An Introduction to Leanban™

A Net Objectives White Paper
TABLE OF CONTENTS

The Leanban Advantage........................................................................................................1
What Is Leanban?..................................................................................................................2
  Scrum, XP and Kanban can be considered as Partial Manifestations of Lean..................3
  Leanban as an Integrated Approach..............................................................................3
Implementing Leanban ........................................................................................................4
  Moving from Scrum to Leanban .....................................................................................4
  Moving from Kanban to Leanban ....................................................................................4
  Moving from Waterfall to Leanban .................................................................................5
Leanban Practices for Multiple Teams..............................................................................5
  Coordinate the work with backlog management ...........................................................5
  Coordinate the teams with a common cadence ..............................................................5
  Consider creating temporary cross-functional teams for the duration of the project......5
Where to Start ....................................................................................................................6
Roles in Leanban ................................................................................................................6
Leanban Practices ..............................................................................................................7
  Adopting and Abandoning Practices ..........................................................................9
  Summary of Practices .................................................................................................10
Summary ............................................................................................................................12
Appendices ........................................................................................................................12
  Artifacts in Leanban .....................................................................................................12
  Recommended Resources .............................................................................................12
  Glossary of Terms .......................................................................................................12
THE LEANBAN ADVANTAGE

Leanban is a team-level offering that makes higher level Lean-Agile tenets actionable on a day-by-day basis. It involves a number of concepts that everyone must learn.

Let’s begin with why you should care about Leanban. What advantages does it provide? Then we can turn to understanding what Leanban is.

Leanban provides a consistent approach to implementing the Minimum Business Increments that have been selected for development.

Here are some of the important advantages of Leanban.

- **Leanban is business driven.** All teams must focus on delivering value as defined and prioritized by the Business. It increases quality, reliability, and velocity.
- **Leanban suggests core practices.** Teams get started quickly by following a core set of practices and then Lean-Thinking to adjust practices and add new ones based on their needs.
- **Leanban is tailored to each team’s situation.** Each team’s situation is unique. Leanban considers the team’s current situation, including whether they are cross-functional, require iterations, and how disciplined they are.
- **Leanban provides a consistent approach across the organization.** This enables individuals to move around easily, facilitates learning across teams, and improves management understanding.
- **Leanban takes a systems-approach.** Teams deliver business value by working in concert; they must not simply focus on their own work. Leanban provides the mindset for coordination across teams.
- **Leanban provides teams a way to evolve to better practices as they learn.** Leanban provides guidance to teams to evolve as practices become inappropriate. It promotes positive change based on achieving the desired outcomes of a practice.
- **Leanban is based on proven principles.** Building on good practices improves professionalism and avoids dogma.
- **Leanban facilitates change.** Leanban supports change at a sustainable pace guided by Lean practices. It is important to determine what degree of change is appropriate without forcing change or avoiding change.
WHAT IS LEANBAN?

Scrum and eXtreme Programming (XP) were the first generation of Agile approaches. Kanban\(^1\) and the Kanban Method were the second. Each of these are consistent with subsets of Lean-Thinking; each emphasizes different Lean principles and each manifests Lean to different degrees.

Leanban is the third generation approach to Lean-Agile at the team-level. It makes higher level Lean-Agile principles actionable on a day-to-day basis. It differs from its predecessors in that from the beginning, it explicitly manifests Lean principles while also incorporating proven Agile practices. Leanban is not a mere hybrid of Scrum and Kanban. It springs directly from Lean principles and it uses Scrum and Kanban and other approaches as they help to manifest Lean. Leanban encapsulates Agile principles and practices with Lean principles and practices.

Leanban is adaptive. It solves the important dilemma of needing a consistent approach while enabling each teams to tailor their process to fit their contexts. It uses Lean-Science as an umbrella and provides a core set of practices that will work for virtually all teams. Leanban provides guidance to the teams on how to tailor other practices as required by their own, unique, situation. Because Leanban is based on a combination of principles and practices, it includes advice on how to adopt new practices when current ones become less than optimal.

Leanban can be thought of as a team-level approach based on Lean principles with integrated Scrum, XP and Kanban practices.

At the team level, Lean recommends these practices:

- Improve flow across the entire value stream.
- Collaborate to the greatest extent possible to remove delays. Create cross-functional teams as much as possible. Of course, some skills must be shared between teams.
- Work together within the team without delay. As much as possible, use colocated, cross-functional teams.
- Work on small batches to improve feedback. Decompose work into small pieces of functionality that can be validated (often called vertical slices).
- Use a shared backlog when multiple teams need to work on related features at the same time.
- Manage work-in-process to improve flow.
- Build quality in. Use test-first methods at least at the acceptance test level. Continuously improve your methods.

\(^1\) In this paper, the term, “Kanban,” refers both to “Kanban” and “the Kanban Method.” Kanban is the term used generally to mean Lean-Thinking applied to software development by managing workflow with a kanban system. The Kanban Method is a variant of this developed by David Anderson that limits itself to improvement via Kaizen. For more information, refer to the Net Objectives’ article, De-Mystifying Kanban: www.netobjectives.com/Demystifying-Kanban.pdf
Scrum, XP and Kanban can be considered as Partial Manifestations of Lean

It is worth noting how each of these approaches incorporate several aspects of Lean-Thinking. Understanding this provides credibility that a Lean-Thinking based approach is effective and provides insights into how to switch between the practices available as required by your context.

**Scrum** works because it follows core Lean principles. Examples include:

- Cross-functional teams are a cornerstone of Lean (called “work-cells” in Lean). They remove delays in that if one team member requires another, they are readily available.
- Daily standups enable people to do micro-planning to ensure that other team members will be available with little or no delay.
- Sprints limit Work-in-Process (WIP) by not allowing teams to have more work than will fit into the length of the sprint.
- The focus on being done at the end of a sprint shortens cycle time.
- The end of sprint retrospection is consistent with the Lean mantra to always be improving.

**eXtreme Programming** (XP) does the following and in fact goes even further. Examples include:

- Paired-programming limits WIP more since fewer items are in play as teammates collaborate more.
- Test-First methods directly incorporate the concepts of building quality in.
- Continuous integration manifests the idea of creating systems that help create quality products.

**Kanban** follows the Lean practices of using visual controls, managing WIP and using Kaizen to learn. Although the Kanban Method ignores team-structure, most Kanban thought leaders outside of Lean-Kanban University (the main proponents of the Kanban Method) suggest creating teams when possible.

**Leanban as an Integrated Approach**

It is not sufficient merely to try to integrate Scrum and Kanban. It requires incorporating their principles and mindsets to meet the needs of the current context. There are three reasons why this is important.

- There are some practices not in Scrum or Kanban that virtually all teams should incorporate.
- We want a single mind-set from which to select practices. In other words, all teams should be following the laws of software development as well as being management friendly.
- When selecting Agile methodologies, the best approach is tailored to meet the current context.

Leanban uses Lean-Thinking to guide teams to incorporate Agile practices into their work. Lean-Thinking reconciles the mindsets behind the various Agile methodologies: what is common, what can be incorporated from one to the other, and what is unique.

The figures below illustrate how this would apply to the three primary Agile methods. The first figure shows the practices of Scrum, Kanban and XP as commonly understood. There is not much overlap. The second figure shows the greater overlap and the distinctives that result from using Lean-Thinking.

- Whether you use estimation and velocity
- Whether you can achieve cross functional teams
• Whether you have iterations or are purely flow based
• How you start: By creating cross-functional teams and the roles of Product Owner and Team Agility Master (Lean-Scrum) or by starting where you are and looking for organizational changes that are easy to achieve (Lean-Kanban)

Start from what we have learned from Scrum, Kanban and XP and using those practices that all teams will value from. But come from Lean principles in doing so to both decide upon practices that are not universal and to help create additional practices as they are needed.

IMPLEMENTING LEANBAN
How you start with Leanban depends upon where you are. Are you currently using Scrum or Kanban? Or is Leanban going to be your first transition to Agile? Here are some considerations.

Moving from Scrum to Leanban
Moving from Scrum to Leanban is accomplished by adding the following practices:

• Create an explicit workflow within the sprint (that is, discuss how stories are to be done)
• Manage WIP within this workflow
• Implement Automated Test-Driven Development (ATDD) to some degree. At a minimum have test specifications for a story prior to its implementation.

Moving from Kanban to Leanban
Moving from Kanban to Leanban is accomplished by adding the following practices:

• Use estimation and velocity if needed
• Break work items down into small stories at the beginning of the development value stream
• Create cross-functional teams to the extent possible
• Implement Automated Test-Driven Development (ATDD) to some degree. At a minimum, have test specifications for a story prior to its implementation.
Moving from Waterfall to Leanban
Moving from Waterfall to Leanban is accomplished in this way:

- Use multi-functional teams or creating an explicit workflow across the various groups involved in product definition, construction, and delivery to the market.
- Follow the steps in “Moving from Scrum to Leanban” or “Moving from Kanban to Leanban.”

Leanban Practices for Multiple Teams
Leanban focuses on individual teams. There are times when these teams must coordinate with each other. Lean-Agile prescribes three practices that are essential for coordinating the Leanban teams and enabling quick feedback.

Coordinate the work with backlog management
It is important that teams work in a coordinated fashion. If one team builds a piece of a feature while other teams work on other segments when time comes to integrate things it is highly likely that the components will not work as well as expected. The situation now is that the teams need to collaborate but the team that built their component first is now busy on other things and the delay between when they first wrote their code and now means it will take them longer to fix anything that is wrong.

It is essential that teams work in a coordinated fashion to both improve collaboration and eliminate the delays between getting out of synch with related teams and discovering errors in understanding what is happening. One method to achieve this is to take the product backlog and have the product owners of the teams create coordinated backlogs for their teams so every team is working on related features at the same time.

Coordinate the teams with a common cadence
Teams must use the same schedule for delivering working code. Maintaining a common cadence allows the teams to discover and resolve integration errors without delay. The best cadence for integrating code from multiple teams is continuous integration. When that is not possible, a common cadence across all teams enables consistent points of synchronization.

Consider creating temporary cross-functional teams for the duration of the project
Cross-functional teams are extremely powerful. They are more efficient and are usually more effective because the resulting collaboration leads to more creative ideas. Of course, it is not always possible to create permanent cross-functional teams. It may be that members of one team are spending most all of their time supporting another team. In that case, it can be useful to assign them temporarily to the team they are supporting. This type of relationship often happens when one team is building or modifying a shared component on behalf of another. Temporarily dedicating a few members of a supporting team to join a team that they support can remove the dependency on the supporting team and allow the rest of that team to focus on its own work.
Where to Start

Here are the main questions when starting with Leanban:

- Can you create cross-functional teams?
- Do you need to plan ahead or just take things as they come?

Use the diagram to decide where to start.

ROLES IN LEANBAN

This section mentions the roles in Leanban. The following roles are required for Leanban:

- Product Owner (PO)
- Team Agility Master
- Programmer and Tester

Additional roles can include:

- Product Manager
- Specialized roles

Use whatever terms you want for these roles. For example, some companies who have a Technical Program Manager role which essentially combines the roles of Team Agility Master and Product Owner.

**Product Owner.** The Product Owner represents all customers and manages the product backlog. Responsibilities include sequencing the product and iteration backlog; assisting the team in decomposing work items into stories; ensuring that items are being worked on in the proper order; and making the business value from which the stories sprang clear to the team.

**Team Agility Master.** The Team Agility Master role combines coaching, facilitation, and focusing on continuous improvement by the team. The Team Agility Master shepherds the team, creates a trustful environment, facilitates team meetings, asks the difficult questions, removes impediments, makes issues and problems visible, keeps the process moving forward, and socializes agility within the greater organization.  

**Development Team.** The development team includes Business Analysts, programmers, and testers. There may or may not be different people filling the role of programmer and tester. The separation is due more to the fact that people already often have these roles and it may be disconcerting to change them. The roles relate more to the skills people have. Both roles are responsible for developing quality code, quickly, reliably and sustainably.

---

This goes beyond traditional Scrum Master role as defined in Scrum by combining Agile tenets with Lean Thinking. For this reason, Net Objectives prefers to call this role the “Team Agility Master.”
Here are some important concepts:

- **The Team** estimates size of backlog items, makes design and implementation decisions, commits to increments of deliverable software, and delivers it. The team tracks its own progress, is autonomous and self-organizing, dynamically adjusts its plans as needed/appropriate, and is accountable as a team for delivering as promised.
- **A Swarm** is a temporary group of team members that works together on one story to bring it to completion.
- **Testers** seek to discover why defects are happening as well as testing for defects.

**Product Manager.** This role aligns Business with development/IT. The Product Manager forms the product vision, improves ROI, manages customer and stakeholder expectations, road-mapping and release planning, prioritizes the product backlog, provides clear, testable requirements to the team, collaborates with both customer and team to ensure goals are met, and accepts product at the end of each iteration. Note: This role is probably outside of the scope of Leanban but it is good to know about.

Product Managers spend about 80% of their time focused on markets and customers and 20% on teams. Product Owners spend 80% of their time focused on teams and 20% on markets and customers.

**Specialized Roles.** Leanban recognizes that there are often specialized roles that may not be available to all of the teams. Examples include Architect, Graphic Design, and Analytics. We would prefer not to have these specializations because they often cause bottlenecks; however, they are often unavoidable or can only be avoided at exorbitant costs. The work for people filling in these roles will usually need to be managed via a personalized Kanban board.

**LEANBAN PRACTICES**

Leanban looks across software methods to discern fit-to-purpose and to apply practices thoughtfully. It does not insist on purely following one approach when that means being limited in using good practices from others. Borrowing from Scrum, Kanban, and XP, Leanban prescribes these core practices:

- **Small batches.** Work items being worked on directly are small (1-3 days). The amount of work being done across the value stream is organized to provide quick feedback. This requires that the started work items be as small as possible and focus on quick business value delivery and/or quick feedback.

  Leanban follows the mandate of focusing on the delivery of business value in small increments. The workflow starts with the definition of the Minimum Business Increment that can be built, deployed and consumed. Each MBI is broken down into vertically sliced features and added to a common backlog. The different teams pull items from the backlog, break them down into stories, and implement the stories in small batches.

  At the team level, each story typically should take no longer than three days to complete from start to finish. Stories that will be worked on by a more than a single team member can be larger than stories that will be worked on individually.
• **Self-organizing teams.** Although teams must work within the context of the overall organization, they should still self-organize their own work within that context.

• **Daily stand-ups.** The team has a stand-up meeting every day, in the same place and at the same time. All team members need to be present as well as the Team Agility Master. Who leads is at the discretion of the team. All work items in progress are quickly reviewed.

• **Focus on completion of work items not starting new ones.** Team members attempt to complete open items instead of opening up new ones. This enables the team to maintain a meaningful cadence. It also limits WIP naturally.

• **Make all work visible.** Anything done by the team is visible on the board or in a tool in a manner that the following aspects can be seen: Its status, if it is blocked, and who is working on it.

• **Include management in what the team is doing.** Take the attitude that their understanding will be helpful. It is a holdover from the early days of Agile to keep management at arms-length. Management is as much likely to be committed to the work being done as the teams.

• **Make all workflow rules explicit.** Team members follow a well-defined sequence of events by which collaborative problem-solving is attained. Everyone on the team knows what the team’s work policies are and agrees to follow them. The policies are not necessarily written down. They should be reflected on the board (virtual or physical). These policies can be changed at any time by a team-wide decision.

Explicit workflow rules help people understand their responsibilities. They also allows for trying things out and seeing if improvements can be made. They are the basis for the Lean approach, “Plan-Do-Check-Adjust,“ as well as WIP management.

People discuss the workflow not to limit their creativity but to understand what everyone is doing. In other words, discussions ensure alignment.

• **Explicitly manage WIP throughout the process.** Establish limits on the number of items that can be being worked on. WIP management can include limiting:
  - The number of projects people are working on
  - The amount of work that will be done in the iteration
  - The number of items in process at any one moment
  - The number of items allowed at any step in the workflow

• **WIP limits can be refined gradually.** You can start high and then lower them to the appropriate levels so as to remove delays in the workflow and to identify problems quickly.

• **In high-level planning, use estimation and velocity.** In a development or IT group, planning can be essential. Using estimation and velocity enables better inter-team coordination through more informed projections of when work will be completed. Estimates enable some amount of high-level planning even if they cannot be used for managing work.

• **Minimize cycle time.** In a maintenance organization that deals with urgent production bugs, minimizing cycle time is more important than estimating and measuring effort.
• **Use Acceptance Test-Driven Development (ATDD).** Acceptance Test-Driven Development is one of the best practices to be used. There are different degrees to which you can adopt it, but it should be used to some degree.³

In addition to these core practices, your situation may call for other or additional practices including:

• **Use cross-functional teams as much as possible.** Using cross-functional teams is one of the best practices available; however, they are not always easy to achieve.⁴

• **Start by creating cross-functional teams if possible.** Balance this with the reality that too much organizational change can cause problems. If you cannot create cross-functional teams or are worried about changing team structure, start with where you are and use a flow model. From this starting point you can continue to improve your teams’ organization as people become more familiar with the principles of Lean.

• **Use a common cadence with or without iterations.** You may use iterations for planning and/or for the discipline it provides. If you choose not to, you must use cadence in order to provide opportunities to synchronize with other teams and roles.

Then, use Lean to drive your particular approach:

• **Lean-XP.** Test-first; strive for TDD when you can (starting with ATDD); paired-programming; strive for continuous integration and automated testing.

• **Lean-Scrum.** Cross-functional teams if you can; use iterations to provide disciplines and a common cadence.

• **Lean-Kanban.** Create teams to the greatest extent possible; adopt a common cadence.

**Adopting and Abandoning Practices**

Few practices are universally applicable. Variations in software development needs are too large for universality to prevail. Unfortunately, most approaches don’t specify when their practices should be applied and how to abandon them and adopt new ones when necessary. People typically need guidance when beginning to adopt a new practice. This guidance should be based, at least in part, on the situation people are in. Examples include having cross-functional teams and working on too many things.

After people have started and are becoming at least competent at a beginner’s level they will want to take more control of their approach and tweak it to their specific context. This requires thinking about the objectives for the practice and then determining whether they can be realized in a different way. It is fine to move from one practice to another as long as the team honors the purposes and objectives of each. The danger of not understanding the “why” behind the “how” is that the team might abandon effective practices just because they are difficult. Naïve assumptions about practices create problems.

³ For more information, see [www.netobjectives.com/atdd](http://www.netobjectives.com/atdd) and [www.netobjectives.com/blogs/how-start-acceptance-test-driven-development](http://www.netobjectives.com/blogs/how-start-acceptance-test-driven-development)

⁴ For more detail, see [www.netobjectives.com/files/resources/articles/ValueOfTeamsAndHowToCreateThem.pdf](http://www.netobjectives.com/files/resources/articles/ValueOfTeamsAndHowToCreateThem.pdf)
For example, some teams have difficulty closing out their iterations and decide to abandon iterations and call it “Kanban.” That is not a good practice and it is not Kanban. First, the team must understand the objectives for iterations, such as:

- Small work items for better planning
- Consistent cadence
- Keeping team members in close collaboration
- Providing helpful pressure to ensure that work is finished.
- Team discipline
- Ensuring testing does not lag behind programming because stories must be completed within the iteration

If a team has abandoned iterations and their testing lags behind programming, they may benefit from adopting iterations within Leanban.

**Summary of Practices**

*Table 1. Alternative methods of getting value*

<table>
<thead>
<tr>
<th>Practice</th>
<th>Value Provided</th>
<th>Alternative Method of Getting Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Time-boxing</strong></td>
<td>Cadence for input, output, demonstration, and retrospection</td>
<td>Can have independent cadences. Must bring discipline to each story since they may take longer than should without it. Use small batches / stories. Use visual controls throughout workflow. Measure velocity via cadence. Plan ahead if valuable. Take a value-centric approach.</td>
</tr>
<tr>
<td></td>
<td>Discipline</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Small batches</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Visibility in and out</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Velocity</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Planning method</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Focus</td>
<td></td>
</tr>
<tr>
<td><strong>Cross-functional team</strong></td>
<td>Limits WIP</td>
<td>Attending to flow while using as close to a true team structure as possible can achieve these values</td>
</tr>
<tr>
<td></td>
<td>Reduces hand-offs</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Improves feedback</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Short term delays in workflow</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Improves collaboration</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Improves learning</td>
<td></td>
</tr>
<tr>
<td><strong>Product Owner</strong></td>
<td>Reduces unneeded features</td>
<td>An equivalent “one-voice” is needed regardless of method.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Finish stories quickly</strong></td>
<td>Minimizes delays</td>
<td>Time-boxes or discipline to complete stories quickly. Decompose to small stories.</td>
</tr>
<tr>
<td></td>
<td>Reduces WIP</td>
<td>INVEST</td>
</tr>
<tr>
<td></td>
<td>Provides quicker</td>
<td></td>
</tr>
<tr>
<td><strong>Remove structural impediments</strong></td>
<td>Minimizes delays</td>
<td>Manage WIP</td>
</tr>
<tr>
<td></td>
<td>Provides quicker feedback</td>
<td>Kaizen / intentional process improvements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shorten feedback cycles</td>
</tr>
<tr>
<td>Practice</td>
<td>Value Provided</td>
<td>Alternative Method of Getting Value</td>
</tr>
<tr>
<td>--------------------------</td>
<td>--------------------------------------</td>
<td>-------------------------------------</td>
</tr>
<tr>
<td>Balance workload</td>
<td>Reliable flow of value</td>
<td>Pull work based on velocity</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Manage WIP</td>
</tr>
</tbody>
</table>

**Table 2. What practices achieve**

<table>
<thead>
<tr>
<th>Practice</th>
<th>What It Achieves</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit workflow</td>
<td>Enables everyone to know what’s happening. Facilitates learning.</td>
</tr>
<tr>
<td>Daily Stand-ups</td>
<td>Keeps people informed (often not needed if collocated).</td>
</tr>
<tr>
<td>Make everything visible</td>
<td>Facilitates learning and management. Detect challenges.</td>
</tr>
<tr>
<td>Common cadence / sprints</td>
<td>Enables early synchronization of different teams.</td>
</tr>
<tr>
<td>Build incrementally and iterate on the increments</td>
<td>Short feedback cycles and learning.</td>
</tr>
<tr>
<td>Focus on finishing</td>
<td>Avoid too much WIP and look for opportunities to collaborate.</td>
</tr>
<tr>
<td>Do continuous integration</td>
<td>Detect out-of-synchronization errors.</td>
</tr>
<tr>
<td>Estimate work items and compute velocity (unless a maintenance group)</td>
<td>Validates understanding of items being worked on by the teams. Facilitates planning.</td>
</tr>
<tr>
<td>Work in small batches</td>
<td>Faster feedback. Easier to avoid workflow delays. Enables people moving around as needed.</td>
</tr>
<tr>
<td>Use small stories</td>
<td>Faster feedback. Easier to avoid delays. Enables people moving around as needed.</td>
</tr>
<tr>
<td>Manage Work-in-Process (WIP)</td>
<td>Eliminate delay and speed up feedback.</td>
</tr>
<tr>
<td>Create cross-functional teams to the extent possible</td>
<td>Eliminate delay, speed up feedback and learn faster.</td>
</tr>
<tr>
<td>Use test-first methods</td>
<td>Better understand what is needed and convey this better. Improve collaboration between dev and test. Facilitate automation of test.</td>
</tr>
<tr>
<td>Paired programming</td>
<td>Collaboration, shared knowledge of code base, and increased discipline.</td>
</tr>
</tbody>
</table>
SUMMARY
Too many Agilists are arguing for one position (Scrum) or another (Kanban). Both have advantages but both leave things out. It shouldn’t be a discussion of one or the other, but rather, what are the principles we should be driving from and what practices should we implement to achieve our goals. Leanban takes a holistic and scientific approach that enables and organization to have consistency of objectives across their teams while enabling the teams to customize their approach to their own situation.

APPENDICES
Artifacts in Leanban
Leanban uses the standard artifacts used in both Agile and Scrum. In particular:

- Capabilities, MBIs, features, stories, tasks
- A Kanban-style board (these are more effective even when doing iterations or sprints)
- A product backlog
- A iteration backlog if iterations are being used
- An impediment backlog
- Burn-down and burn-up charts as appropriate
- Cumulative Flow Diagrams
- Scatter diagrams

Recommended Resources
Here are helpful resources related to Leanban

- Resistance is not to change (blog): www.netobjectives.com/blogs/resistance-not-change
- Technical practices for the Lean-Agile Team (training): www.netobjectives.com/training/technical-practices-lean-agile-team

Here are some recommended books


Glossary of Terms
Here are a few important terms in Leanban.

Backlog a list of work to be completed that has various degrees of commitment associated with it depending upon where it is being used. There are three main backlogs in Leanban:
- **Portfolio backlog.** This is the current list of work that is projected for completion at the portfolio level. It is not committed to, but should be ordered in the expected order of completion based on value expected and cost of completion.

- **Product backlog.** This is the current list of work that is projected for the product being built. This is organized around MBIs and the work may on the product backlog may stop when the business sponsor determines that enough value has been delivered.

- **Iteration backlog.** This is the list of work committed to be done in the current iteration.

**Cadence** is the flow or rhythm of events, especially the pattern in which something is experienced. Cadence in Leanban is the rhythm of when backlogs are refreshed, work is readied for synchronization, retrospections are done, demos are presented, and teams reflect on their work.

**Cross-functional teams** are teams that have all of the capabilities to deliver the work they’ve been assigned. Team members can specialize in certain skills but as a whole, the team is capable of delivering what they’ve been called on to build.

**Iteration.** A time-boxed event (generally two-weeks) that serves as the timeframe for planning, work, demo, delivery and retrospection.

**Minimum Business Increment** (MBI) is the smallest piece of functionality that can be delivered that has value to the business in that it:

- adds value for the customers of the business
- provides valuable feedback that the right functionality is being built
- provides valuable feedback that the functionality is being built the right way
- provides functionality that can be verified as an increment that can be delivered
- enhances the ability of the organization to deliver value in the future

**Sprint** is Scrum’s term for iteration.

**Time-box** is an allocated period of time allowed for getting work done. This means that we commit to completing at a certain time, typically with a certain amount of effort (people and resources). In other words, we commit to getting as much work done as possible within this timeframe.
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

Although some of our consultants are well known thought leaders, Net Objectives’ consultants are actually a team. Most of them are authors. All of them are contributors to our approach.

Our Workshops and Courses

Transformation Bootcamp  
Lean-Agile Product Management  
Leanban  
Lean-Agile Executive Leadership  
Lean-Agile Middle Management  
Lean-Agile Team Leadership  
Acceptance Test-Driven Development  
Sustainable Test-Driven Development  
Design Patterns for Agile Developers  
Emergent Design  
Essential Skills for the Lean-Agile Developer  
Advanced Software Design  
Leading SAFe®  
SAFe Program Consultant

Our Books and Resources

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

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