, with agile methodologies requirements and scope evolve over the life of the project. One of the key tools to uncover requirements on an agile project is the use of user stories. By definition, "a user story represents a small, concise statement of functionality or quality needed to deliver value to a specific stakeholder [10, p. 359]." The user story captures who (a user role), what (a necessary action, feature or quality) and why (the benefit or value received by the user). A typical format for writing user stories is shown in Figure 7.

Figure 7 – User Story Format

Examples of user stories are as follows:

- As a Facebook user, I want to update my profile so that potential friends may find me.
- As a telecommuting employee, I want to go seamlessly from my cell phone to my office phone while staying on the conference call so that I don't have to leave and then re-enter the conference.
- As a production supervisor, I want to have a weekly summary report on yield and quality by crew so that I can save time.
- As a control room operator, I want a mix tank low level alarm so that I know when to add more chemicals.

User stories can be helpful early in the engineering design phase of a project, since it helps focus the client on needs, which are project requirements. In fact, a common synonym for architectural programming, which is the research and decision-making process that identifies the scope of work to be designed, is "functional and operational requirements" [11]. Programming is the responsibility of the owner, but the requirements provided by the owner can vary from vague to very specific.

Owners sometimes focus on scope without first considering project requirements. An example of this occurred at a pharmaceutical company. The client representative, Richard, wanted to construct a new breakroom for production personnel, since the existing cafeteria was too far away. Richard identified 600 square feet (SF) of unused warehouse space that could be renovated for the breakroom. But Richard was providing the "how", the square footage. The architect and project manager moved the conversation back to the "what", which is how many production personnel will use the breakroom on average and peak. That's an important

requirement since states have building codes that restrict the number of people in a given area. For example, there is a sign in the entry area of restaurants indicating the maximum number of occupants. For Richard's project, over 1,000 SF was needed based on the planned number of production personnel that would be using the breakroom. If the breakroom had been built using the available 600 SF, the project would have failed once the local fire chief inspected the breakroom and saw the overcapacity situation.

On agile projects, a list of user stories is developed early in the project, and typically 60-80% of the user stories can be uncovered. The remaining user stories will evolve over the life of the project. The list of user stories becomes the product backlog, and the product owner has the responsibility to prioritize this list. The team, working with the product owner, uses the list of prioritized user stories to decide what work will be done during the next iteration and the product functionality that will be delivered.

User stories can be useful early in the engineering design phase of a construction project, when the owner has not done architectural programming. The difference is that user stories cannot evolve over the life of the project. All user stories need to be uncovered before design can be completed.

Scrum

Scrum is probably the most popular framework for implementing agile. With scrum, the product is build using a series of fixed-length iterations, called sprints, which are typically two or four weeks. Figure 8 shows the various meetings held during the two-week sprints.

	Monday	Tuesday	Wednesday	Thursday	Friday
Sprint 1	Sprint Planning	Daily Scrum	Daily Scrum	Daily Scrum	Daily Scrum
	Daily Scrum Story Time	Daily Scrum	Daily Scrum	Daily Scrum	Sprint Review Retrospective
Sprint 2	Sprint Planning	Daily Scrum	Daily Scrum	Daily Scrum	Daily Scrum
	Daily Scrum Story Time	Daily Scrum	Daily Scrum	Daily Scrum	Sprint Review Retrospective

Figure 8 – Sprint Meetings

Sprint Planning Meeting

The sprint planning meeting is held at the beginning of each sprint. At this meeting, the team, working with the product owner, decides on the work that will be done during the sprint and the product functionality that will be delivered. The meeting starts by the product owner reviewing the prioritized user stories and any other backlog items such as documentation or reports. The team decides on the work that can be accomplished during the iteration, and develops detailed plans including tasks, resources, hours and dates. For a two-week sprint, the planning meeting is less than four hours. Sprint planning is analogous to the two week look-ahead plan used on some construction projects.

Daily Scrum

The daily scrum should be held at the same time every day, and most teams hold the daily scrum first thing in the morning. The meeting should take less than 30 minutes, and an effective technique is not providing chairs since sitting can promote long discussions. Each team member reports on work accomplished since the last meeting, work that will be done by the next meeting, and any obstacles and/or issues impeding progress. The daily scrum should not be used to resolve issues. Any issues should be handled immediately after the meeting by the people who can solve it. The daily scrum is just as applicable on construction projects. The engineering design team and the construction team can use a short daily meeting to discuss accomplishments, plans and issues.

Story Time

In the middle of each sprint, time is reserved for a meeting called story time, which is used for grooming the user stories backlog. The purpose of this meeting, which is at least one hour, is to develop and/or update estimates on the amount of work for newly identified user stories. The team also identifies any user stories too large for a single iteration, so that the product owner can break the user story into multiple smaller user stories. Since story time is intended to 'groom' the list of evolving requirements, it is only applicable on construction projects when architectural programming is being done early in engineering phase.

Sprint Review

A brief review, typically one or two hours, is done at the end of each sprint with the client and other key stakeholders to show the work accomplished during the sprint. On agile projects, product is delivered at the end of each sprint, and the workable product is demonstrated and feedback obtained from the client. The feedback may result in changes needed for the demonstrated product, and the changes are added to the product backlog. For construction projects, client review meetings are equivalent to the sprint review. For example, a client review meeting may include review of completed engineering design drawings, or inspecting installed equipment and piping.

Retrospective

At the end of each sprint, the team holds a retrospective, which allows for learnings and insights to be captured. The goal for each retrospective is to identify one or two specific things to improve for future sprints, and to create an action plan to implement those changes. The retrospective typically ranges from one to two hours. The retrospective is essentially a lessons learned meeting. Lessons learned are commonly done at the end of any project, but can also be done during the project.

In summary, the scrum meetings are applicable to construction projects. Each sprint has a planning meeting which is analogous to the two week look-ahead plan used on some construction projects. The daily scrum used with agile can also be used on construction projects to discuss work accomplished, work that will be done by tomorrow, and any obstacles and/or issues impeding progress. Agile uses story time to 'groom' the list of evolving requirements, but it is applicable on construction projects when architectural programming is being done early in engineering phase. The sprint review is analogous to client review meetings for construction projects. The agile retrospective is essentially a lessons learned meeting.

Agile Estimation Techniques

The reason for creating estimates is to provide some measure of predictability for cost and schedule. Asking the person assigned to a task how many hours it will take has two problems. First, the person is probably not good at estimating; and second, the person will give you an answer anyway. This results in incorrect schedules and budgets.

Most people are really bad at estimating how long things will take, but people are good at relative sizing. For example, a person can pick the taller building when one building is two stories and the other building is five stories. However, as things get bigger our ability to perceive the differences in size decreases. Picking the taller building when one building is 92 stories tall and the other 94 stories tall would be very difficult.

The Fibonacci number sequence can be used for estimating project work since the sequence "increases at about the same rate at which humans are able to easily perceive differences [7, p. 124]." The Fibonacci sequence is comprise of integers where each number is equal to the sum of the previous two numbers: 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, and so on.

With agile, a two-step process is used to develop work estimates. The first step is to arrange all user stories by relative size, with the smallest on the left and largest on the right. The smallest user story is assigned a value of 1. The team then scans to the right looking for the user story that is about twice as much work, and that user story is assigned a value of 2. The team then scans again to the right looking for the user story that is about twice as much work and that user story is assigned a value of 3. The process continues until each user story has an assigned value. Typically, the Fibonacci sequence is used up to 34 or 55. Any user story larger than these

values is called an epic, too large to be effectively done within one iteration, and the epic is broken into multiple, smaller user stories. Once done, all user stories with a specific value are grouped under that number as shown in Figure 9. The numbers are called story points, or just points, and by definition are a relative unit of measure for the amount of work needed to complete a user story [7, p. 122].

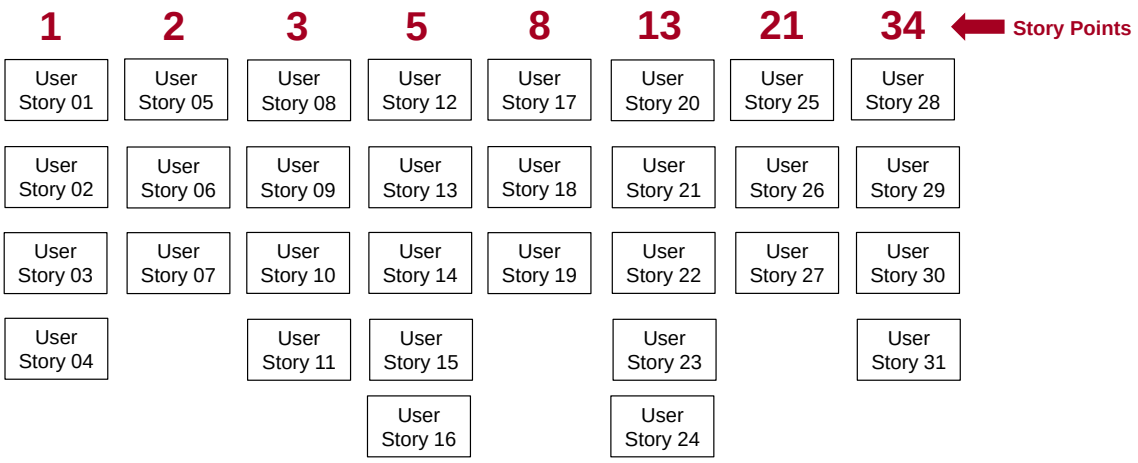

Figure 9 – Estimating Work with Fibonacci Numbers

The last step in agile estimating is to complete work for some user stories and measure how long they actually take. Using the list of user stories in Figure 9, let's assume the agile team completes user stories 1, 5, 6, 12, and 28 during the first two-week sprint, which means 44 story points were completed. For the second two-week sprint the number of story points completed is 48. By definition, the velocity is 46, which is simply the average number of story points completed per sprint. For the sample project shown in Figure 9, there is a total of 335 story points. With a velocity of 46 story points per sprint, the implication is that eight sprints are needed to complete the project (335/46). However, recall that on any agile project typically only 60-80% of the user stories are uncovered early in the project, and the remaining requirements evolve over the project life. Assuming only 60% of the requirements are defined for the project, the number of expected story points is 558 (335/0.6), and the number of sprints needed to complete the project is twelve (558/46).

A simple and logical extension to the agile estimating process is calculating hours. For the sample project from Figure 9, if the team consists of seven full-time team members, that equates to 280 hours per week, and 560 hours per two-week sprint. Therefore, each story point is equivalent to 12.2 hours of effort (560/46).

On construction projects the use of story points is not a useful concept. Estimating quantitative work such as installing 100 feet of 8" water main or constructing a retaining wall is easy using estimating database information. However, the use of a set of modified Fibonacci numbers, along with an agile estimating technique called planning poker [12, p. 56] can be very effective for estimating non-quantitative tasks such as design calculations and schedule preparation.

To use planning poker, each participant is given a set of cards with a modified Fibonacci sequence of 1, 2, 4, 8, 16, 24, 40, 64, 80 and 120 hours. The technique consists of the following steps:

1. The task owner provides a short overview of the task scope
2. The team asks questions and discusses to clarify the task scope
3. Each team member lays a card down with their estimate of hours
4. Team members with low or high estimates offer their reasoning and the team discusses
5. Steps 3 and 4 are repeated until consensus is reached

Typically, consensus is reached within three cycles. If the team believes a work item is greater than 120 hours the task should be broken down into smaller components.

Agile Information Radiators

One common characteristic of scrum teams is co-location. The scrum team work area will have walls covered with hand-drawn charts, and a task board of sticky notes indicating tasks not started, tasks in progress, and completed tasks. These low technology tools are known as information radiators [7, p. 102], and a radiator example is shown in Figure 10.

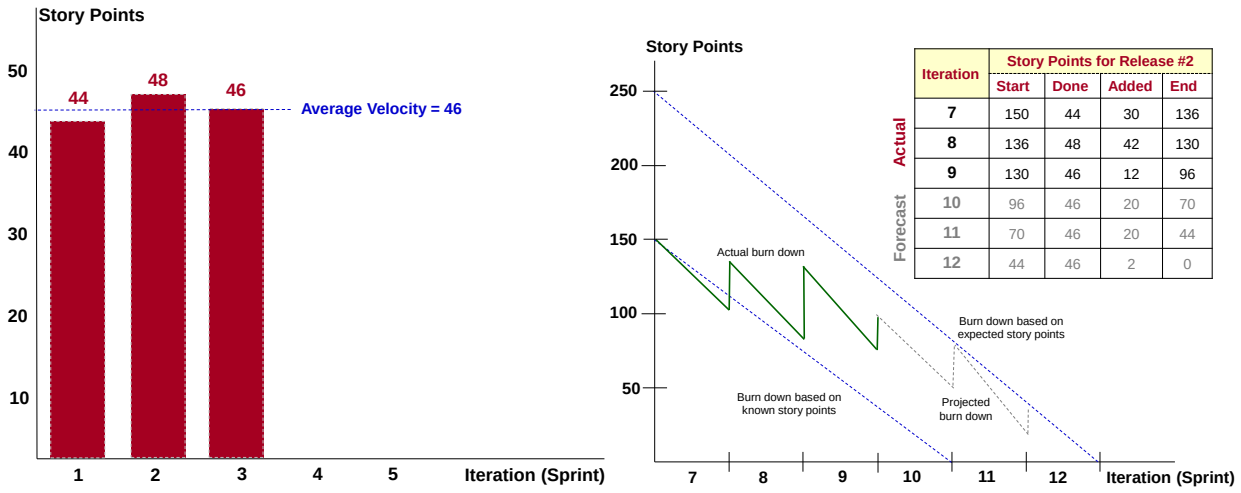

Figure 10 – Agile Information Radiators

Velocity Histogram

As discussed earlier in this paper, story points are relative measures of the effort and size of a task. A task worth two story points is twice the work compared to a task worth one story point. Using agile estimation techniques each task is assigned a story points value. By definition, the number of story points completed during a sprint is called the velocity. A histogram can be used

to plot the velocity for each sprint as shown in Figure 10, and the data from the first three iterations can be used to calculate an average velocity of 46 for the project.

Story points and iterations are concepts that cannot be used on construction projects, so the velocity histogram is not an applicable tool. But since the velocity histogram measures the amount of work completed, in units of story points, the comparable technique for construction projects is earned value management. Earned value is the percent of the total budget actually completed at a point in time [13, p. 2], so earned value also measures the amount of work completed. A histogram of earned value earned for each time period can be generated using monetary units for the vertical axis, and that is equivalent to the velocity histogram.

Burndown Chart

Books and websites are evenly divided on whether the term is burn down or burndown; this paper will use the single word version. A burndown chart displays the remaining work for a specific iteration. The remaining work is typically shown as story points, but as discussed earlier, story points can easily be converted into hours.

The burndown chart is also used to show the remaining work for a release, which consists of multiple iterations. Figure 10 shows the burndown chart for the second product release on a project. The chart shows actual values for iterations 7, 8 and 9; and forecast values for iterations 10, 11 and 12.

Release #2 consists of up to six iterations with 150 defined story points. Note the use of two baselines. The first baseline uses the known 150 story points, and the previously calculated average of 46 story points per iteration. Based on this information, the release can be completed in four iterations (150/46 = 3.3).

However, on agile projects requirements and scope evolve, so typically only 60-80% of the user stories are known at the start of a release. Assuming the worst case of 60% known requirements, the number of expected story points is 250 (150/0.6). Using the calculated velocity of 46, the number of iterations needed is six (250/46 = 5.4). Therefore, the second baseline shown is based on the expected story points and six iterations. Note with the sample release burndown chart the addition of story points before the start of the next iteration as new requirements and scope are defined.

The burndown chart is frequently used on constructions projects, but hours are used instead of story points for the vertical axis. The burndown chart can be used to show planned, actual and earned hours. The biggest difference with construction projects is that new hours are not added to the chart unless there is an approved scope change.

Kanban

Agile lifecycle models, such as scrum, are based on iterative and incremental development. The exception is kanban, which was developed by Toyota in the late 1940s. Kanban focuses on the continuous flow of work, and does not use fixed-length iterations like scrum. The key metric with scrum is velocity, which is the number of story points completed during the iteration, and this provides a measure of the amount of work done. The key metric with kanban is cycle time, the amount of time it takes for a unit of work to travel through the project process.

Kanban is a Japanese word and roughly translates to “card system”. The cards were used to create a “pull” system in the production line, rather than the common approach of letting material queue up for the next step in the process. For example, the team that attaches the doors to the car frame delivers a card, or "kanban", to the team that assembles the doors to signal they will soon need more doors. This signals the door assembly team to make more doors. Although the signaling technology has evolved over the years, this system is still at the core of "just in time" manufacturing [14].

Applied to projects, kanban matches the amount of work to the team’s capacity by limiting the work in progress (WIP) at each step in the process. Once the team completes a task, they start work on the next task from the prioritized task backlog. The product owner is able to re-prioritize the “to do” list as new user stories are developed without disrupting the team. Provided the most important user stories are at the top of the “to do” list, the agile team is assured of delivering maximum value to the business. The WIP limits highlight bottlenecks in the project process due to procedures, people and/or equipment problems. The plan-do-check-act (PDCA) cycle is then used to eliminate bottlenecks and improve the project process and cycle time.

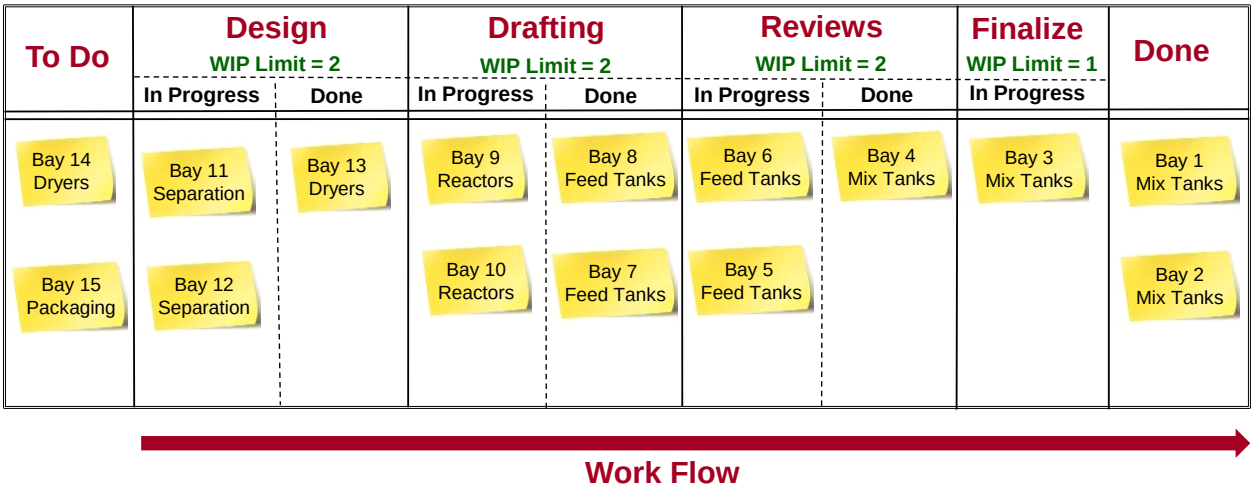

Figure 11 – Kanban Signaling Board for Engineering Project

A key tool with kanban is the task board, also known as the signaling board, shown in Figure 11. The signaling board shows work not started, work done and work in progress and done for each

step in the process. For example, a software development project could have process steps of design, built and test.

The signaling board is a tool that can be applied to construction projects. The example in Figure 11 is for the engineering design phase for a new chemicals manufacturing facility. The process steps are design (equipment and pipe sizing), drafting, reviews and finalize. With kanban, work in progress limits are placed on each process step in order to match the amount of work to the capacity of the team.

Conclusion

The waterfall life cycle model is plan-driven. Requirements are baselined early in the project, and a project plan is prepared to meet the requirements. In contrast, agile methodologies are change-driven. Requirements and scope evolve over the life of the project, and increments of product are delivered in short time-frame boxes.

Agile cannot be used for an entire construction project. Agile can be applied to part of a project such as developing production control software for a new manufacturing facility. In addition, an incremental waterfall model can be used to closely mirror the advantages of faster delivery of useable product with the agile approach.

Planning on agile and waterfall projects are similar. The agile release plan resembles a project plan. The difference is that scope, schedule and budgets are established at a detailed level on waterfall projects, while done at a summary level on agile projects.

Many of the agile meetings and tools are also utilized on construction projects, but with different names. There are agile concepts that will add value on construction projects, such as the use of Fibonacci numbers, along with a technique called planning poker, for estimating non-quantitative tasks such as design calculations and schedule preparation. Your projects organization can become more agile by implementing the applicable agile tools and techniques discussed in this paper!

References

1. Project Management Institute, *A Guide to the Project Management Body of Knowledge, Fifth Edition*, 2013
2. Stephenson, H., 2015, *Total Cost Management Framework, Second Edition,* AACE® International
3. Project Management Institute, *Skills Employers Need from Project Professionals,* December 17, 2014
4. Wikipedia website, *Winston W. Royce,* https://en.wikipedia.org/wiki/Winston_W._Royce
5. Agile Methodology website, W*here Did Agile Come From,* http://agilemethodology.org/

6. Course Hero website, *The Standish Group 1995 Chaos Report,* https://www.coursehero.com/file/8503917/chaos-report-1995
7. Sims, Chris & Johnson, Hillary Louise, 2011, *The Elements of Scrum,* First Edition, Dymaxicon
8. Wikipedia website, *Agile Software Development,* http://en.wikipedia.org/wiki/Agile_software_development
9. *Manifesto for Agile Software Development,* http://www.agilemanifesto.org/
10. *A Guide to the Business Analysis Body of Knowledge, Version 3.0*, International Institute of Business Analysis, Toronto, Ontario, Canada, 2015
11. Whole Building Design Guide (WBDG) website, *Architectural Programming,* http://www.wbdg.org/design/dd_archprogramming.php
12. Cohn, Mike, 2006, *Agile Estimating and Planning,* Twelfth Edition, Pearson Education, Inc.
13. Lukas, Joseph, 2008 AACE International Transactions, *Earned Value Analysis – Why it Doesn't Work*
14. Atlassian Website, *A Brief Introduction to Kanban,* https://www.atlassian.com/agile/kanbann

Joseph A. Lukas, PE CCP
PMCentersUSA
joe.lukas@pmcentersusa.com