Appendix B: A Model of Lean-Agile Software Development

As far as the laws of mathematics refer to reality, they are not certain; and as far as they are certain, they do not refer to reality. —Albert Einstein

We have always found it useful to create a model of our thinking. This both sharpens our understanding of things as well as gives us frequent opportunities to check its validity. While one should not work off a model as truth without understanding it, very often, the extra awareness that a model brings is useful—it allows us to take advantage of our intuitive knowledge by bringing it to the surface. As with all models, this one will change. Please refer to the book's web-site (www.netobjectives.com/lasd) to get the latest version.

Something as complex as Lean thinking has many different ways to view it and each of these may have several levels of understanding. These perspectives are not truly orthogonal, but rather build on each other. Figure B.1 illustrates how practices build on knowledge, which builds on attitudes, which build on perspectives and principles, which builds off foundational thinking.

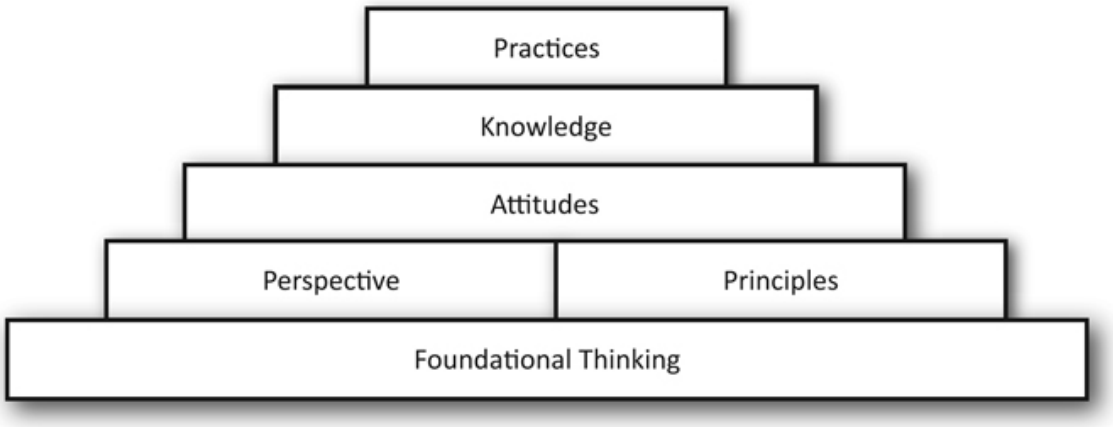


Figure B-1. The building blocks of the model

Foundational Thinking of Lean

This is the underlying belief system that Lean is based on. Much of this work comes from W. Edwards Deming.

- Most errors are systemic in nature.
- People are basically good and want to do a good job (e.g., respect people).
- Businesses will do best when they maximize the value they give to their customers.

These create the possibility and goal of achieving continuous process improvement through a combination of people doing the work while being coached and led by management.

Perspectives

You can think of perspective as how to look at things. Perspectives in themselves don't tell you how things work. But if you don't attend to the right things you often lose a lot of power. We often see things but don't understand how important they are or we otherwise misinterpret them. The perspectives of Lean are a combination of Deming's system of profound knowledge, blended with Toyota's focus on Just In Time (JIT).

- Look at time, not resource utilization.
- Think of the development process as fast-flexible-flow.
- Lowering buffers between steps increases visibility into the process.
- The best way to eliminate waste is do not build what you don't need.
- Your process is your baseline for change.
- View impediments to flow as waste.

Principles

Principles come in two flavors—principles as laws and principles as guidance. There is obviously a relationship between these, so there is some redundancy in the listings.

Principles (Laws)

The following are what we think of as laws of development. Break these at your own peril and incur waste. That does not mean you never break them. There are times where emergencies call for action that is not optimal. But understand that a cost is being incurred.

- Shortening cycle time reduces waste and increases quality.
- You tend to get waste and lower quality when you increase the time between:
    - When you need information and when you get it
    - When you make an error and when you discover the error
- Making decisions too early increases the risk of waste.
- Excess work-in-process (WIP) increases both risk and waste.
- Impediments to flow cause waste.
- Increasing the number of concurrent projects without increasing resources working them increases the length of the projects.
- Large batches cause waste.
- Switching from one task to another such that thrashing occurs causes waste.
- Working on more than one project at a time decreases a person's efficiency.
- Ignoring risk may cause waste.
- Delivering value quickly increases ROI.

Principles (Guidance)

These are principles to follow. That is, these give us guidance. Those that are in bold-italics are the seven principles detailed in (Poppendieck and Poppendieck 2003).

- **Optimize the whole.**

  - Look to shorten cycle time from concept to consumption.
  - Do not make local improvements at the expense of total cycle time.
- **Eliminate waste.**
  - Limit work to capacity.
  - Eliminate delays waiting for people or information.
  - Eliminate the delay from making an error until detecting it.
  - Focus on eliminating the cause of errors.
  - When something impedes the team, find a way to eliminate it.
  - Have teams work on one project at a time.
- **Create knowledge.**
  - Look to the system for errors.
  - Follow the scientific method to see how to improve your process.
  - Select the most important things to work on.
  - Define your workflow as much as feasible so it can be used as the baseline for change. This also creates visibility to management.
- **Build quality in.**
  - Quality problems are often caused by delays in the work flow. Removing such delays improves quality and increases speed of delivery while decreasing cost.
- **Defer commitment.**
  - Make decisions at the appropriate time.
  - Make decisions reversible if possible.
- **Deliver Fast.**
  - Organize product enhancements into Minimal Marketable Features.
  - Follow the guidance for "eliminate waste" to remove delays.
- **Respect people.**
  - Have the people with the greatest knowledge of the problem at hand make the decisions regarding it.
  - Improve your culture by improving your management systems.
  - Set a goal of continuous process improvement where the people doing the work improve the process of the work being done

Attitudes

Attitudes are important. They help define how we look at things and whether we consider something valuable. Attitudes are a result of our belief system and affect how we look at things.

- Management is important. Management needs to set outcomes for teams, allowing the teams to figure out the best way to achieve them.
- Have a goal of delivering as much value as possible in as short a period of time.
- Focusing on removing delays by eliminating waste will raise quality and lower costs.
- Always go to root cause to find and solve impediments.
- Do not let errors go by without fixing them, or at least noting them, so the root cause can be fixed later.

Knowledge

This is knowledge based from experience. We could also call this lessons learned.

- If you have long test and fix cycles you are not testing early enough.
- If you have requirements churn you are doing too much of requirements early.
- Optimizing components of your process without attending to the whole may result in waste.
- Focusing on lower costs typically results in lower quality and longer cycle times.
- Focusing on only quality may result in longer cycle times with little value to the customer.
- Focusing on achieving speed through eliminating delays will shorten delivery time, raise quality and lower costs..
- People doing the work have a greater appreciation of it than those managing it.
- A high level of work-in-process (WIP) often indicates lots of thrashing.

Practices

There are, of course, many practices that Lean suggests doing. However, you have to be very careful about following practices. Practices make sense only within a particular context; always ensure that the practices being followed are sensible for the context at hand. Knowing practices, however, is a good starting point. Using principles to guide practices in unfamiliar contexts is often a good way to create new practices.

- Use value-stream maps to see delays.
- Manage with visual controls.
- Build software in stages (iterations).
- Do continuous process improvement.
- Move testing up to the start of the development process.
- Select stories to work on to minimize risk (note, the biggest risk is often building what you don't need).
- Use Minimum Marketable Features to manage release cycles.
- Have cross-functional teams take on projects until they are completed and then move on to another.
- Go to the “gemba” (that is, go to where the work is being done).

Just a Beginning

The model presented here is just a beginning. Lean product development is not new. As our understanding of Lean unfolds, this model will be refined. There is already much more available. We are very excited about Don Reinertsen’s book, *Principles of Product Development Flow: Second Generation Lean Product Development* (2009) in which he lays out 175 principles of Lean product development, organized into the following areas:

- Economic

- Queuing
- Variability
- Batch Size
- WIP Constraint
- Flow Control
- Fast Feedback
- Decentralization

This work presents a extensive model that Don began with his excellent book, *Managing the Design Factory* (1997), which is a must-read for any Lean product developer.

We will continue to post what we learn on the web site for this book. You can find it at www.netobjectives.com/lasd.

References

Poppendieck, Mary, and Tom Poppendieck. *Lean Software Development: An Agile Toolkit.* Boston, MA: Addison-Wesley Professional, 2003.

Reinertsen, Donald. *Managing the Design Factory: A Product Developers Toolkit.* New York: Free Press, 1997.

—. *Principles of Product Developmen Flow: Second Generatilon Lean Product Development.* New York: Celeritas Publishing, 2009.

Business-Driven Software Development™

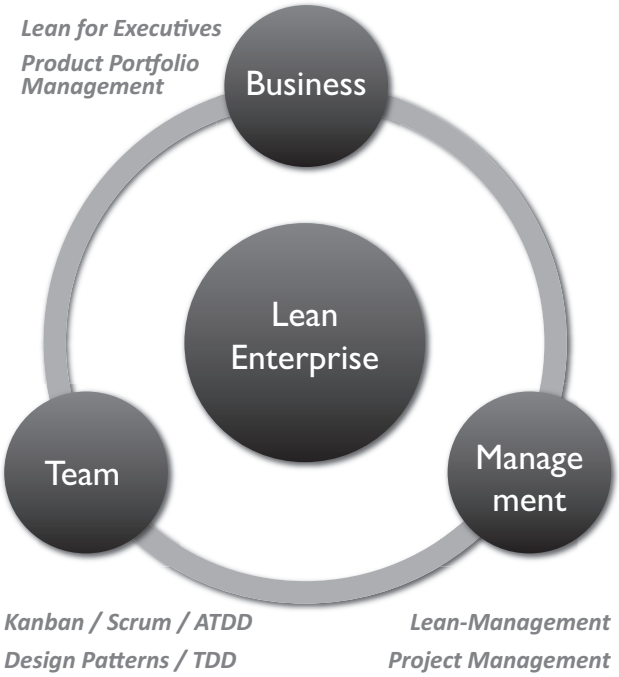
Assessments, Consulting, Training, and Coaching

Business-Driven Software Development (BDSD) 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. BDSD has built a reputation and track record of delivering higher quality products faster and with lower cost than other methods

BDSD 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.

BDSD 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. In BDSD:

- **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

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 (e.g., the team when that is not the main problem) or using the wrong method (e.g., Scrum, just because it is popular). Net Objectives is experienced in all of the Agile team methods (Scrum, XP, Kanban, Scrumban) and integrates business, management and teams. This lets us help you select the right method for you.

Assessments

See where you are, where you want to go, and how to get there.

Business and Management Training

Lean Software Development

Product Portfolio Management

Enterprise Release Planning

Productive Lean-Agile Team Training

Team training in Kanban, Scrum

Technical Training in ATDD, TDD, Design Patterns

Roles Training

Lean-Agile Project Manager

Product Owner

For more information about Business Driven Software Development Services
Contact Mike Shalloway at Net Objectives: mike.shalloway@netobjectives.com / 404-593-8375
NetObjectives.com / 1-888-LEAN-244 (888-532-6244)