Principles of Product Development Flow
Second Generation Lean Product Development

Donald G. Reinertsen; Celeritas Publishing, 2009, 294 pages
ISBN 978-1-935401-00-1

Book report
Jim McDonough, October 26, 2012

Review

This book is a comprehensive analysis of the concepts and principles associated with developing products. Throughout the book the author offers enumerated Principles of Flow – 175 in total – summarized and indexed in an appendix for easy reference, with each principle thoroughly explained in the text. A recurring theme is the distinction between product development and manufacturing, particularly to the extent that some techniques which work well in Lean manufacturing settings are not transferable to product development. It is full of metaphors, anecdotes and everyday examples driving home the important points.

The enumerated Principles of Flow are comparable to axioms in the field of mathematics or to design patterns in the field of software development. In each case the axiom, design pattern or Principle of Flow has a name which makes it easy to convey an entire concept with a single word or phrase when used between practitioners who understand these names. Just as the transitive axiom[1] is universally understood amongst mathematicians and the decorator pattern[2] is recognized amongst software designers, so too is the D18: Frequency Response Principle[3] recognized and understood by those familiar with the Reinertsen's principles of product development flow.

Though the first few pages contain single-paragraph accolades from reviewers of the draft, there is no introductory Foreword or Preface, but begins directly with Chapter 1, in which the eight major themes are presented, setting the stage for the subsequent eight chapters. Each subsequent chapter begins with a section of introductory paragraphs on the major theme, followed by sections identifying and explaining the associated principles, and ends with a Conclusion section where the major points of the chapter are summarized.

1 – The Principles of Flow

In Chapter 1, Reinertsen immediately launches into a scathing indictment on the prevailing management practices used in product development:

> "I believe that the dominant paradigm for managing product development is fundamentally wrong. Not just a little wrong, but wrong to its very core. It is as wrong as we were in manufacturing, before the Japanese unlocked the secret of lean manufacturing. I believe that a new paradigm is emerging, one that challenges the current orthodoxy of product development,. I want to help accelerate the adoption of this new approach. I believe I can do this by helping people understand it. That is the purpose of this book." (p. 1)

A stark contrast is drawn between the official procedure product developers are expected to follow and the actual procedure they use:

---

1 Transitive axiom stipulates: If a = b and b = c, then a = c; see http://www.quickmind.net/mathpathways/algebra1/content/algebra1/topics/axiomequ.htm
2 See http://en.wikipedia.org/wiki/Decorator_pattern
3 D18: Frequency Response Principle: We can't respond faster than our frequency response. (p. 261)

> "Officially, many product developers have phase-gate processes. Work is divided into mutually exclusive phases separated by gates. One phase must be complete before the next one can begin. …
>
> What really happens is quite different. When I survey managers in my product development courses, 95 percent will admit that they begin design before they know all requirements. In fact, the average product developer begins design when 50 percent of requirements are known. They simply do not publicize this to management. Instead, they engage in a time-honored ritual of asking for permission to proceed. There is a tacit 'don't ask, don't tell' policy. Managers politely avoid asking if the next phase's activities have already begun, and developers discreetly avoid mentioning that they have already moved on. In practice, sensible behavior prevails, despite the presence of a dysfunctional formal procedure." (pp. 1-2)

This refreshing candor continues throughout the book. The problems with the current orthodoxy are listed (p. 4):

1. Failure to Correctly Quantify Economics
2. Blindness to Queues
3. Worship of Efficiency
4. Hostility to Variability
5. Worship of Conformance
6. Institutionalization of Large Batch Sizes
7. Underutilization of Cadence
8. Managing Timeliness instead of Queues
9. Absence of WIP Constraints
10. Inflexibility
11. Noneconomic Flow Control
12. Centralized Control

Reinertsen proceeds to provide details for each problem, highlighting the importance of having a foundation based on economics, and suggests the lack of correctly quantifying economics is the root of many problems experienced with current product development processes.

2 – The Economic View

This chapter explains Reinertsen's 21 Economic Principles. Concepts of economics are explored in a style easy to comprehend by the novice, presenting an economic framework in which to perceive a project. He presents a convincing argument for having a common denominator for making business decisions, enabling organizations to make apples-to-apples comparisons on the risks and benefits of various options. His suggestion: profit. Decisions on what to do and what to change become easier and corresponding action plans have a better chance of success once the value of options are converted into the same currency of "profit".

Reinertsen proposes using "cost of delay" as one of the fundamental measurements for making informed decisions on improving the process of product development, arguing that it has the added benefit of accelerating the ability of management to make decisions.

> "In practice, since 85 percent of companies do not quantify cost of delay, it is worth highlighting this particular sensitivity. We simply have no business trading money for cycle time if we do not know the economic value of cycle time." (p. 31)

A diagram portraying the relationship between cost and batch size is used to show that optimal batch size is represented as a U-curve determined by comparing "holding cost" against "transaction cost", and that merely approximating the optimal batch size is acceptable as a substitute for exact precision since slight

moves in either direction away from this point do not make significant differences due to the flat bottom of the U-curve on which it is based.

3 – Managing Queues

This chapter explains Reinertsen's 16 Queueing Principles. He covers some queuing theory, the effects and behavior of queues and offers reasons why queues matter, pointing out the difficulty product developers often have in perceiving that queues even exist, suggesting that unfounded assumptions about higher levels of capacity utilization usually adversely affect cycle times and profit.

Much of this chapter deals with the economic damage wrought by having large queues of work waiting to be processed, with examples of how the feedback delay resulting from queues enables bad design assumptions to proliferate until the feedback about the design assumptions finally reaches the designer.

> "Queues are the hidden source of most development waste. They are concealed from view in our financial systems. Because they are composed of information, not physical objects, they are hard to spot in the workplace. Queues increase cycle time, expenses, variability, and risk. They slow feedback, reduce quality and decrease motivation. Reducing queues is the key to improving product development performance." (pp. 82-83)

To aid in understanding the concepts of queue dynamics, some everyday examples of queues are presented using airport check-in lines and freeway traffic patterns.

4 – Exploiting Variability

This chapter explains Reinertsen's 16 Variability Principles. Variability is one area where comparison with manufacturing would lead product developers to wrong conclusions; the author asserts those successes in reducing variability in manufacturing processes do not necessarily translate into similar successes in product development processes. In product development, variability often presents opportunities to be exploited.

Reinertsen recognizes there are some aspects of variability in product development which present risk and should be reduced. He offers repetition as one technique to achieve this, citing the example of daily build cycles used in the software development industry, which, due to frequent repeated use, is a good candidate for process optimization to reduce its variability.

Another technique is to move the variability to its lowest-cost process stage. The author explains this using a simple example of how variability is managed in the air transportation industry:

> "For example, in air traffic control systems, the landing rate at a busy airport is typically a variable bottleneck. To ensure we load this bottleneck effectively,we have a small inventory of planes in a holding pattern waiting to land. However, it is very wasteful to fly planes around in circles. We can hold additional inventory en route to the destination by slowing them in flight. This ensures they spend very little time circling.
>
> "However, holding inventory in transit is still not the cheapest place to hold inventory. We hold the bulk of the inventory on the ground. It is safer and cheaper to have planes wait before takeoff than in any other place in the system. By doing so, we have coupled the downstream variation of the landing process into cheaper upstream stages. Instead of letting planes circle in a holding pattern for varying amounts of time, we let them wait varying amounts of time to take off. This drives variance into the cheapest stage of the process." (p. 107)

Diagrams of payoff functions reinforce the concepts embodied in many of the principles presented in this chapter.

5 – Reducing Batch Size

This chapter explains Reinertsen's 22 Batch Size Principles. The science behind batch size is covered as well as ways to manage it. The author states many product developers are oblivious to the impact batch size has on their work and proceeds to present the benefits of reducing batch size. In explaining how batch size reduces risk, a very good example is used showing how modern technology benefits from small batch sizes:

> "It is not an accident that a packet switching network like the Internet breaks its messages into many small packets. Consider what would happen if we sent a 10 megabyte file as a single large packet and we had one error per million bytes. We would average 10 errors in our transmission. We would have one chance in 22,026 that the file would reach the destination without an error. In contrast, if we break this large file into 10,000 packets of 1,000 bytes apiece there is only 1 chance in 1,000 that an individual packet will become corrupted. We will have to resend an average of 10 of the 10,000 packets , an overhead of 0.1 percent.
>
> "Put another way, if we didn't break messages into smaller packets, our overhead would be 22 million times higher. We achieve such high reliability by using small packets that the Internet can actually forward packets between nodes without error checking at the network level." (p. 114)

Included is a section on some of the ways to design a product development process to facilitate small batches. Some of the challenges with large batches are highlighted and Reinertsen implies that consideration of batch size in product development is still in its infancy.

Here again we see the diagram first presented in chapter 2 showing the relationship between cost and batch size, now expanded to show how changes in holding costs or transaction costs will affect the optimum batch size.

6 – Applying WIP Constraints

This chapter explains Reinertsen's 23 WIP Constraint Principles. Examples of work-in-process (WIP) constraints are given along with the economic reasons for controlling WIP. He describes the effects and benefits of applying constraints on WIP and councils on how to prepare for and respond to emerging queues, noting that a problem with a queue often requires more time to resolve than it took to occur.

Using an analogy based on naval warfare, Reinertsen points out that virtually every company has a few workers who are uniquely suited to resolving difficult problems, but their ability to respond quickly to such problems is compromised when they become loaded with work to the point of full utilization, a temptation to which many in management succumb precisely because these workers are so good at what they do.

He speculates on how far we have to go for this notion to become commonplace in product development:

> "Today, less than 1 percent of product development processes are managed with WIP constraints. Yet, once we recognize that queue size determines cycle time, we can easily use WIP constraints to control cycle time. This requires a basic shift in mind-set, a mind-set change that is very difficult for companies that believe a full pipeline is a good thing." (p. 166)

7 – Controlling Flow Under Uncertainty

This chapter explains Reinertsen's 30 Flow Control Principles. Concepts such as Congestion, Cadence, Synchronization and Sequencing Work are introduced and examined for how they contribute to flow. He

suggests that knowing the size of a queue is not as important as knowing how long it will take for work to move through the queue.

Reinertsen uses the example of highway traffic flow to help explain the concept of Congestion, showing the relationship between speed and density as a parabola, and that optimal flow occurs at a point where the trajectories for these two characteristics intersect. One particularly interesting phenomenon is presented called Congestion Collapse, an abrupt drop in throughput at high levels of utilization, known to daily commuters as gridlock:

> "When the internet becomes congested, it takes longer to deliver packets. Senders wait a certain amount of time to receive an acknowledgment. If they do not receive an acknowledgment within this time period, they assume the packet was lost and resend it. At heavy loads, the number of resent packets increases, but this increases loads even more. The extra load causes router buffers to overflow, causing even more senders to lose packets. As ever increasing numbers of senders transmit extra packets, loads rise even higher. Eventually, the network is 100 percent utilized with zero throughput." (p. 173)

The concept of Cadence is explained using a simple yet engaging example of a large city bus system, where it is shown how lack of a cadence leads to sub-optimization of the system, resulting in unpredictable waiting times at bus stops.

In the section on Synchronization it is demonstrated how synchronous processing yields savings in both time and cost over unsynchronized processing and how synchronized batches can decrease queue size.

Because it deals with tasks which represent both non-homogeneous durations and non-homogeneous costs of delay, Reinertsen uses a hospital emergency room as a model to describe the concept of Sequencing Work, showing how the cost of delay is used to determine the sequence of providing care to multiple patients with dissimilar needs.

He presents the benefits of Late Binding, a concept based upon the idea that committing resources to long horizons has the undesirable effect of compounding the variances from all preceding tasks. A diagram shows the comparison between a single 8-month commitment horizon with a consecutive series of eight 1-month commitment horizons. These 1-month horizons will be recognized by those familiar with the Agile process known as Scrum, where sprints typically are no more than 30 days in duration, after which decisions are made for committing resources to the next sprint.

8 – Using Fast Feedback

This chapter explains Reinertsen's 24 Fast Feedback Principles. Topics covered here include selecting metrics for product development, the effects of human behavior, ideas from control systems engineering and approaching feedback and control from an economic perspective. He shows how to operate in an environment which is constantly changing, noting that in product development it is prudent, upon determining there is a deviation from the original plan, to reevaluate goals and adjust accordingly than to stick with a static plan.

He shows how to capitalize on unexpected opportunities as they present themselves, with an example of how automobile manufacturers, sticking with their original plans, failed to provide MP3 player compatibility until years after smaller companies took the initiative to supply this.

The benefits of fast feedback are presented through a diagram depicting a reinforcing cycle, where faster feedback leads to less accumulated variation, which leads to lower in-process inventory, which leads to faster flow time, which leads to even faster feedback.

Reinertsen argues the case for enabling a larger segment of the workforce to make decisions by controlling only the economic logic behind decisions rather than the decisions themselves, and suggests

that a team in which its members share the same work location makes for the best arrangement to facilitate all aspects of communication. In deciding which metrics would be most useful, he suggests this will depend on which goals are considered important, offering some guidance in how to choose.

9 – Achieving Decentralized Control

This chapter explains Reinertsen's 23 Decentralization Principles. This begins with some background on how the United States Marine Corps incorporates decentralization into its warfare models. Reinertsen discusses achieving a balance between centralization and decentralization, and shows how to approach making the decisions necessary to arrive at the right balance.

He uses fire fighting as a familiar example of establishing a balance between centralized and decentralized control, explaining how fire extinguishers are placed at strategic locations and are expected to be used to combat small fires without requiring the user to request permission first, but that large fires are fought by trained fire fighters who arrive from a centralized location in fire trucks and have at their disposal more robust fire fighting equipment.

Other topics covered include maintaining alignment of work efforts with decentralized control and both the technological and human aspects of decentralization. He emphasizes the ability of an organization to change direction quickly as well as developing its workforce members to become comfortable taking the initiative:

> "The more we encourage initiative, the more chance we have to provide positive reinforcement for this initiative. The more chances people have to use initiative, the more they will feel this is not a risky thing to do." (p. 263)

Summary

This is an excellent book for educating the reader about the principles associated with product development and for explaining how corporate policies affect the flow of that work. The occasional use of statistical mathematics might offer a challenge to some readers, but Reinertsen succeeds in raising awareness of the considerations for maximizing flow in a product development environment. The assignment of enumerated identifiers to each of the 175 principles, such as Q14, B5 and FF23, not only categorize the principles but provides for a short-hand method of communication amongst those who become familiar with the principles and can refer to them by these identifiers.

Notable passages

- "Once we quantify the cost of delay, we become aware of the cost of queues. Once we recognize the cost of queues, we are motivated to measure and manage them. Without a cost of delay, queues appear to be free and therefore, unworthy of attention." (p. 6)

- "... making activities more efficient is much less important than eliminating *inactivity*. In product development, our greatest waste is not unproductive engineers, but work products sitting idle in process queues." (p. 33)

- "... when product developers choose to operate their processes at high levels of utilization, they create unnecessary and wasteful variability in their processes. It is important to realize that this variability is a self-inflicted wound." (p. 64)

- "In product development, fast feedback also provides disproportionate economic benefits because the consequences of failures usually increase exponentially when we delay feedback. Each engineering decision acts as a predecessor for many subsequent decisions. The number of

dependent decisions generally grows geometrically with time. Consequently, a single incorrect assumption can force us to change hundreds of later decisions. When we delay feedback, rework becomes exponentially more expensive. With accelerated feedback, we can truncate bad paths before they do significant economic damage." (p. 114)

- "When people see that their actions cause consequences, it changes their behavior. They feel a sense of control, and this causes them to take even more control of the system. Instead of being the victim of a vast monolithic system, they become aware of a subdomain where they have power and influence.

  "Victims often can remain victims because they assume they have no control over outcomes. They have the steering wheel in their hands, but they don't turn it. When people begin to believe that the steering wheel they are holding can actually turn the car, they start steering the car. Thus, the human effects of fast feedback loops are regenerative. Fast feedback gives people a sense of control; they use it, see results, and this further reinforces their sense of control" (p. 232)