

Process Optimization by Design

David N. Card
DNV ITGS

Dahai Liu and Stephen Kropp
Embry-Riddle Aeronautical University

David Card is the managing director of the US operations of Det Norske Veritas Information Technology Global Services. He spent one year as a Resident Affiliate at the Software Engineering Institute and seven years as a member of the NASA Software Engineering Laboratory research team. Mr. Card is the author of *Measuring Software Design Quality* (Prentice Hall, 1990), co-author of *Practical Software Measurement* (Addison Wesley, 2002), and co-editor *ISO/IEC Standard 15939: Software Measurement Process* (International Organization for Standardization, 2002). He is a Senior Member of the American Society for Quality.

Dr. Dahai Liu is an assistant professor in the Human Factors and Systems department at Embry-Riddle Aeronautical University. He is a member of HFES, IIE, AIAA and INCOSE. Dr. Liu specializes in the areas of human computer interaction and systems engineering. His experience includes human system performance modeling, quality, simulation and usability engineering.

Stephen Kropp holds a Bachelor of Science Degree from Embry-Riddle Aeronautical University (ERAU) in Aeronautical Science with a minor in Business Administration and is currently working on a Master's Degree at ERAU in the Human Factors and Systems Engineering program. He currently works as a Systems Analyst in the Information Technology Middleware Services Department at ERAU

Abstract

The “Lean” paradigm for process improvement is becoming increasingly popular in software and systems engineering, as well as manufacturing. However, the Lean influence arrives most often as redesign and rework in the form of a Kaizen event. This article explains how the Lean principles can be used to help design good processes from the start. Moreover, since Lean is based on queuing theory, processes designed in this manner can be optimized through simulation. The article demonstrates the application of these concepts with results from a simulation study of an industrial software process. It also discusses the relationship of Lean concepts to Agile and the CMMI.

Introduction

Many different approaches to process improvement have found their way into the software and information technology industry in recent years. Lean [1] is a current focus of discussion. Lean thinking evolved from concepts employed at Toyota beginning around 1950. Lean derives from experience in manufacturing, but has been applied

successfully in service and software processes [2]. The five widely recognized Lean principles [1] are as follows:

- Value – identify and focus on the value to be produced for the customer
- Value Stream – manage the process (stream) producing the value
- Flow – maintain a smooth flow of work products through the process
- Pull – only produce work products when the next process step or customer is ready
- Perfection – make each work product defect free, including intermediate work

Each of these principles includes both a cultural and technical component. Although this author recognizes the importance of culture to successful process improvement, this article focuses on the technical component of Lean, which is often misunderstood and misapplied. The cultural components include language, national culture, and general adoption challenges. While these are important, too, they can only be addressed once the technical foundation has been properly understood.

Typically, Lean principles are deployed through Kaizen events where processes (often laboriously established through CMMI, ISO 9001, and Six Sigma-based approaches) are reworked to remove “waste”. A Kaizen meeting focuses on improving a selected portion of the value stream, or process, by identifying obstacles to realizing the Lean principles. Fortunately, those organizations just beginning the process improvement journey have the option of building the Lean principles into their processes from the start through “Process Optimization by Design”, rather than through later Kaizen working meetings.

Lean and Process Definition

Many people have come to believe that Lean is about the volume or quantity of process description. That is, it is commonly believed that a Lean process is one that is defined in a few pages of text and diagrams, rather than a more lengthy document. Actually, the word “lean” in the process context, refers to the amount of inventory (parts and materials in storage) and work in progress (work products underway, but not yet completed). Lean does not refer the volume (e.g., number of pages of definition) of the process that transforms them. A Lean process is one that operates with minimal inventory and work in progress. That may require an extensive process description, or not. Lean stresses developing process descriptions that implement the Lean principles, not arbitrarily reducing page count.

When organizations do focus on reducing the volume of their process descriptions, they often think only of the organizationally defined process. That is, they seek to remove steps or activities from the general process requirements. These are only a small part of the effective process volume of an organization. In most organizations, the organizationally defined process gets extensively tailored for individual projects. Thus, each element of the organizational process has many local instantiations. The result is that most of the volume of processes in an organization results from project-specific

tailoring. Standardizing, or reducing the amount of project-specific tailoring and alternatives, usually is a more effective method of reducing the real volume of processes.

Lean is about the amount of work in progress, not volume of processes. What does this mean in the context of software development? Examples of work in progress include:

- Requirements that have been approved, but not designed
- Design units that have been approved, but not coded
- Code that has been approved, but not tested

Thus, Lean is about reducing the amount of these kinds of items – thus, making production of software flow continuously. The simulation model discussed in a later section helps to show how reducing work in progress leads to greater efficiency and higher throughput.

This notion of Lean matches the vision of Agile development. The Waterfall model is the antithesis of Lean. A typical Waterfall model requires that all requirements analysis be completed before design, all design be completed before code, all code be completed before testing, etc. Thus, the Waterfall model maximizes work in progress. For example, all of the work done to date for each phase is work in progress because it cannot proceed further into production.

The Lean view of process indicates that the Waterfall model is the least efficient, although it requires the least discipline to execute. True Agile development requires more discipline than Waterfall development. For example, Lindvall and Muthig [7] report that configuration management (a basic development discipline) is especially difficult for large Agile projects. Unfortunately, many practitioners view Agile approaches as an excuse for dispensing with discipline, usually resulting in poor quality as well as lower efficiency.

Lean and Queuing Theory

The approach of Process Optimization by Design focuses on the implications of the queuing theory underlying Lean thinking. The performance of a queue can be described mathematically, as follows:

$$T = \left(\frac{V_w^2 + V_p^2}{2} \right) \left(\frac{u}{1-u} \right) C$$

Where:

T = actual cycle time

V_w = variability in work (task) size

V_p = variability in process performance

u = utilization

C = theoretical cycle time to process one unit of work

Figure 1 shows the interactions among these terms. The curves represent the relationship between utilization and throughput for different levels of variability (V_w plus V_p). At 100 percent of capacity cycle time becomes infinitely long. However, processes with low variability (represented by the lower curves in Figure 1) can operate closer to 100% capacity without experiencing dramatic degradation of performance. Interestingly, most software organizations have little insight into the capacity of their processes and don't actively manage it. Managers often aim, unrealistically, for 100% (or higher) utilization.

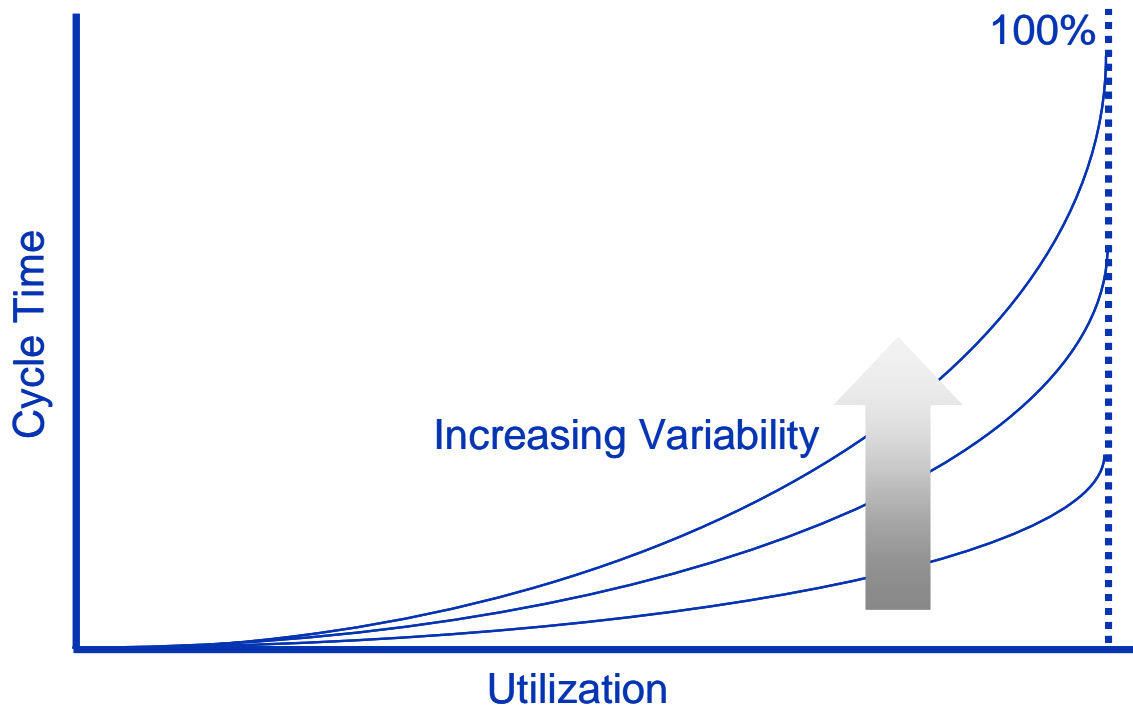


Figure 1. Process Throughput for Various Conditions (from [2])

This model helps to explain the phenomenon of the “mythical man-month” [5], in which adding staff to a project that is late, tends to make it later. Traditionally, this has been attributed to the increased communication overhead associated with larger teams. However, the queuing model suggests other factors may be more important. Adding capacity (in terms of staff) to the project reduces the theoretical cycle time (C) for processing one item of work. At the same time it results in a reallocation of the work among the staff (increasing V_w) and redeployment of processes (increasing V_p). Since the variabilities are power functions and the capacity (theoretical cycle time) is linear (see equation above), the effect of increased variability can easily exceed the benefit of an increase in resources.

Other quantifiable factors also contribute to the performance of queues (although we will not go into the arithmetic behind them here). Thus, queuing theory defines the characteristics of an effective and efficient process, including:

- Balance the work load across subprocesses
- Break work into small packages
- Manage process utilization
- Control process variation
- Control task variation
- Minimize hand-offs and waiting

These characteristics can be built into a process during its initial definition, regardless of the specific modeling or representation technique employed in the definition (e.g., IDEF, ETVX), or applied during subsequent updates to definitions.

Moreover, these characteristics are measurable properties (or performance parameters) of the process. They help to describe its local performance and determine its overall throughput. During initial process definition, many of these performance parameters may be estimated in order to make design decisions. Those decisions can be supported with simulation studies, as described in a subsequent section. Later, during process execution, these parameters should be measured to guide subsequent process management decisions. Consequently, the Lean paradigm links simulation, measurement, and process definition together.

This article reports the results of a discrete simulation study of process performance. The study employed the Arena modeling package (Rockwell Automation) and generic industry data. The simulation results show how decisions about task size and work balance affect overall process performance. The article demonstrates that a systematic process design approach incorporating Lean concepts and simulation makes it possible to optimize the quality and throughput of processes during their design.

Relationship of Lean to Agile

A simple comparison of the five recognized principles of Lean [1] with the twelve principles of the Agile Manifesto [3] suggests some commonality. In particular, six Agile principles map to three of the Lean principles, as shown below:

Value

- Our highest priority is to satisfy the customer.

Perfection

- Continuous attention to technical excellence and good design enhances agility.
- At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly

Flow

- Deliver working software frequently.
- Working software is the primary measure of progress.
- The sponsors, developers, and users should be able to maintain a constant pace indefinitely

Despite the alignment of two Agile principles with the Lean Principle of perfection, many implementations of Agile methods have been criticized for failing to pay sufficient attention to quality (i.e., perfection), resulting in software that does not satisfy the customer.

The Lean principles of Pull and Value Stream are not incorporated into most Agile methods. This is not surprising given that the Agile Manifesto downgrades the importance of processes (i.e., Value Stream) relative to other contributors to software development. Pull is a property that can only be understood in the context of a specific process. In contrast, Lean views the process as the mechanism for achieving perfection, optimizing flow, and responding to pull.

Many practitioners use Agile as a rationale for avoiding process discipline. However, a Lean process requires disciplined performance to coordinate the continuous flow of work through it. An effective implementation of Agile methods also requires discipline. Proponents of Agile methods might benefit by studying the Lean principles of Pull and Value Stream and giving these factors additional weight.

Simulation of Process Design

The effects of the Lean principles on performance can be demonstrated through discrete simulation modeling. The authors are conducting two phases of simulation studies. The first phase involves a simplified process characterized by industry data from published sources. The second phase will study an actual industrial process using data obtained from real projects developed using an actual process.

Figure 2 shows the simple process segment simulated in Phase 1 of our study. While similar to industry practices, it is a substantial simplification of a real-world process.

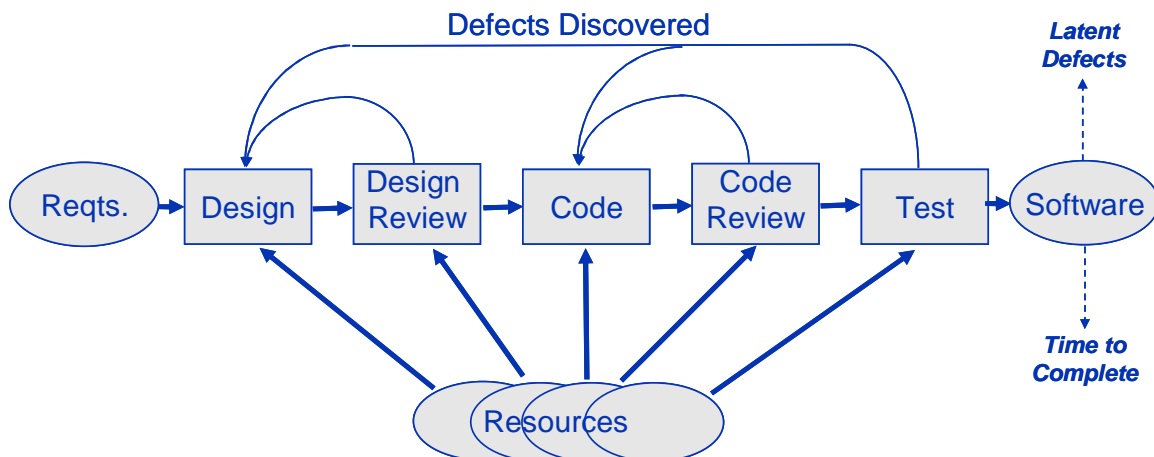


Figure 2. Description of Simulated Phase 1 Process

This process includes five activities. The activities process a fixed set of requirements using a fixed pool of resources (personnel). Each requirement results in one or more units

of work called a Design Unit. At each stage, iteration may be required due to defects inserted in the Design and Code activities, and then found in the Design Review, Code Review, and Test activities. The performance of this process is evaluated in terms of the time required to complete processing all requirements and the latent defects remaining.

Design (work) Units are grouped into packages for implementation. All the units of a package flow through the process together. This process was simulated under two different conditions of work package size (or scenarios):

- Large packages with an average size of 10 Design Units
- Small packages with an average size of 5 Design Units

This initial simulation study employed the Arena modeling package (Rockwell Automation) [4] and generic industry data. Approximately 1000 work units were processed in each simulation run. The simulation was repeated 50 times for each scenario. Performance was evaluated in terms of cycle time (duration), effort expended, and latent defects. Table 1 summarizes the results.

Table 1. Initial Simulation Results

Performance Measure	Small Design (Work) Packages	Large Design (Work) Packages	Improvement (%): Large to Small
Cycle Time per Unit	0.79	0.93	17
Effort per Unit	12.29	14.34	17
Latent Defects	0.05	0.07	25

The table shows significant improvements on all three performance measures are obtained simply by breaking the work into smaller packages for processing. This is a principle of both Lean and Agile, although as previously reported, configuration management [7] becomes a significant challenge with large Agile (and Lean) projects.

The Phase 1 results are sufficient to show that work package size can have an effect on performance. Phase 2 will provide more accurate information on the effects of this and other factors in a real industrial setting. This will demonstrate the value of simulation in helping to design processes and work flows.

Process Optimization by Design

Process Optimization by Design is an approach to implementing the Lean principles at the start of process definition activities, rather than as later improvement rework. This approach involves five steps, as described in Figure 3.

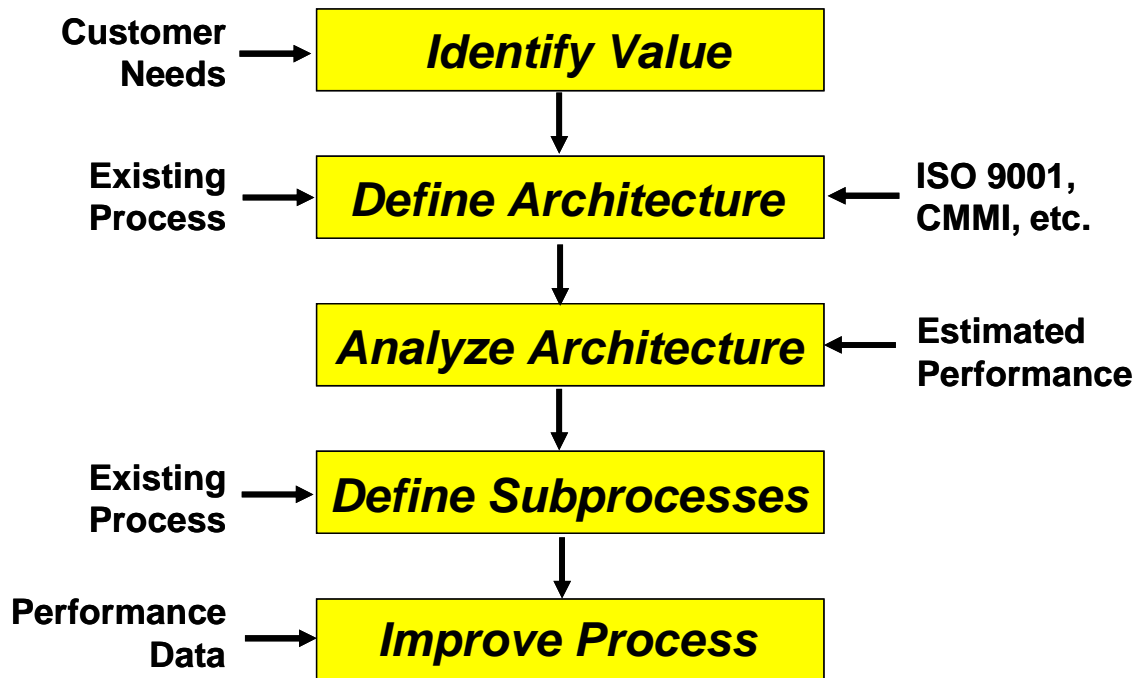


Figure 3. Process Optimization by Design

The five steps and techniques commonly used to implement them are as follows:

- **Identify the value** to be delivered by the process. Obviously, a software process is intended to deliver software, but what are the “critical to quality” characteristics of that software? These are the factors that the process should seek to optimize. Quality Function Deployment (QFD) may be useful here.
- **Define the architecture** of the overall process. The architecture includes the specification of basic components, connections, and constraints. Many process design methods focus on identifying individual components and say little about the architecture. IDEF0 is a process description technique often employed to describe the high-level organization of processes.
- **Analyze the architecture** to identify bottlenecks and optimum operating conditions. Lean principles and the queuing model help to evaluate how well this process will operate. Many factors, such as task size and task effort, will have to be estimated at this stage. The authors recommend discrete process simulation at this stage.
- **Define the subprocesses.** Techniques such as Entry-Task-Verification-Exit (ETVX) may be useful for documenting individual subprocesses within the overall architecture. The details on individual activities must be specified in sufficient detail to guide implementers.

- **Improve the process.** Factors that were estimated while evaluating the architecture should be measured and analyzed during process execution to identify improvement opportunities. Conventional Six Sigma methods are useful for analysis. McGarry, Card, et al. [8] provide general measurement guidance.

The focus on analyzing the process architecture is the most significant contribution of Process Optimization by Design. Many sources recommend developing a process architecture (e.g., [6]). However, few provide any substantial guidance on evaluating the process architecture. Process Optimization by Design views the performance of the process architecture in terms of the Lean heuristics, as well as from the perspective of simulation. Process designers get meaningful feedback on the likely performance of their designs before investing in detailed subprocess specifications.

Summary

Lean Principles can be applied during initial process design as well as during Kaizen and other after-the-fact improvement activities. Simulation of process designs help to identify bottlenecks, evaluate alternatives, and adjust operational policies. The “Process Optimization by Design” approach incorporates Lean principles, established process definition techniques, and simulation to help ensure the definition of close to optimal process designs.

Future work, i.e., Phase 2 of the simulation study will apply this approach to specific industry processes and projects, rather than working with a generic process and data. This will provide examples of specific decisions based on this approach.

References

- [1] J. Womack, D. Jones, and D. Roos, *The Machine that Changed the World*, Harper, 1991
- [2] P. Middleton and j. Sutton, *Lean Software Strategies*, Productivity Press, 2005
- [3] Principles behind the Agile Manifesto, www.agilemanifesto.org, 26 June 2007.
- [4] W. Kelton, R. Sadowski, and D. Sturrock, *Simulation with Arena, 4th Edition*, McGraw Hill, 2007
- [5] F. Brooks, *The Mythical Man Month*, Addison Wesley, 1975
- [6] M. Crissis, M. Konrad and C. Schrum, *Capability Maturity Model – Integration, 2nd Edition*, Addison Wesley, 2007
- [7] M. Lindvall, D. Muthig, et al., Agile Software Development in Large Organizations, *IEEE Computer*, December 2004
- [8] J. McGarry, D. Card, et al., *Practical Software Measurement*, Addison-Wesley, 2002