Managing Iterative Software Development Projects

KURT BITTNER
IAN SPENCE

OBJECT TECHNOLOGY SERIES
BOOCH
JACOBSON
RUMBAUGH
ADDISON-WESLEY
SERIES EDITORS

Managing Iterative Software Development Projects

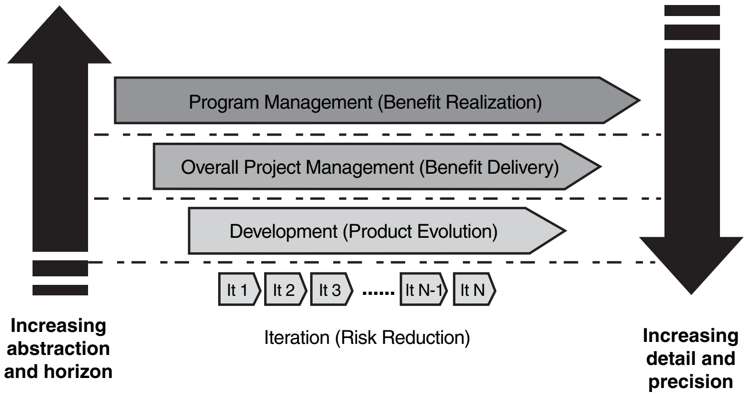
Program Management (Benefit Realization)
Overall Project Management (Benefit Delivery)
Development (Product Evolution)
It 1
It 2
It 3
......
It N-1
It N
Iteration (Risk Reduction)
Increasing abstraction and horizon
Increasing detail and precision

Figure 5-2 *The levels of abstraction and detail vary across the layers.*

The lower the layer, the shorter the time horizon and the more detailed, accurate, and precise the plans. A program plan may well stretch for a number of years, so it will be quite abstract and imprecise about long-term estimates. On the other hand, an iteration plan is focused on the next few (typically four to six) weeks and will be very precise about what exactly is to be done.

Traditionally, iterative project management guidance has focused solely on the lower two layers, and where it has attempted to address the upper layers, it has disregarded many of the important management and financial controls that are integral to controlling a business and that are at the heart of more traditional management approaches such as PMBOK[5] and PRINCE2.[6] It has also tended to give an unwarranted precedence to the development of new and innovative software solutions while disregarding the impact of business change and other organizational issues that often derail technically sound solutions.

Failure to discuss the impact of iterative development on the top two management layers has often made it difficult to sell iterative development to senior managers. Without support from senior management, iterative approaches can come into conflict with the programs and projects that they are supposed to support. For many senior managers the apparent lack

[5] The *Project Management Body of Knowledge (PMBOK),* developed by the Project Management Institute (http://www.pmi.org).

[6] PRINCE (**PR**ojects **IN** **C**ontrolled **E**nvironments) is a structured method for effective project management. See http://www.prince2.com.

of management control advocated by many "agile" approaches is a cause for concern. They usually view those approaches as being in opposition to the visibility and foresight they need to ensure that project budgets are well spent.

We include the overall project layer in our discussions because iterative development has major implications for the planning and management of this layer. The nature of the planning, monitoring, and control undertaken for an iterative project is fundamentally different from that undertaken for a traditional project. The feedback provided by iterating will have effects far beyond the development layer. For iterative development to be effective, the overall project managers must also adopt a progressive, adaptive approach to project management and planning.

This is also the layer that is most affected by the layering itself. Project managers steeped in traditional approaches often cling to detailed planning as the primary mechanism for risk reduction, even though detailed plans offer no capability to mitigate risk. The growing consensus is that risk is only mitigated through the demonstration of results. To reduce risk and achieve better results, the overall project managers must be prepared to delegate and allow detail to be pushed down through the development layer to the iteration plans, where agile approaches are adopted.

The program management layer is isolated from the iterations by the middle two management layers, making it much more stable and unaffected by the particular techniques used to develop the products required to realize business benefit from its investments. There is, however, a role for iterative project management techniques in the management of programs, particularly those focused on technological innovation and the development of software-intensive products. We return to this subject in Chapter 10, "A Scalable Approach to Managing Iterative Projects," when we consider scaling up iterative project management techniques for use on large projects and programs. For now we focus on the integrated planning and management of the three project management layers.

Positioning the Unified Process Lifecycle

In Chapter 3, "Controlling Iterative Projects," we saw how important the Unified Process lifecycle is as a framework for controlling iterative development. As shown in Figure 5-3, each application of the Unified Process is known as an *evolution* and produces a major release of the product.

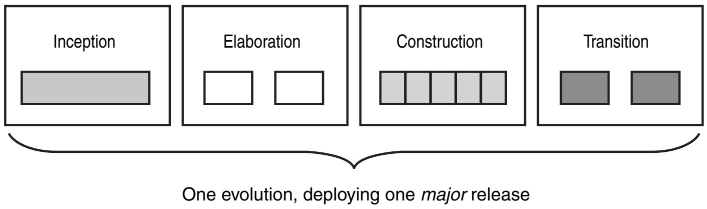

Figure 5-3 *An evolution is a single pass through the Unified Process lifecycle.*

The number and sizes of iterations shown here are purely illustrative and are not indicative of the relative lengths and numbers of iterations required by the phases, as we discuss in Chapter 7, "Evolution and Phase Planning."

Very few software development projects deliver a complete solution in a single release in the manner shown in Figure 5-3. Most projects deliver their solution as a series of major releases, each of which delivers benefit to their customers. These evolutions are *more or less* sequentially ordered[7] projects, each one building on the previous evolution as shown in Figure 5-4.

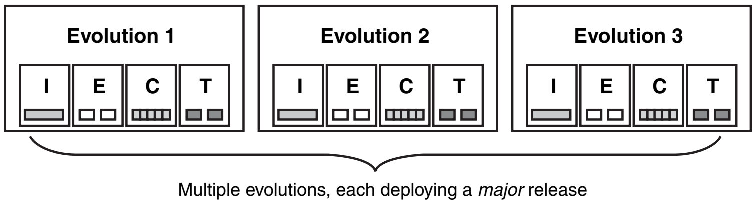

Figure 5-4 *A solution delivered through several evolutions*

Note that this is a Unified Process evolution for each major release (Release 1, 2, 3, and so on), not for each minor bug fix or point release. During the transition of Release 1, minor enhancements, fixes, and emergency releases may be required (Releases 1.1, 1.2, 1.3, and so on). These would be the result of undertaking transition iterations as part of the first evolution.

There are cases where only a single evolution is required, but these are rare. From an IT supplier's perspective, this can appear to occur in outsourcing environments where a customer gives the supplier one well-defined evolution to produce and then brings the other evolutions in house. From the customer perspective, this initial delivery is still the first of a series.

7 We say *more or less* because there is often some slight overlap between the completion of one evolution and the start of the next. For example, the Inception phase of the second evolution might overlap with the Transition phase for the first evolution.

The overall project layer is responsible for identifying and coordinating the development of an appropriate set of evolutions to meet its goals.

Layering the Plans and Milestones

Some people in the agile methods community argue that there is no need for planning beyond the current iteration, let alone the current evolution. As the quote from John Harvey-Jones at the beginning of this chapter observes, developing is more fun than planning, but planning is necessary to know where you are headed. The iterative approach enables the team to do something concrete within the current iteration, but without higher-level plans to illuminate the future, the current iteration effort could turn out to be misapplied and worthless in the end.

Given the complexity of coordinating multiple releases and possibly more than one development team, a mechanism is needed to simplify and focus the plans. We exploit the management layers to provide a simple set of plans; one for each layer, each focused on a different set of issues. This separation of concerns has many benefits, including the following:

- Reducing the management overheads by keeping the plans and control mechanisms simple and focused
- Keeping the detail focused on the short-term where it is required
- Providing plans with both the breadth and depth required to satisfy all the stakeholders in the project
- Enabling managers to manage and developers to develop

Figure 5-5 shows a simplified view of the planning elements at each of our project layers. The overall project is made up of development projects, each developing an evolution of the product to be produced. Each evolution is managed iteratively using the lifecycle of the Unified Process. Each Unified Process phase consists of iterations, which in turn are made up of planned and executed tasks.

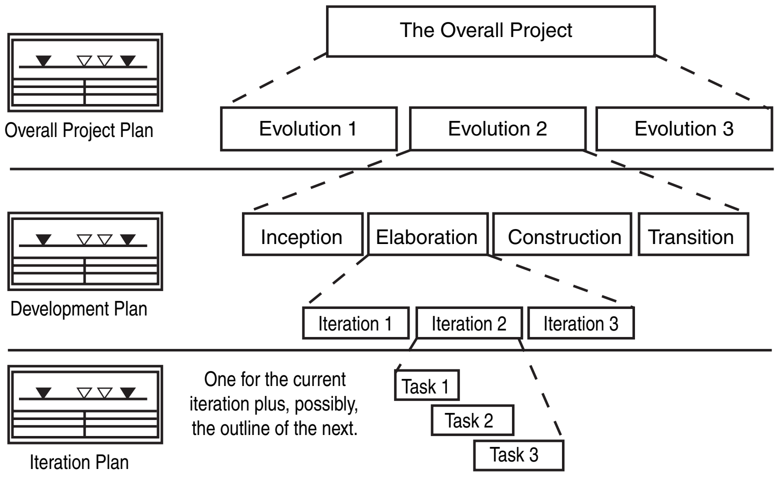

Figure 5-5 *Layering of project plans*

Although Figure 5-5 seems to suggest that there are a lot of plans, only three are active at any one time, one for each layer: (1) the overall project, (2) the current evolution, and (3) the current iteration.[8] The formality of these plans varies with the needs of the project, but in general each plan can be (and probably should be) no longer than a few pages.[9] We shall shortly see how this is possible, but first let's talk about each type of plan.

- **The Overall Project Plan**—For each project there is a single overall project plan. This overall plan is actually quite brief—it lays out the number of evolutions and releases needed to deliver the solution, including milestone dates, functionality delivered, risks to be addressed, and overall resource levels needed. The overall resources used and the business benefit delivered tie back to the Business Case developed for the

[8] In many ways this layering is akin to that which occurs when defining service contracts where plans need to be versioned and changes formally approved. Service contracts typically include a Master Agreement (akin to the Overall Project Plan), a number of Project Descriptions (akin to the Development Plans), and detailed Statements of Work (akin to the Iteration Plans) in order to restrict change control to the appropriate level. A similar separation of concerns is in action here.

[9] In this case we are purely talking about the plan itself, not the full set of project management documentation. Some organizations require a larger set of technical management plans (risk management, quality management, and so on) to be included in the planning document. Typically this makes the planning document longer than a few pages.

solution. The overall project plan acts as the roadmap for the project as a whole. Producing the overall project plan is the subject of the next chapter.

- **The Current Development Plan—**There is a single development plan for the current evolution that lays out the lifecycle milestones and goals for each phase, identifying the number and purpose of the iterations they contain. Where staffing levels vary across phases, the development plan lays out how many people are needed and what skills they need. As this evolution progresses and its results start to become apparent, the planning of the next evolution starts.[10] Producing the development plan for an evolution is the subject of Chapter 7.

- **The Current Iteration Plan—**There is a single iteration plan for the current iteration. This is a detailed plan describing what will be done during the iteration. The iteration plans tend to be the most susceptible to change because the iteration is where the real work happens, but because we do not plan every iteration at the start of the project, changes to the iteration plans do not ripple disruptively through the entire set of project plans. As this iteration progresses and its results start to become apparent, the planning of the next iteration starts. The production of iteration plans is the subject of Chapter 8, "Iteration Planning."

  Many iterative project managers create a very rough draft of the plan for the next iteration alongside the iteration plan for the current iteration. The plan for the next iteration can then evolve as the current iteration is executed and its results become apparent. In this case at the end of the current iteration, the draft plan can be modified to become the plan for the next iteration rather than having to generate it from

[10] Although it is tempting to collapse the development plans into the overall project plan, our experience is that this causes far more problems than the benefit of a slightly reduced number of documents warrants. It also makes the overall project plan more volatile because it needs to be changed every time the development plans are tuned.