

Lean Software Development

Christof Ebert, Vector Consulting Services

Pekka Abrahamsson, Free University of Bozen-Bolzano

Nilay Oza, University of Helsinki



LEAN DEVELOPMENT IS a product development paradigm with an end-to-end focus on creating value for the customer, eliminating waste, optimizing value streams, empowering people, and continuously improving (see Figure 1¹). Lean thinking has penetrated many industries. It was first used in manufacturing, with clear goals to empower teams, reduce waste, optimize work streams, and above all keep market and customer needs as the primary decision driver.²

This *IEEE Software* special issue

addresses lean software development as opposed to management or manufacturing theories. In that context, we sought to address some key questions: What design principles deliver value, and how are they introduced to best manage change?

Steady Evolution

Famous author Antoine de Saint-Exupery once remarked that perfection is not about whether there's nothing left to be added, but if there's nothing left to be cut out. It's an old wisdom from

software development that most features in any product don't add value, but rather create unnecessary cost and complexity, thus gradually deteriorating the product. There are two notable examples of this type of development. The initial Netscape browser had grown its complexity so fast that it couldn't be maintained anymore, thus the company lost a huge market share and technology lead within a few years. More recently Nokia's Symbian 60 smartphone platform was abandoned due to its having grown beyond control, making it

harder for new innovations to be implemented. Across industries and independent of whether it's general-purpose, IT, or embedded software, between 30 and 50 percent of all features are unnecessary and add overhead.¹

The demand for cost savings in recent years has forced the software industry to look at lean methods with the ambition of applying them to "software production."³ Although some claim that principles from other fields can't apply to a creative and design-oriented discipline such as software development, many studies have proven the simple wisdom that we all benefit from empowered and motivated teams, we build our products faster and with better quality if market strategy is understood and requirements changes managed, we learn from previous defects, we can emphasize repeatable processes, and we should build from high-quality components. The software industry is now poised to transform to customer-centric development, which translates into both reducing the total life-cycle cost and increasing efficiency and effectiveness by eliminating waste and adding value.

With the onset of 2009's worldwide economic crisis, many companies started looking for lean principles to make their software development efforts more efficient. Entire business processes can be based on lean principles, with eBay serving as a prominent example. eBay continuously improves and reinvents its business model, anticipating and creating market needs—it started as an auctioning platform, then moved to a marketplace with independent stores, and now provides the leading payment services, while always promoting what adds value to its users.

Today, companies find themselves looking to lean development also to compensate for an increasing shortage of talented engineers. In a 2010 survey

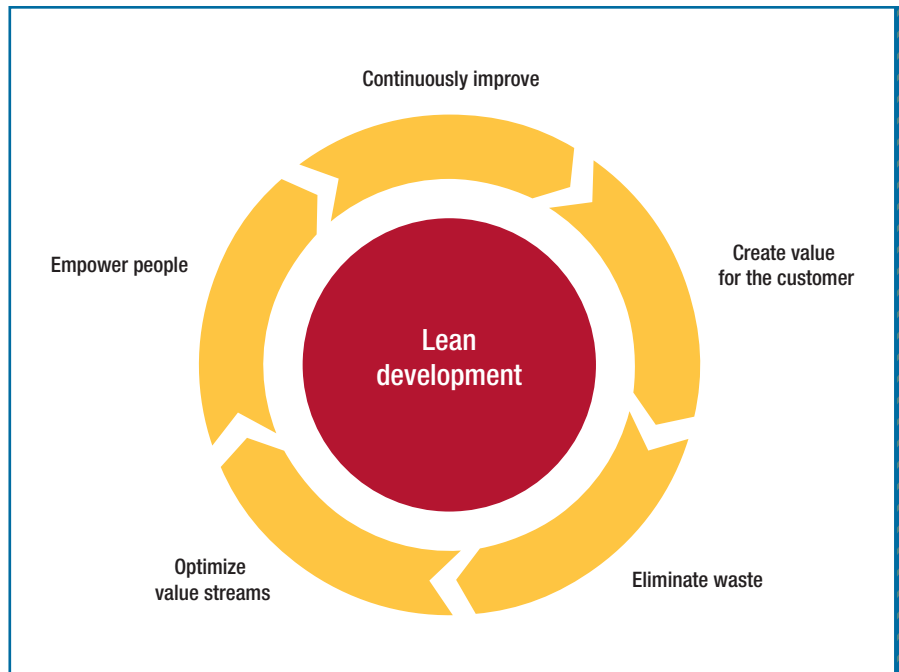


FIGURE 1. Lean product development cycle with the five elements of lean development, starting with value orientation (upper right), then reducing overheads, improving interfaces, and finally empowering developers in order to achieve continuous improvement.

by Forrester, 35 percent of organizations polled described their primary development method as based on agile and lean principles.⁴ Forrester interviewed the development managers of companies such as HP, IBM, and Microsoft and found they had all adopted practices to be more flexible and closer to their customer and the market needs. Yet only 39 percent of organizations surveyed consider their respective implementations to be mature.⁴

From the software perspective, the path to lean started with agile methods in coding. Over the past decade, most companies introduced agile development to address efficiency and effectiveness. Although it helped tremendously with user-centric and iterative development, its enterprise-level applicability remains stagnant. Too often, agile practices are limited to teams or projects, focusing too much on the

short term, such as reducing documentation and unnecessary features, only to find later that overall life-cycle cost were negatively impacted. Lean development attempts to bridge this gap. In fact, when Mary and Tom Poppendieck wrote their first book on lean software development 10 years ago,³ it was tightly connected to agile software development. Recently, much more diversity has been introduced, from user needs and workflow analysis to overall performance measurement.

But there's a dark side in this fast move toward lean software development, as we've faced in our own experience. Too often, the expectations surrounding the hype are exaggerated, and lean principles are only superficially introduced, such as a specific method without exactly knowing what to expect from it or considering its limits. Consequently, results of such unmanaged

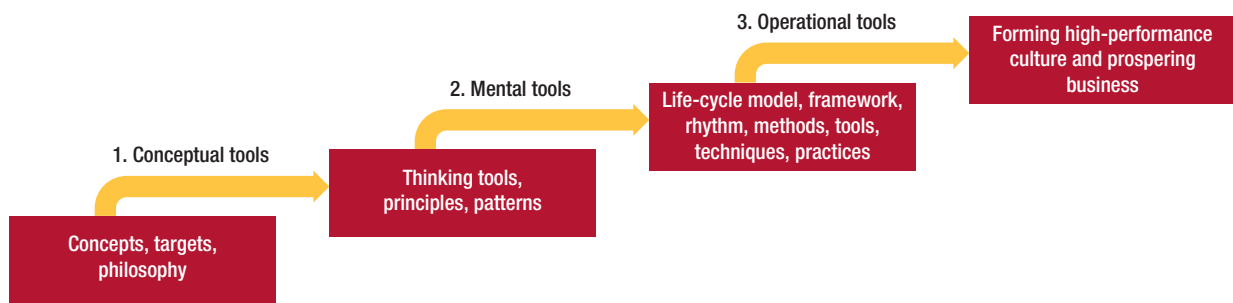


FIGURE 2. A framework for method adoption, starting with initial concepts such as “we will reduce waste,” to understanding how this is approached such as prioritizing requirements, on the way up to making it operationally feasible, such as clarifying how priorities are set.

change to lean are meager and unsatisfactory, and engineers feel frustrated. Superficial introduction of a method is a much wider phenomenon than you might think. Unfortunately, a majority of method-deployment programs in software industry today are trapped in this fallacy of thinking that operational tools are sufficient for improving development. Looking to industry, we see that about half of all culture- and method-change programs fail.¹ Sustainable change needs professional change management beyond simply selecting a method and trying to practice it. Figure 2 highlights the necessary components for successful, company-specific method development and deployment.⁵

Obviously, some lean principles just don't fit into software development. Flow and queuing theory, for instance, aren't at all practical for normal business, so we should be careful not to overemphasize mathematical analyses in development processes. Such challenges offer more than enough reason for *IEEE Software* to examine the usage of lean principles in software development. The intangible nature of software, developers as knowledge workers, and the difficulty in defining flow in software development make the application of lean principles and practices challenging. Thus, although lean practices such as eliminating waste or improving

interfaces and workflows are sometimes adopted in addition to or in combination with agile methods, we're still in the early phases of truly understanding how “lean methods” impact software development. Nevertheless, the need is there across IT and software industries: lean conferences are born, lean software books are selling, and organizations are keenly adopting lean principles.

In This Issue

For this special issue, we received 23 submissions from all over the world. Based on the feedback of our expert reviewers, we ultimately selected five articles from industry and academia that we thought could best address lean development with its many facets and yet provide hands-on perspectives. A key selection criterion was to facilitate technology transfer to the environments of our readers.

The first article by Mary Poppendieck and Michael Cusumano, entitled “Lean Software Development: A Tutorial,” provides thorough insights into lean software development—where it comes from, what it means, how it relates to agile development methods, and its outlook in the future. If you intend to embark on a lean journey, this tutorial is an excellent starting point.

“Making Architecture Visible to Improve Flow Management in Lean Software Development,” by Robert L.

Nord, Ipek Ozkaya, and Raghvinder Sangwan, shows how lean principles are used in architecture development. The authors highlight the role of architecture and how it can help improve the flow in lean software development. This article will help you better examine and improve flows in your own lean setting through the focal point of architecture.

In “Lean Solutions to Software Product Management Problems,” Andrey Maglyas, Uolevi Nikula, and Kari Smolander focus on the path from development to product strategy. Their article, based on a systematic study in several companies, presents five problems that could be avoided or solved by adopting specific lean approach to software product management. It brings concrete problems to the surface along with specific solutions for them.

Dag Sjøberg, Anders Johnsen, and Jørgen Solberg take an empirical approach in their article, “Quantifying the Effect of Using Kanban vs. Scrum: A Case Study.” Using more than 12,000 work items collected over the years 2009–2011, the authors demonstrate how the effect of processes or methods emphasizing scrum versus Kanban can be evaluated and compared on the basis of objective data.

Finally, in “A Business Case for Feature-Oriented Requirements Engineering,” Arnold Rudorfer, Tobias

Stenzel, and Gerold Herold outline the journey of a business unit in the Imaging and Therapy Division of Siemens while introducing feature-oriented requirements engineering and developing a business case to understand the benefits for the development organization. Their article emphasizes the relevance of a good business case as a success factor for successful change in industry.

The five core principles of lean product development (that is customer focus, waste reduction, team empowerment, work stream efficiency, and continuous improvement) were coined years ago, but we still lack a coherent set of features applicable to lean software development. For example, while some people reduce “lean” to results (less waste), others look primarily to a single method (value stream analysis versus team empowerment). This doesn’t help in practice, as it creates method fights similar to the early days of object orientation or agile software development. If everything is called “lean,” and different methods from agile to project management are mixed ad hoc, confusion results both in science and practice, but this creates no sustainable improvements in software development.

With this special issue, it’s our goal to build a foundation and facilitate alignment on what “lean” means within our community. Essentially, to advance lean development, empirical studies must be conducted across industries and value must be measured. Or in the words of before-cited Antoine de Saint-Exupery, “Grown-ups like numbers.” The selected articles merge the mutual interest of practitioners and academic researchers to be more successful in the market by applying lean thinking. In bringing state-of-the-practice knowledge on lean principles to software development, we hope this special issue improves your day-to-day

ABOUT THE AUTHORS




CHRISTOF EBERT is managing director at Vector Consulting Services. He supports clients around the world to improve product strategy and product development and to manage organizational changes. Contact him at christof.ebert@vector.com.



PEKKA ABRAHAMSSON is a vice dean and a full professor at the Free University of Bozen-Bolzano. His research interests are in lean and agile development, software innovation and empirical software engineering. Contact him at pekka.abrahamsson@iee.org.



NILAY OZA is a project director at the University of Helsinki. Previously, he was with VTT Technical Research Centre of Finland. His research experience is at the crossroads of enterprise transformation, Internet technologies, and business of software. Contact him at [nilay.oza@cs.helsinki.fi](mailto:oza@cs.helsinki.fi).

development and stimulates new approaches to adding more value with less effort. 

References

1. C. Ebert and R. Dumke, *Software Measurement*, Springer, 2007.
2. J. Womack, D. Jones, and D. Roos, *The Machine That Changed the World: The Story of Lean Production*, Simon & Schuster, 2007
3. M. Poppendieck and T. Poppendieck, *Lean Software Development: An Agile Toolkit*, Addison-Wesley Professional, 2003.
4. D. West and J.S. Hammond, *The Forrester Wave: Agile Development Management Tools*, 2010.
5. J.-P. Tolvanen, *Incremental Method Engineering with Modeling Tools: Theoretical Principles and Empirical Evidence*, Univ. Jyväskylä, 1998; http://users.jyu.fi/~jpt/Tolvanen_dissertation.pdf.



Selected CS articles and columns are also available for free at <http://ComputingNow.computer.org>.

IEEE Software

FIND US ON
**FACEBOOK
& TWITTER!**

[facebook.com/
ieeesoftware](https://www.facebook.com/ieeesoftware)

[twitter.com/
ieeesoftware](https://twitter.com/ieeesoftware)