TOWARDS DEVELOPING ACCELERATORS IN HALF THE TIME

D. Reinertsen#, Reinertsen & Associates, Redondo Beach, CA 90277, U.S.A.

Abstract

New methods are transforming the way organizations develop complex systems which contain advanced technology. In the past, organizations used sequential development processes which required one stage to be entirely complete before the next one started. Now they are shifting to processes that handle work in small batches. These small batches accelerate valuable feedback and reduce the size of in-process queues. Smaller queues unlock simultaneous improvements in quality, efficiency, and cycle time. This new approach is sometimes called lean product development because of its similarity to the ideas of lean manufacturing. This paper will introduce readers to some of the key concepts underlying lean product development and explain their relevance to the design of systems like particle accelerators.

BACKGROUND

The 20th century paradigm for product development used large batch sizes and sequential processes. All requirements were identified before design activities began, all design was completed before systems were built, and the entire system was built before testing commenced. At the time this appeared quite organized and logical—now we know that this approach delivers the benefits of technology both inefficiently and slowly.

Sequential processes cause each activity to be on the critical path of the project for the maximum theoretical amount of time. This lengthens the schedule and delays critical feedback between stages. Slow feedback causes us to pursue unproductive paths longer, which hurts efficiency, cycle time, and quality.

Sequential processes transfer 100 percent of the work product of each stage in one large batch. This method, which is humorously called, "the elephant traveling through the boa constrictor," intrinsically leads to overloads and delays.

The large batch approach also has a more insidious side effect. It leads to a destructive form of regenerative feedback that massively increases development cost, development cycle time, and risk of obsolescence. This phenomenon, called gold-plating, was first described in the Packard Commission report of June 1986. [1] This report pointed out how long schedules inexorably cascade into even longer and more expensive schedules.

Gold-plating occurs in projects that combine large scope and sequential development methods. As the scope of a project grows, its budget and cycle time increases. But large budgets increase financial risk, so these projects require increased scrutiny and involve multiple organization to syndicate their larger risk. This, in turn, further increases cycle time. As cycle times increase another dangerous feedback cycle begins. Because the project is aiming at a distant planning horizon, its performance target becomes more uncertain. The natural response under these circumstances is to aim at the worst case requirements—nobody wants to fund a huge project that becomes obsolete before it is turned on. These high-end performance goals then lead to even more scope, larger investment, and longer cycle times.

In some cases, the sheer size of the project makes it "too big to fail (TBTF)." Such TBTF projects are guaranteed funding, a fact that is not unnoticed by smaller and less well-funded projects. These smaller projects seek ways to become part of the TBTF project, because this will guarantee their funding. The TBTF project begins to act as a magnet accreting even more scope, more risk, and longer schedules.

This disaster scenario arises from two primary causes: large initial scope and a sequential process that leads to long cycle time and consequently a long planning horizon. By using a more concurrent process we can reduce gold-plating, even when we can't change the scope of the project. Organization are now choosing this second path. They realize that large batch, sequential processes inexorably lead to slow, expensive development. The purpose of this paper to describe some of the new methods, explaining how and why they work.

THE NEW PARADIGM

The new paradigm departs from many past practices, but this paper will focus four critical shifts. They are illustrated in Figure 1. The key elements are decentralized control, a flow-centric approach, small batch size, and the use of fast feedback.

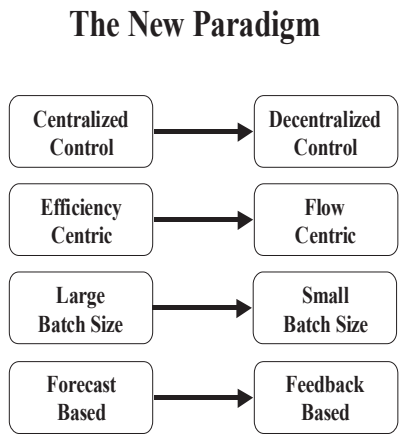

Figure 1: The New Paradigm

#don@reinertsenassociates.com

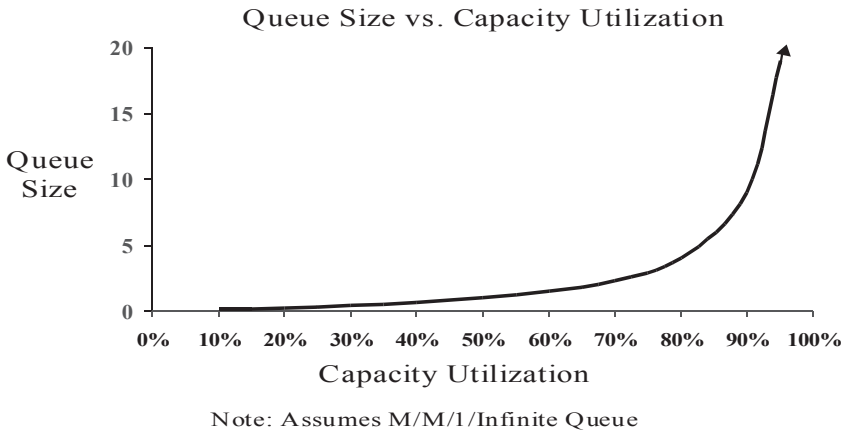

Figure 2: Queue size rises rapidly with increased capacity utilization.

DECENTRALIZED CONTROL

The new paradigm uses decentralized control to achieve fast decisions. Such decentralization mobilizes workers at the lowest level of the organization to make decisions. To work, this decentralization of control must be accompanied with mechanisms to maintain alignment among these local decisions. Such alignment is achieved by giving lower level workers good decision support information, which enables them to make good economic choices without requiring management participation in these decisions. Boeing used such an approach on the Boeing 777. The weight of this plane was a critical objective with enormous economic significance. An overweight plane would breach contractual performance guarantees and cost hundreds of millions of dollars. Classic management methods would elevate such decisions to high levels in the organization. Boeing used a different approach: 5000 engineers were authorized to trade weight for product cost whenever a pound of weight could be eliminated without raising product cost by more than $300. This carefully calculated trade-off rule enabled Boeing to delegate decision-making authority to their lowest level engineers. Because the trade-off rule was controlled by management, management did not need to participate in every decision. Boeing was able to influence the thousands of little decisions without delaying these decisions. Such decentralized control is vital on large projects, because all decisions cannot be funneled through the project manager.

These trade-off rules can exist in many forms, but when development cycle time is of high value, one of the most important decision rules is known as, "Cost of Delay (COD)." COD is the economic value of a one month delay in the delivery of a project. This can be calculated using an economic framework that expresses the value of individual project objectives. Such a framework enables fast, fact-based, decentralized, and transparent project decisions. Systematic methods for calculating COD have been available for twenty years [2].

FLOW CENTRIC

A second aspect of the new paradigm is that it is flow centric rather than efficiency centric. An efficiency centric approach strives to to keep all resources busy 100 percent of the time. This can work well when the work is highly predictable, since overloads are unlikely. However, most advanced development work is not completely predictable. Outcomes are at least partially stochastic and planning must be done using the mathematics of stochastic processes, not the mathematics of deterministic processes. This mathematics is called queueing theory, and it tells us that when we load a process with variability to high utilization we will experience long delays. As shown in Figure 2, a process with variability will experience geometrically larger queues as we approach 100 percent resource utilization.

These queues are important because they contribute nothing to the value of the project, they are pure wasted time. Many organizations believe they have no waste when their resources are highly utilized, but busy workers simply create the illusion of progress. In reality, high levels of utilization cause valuable work products to sit in queues waiting to access these highly utilized resources. In a traditional process it is not unusual to see work products spend 80 percent of their lifetimes waiting idle in queues. Put another way, if these processes could eliminate their queues their cycle time would be 5 times faster with no change in worker productivity. Contrary to what many people believe, a product development

process is never optimized when it operates at a high levels of efficiency.

SMALL BATCH SIZE

A third attribute of the new lean approach is its use of small batches. The batch size in a development process is amount of information that is moved from stage to stage. For example, programmers used to write code for months before it was tested at a system level; it might take years before this code was deployed. Today's web-based companies can conceive a feature, code it, test it, and deploy it in less than 24 hours. This has occurred because they have learned to exploit small batch sizes.

The key to enabling work in small batch sizes is to reduce the transaction costs associated with each batch. In small-batch software development transaction costs have been lowered by orders of magnitude by investing in test automation and continuous deployment technologies. This follows the same pattern that was used in lean manufacturing where set-up time reduction created huge improvements in cycle time, quality, and efficiency.

Batch size reduction is the method of choice for reducing cycle time in modern development processes. In contrast, traditional developers pay virtually no attention to batch size.

FEEDBACK-BASED

Finally, the new approach exploits rapid feedback. It recognizes that each step in development gives us information that alters the conditional probabilities of the outcomes in the next step. If we exploit this emerging information we can adjust our development path to make it more efficient and faster.

Fast feedback loops give us the ability to truncate unproductive paths quickly, which unlocks resources for other purposes. By quickly truncating unproductive path we can improve efficiency with no change in our success rates.

In contrast, the traditional approach to development focusing on up-front planning rather than adaptation. We tried to forecast everything and failed to do this accurately.

PRODUCT ARCHITECTURE

This new small batch size paradigm works best when the product architecture exploits it. This can be done by making explicit architectural choices that facilitate the parallel development of subsystems. Thus, architecture becomes a management tool, not just a technical solution for a technical design problem.

Certain architectural choices dramatically affect the speed of development. First, whenever we can reuse subsystems from previous designs this reduces queueing in our development process. It does so because it lowers capacity utilization and because it reduces variability.

Second, we will be able to develop subsystems in parallel when they have well-defined interfaces. This prevents them from interacting with each other.

Third, we can make system integration much easier when we concentrate variability in a limited section of our system and buffer these subsystems from other subsystems using generous interface margins. Parallel development requires reducing dependencies so that subsystems can progress independently.

Finally, a product architecture suited for parallel development will consist of subsystems that can be tested before the rest of the system is available. This is accomplished by designing these subsystems to support their own testing and by creating strong testing infrastructure.

RELEVANCE TO ACCELERATORS

The methods described in this paper work well for almost any project, but certain conditions make them particularly valuable. These methods inherently reduce queue size which leads to fast flow-through times. This is valuable when fast development is necessary. These processes also create fast feedback. Such feedback is particularly useful when operating near the limits of technology. And finally, these processes with their limited queues and small batches are more efficient. This is useful when unlimited funding is not available. This combined benefit of speed, quality, and efficiency is valuable for accelerator projects.

Importantly, these methods work particularly well for large complex systems. In fact, the more advanced the technology the higher the variability, and the higher the variability the more a stochastic approach applies.

The ideas behind this talk are described in more detail in reference [3].

REFERENCES

[1] President's Blue Ribbon Commission on Defense Management. “A Quest for Excellence: Report to the President.” June 1986, p. 47

[2] P. G. Smith and D.G. Reinertsen, “Developing Products in Half the Time: New Rules, New Tools.” New York: John Wiley & Sons, 1997.

[3] D. G. Reinertsen, “The Principles of Product Development Flow: Second Generation Lean Product Development.” Redondo Beach CA, USA: Celeritas Publishing, 2009