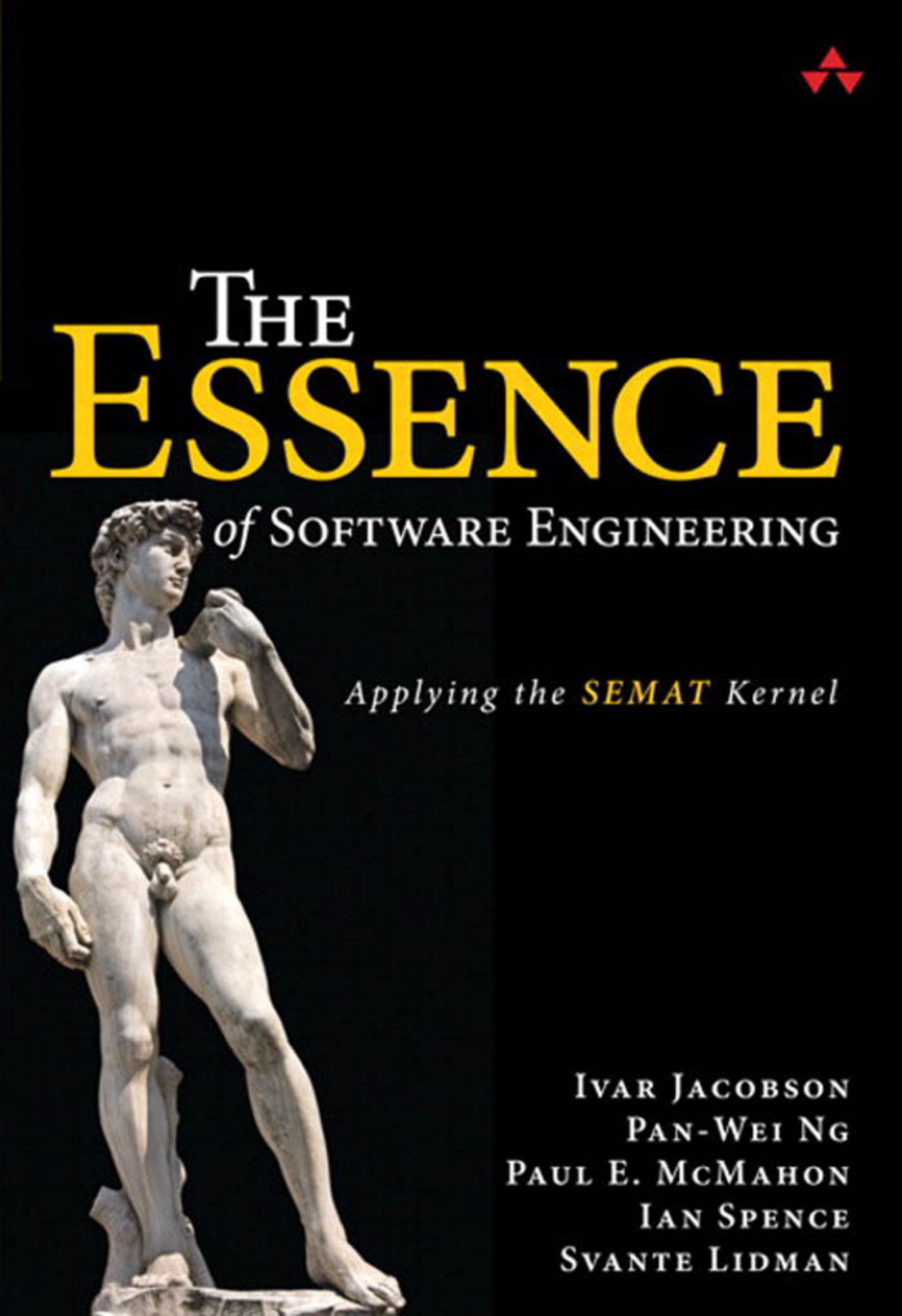
THE
ESSENCE
of SOFTWARE ENGINEERING
Applying the SEMAT Kernel
IVAR JACOBSON
PAN-WEI NG
PAUL E. MCMAHON
IAN SPENCE
SVANTE LIDMAN

The Essence of Software Engineering

This page intentionally left blank

7

Running Iterations with the Kernel: Plan-Do-Check-Adapt

The main character in our story is Smith. He had to develop a mobile application to browse a social network (e.g., Facebook, Google+) offline. The idea for this application came from Angela, Smith's customer representative from the marketing department. The application was to cache contents from the user's social network in the user's mobile device, thereby making it possible, as an example, to look at photos without a network connection.

We will look at how Smith and the team perform their development iteratively by walking through one of their iterations.

7.1 TERMINOLOGY USED

The subject of this book is how the kernel can help software professionals in everyday situations. To keep it short we have left out many things that are already well known regarding agile planning and execution. However, to be clear the following terminology is used throughout the book when we discuss iterative working.

- *Iteration:* An iteration is a timebox intended to work with a stable increment of a software system. The length of an iteration is typically two to four weeks and can involve all kinds of activities, from requirements elicitation to the deployment of the resultant software system. The software system, and the team's understanding of it, is typically grown over a number of iterations.
- *Iteration objective:* As well as having the objective to work with the next stable increment of the software system, you may want to define additional objectives to be achieved in the iteration. These can include getting some key risk under control, demonstrating completely new functionality, establishing a new way of working, and so on. The iteration objectives can often be tied to progressing one or more of the kernel alpha states.
- *Iteration backlog:* When the team agrees on what is to be done in the iteration, they add the objectives to the iteration backlog. The iteration backlog can then be refined with more detail by adding the tasks necessary to achieve the objectives.
- *Task:* A task is a portion of work that can be clearly identified, isolated, and then accepted by one or more team members for completion. Each iteration objective is broken down into one or more tasks for the team to manage in the iteration backlog.

7.2 PLAN-DO-CHECK-ADAPT

Developing your software iteratively is like taking a journey in your car. You need to know where you are, where you are heading, how much fuel you have, and how much farther you have to go before reaching your destination. You adapt to the road

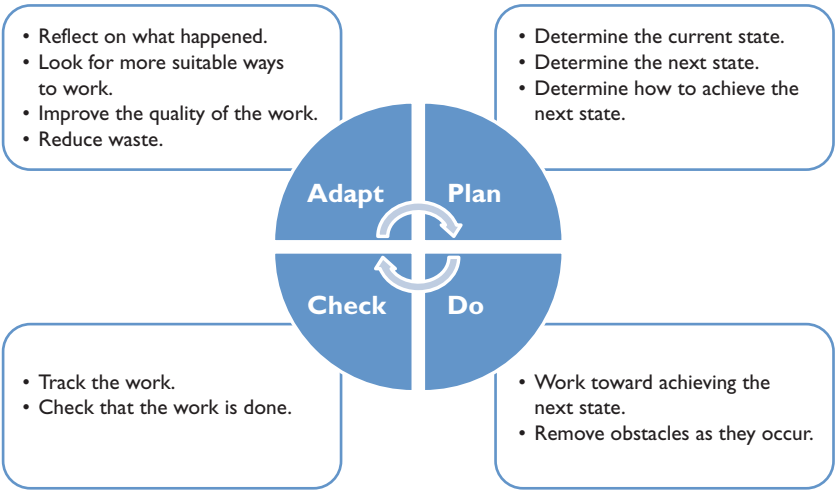

Figure 7-1 Plan-Do-Check-Adapt cycle

We have modified Deming's PDCA cycle by replacing Act with Adapt, as this is more descriptive of the intent.

conditions, traffic, and weather as you drive. You are continually planning, doing, checking, and adapting (see Figure 7-1). This is how Smith and his team ran their iterations.

- *Plan:* First, Smith's team determined the current state of the whole endeavor by determining the current state of each alpha. Then, based on this, they defined what alphas and states to progress in the coming iteration. They then planned how to achieve the target states by identifying tasks to be completed to achieve these states. This enabled them to tie in the detailed day-to-day work of the team with the progress of the endeavor as a whole. If the effort to complete the tasks exceeds that available in the iteration, then it will take more than a single iteration to achieve the objectives and the target states.
- *Do:* With each iteration, Smith's team worked on the identified tasks to progress the endeavor as a whole toward the

target states. This involved writing code, testing, setting up environments, discussing requirements, writing documentation, and so on.

- *Check:* Smith's team tracked the objectives and tasks to make sure they were completing what they had planned (or they replanned as required) and that they followed their agreed-on way of working.
- *Adapt:* Smith's team reviewed their way of working, identified impediments, and found better or more suitable ways of doing things. This often resulted in changes to their plans and their way of working.

7.3 SETTING THE SCENE

The company had very little in the way of formal processes. It relied on having skilled and creative individuals. This had worked well in the past for experienced teams. But the company was growing and there were many new hires. The new hires, mostly fresh out of university, had good technical skills—for example, in programming languages—but were less equipped in other aspects of software development—for example, working with stakeholders to gain agreement on requirements.

When starting out, Smith's team had only two developers, Tom and himself. Both were familiar with the kernel. They were later joined by two other developers, Dick and Harriet, who were new to the job and had no previous knowledge of the kernel.

Success to Smith meant more than functionality, schedule, and quality. It also meant growing his new team members' competence in the different areas of the kernel.

At the outset the team members had different perspectives on their responsibilities, which we illustrate with the help of the kernel (see Figure 7-2).

	Angela	Dave	Smith	Tom	Dick and Harriet
Opportunity	✔	✔	✔		
Stakeholder	✔	✔			
Requirements	✔	✔	✔	✔	✔
Software System	✔	✔	✔	✔	✔
Work		✔	✔	✔	
Team			✔	✔	
Way of Working			✔	✔	✔

Figure 7-2 Views of the kernel of different participants

- *Angela, the customer representative:* Her main responsibility was to ensure the software system addressed the opportunity through continuous involvement of stakeholders (her department colleagues, and department head).
- *Dave, the department manager:* His primary concern was to make sure something of value was delivered on time and within budget.
- *Smith, the senior member of the development team:* He was the only developer with full knowledge of the company, its products, and its way of working.
- *Tom, an experienced developer Smith had worked with before:* He knew the kernel but had little knowledge of the best way to identify opportunities or work with the stakeholders.
- *Dick and Harriet, new developers:* They loved to code and to write good software. However, they were not up to speed on the company's way of working.