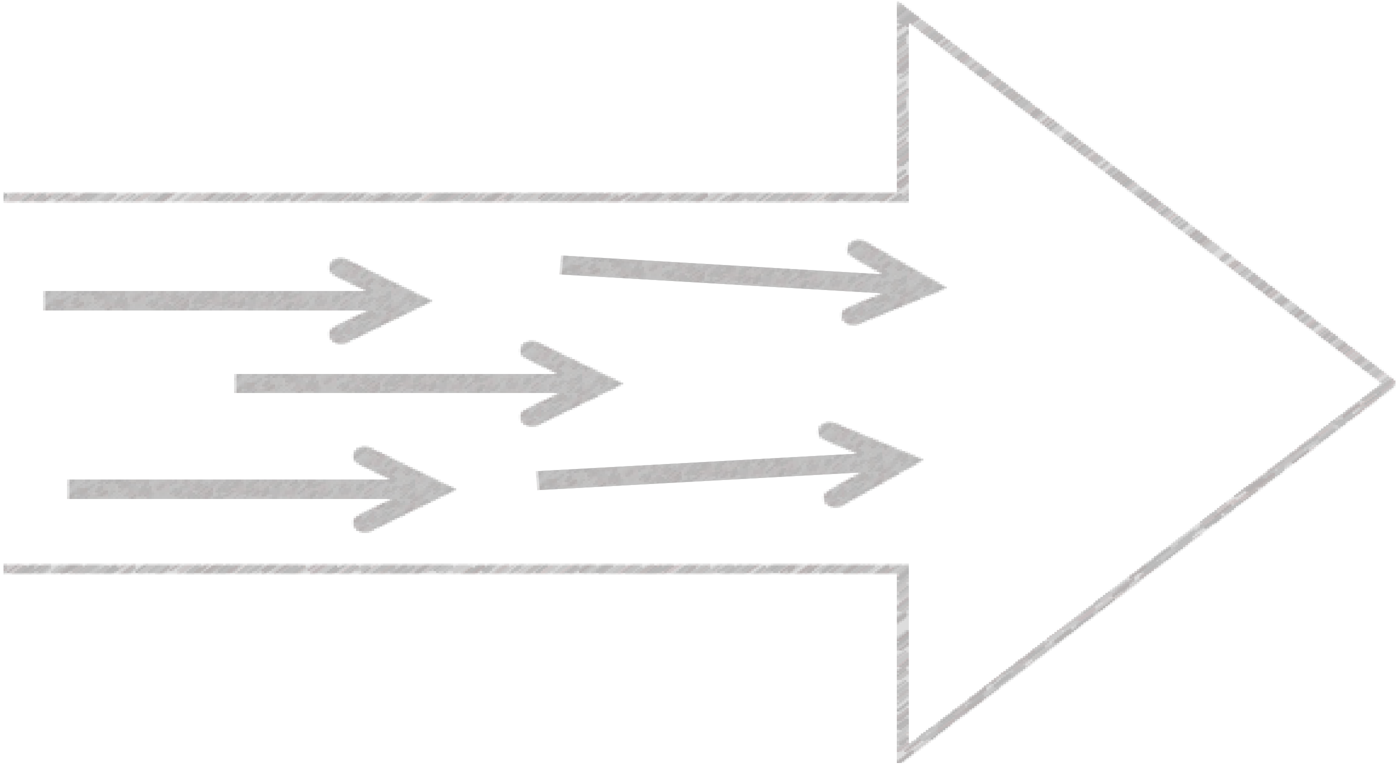
Net Objectives
ESSENTIAL WHITE PAPERS

Aligning Multiple Teams with Lean-Agile Thinking

by Al Shalloway



Aligning Multiple Teams with Lean-Agile Thinking

by Al Shalloway

A Net Objectives Essential White Paper

Net Objectives Press, a division of Net Objectives
1037 NE 65th Street Suite #362
Seattle, WA 98115-6655
404-593-8375

Find us on the Web at: *www.netobjectives.com*

To report errors, please send a note to *info@netobjectives.com*

Published by Net Objectives, Inc.

Notice of Rights

Notice of Liabilities

The information in this book is distributed on an "As Is" basis without warranty. While every precaution has been taken in the preparation of this book, neither the authors nor Net Objectives shall have any liability to any person or entity with respect to any loss or damage caused or alleged to be caused directly or indirectly by the instructions contained in this book or by the computer or hardware products described in it.

10 9 8 7 6 5 4 3 2

Printed in the United States of America

INTRODUCTION

Software development and IT shops around the world are embracing Agile methods. For teams and smaller organizations, the results have been impressive. And yet for larger IT organizations and software development organizations, success has been elusive. Rather than productivity and profit, they experience mediocre results or worse. Why is this?

It is a question that some within the Agile community are now starting to address. A big part of the issue is that, at the level of the enterprise, Agile has lacked a solid systems perspective. It is not enough to try to build up team by team, with management trying to remove impediments along the way; instead, it requires an integration of business stakeholders, middle management, and effective teams. It requires both systems thinking and the discipline that systems thinking demands.

This article describes three key principles of Lean Thinking for software development. It applies these to the value stream (the name Lean gives the workflow from "concept" to "consumption"). It also describes three disciplines Lean-Agile teams will need to follow to keep value flowing. Finally, it illustrates how Lean Thinking guides Agile enterprises in addressing challenges in their context. Lean-Agile lays out a different, more disciplined approach for scaling Agile.

THE DISCIPLINE OF LEAN-AGILE

"Disciplined Agile" may sound like an oxymoron and has certainly been controversial for some in the Agile community, but it is essential for sustained success. Discipline does not mean "heavy handed" — we all know that too much management, over-planning, over-design, and overly large projects are not effective. However, undisciplined teams that use Agile as a justification to avoid doing what is necessary are also not effective ... and, by the way, are not Agile.

While discipline in Agile is required to improve performance at the team level, it is even more essential at the enterprise level. It provides a pathway for scale and sustainability.

Instilling discipline and coordination across Agile teams requires an extension to classic Agile and its team-centric point of view. It requires extending Agile with Lean principles and practices. My colleagues and I call this extended Agile mindset "Lean-Agile."

Consider the classic approach to Agile across teams. You start Agile with one team, then add another and another and coordinate them with a "team of teams." The problem is that coordinating teams with a team-of- teams approach tends to not work the more teams there are. Getting a few team members working together within one team is considerably different than getting a few teams working together. The reasons for this are intrinsic to the differences between intra-team dynamics and inter-team dynamics. Cross-team dynamics are deceptively difficult.

Lean-Agile takes a different approach. Lean-Agile uses systems thinking. Its focus is on the incremental delivery of business value by attending to the entire value stream as one system. It says that management needs to provide the big picture while teams implement within that context. Lean-Agile will suggest changes to the workflow to remove delays in receiving feedback, detecting errors, using information, and ultimately, delivering value.

Alan Shalloway is the founder and CEO of Net Objectives. With over 40 years of experience, Alan is an industry thought leader in Lean, Kanban, product portfolio management, Scrum and agile design. He helps companies transition to Lean and Agile methods enterprise-wide as well teaches courses in these areas. Alan has developed training and coaching methods for Lean-Agile that have helped Net Objectives' clients achieve long-term, sustainable productivity gains. He is a popular speaker at prestigious conferences worldwide. He is the primary author of Design Patterns Explained: A New Perspective on Object-Oriented Design, Lean-Agile Pocket Guide for Scrum Teams, Lean-Agile Software Development: Achieving Enterprise Agility and Essential Skills for the Agile Developer. Alan has worked in literally dozens of industries over his career. He is a co-founder and board member for the Lean Software and Systems Consortium. He has a Masters in Computer Science from M.I.T. and a Masters in Mathematics from Emory University. You can follow Alan on twitter @ alshalloway

Let's look at these key principles in turn to see how they effectively extend Agile to work at scale.

SYSTEMS THINKING

Systems thinking is the process of understanding how parts of a system influence one another within a whole. In software development, these parts are product selection, product prioritization, requirements, architecture, design, code, test, quality assurance, delivery, integration, management, HR, and more. Systems thinking suggests we look at the development workflow in its entirety, seeing how one part affects the others. Local optimizations without this view may have little or even a harmful impact on the whole.

Lean offers a particular way to look at the system: the value stream. The value stream is the flow of work from when an idea is first conceived through implementation, deployment, and eventual use. The time from start to finish is called "cycle time." Lean Thinking says that actions that shorten cycle time are usually good and those that lengthen cycle time are probably not. Lean Thinking looks for ways to remove delays, which also results in eliminating unnecessary work. This leads to improved quality and lowers costs.

When multiple teams and products are involved, this holistic approach means looking at the entire book of work. For example, it may be that delaying one project is worthwhile if another project can deliver value greater than the cost of delaying the first project. This gives us helpful questions for deciding whether to start a new project: "Will this new project add to the value being delivered?" and "Will adding this project slow down or negatively impact the ability of existing projects to deliver value?"

MANAGEMENT OVERSIGHT (CREATING THE CONTEXT FOR THE TEAMS' WORK)

Having "self-organizing teams" is a very good Agile practice. It works at the team level because teams are able to apply their own local knowledge to adapt to their conditions. Lean says that, at scale, this is not sufficient and that a broader perspective is also required. Management provides this perspective by creating the context within which teams work together. When several teams are involved, a lack of management oversight inevitably leads to a lack of team coordination, which leads to wasted effort.

This enterprise awareness helps at the project level too. Sometimes teams are able to continue providing value on their project, but other projects provide opportunity for greater value. The decision to switch to the other project must come from someone with eyes on the whole portfolio. Management provides this oversight to help a team stop its current project, at an appropriate place, in favor of a more valuable one.

Another area where management oversight helps is in continuous improvement. Lean Thinking suggests that management plays a crucial role in improvement: this includes both improving the environment in which teams work and acting as a coach, often by questioning the teams about their process and helping them think about ways to improve. This is management via leader- ship and coaching.

REMOVING DELAYS

Removing delays is a central tenet of Lean. In manufacturing, they call this "just-in-time" — making things just before they are needed. In software development, it means to not do work before it should be done. For example, don't create requirements until you are ready to work on them, nor write code until you are ready to test it.

All Agile methods work well in lowering delays in the workflow by advocating small batch sizes, shortening feedback loops, and avoiding work before it is needed. This removes delays, which is essential because of the extra work these delays create. For example, consider how delays between coding and test affect developers. It takes relatively little time for developers to fix bugs they are immediately told about compared to bugs they discover weeks later. Testing up front is a Lean-Agile way to remove delays.

PUTTING THEM TOGETHER

Lean-Agile puts these three principles together — systems thinking, management creating the context for the teams, and removing delays — to provide a solid and proven approach to enterprise agility. This requires honoring:

- Your understanding of the best way to work on things when working alone
- The agreements between different roles as to how they are to work together
- The decisions of those who are in a better position to make them

LEAN-AGILE AND THE VALUE STREAM

Multi-product, multi-team agility can be achieved by attending to flow and ensuring everyone agrees on who makes certain decisions at certain places in the value stream. Before going into how this works, let's first look at the reason for focusing on the value stream.

Most companies are organized hierarchically even though work flows laterally across the organization (see Figure 1). People are managed vertically; value flows horizontally.

UTILIZATION VS. FLOW OF VALUE

The problem is that managers will manage what they can see. In this case, they are going to manage employee workload, productivity, and quality of work. While this sounds reasonable, it is more important to manage the time to market of what is being built, the effects of upstream groups on their staff, and the effects their staff have on downstream groups. Focusing on people rather than on the workflow results in a lack of cooperation.

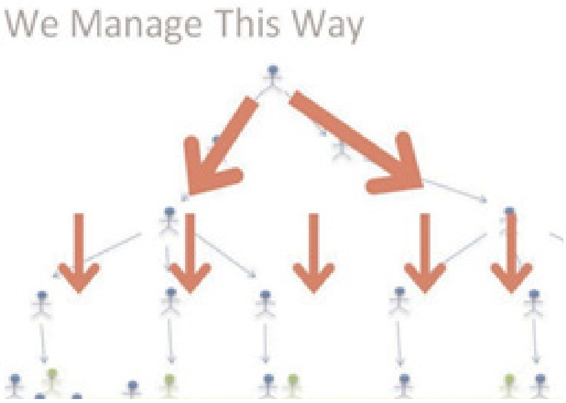


Figure 1 — Management and workflow are orthogonal.

Consider the workflow for a project illustrated in Figure 2. Someone starts on the project only to hand it off to someone or to wait for someone for information. Although people are always busy, work starts and stops. There is activity on a work item, and there is waiting until someone can work on it (see Figure 3).

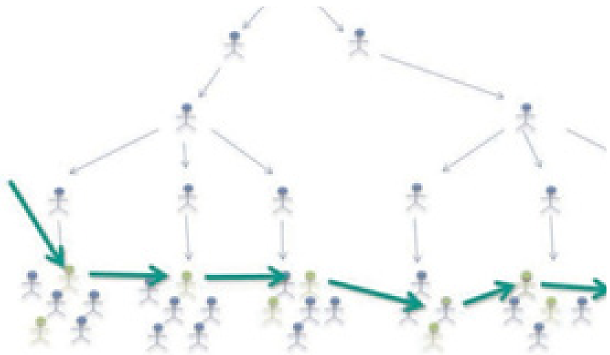


Figure 2 — The workflow across the organization.

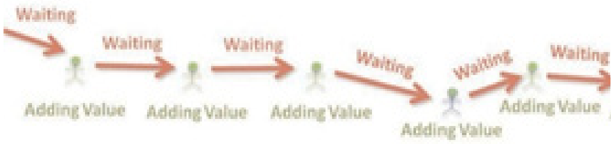


Figure 3 — Adding value and waiting time in a value stream.

TIME TO MARKET

The important question to ask is, "What percentage of time is spent working on the item and what percentage is spent waiting?" The numbers may surprise you. In traditional organizations, people may work on four to six projects at a time. This suggests that, on average, they cannot devote more than 20% of their time to any one project. This in turn means that, on average, 80% of the time spent on any particular work item is devoted to waiting for someone to pick it up. And that is delay! Even though people are fully busy, value is not flowing smoothly and quickly through the value stream.

Does this number seem too large? If you are not tracking this, how would you know? In too many organizations, no one is managing this.

Lean Thinking tells us we are creating a problem by an improper focus on employee utilization rather than on the flow of value. Yes, it is a good goal to make sure people are working. But it is counterproductive when doing so adds additional delays in the workflow. That is just what happens when you have too many projects going at once, even if people are fully busy.

We start with good intentions, allocating people to the most important projects. At some point, the ideal teams are set up. Then a new project comes along. If someone seems to have some "spare capacity," they get assigned to the project even if this will require them to ask for help from key people who are already totally booked up. Key people are the ones who have essential subject matter expertise, certain skill sets (such as architecture), or happened to be around years ago when your now- legacy system was developed. Too often, these individuals are already overloaded. Then the new project starts, and they are asked for more help. Now they are seriously overutilized, and the delays in workflow start. And so it goes in an endless, downward cycle. The net result is that your best people are made to work in the most ineffective manner. This leads to ...

Discipline #1: Stakeholders Cannot Start More Projects than the Development Organization Has the Capacity For

Stakeholders are required to identify and prioritize those projects that are most important, that provide the greatest value. Teams pull from this prioritized list when they are ready for work.

However, this is only one part of the big picture. The teams have to pull the work they are going to do in unison so that one team doesn't finish its work, have to wait on another team to finish, and then thrash as the teams try to integrate their respective pieces. This leads to...

Discipline #2: Teams Capable of Delivering the Selected Value Must Work in Unison

There must be enough capacity to build the entire increment of work before pulling from the product queue.

BUILDING AND INTEGRATING ACROSS THE FEATURE LEVEL

Multiple teams working at the same time is good, but it is insufficient. This is often the case when companies are large and have component teams. Consider the situation shown in Figure 4. There are a hundred or so developers organized around two or more product lines. Call them Product Line A and Product Line B. Each of these applications has its own component team that [develops shared methods.] There is also a component team that works across applications.

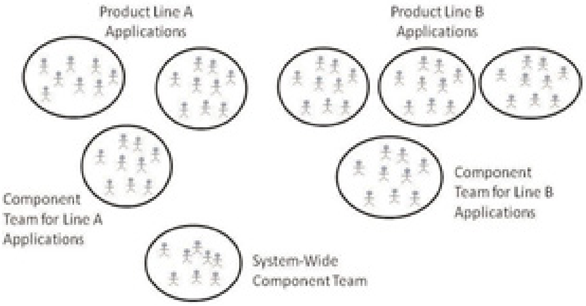

Product Line A Applications
Product Line B Applications
Component Team for Line A Applications
Component Team for Line B Applications
System-Wide Component Team

Figure 4 — Organization of company with component teams for two different product lines.

One solution would be to create cross-functional teams with people from each application, from the component team(s) for the application, and from the component team(s) that run across applications. This is a great approach if you can do it, but too often, it is just not possible.

Suppose Agile teams must work together (as shown in Figure 5) and over several sprints. This was the situation I saw at one client. These were highly functioning Scrum teams that were delivering their work well in each sprint and yet were not able to deliver the overall feature in an efficient manner. While this was confusing at first, a little value stream mapping helped to reveal what was happening.

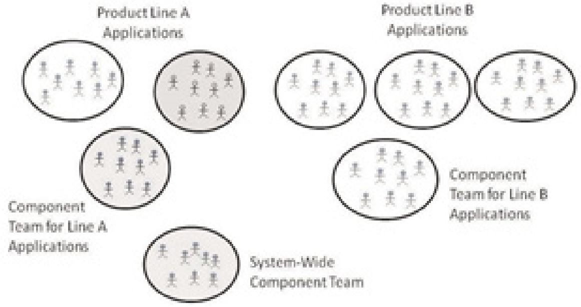

Product Line A Applications
Product Line B Applications
Component Team for Line A Applications
Component Team for Line B Applications
System-Wide Component Team

Figure 5 — Teams collaborating together.

Figure 6 shows how work of a feature was spread out across these teams. Each team got a backlog consisting of their part of the feature. The teams would then take these backlogs and work at their own discretion. Figure 7 shows the first sprint. Note that the different shades on the backlog represent pieces dependent upon each other. While each team told the other teams what they were doing, little coordination actually took place.

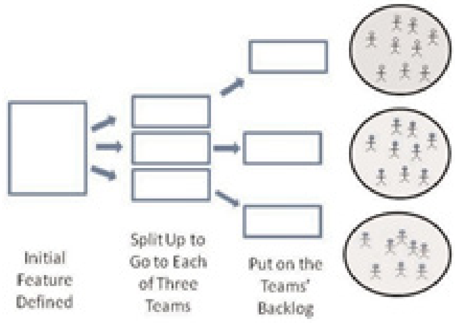

Initial Feature Defined
Split Up to Go to Each of Three Teams
Put on the Teams' Backlog

Figure 6 — How work is assigned to the three teams involved.

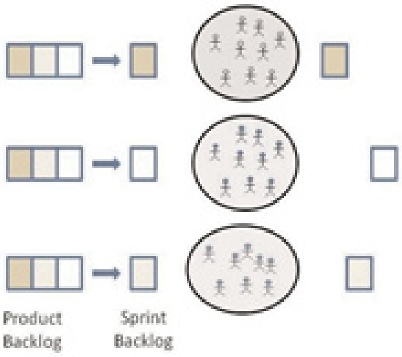


Figure 7 — Work selected for first sprint.

What was happening was that each team chose stories from their backlog based on what would make them most efficient. After selecting what to work on, they coordinated with the others. But the damage had already been done. They were unable to do any significant integration after the first sprint because the completed pieces did not represent end-to-end functionality. This is shown in Figure 8.

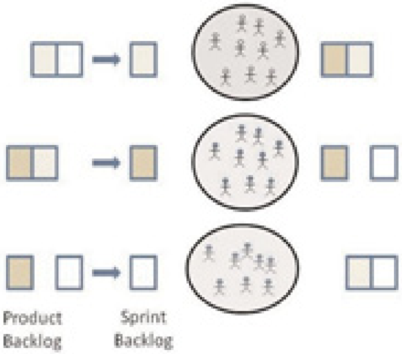


Figure 8 — Work selected for second sprint.

And so it continued until the teams completed all of the work for the feature (Figure 9). Now, they were finally able to integrate their work. The challenge was that there had been several sprints' worth of work on these different parts of the system without any meaningful integration. It was almost a certainty that more work would have to be done. Thrashing ensued. Integration took longer than if they had been doing it on a continual basis. That is delay and waste.

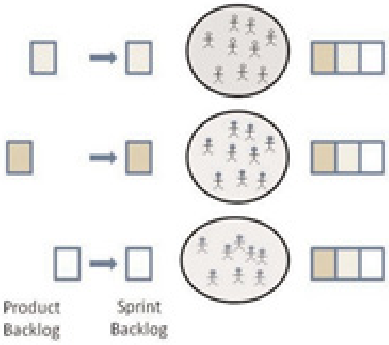


Figure 9 — All parts of feature have been done across the teams.

They incurred another type of delay as well. Since they had not done integration, they were not able to show the functionality to the customer to get feedback. The feedback loop was much longer than the length of the sprint. And that is delay and waste.

To solve this problem, Lean Thinking says we want both short cycle time (the time from conception to consumption) and to have as few delays as possible along the way. This means we want quick feedback loops along the entire value stream.

Of course, achieving quick feedback is an Agile discipline. We strive to deliver working software at the end of every sprint. In the example above, while working software was delivered at the team (Agile) level, it was not being delivered at the enterprise (Lean) level. It is the team of teams that should be delivering working software every sprint. Therefore, instead of giving the teams their backlogs independently, the work should be divided such that the work done each sprint will deliver a piece of functionality that can be demonstrated to the customer.

This is not an individual team decision, and it leads to...

Discipline #3: Teams Must Let Someone Who Sees the Bigger Picture Decide What They Should Be Working On

Taking this approach leads to a different value stream. Compare the improved value stream in Figures 10-12 with the original value stream in Figures 7-9. The new value stream enables demonstration of the software at a much quicker pace, resulting in shorter feedback cycles and less integration work. It also allows all of the teams to have greater vision of what they are building.

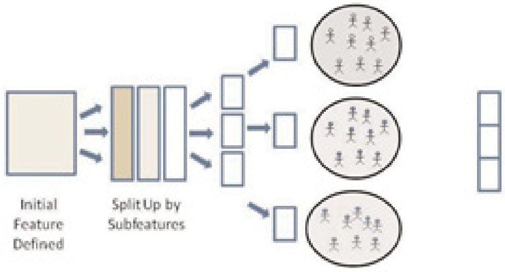


Figure 10 — Work selected for first sprint to enable feedback on one slice of functionality.

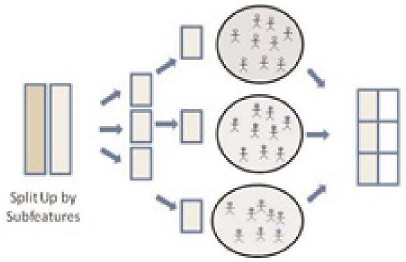


Figure 11 — Work selected for second sprint to enable feedback of another slice of functionality.

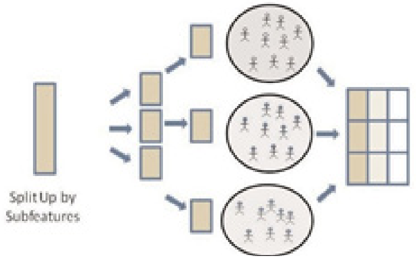


Figure 12 — Work selected for third sprint to enable feedback of last slice of functionality.

OPTIMIZE THE WHOLE (MACRO-) TEAM

Does this mean we are just creating a bigger team? Perhaps. Instead of three teams that have to work together, there is now one larger team working together. The important thing to note is that it was Lean Thinking that guided us here to address a problem in this context. It was clear that building software that couldn't be demonstrated was the problem.

Interestingly enough, the optimal solution would be to create three cross-functional teams from the teams being discussed. Each would then build a piece of functionality, but a single backlog would still be used to ensure that the teams were working together in the best way.

In making decisions on how to develop software, Lean's mantra of "optimizing the whole while attending to flow" provides essential insights. In this case, instead of focusing on how each team works individually, we must consider the macro-team, the larger team actually building the software. How will it best develop soft- ware with the shortest feedback loops possible? It may seem surprising that competent teams couldn't see this. We have found, however, that the focus on Scrum teams consistently obscures the bigger picture that Lean's systems thinking naturally brings to the fore.

SUMMARY: THE PATTERN OF SUCCESS

Software development should be about delivering value quickly. It requires a relentless focus on removing delays, whether they come from too much work-in- progress, waiting, testing or poor workflow. The larger the organization, the more teams that are involved, the more difficult this challenge becomes. Trying to scale up from individual teams to the enterprise rarely works. Lean Thinking is naturally focused on enterprise-level issues. Agile methods informed by Lean Thinking lead to the disciplined type of Agile required for the Agile enterprise. We must remember that Agile is not about team- iterative development, but rather about the incremental delivery of business value.

OTHER ARTICLES OF INTEREST

Go to *www.netobjectives.com/articles* to see the following articles:

- The Business Case for Agility
- Why Tailored Agile Transformations Are More Effective, Less Expensive and Less Risky

Check out the Net Objectives Portal at *portal.netobjectives.com* where an extensive amount of online self-study is available.

ROADMAP TO SUCCESS

YOUR ROADMAP TO LEAN-AGILE SUCCESS

Growing numbers of organizations are realizing the need to become more Agile. Some are weighing the risks and benefits and seeking guidance. Others are implementing initiatives and are looking for ways to improve their return on investment.

The road to Lean-Agile success has become less risky as the early adopters have paved the way for the next generation of Lean-Agile methodologies and practices that solve the common problems, and transcend the limitations that early adopters have struggled with.

Net Objectives has been a thought leader in each of the Agile methods of the past decade. This uniquely enables us to provide the most effective approach to our clients' needs.

For more than a decade, Net Objectives has been training and facilitating large and small organizations to achieve agility.

We serve organizations at the team, management, and enterprise level with comprehensive organizational consulting, coaching, and training.

We do not promote one method as most other firms do – rather we pull from a broad knowledge base to offer an approach tailored to your situation.

UNDERTAKING YOUR TRANSITION TO AGILE

There is no one method that guarantees success at the team level. Our full assessment services will answer these questions to help you to determine which to choose.

- Do cross functional teams already exist and if not, how difficult will it be to create them?
- Are certain staff essential for multiple teams
- How many concurrent projects are teams working on at one time?
- What challenges face the organization in integration and deployment?

Throughout your transition, Net Objectives will help ensure everything is in place through appropriate Lean-Agile training:

- **Teams** are capable of delivering value quickly with high value
- **Businesses** are capable of selecting, sizing and prioritizing business capabilities to be developed
- **Management** takes responsibility for improving the value stream and removing impediments facing teams

Our coaches enable your teams with skills, and competencies to leverage the power of agility as part of your value stream. Our consultants collaborate with management, stakeholders, executives, and experts to provide insight and guidance from the organizational view.

UNDERSTANDING AGILE

The first step toward success is drawing a clear distinction between enterprise agility, and team agility. The benefits of Agile at the team level are very different than benefits at the enterprise level. The paths to success and the challenges presented are also very different.

Enterprise agility enables an organization to effectively respond at the enterprise level to changing business needs while reliably delivering business value.

Team agility is a component of that capability – a component – and not the equivalent of enterprise agility. This understanding is essential since team methods alone cannot deliver enterprise level benefits.

First generation methods made the assumption that team agility translated to the enterprise. This has been a costly simplification.

Many organizations have attempted to achieve enterprise agility simply by creating more Agile teams. This often starts well, but usually ends up being impeded by enterprise level problems that team solutions do not solve.

The next generation of Lean-Agile openly acknowledges practical truths, limitations, and organizational structures required to fulfill the needs of the entire Lean -Agile enterprise.

From assessment and planning to pilot and rollout, our goal is to facilitate your organization with custom approaches and solutions that are appropriate to your needs, structure, and goals. Let us show how the next generation of Agile can benefit your organization.

Drive from Business Value

BUSINESS-DRIVEN SOFTWARE DEVELOPMENT

Business-Driven Software Development is Net Objectives' proprietary integration of Lean-Thinking with Agile methods across the business, management and development teams to maximize the value delivered from a software development organization. This approach has a consistent track record of delivering higher quality products faster and with lower cost than other methods.

Business-Driven Software Development goes beyond the first generation of Agile methods such as Scrum and XP by viewing the entire value stream of development. Lean-Thinking enables product portfolio management, release planning and critical metrics to create a top-down vision while still promoting a bottom-up implementation.

Our approach integrates business, management and teams. Popular Agile methods, such as Scrum, tend to isolate teams from the business side and seem to have forgotten management's role altogether. These are critical aspects of all successful organizations. Here are some key elements:

- Business provides the vision and direction; properly selecting, sizing and prioritizing those products and enhancements that will maximize your investment.
- Teams self-organize and do the work; consistently delivering value quickly while reducing the risk of developing what is not needed.
- Management bridges the two; providing the right environment for successful development by creating an organizational structure that removes impediments to the production of value. This increases productivity, lowers cost and improves quality.

BECOME A LEAN-AGILE ENTERPRISE

Involve all levels. All levels of your organization will experience impacts and require change management. We help prepare executive, mid-management and the front-line with the competencies required to successfully change the culture to a Lean-Agile enterprise.

Prioritization is only half the problem. Learn how to both prioritize and size your initiatives to enable your teams to implement them quickly.

Learn to come from business need not just system capability. There is a disconnect between the business side and development side in many organizations. Learn how BDSD can bridge this gap by providing the practices for managing the flow of work.

WHY NET OBJECTIVES

While many organizations are having success with Agile methods, many more are not. Much of this is due to organizations either starting in the wrong place, such as focusing on the team when that is not the main problem, or using the wrong method, such as using Scrum or kanban because they are popular.

Net Objectives is experienced in all of the Agile team methods (Scrum, XP, Kanban) and integrates business, management and teams. This lets us help you select the right method for you.

LEARN TO DRIVE DEVELOPMENT FROM THE DELIVERY OF BUSINESS VALUE

What really matters to any organization? The delivery of value to customers. Most development organizations, both large and small, are not organized to optimize the delivery of value. By focusing the system within which your people are working and by aligning your people by giving them clear visibility into the value they are creating, any development organization can deliver far more value, lower friction, and do it with fewer acts of self-destructive heroism on the part of the teams.

THE NET OBJECTIVES TRANSFORMATION MODEL

Our approach is to start where you are and then set out a roadmap to get you to where you want to be, with concrete actionable steps to make immediate progress at a rate your people and organization can absorb. We do this by guiding executive leadership, middle management, and the teams at the working surface. The coordination of all three is required to make change that will stick.

OUR EXPERTS

Net Objectives' consultants are actually a team. Some are well known thought leaders. Most of them are authors. All of them are contributors to our approach.

Al Shalloway

Alan Chedalawada

Guy Beaver

Scott Bain

Max Guernsey

Luniel de Beer

SELECTED COURSES

Executive Leadership and Management

Lean-Agile Executive Briefing

Preparing Leadership for a Lean-Agile/SAFe Transformation

Product Manager & Product Owner

Lean-Agile Product Roadmaps

PM/PO Essentials

Lean-Agile at the Team

Acceptance Test-Driven Development

Implementing Team Agility

Team Agility Coaching Certification

Lean-Agile Story Writing with Tests

Technical Agility

Advanced Software Design

Design Patterns Lab

Effective Object-Oriented Analysis and Design

Emergent Design

Sustainable Test-Driven Development

DevOps

DevOps for Leaders and Managers

DevOps Roadmap Overview

SAFe®-Related

Implementing SAFe with SPC4 Certification

Leading SAFe® 4.0

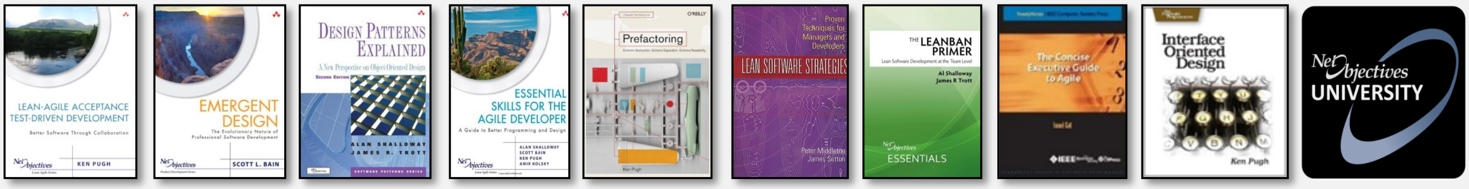
Using ATDD/BDD in the Agile Release Train (workshop)

Architecting in a SAFe Environment

Implement the Built-in Quality of SAFe

Taking Agile at Scale to the Next Level

OUR BOOKS AND RESOURCES


LEAN-AGILE ACCEPTANCE TEST-DRIVEN DEVELOPMENT
KEN PUGH
EMERGENT DESIGN
SCOTT L. BAIN
DESIGN PATTERNS EXPLAINED
ALAN SHALLOWAY
JAMES R. TROTT
ESSENTIAL SKILLS FOR THE AGILE DEVELOPER
Prefactoring
LEAN SOFTWARE STRATEGIES
THE LEANBAN PRIMER
ESSENTIALS
The Concise Executive Guide to Agile
Interface Oriented Design
Ken Pugh
Net Objectives UNIVERSITY

CONTACT US
info@netobjectives.com
1.888.LEAN-244 (1.888.532.6244)

LEARN MORE
www.NetObjectives.com
portal.NetObjectives.com