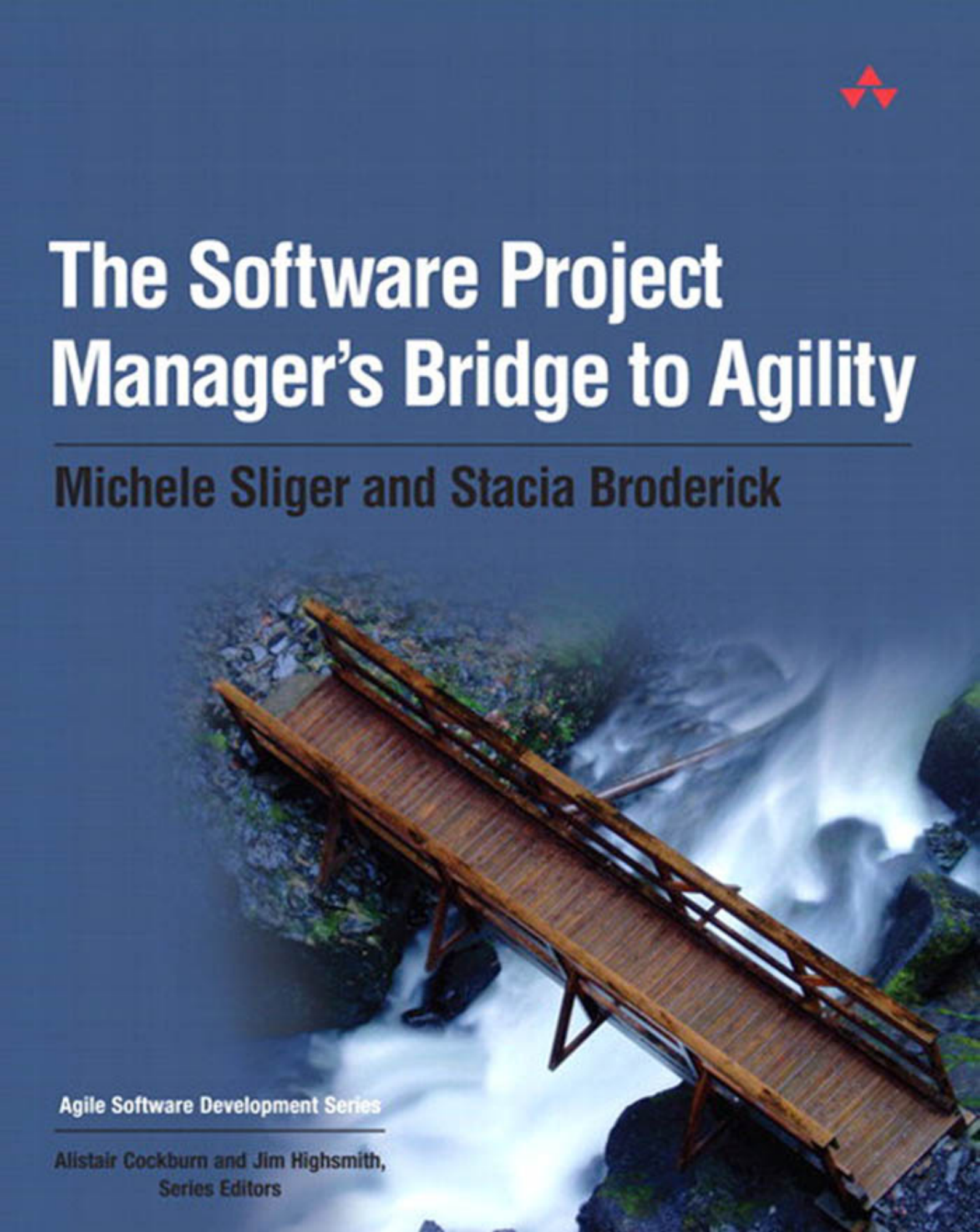

The Software Project Manager's Bridge to Agility

Michele Sliger and Stacia Broderick

Agile Software Development Series

Alistair Cockburn and Jim Highsmith,
Series Editors

The Software Project Manager's Bridge to Agility

features in the product backlog to be written as "user stories," which is a format for writing requirements from a user's perspective.[5] For example, rather than stating "the system shall…," a user story would say "as a <type of user> I want to be able to do <feature/function> so that <I get value>."[6] These features or user stories are then estimated in a unit of measure called "story points"; assigning a story point "estimate" to a user story says, "We think it's about this complex." All other user stories are then assigned story points in relation to the first one that's estimated. This relative complexity is a way of giving an indication of difficulty at a very high level. A sense of how many story points a team can complete in one iteration lends the ability to forecast the number of iterations a set of features will take to implement.

Regardless of your unit of measure, the team's iteration length, or the involvement level of the customer, the agile release plan has a fundamental, critical assumption: that the functionality delivered every iteration is of the highest quality possible. If the team is delivering code that's not tested and integrated every iteration, there remains an indeterminable amount of work due to the unknown product quality, which is risky. However, if the team frontloads as much testing work as possible during the iteration, so that the incremental deliveries are all of the highest quality, the team may only need a short hardening iteration at the end of the release in order to release the product to production. Additionally, it has mitigated the risk of the unknown by testing early and often.

While we're on the subject of different iteration types, many teams schedule an Iteration 0 (zero) at the beginning of the project, and an Iteration H (hardening) as the last iteration in a release (to production) cycle. Iteration 0 is used for project approvals, environment setup, ramp-up, discovery and initial overviews, and design discussions. Iteration H is used at the end to prepare for delivery, and it includes activities such as finalizing training and marketing materials and preparing the golden master or installation/download files. Iteration H is not used as a testing cycle; in fact, analysis, design, and testing should happen continuously throughout the iterations themselves, verified by the customer once he signs off on a story as being "done" for the iteration. Rather, Iterations 0 and H are used as minimal bookends for work that is not explicitly new development. Additionally, and very importantly, we firmly believe in asking our teams to deliver at least one feature in Iteration 0 because we've seen many teams revert to form, utilizing Iteration 0 as purely a design or analysis phase.

Figure 6-2 illustrates two release possibilities—internal and external—when thinking about release plans. When teams stay disciplined in delivering working product every iteration, the organization theoretically has the ability to package up and "release" what's been built. Whether or not the package is actually released to customers depends on a number of factors, but it could just as well be released internally to training or support services or externally to a partner for a limited user-acceptance testing cycle. Regardless of the use, sticking to quality increments of product will provide the organization the agility it desires for whatever need is most pressing.

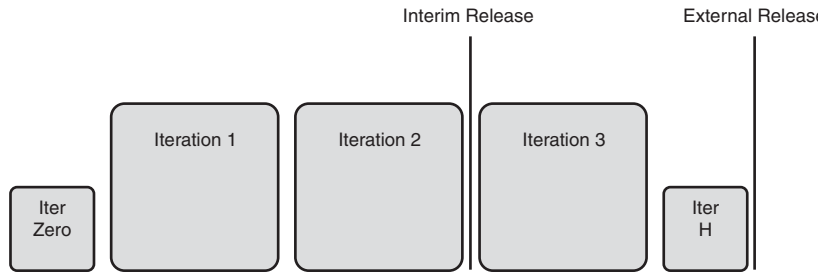

Figure 6-2
A release plan is a high-level plan for a series of iterations. Releases can be internal or external.

Table 6-1 provides a summary comparison of traditional and agile approaches to schedule development at the strategic level. Agile schedule development at the strategic level is referred to as "release planning."

Table 6-1
Schedule Development at the Strategic Level

Traditional	Agile
Create the final project schedule (usually in a tool that shows bar charts, milestones, and a network diagram) and baseline.	Facilitate the collaborative creation of the release plan, resulting in a quarterly high-level plan indicating release goals, features to be completed, and timeboxes.
Update the activity attributes, resources, project plan, etc., as appropriate.	Assist the team in updating the release plan as needed, based on iteration results.

The Release Plan: Schedule Control at the Strategic Level

Schedule control is carried out by knowing the current status of the project schedule, and managing changes in the plan that might impact the schedule. The agile release plan is managed in a similar fashion, except that the plan remains high level instead of containing detailed tasks from beginning to end; additionally, the agile project manager facilitates the customer and the team in making schedule changes, if necessary, based on project visibility. We will tell a story to illustrate this concept.

During a release planning meeting, our fictitious team, whom we'll call "GERT," analyzed its previous work and determined that it could reasonably commit to a pace of ten points per iteration, each iteration being four weeks in length. The product owner really needed the release six months later, and he was interested to know about how many features he would receive in the release. The team went through a collaborative exercise of placing user stories into iterations by using sticky notes and flip chart paper, "filling up" each iteration (or steel box) with no more than ten points' worth of stories in each. The GERT team stopped once it filled the final iteration with features, and realized that it had chosen a subset of the product backlog equal to roughly 60 points (10 points 6 iterations = 60 points). This set of features, called the "release backlog," can be used as a baseline by which project progress is measured. Remember, however, that we want to allow appropriate change as the customer provides product feedback. Thus, "managing" the project schedule in an agile setting is really about making progress visible so that the customer can balance the original product vision along with the necessary changes in scope that arise out of natural product evolution.

Back to our team. After the first iteration, the team only completed seven story points; velocity was lower than initially expected because the team ran into some unexpected environment setup issues. The project manager, team, and customer discussed the situation and decided to extend the release plan to seven total iterations, versus the six originally predicted, in order to account for the lower velocity. At this point, the customer could have just as well reduced the amount of scope in the release backlog, but he

decided that he had a little wiggle room in the overall release end date and could float another iteration. When discovering that actual velocity is higher or lower than expected, the product owner can decide if he wants to reprioritize user stories, drop user stories from the release backlog, or extend the release by *n* iterations.

The GERT team completed 12 story points of work in the second iteration. The customer was delighted! Even though the team and the customer were happy about the increase in velocity, they decided to keep the extended date just in case they needed the buffer. Figure 6-3 shows how the team began with a commitment to 60 points over six months using four-week iterations and baselined this. Then the burndown of what the team delivered each iteration was noted, resulting in the decision to commit to a new baseline.

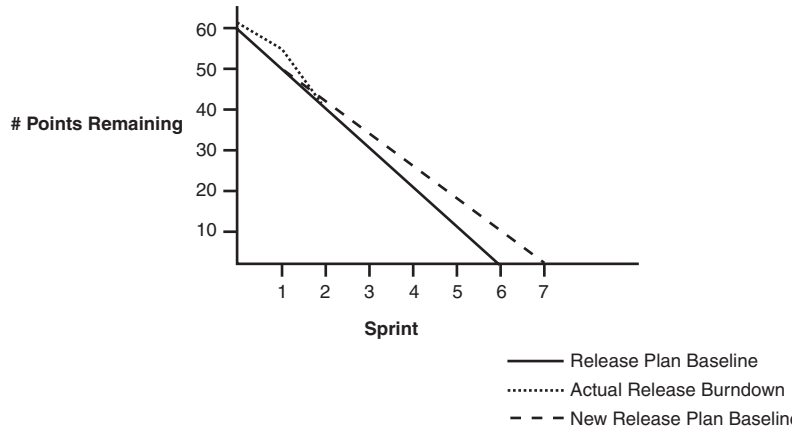

Figure 6-3
This release burndown chart represents the GERT team's progress through iterations 1 and 2 of its project.

Managing a number of iterations in a release plan is dependent on the real-time status of the working product, as well as on the collaboration and negotiation between the customer and delivery team. The agile project manager facilitates this discussion and makes certain that both sides have the information each needs in order to reach a collaborative decision about the release.

As new discoveries are made about the system, any changes that need to be made can be done by adding an item to the product backlog. In Agile, it is expected that these changes are necessary in order to build the right product. The product and release backlogs therefore function as a natural change management system.

Table 6-2 provides a summary comparison of schedule control at the strategic level for the traditional and agile approaches to project management. In agile terminology, schedule control at the strategic level is referred to as "agile release plan management."

Table 6-2
Schedule Control at the Strategic Level

Traditional	Agile
Update the schedule based on approved changes and re-baseline.	Facilitate the review meeting where the team updates the release plan based on the team's velocity and changes in the backlog.
Calculate schedule variance and schedule performance index.	Remind the team of its duty to update the work remaining in the iteration backlog; at the end of every iteration, the release burndown chart is updated. The team also measures its velocity.
Track and document requested changes.	The customer updates the product and/or release backlog.
Identify and analyze recommended corrective action to get the project back on track.	Facilitate a discussion between the customer and the team to discuss any corrective action that should be taken (i.e., add another iteration, add a team, drop features, terminate the project, etc.).

Iteration Planning: Developing the Schedule at the Tactical Level

Iteration planning brings us inside the steel box. It is the meeting in which the team breaks down the items from a prioritized product or release backlog into small tasks and assigns estimates and owners sufficiently so that the team can agree to the iteration plan. It is in this meeting that the activities (tasks) are defined, sequenced, and estimated by the team. Like the steel box, the iteration end date is fixed and padlocked; the team, based on its past performance (that is, velocity) will choose enough work to fill up the box. The iteration planning meeting can take up to eight hours for a month-long iteration, or up to four hours for a two-week iteration.

An iteration planning meeting includes the delivery team, the project manager, and the customer (or customer's representative, often referred to as the "product owner"). Others may attend the meeting if they wish to observe or contribute supplemental information about the features to be built.

The output of this meeting is the iteration backlog, which is a list of the team's tasks and other information necessary to manage its day-to-day tactics in turning product backlog items into working product increments for the iteration. A main difference between the release backlog and the iteration backlog is its unit of measure for the estimates: release backlog items—or features—are measured in story points or other high-level estimates, whereas iteration backlog items—or tasks—are estimated in hours.

Because the timeframe is so narrow—one to four weeks—teams become quite proficient at predicting the work within the iteration. Teams perform the traditional planning processes of activity definition, sequencing, and estimation in the planning meeting and throughout the duration of the iteration as needed, leveraging the daily stand-up meeting to gain valuable insights that may affect the tactics.

Activity Definition

The *PMBOK® Guide* classifies activity definition as "identifying and documenting the work that is planned to be performed." [7] In traditional project planning, a project manager would think about all the tasks that might ever happen during the course of the project as part of documenting the expected work to be performed. In fact, project managers often have at hand their methodology templates or project structures used previously that contain the work breakdown structure and subsequent tasks; the project manager simply inserts the new project name and makes a few adjustments. This reuse of a previous project plan is in line with traditional thinking, that new product development can be predicted and repeated. As we know, planning in an agile project should account for the change that will happen and involves teams working as a cohesive unit, task-sharing their way to iteration success.

Thus, activity definition happens in an agile project when the iteration has arrived in which the feature will be worked on; this breakdown occurs in