Book Review

Managing the Design Factory: A Product Developer's Toolkit
by Donald Reinertsen

The Free Press, 1997
ISBN: 0-684-83991-1
Cover Price: US$35
256 pages

This book provides all the business justifications for using iterative development for any project (not just software) versus other process approaches. It does so in a fair, convincing, and sweeping way, by examining the costs and benefits of:

- Hierarchical management structures versus matrix or project team structures
- Queues and different batch sizes
- Acquiring information in various ways, including feedback gathering within organizations
- Product architecture
- Tools for managing information
- Metrics
- Risk management

Reinertsen describes, in all the dimensions I can imagine, how to organize a company to avoid unnecessary costs and maximize profits, whether you are in a low-margin manufacturing environment or a high-margin, high-risk, R&D environment.

The High Cost of Delays

The book begins by modeling a company's cost and schedule inputs, and the consequent profit output. From this, Reinertsen constructs a model that allows a business to compare the impact of schedule changes and requirements changes on development and unit costs. For most software development activities, schedule changes have the greatest impact on profit, and this provides the justification for investing in tools. Reinertsen
goes on to show that the noniterative processes and business architectures most companies use saddle their development organizations with inevitable schedule delays, because these processes and architectures are optimized for efficiency, not rapid time-to-market. The more efficient the business, the longer the schedule delay. I'll say more about this below.

Another major source of delay -- and high costs -- is a failure to gather and use information in a timely way. "A small change in probability or risk produces a small change in information," explains Reinertsen, so if you tackle only things you are almost certain of, you will not learn much. It is more profitable to tackle things about which you are uncertain first and get the information early. Using the information to correct risky situations and gain more certainty early in a project can help avoid expensive delays that might otherwise occur later on. It can help you plan a realistic length for your iterations as well. There is always tension between trying to control costs by having long iterations, and gaining opportunity to leverage new learning, which favors short iterations. This tension typically results in a compromise on iteration length, which you can calculate.

Reinertsen also looks at the potentially high cost of using feedback for organizational planning. Without delving into Control Theory, Reinertsen shows that feedback loops sometimes mask problems that it would be better to fix at the source rather than moderate at the output.

**Organization and Process**

In his examination of typical organizational paradigms, Reinertsen shows that, although functional hierarchies are the most efficient structure for reducing short-term costs, organizing teams by project is the most efficient approach for reducing elapsed time -- that is, calendar time from project start to completion. Functional hierarchies have queues to keep all the workers busy all the time. This is efficient, but invariably results in delays; people tend not to shift their attention from the task they are working on, even if it is not as critical to the project as other tasks in their work queue. In contrast, project teams tend to focus on the next thing needed to advance the project goals, even if this means workers are sometimes idle or not working in their primary skill area.

To further illustrate this point, Reinertsen provides the anecdote of the proverbial manager who shows up to accelerate a slipping schedule, grumbling that things only get done "while he's around." Well, of course, the manager's pet project gets urgent attention on all the queues in the hierarchical organization. And nobody stops to count the cost for all the other projects, which fall even farther behind.

Reinertsen also examines how to optimize business processes for both one-shot efforts such as R&D and repetitive endeavors, such as a production line. As he discusses patterns for processes, optimal increment size, overlapping increments, sequential versus concurrent processes, and process evolution -- all topics deeply pertinent to the Rational Unified Process,® or RUP® -- he keeps an eye toward maximizing profit.
Architectures are not free. In fact, they cost a lot of money. So do modularity and the other optimal qualities (maintainability, portability, etc.) for which we strive. By understanding how architecture affects costs, positively and negatively, we can determine an appropriate level of architectural investment. Similarly, if we understand the cost and value of reuse, which Reinertsen presents as an architectural problem, we can know how much to invest in it. The bottom line message in this section? Don't spend money on stuff you don't need.

It can be a big waste of money, for example, to create a very formal, detailed requirement specification at the beginning of a project, because requirements inevitably change as the project progresses. And there's another, even greater, business risk: the market inevitably will have changed by the time you finish developing your products. So in a dynamic market it is cheaper, explains Reinertsen, to commence development work and evolve requirements as understanding of the project improves and market conditions change. He also shows the value of prioritizing work on features according to their relative importance.

The final sections of this book show how to determine the value of tools for managing information. When calculating the ROI (return on investment) of tools, most middle managers produce an estimate based on engineering hours saved. This is appropriate for a commodity producer, for whom profit is strictly a function of revenues less costs. For high-tech industries, however, profit is often a function of time-to-market, and a two-month acceleration in schedule might double or quadruple profits. Reinertsen, then, has brought us full circle in considering the company as a profit engine, and making the right choices to maximize its performance.

Reinertsen's thinking is closely aligned with the underlying assumptions in RUP. This is not primarily a software book, yet Reinertsen measures all of these assumptions against the profit motive -- and it's gratifying to see that RUP passes with flying colors.

-Jeremy Haddock
Technical Lead, Strategic Services Organization
Rational Software

For more information on the products or services discussed in this article, please click here and follow the instructions provided. Thank you!