Lean Principles from "The Toyota Way" and Some Software Development Implications

Long term philosophy

1. Base your management decisions on long term philosophy, even at the expense of short term financial goals.

Understanding your company's underlying philosophy will help you make process improvements aligned with your organization.

The right process will produce the right result

2. Create continuous process flow to bring problems to the surface

Strive for a continuous flow of value across the value stream. **Value stream mapping** is a technique often used to look at the whole value stream and identify work done that isn't directly contributing to value. Tom and Mary Poppendiek, authors of several books on Lean Software Development use the phrase **"concept to cash"** to indicate the end-points of the value stream. I like to carry it further to out, starting from the identification of a customer need, to customers filling that need. For commercial software this ultimately leads to the creation of market demand, or market pull.

The Kanban board is likely a portion of the value stream used by your organization.

3. Use the "pull" system to avoid overproduction

Set limits for the number of development items you have in process. **Set limits at each process step where different people with different skills hand off work to each other.** Those downstream only pull work when they have the capacity to accept it.

Using a pull approach with limits will quickly identify groups that are understaffed and groups that are over-producing

4. Level out the workload (heijunka). (Work like the tortoise, not the hare.)

Watch for bottlenecks and buildups in the development of features or stories such as: many items with development complete, but testing not started, or analysts working late nights to finish requirements in an attempt to keep up with development. **Heijunka** (平準化) or smoothing of flow focuses on reducing waste from unevenness, or **Mura** (斑 or ムラ)

*What we call a Kanban board looks much like a **Heijunka Board** created to visualize and balance flow in manufacturing.*

5. Build a culture of stopping to fix problems, to get quality right from the first

This results in looking at the quality of work at every process step. If quality of work upstream results in poor quality downstream, "stop the line" and determine how to fix quality upstream.

In software development practices such as developer, tester, and business analyst collaboration when writing acceptance criteria helps improve requirements quality, and the downstream quality of the resulting software.

6. Standardized tasks are the foundation for continuous improvement and employee empowerment

Standardized tasks doesn't mean rigid process, rather it means a commonly understood process that benefits from everyone's continuous learning.

In an agile environment this often means the adoption of skills such as test-driven development along with the goal the everyone understands and uses the approach in development.

7. Use visual control so no problems are hidden

Toyota principles and common agile practice favor simple visual systems, and reducing reporting to as little as possible – a single sheet of paper for example. Alistair Cockburn coined the term **"information radiator"** to describe simple publicly displayed visualizations.

Visual Kanban boards are an excellent example of keeping process status visible in a simple way. Other agile practices such as burn-up or burn-down charts have the same effect, so long as they are publicly displayed.

8. Use only reliable, thoroughly tested technology that serves your people and processes.

Technology supports people, not replaces them. New, unproven, or unreliable technology threatens flow.

Technology such as continuous integration and automated testing is generally reliable and improves flow. Some approaches to automated acceptance testing result in lots of "false positives" or failures of tests due to minor changes in code or data. Many teams spend way too much time fiddling with technology that promises efficiency but actually results in serious loss of flow.

Add value to the organization by developing your people and partner

9. Grow leaders who thoroughly understand the work, live the philosophy, and teach it to others.

Ideal leaders are expert at doing the work of the people they lead.

The "coach" role in Extreme Programming is an example of leader who understand the work, philosophy and has the responsibility of teaching others.

10. Develop exceptional people and teams who follow your company's philosophy.

An emphasis on individual skills, cross-functional teams, and whole team improvement is a Toyota principle and common to agile approaches.

An interesting difference is the notion that agile values and philosophy are often used to supplement most organizations' philosophy.

11. Respect your extended network of partners and suppliers by challenging them and helping them improve.

At its best, the collaborative posture agile development strikes with customers, users, and software development teams seeks to build strong partnerships and educate all parties.

Continuously solving root problems drives organizational learning

12. Go and see for yourself to thoroughly understand the situation (Genchi Genbutsu, 現地現物);

Collocated teams place cross functional teams together where through reflection they're able to diagnose and solve many of their own problems. The rise of the "agile coach" is an example of skilled individuals going to where the work is happening (**Gemba 現場**, or the scene of the crime) to observe people, and help solve problems. In Toyota, managers are expected to be on the floor with their teams to observe and help solve problems.

When we carry this principle out to the choices we make about what software to build, gemba refers to the place where our customers and users will use our product. A good Scrum product owner will spend lots of time face-to-face with their customers and users learning and observing in order to help build software that solves their problems.

13. Make decisions slowly by consensus, thoroughly considering all options (Nemawashi, 根回し); implement decisions rapidly

The practice of whole team reflection sessions used in agile development is an example of making decisions through consensus.

We likely have a bit more to learn from Toyota on identifying the root causes of problems, and considering many options before proceeding with a solution. Toyota's A3 thinking process for problem solving is often recommended by agile practitioners: http://jeffsutherland.com/scrum/2009/04/scrum-and-a3-process-game-over-for.html

14. Become a learning organization through relentless reflection (Hansei, 反省) and continuous improvement (Kaizen, 改善)

Certainly process reflection and continuous improvement are parts of modern agile development processes.

Toyota's attention to self-reflection, hensei, is a more personal manifestation. Hensei refers to acknowledging your own mistakes and pledging to improve.

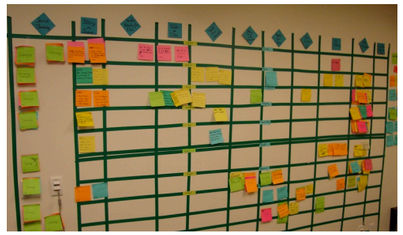
Kanban Board Basics

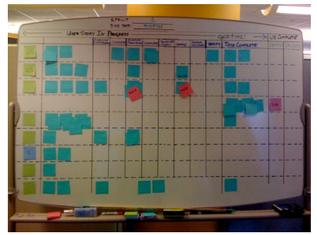
Using Kanban Boards to Visualize Flow and Apply Lean Thinking to Software Development

Kanban boards are a simple way to visualize the flow of a software development process. A kanban board looks like a task-board now commonly used in agile software development, but with one important difference: the limiting of work in progress, or more simply, the number of items that can be worked on in any one process stage.

Why: The ultimate goals for limiting work in progress is to provide **quicker detection of bottlenecks** and mismatches in the rates different groups can supply and consume output to each other. Resolving bottlenecks results in a **smoother flow** and a **greater throughput** of work.

Kanban boards are one simple tool that springs from applying Lean thinking and principles to software development.





Kanban History

A Google Japan search for "kanban" simply returns signs, many of them silly.

Kanban (in kanji 看板 also in katakana カンバン, where kan, 看 / カン, means "visual," and ban, 板 / バン, means "card" or "board") is a concept related to lean and just-in-time (JIT) production. The Japanese word kanban (pronounced [kambaɴ]) is a common term meaning "**signboard**" or "**billboard**". According to Taiichi Ohno, the man credited with developing JIT, kanban is a means through which JIT is achieved.

Kanban is a signaling system to trigger action. As its name suggests, kanban historically uses cards to signal the need for an item. However, other devices such as plastic markers (kanban squares) or balls (often golf balls) or an empty part-transport trolley or floor location can also be used to trigger the movement, production, or supply of a unit in a factory.

In the late 1940s, Toyota was studying supermarkets with a view to applying some of their management techniques to their work. This interest came about because in a supermarket the customer can get what is needed at the time needed in the amount needed. The supermarket only stocks what it believes it will sell and the customers only take what they need because future supply is assured. This led Toyota to view earlier processes, to that in focus, as a kind of store. The process goes to this store to get its needed components and the store then replenishes those components. It is the rate of this replenishment, which is controlled by kanban that gives the permission to produce. In 1953, Toyota applied this logic in their main plant machine shop

A kanban card used to resupply parts in a Toyota manufacturing facility.

Stickies, Index Cards, & Tape

Regular sticky notes work well on whiteboards. But, when using regular walls, or rougher surfaces, try 3M Super Stickies. Index cards and tape, or stick pins work well too.

Different colored stickies are useful to distinguish tasks from stories or features. You could use different colored stickies or cards for bugs, or production issues.

Write with felt-tipped markers so items are easy to read by a group standing around the board.

Painter's tape works well to line your Kanban board.

Use easy to get lo-fi tools to make it easy and fun.

Setting Up a Kanban Board

The Kanban board helps you visualize the flow of work in progress. Setting up a Kanban board can be a great way to understand your process today.

Who: For small teams, less than 10 people, the entire team should participate. For larger teams, senior team members should create the board, then review the board with the entire team.

At its simplest, Kanban boards visualize the process you have today. If you're new to Kanban, it's a bad idea to use the creation of the Kanban board as an excuse to change your process. Start by visualizing your current process.

1 Describe Your Current Process Flow

Together as a team discuss the process flow a regular development item goes through between when it comes to your team, and when it's considered done by your team.

A development item may be a user story, a feature, a bug, or anything else your team might work on.

Your process flow can be composed of any steps your team believes would be valuable to visualize. It's common to separate development and testing into separate steps. Some teams prefer to add steps for writing acceptance criteria before development, and performing story acceptance after testing.

Map your process flow, not your job roles. The same job role may act on work in different process steps such as analysts writing acceptance criteria and later evaluating reacceptance.

2 Create a Kanban Board

Find a large visible wall or whiteboard to create the Kanban board. Painter's tape works well to create columns for process steps, and rows for items in progress.

Find a visible place where the whole team can see the board and meet at the board during daily standup meetings. Flow isn't visible if no on sees it.

Buffers

Between in progress columns such as "in development" or "in test" place a buffer column. Buffer columns are the place completed work is placed, and a place where work can be pulled from if that same work moves into the next track.

Expedite Track

Create a row for urgent work, work that must be started immediately and completed as soon as possible. Create this row above or below the regular Kanban board rows. As a team decide what your agreement will be with each other, and with those placing urgent items onto the board. It's common to agree that when an item is placed in the expedite track that all other work stops until that item is completed.

Task Decomposition

It's common to decompose work items such as user stories into development tasks. Instead of a "story in progress" column create three columns: tasks, tasks in progress, and tasks complete. These three columns visualize the flow of tasks being worked on for the user story.

Many teams find it valuable to create separate columns for testing tasks in progress and complete. Healthy teams allow testers and developers to work simultaneously.

*The Kanban board makes **your** process visible, it doesn't change it. If testers normally work while developers work, construct your board so that you can visualize it. Don't assume that separate columns means one person must finish work before another starts.*

3 Set Work In Progress Limits

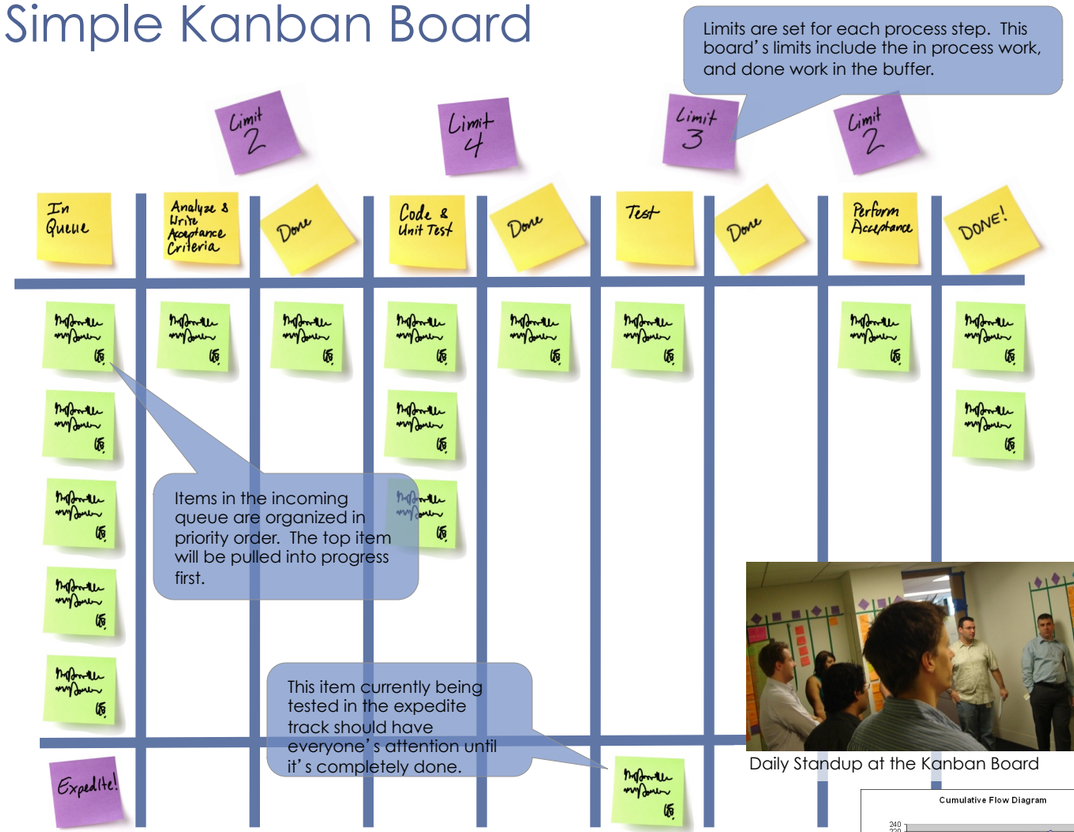
For each process step, set a limit to the number of items you'll allow in progress.

For each process step consider how many individual people are capable of working on an item in the process step. An appropriate work in progress limit might be exactly that number of people. For teams where developers work in pairs, limits are often set to the number of pairs. You may choose to increase limits a bit to start, especially if you're new to Kanban. A good upper limit is 1.5 * the number of people that can perform that process step.

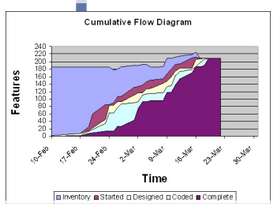
For Kanban boards where large items such as user stories or features use task decomposition to show work in progress, set a limit to the number of large items that can be in progress. A small number such as two to four is a good starting point. Use the method above to set WIP limits for tasks in progress.

Setting limits initially can seem tricky. Don't worry too much about getting them correct. You can and should adjust them as you go.

Simple Kanban Board

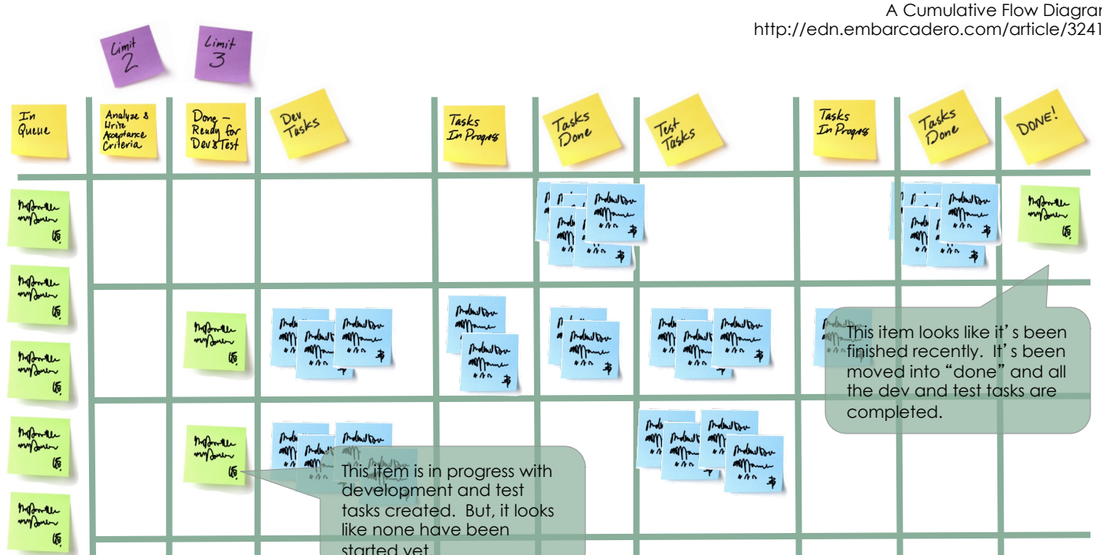


Daily Standup at the Kanban Board



A Cumulative Flow Diagram
http://edn.embarcadero.com/article/32410

Kanban Board With Dev & Test Tasks



Using the Board

Look to the Kanban board for an accurate representation of where work in progress is. Team members use the Kanban board to signal each other that work is ready to pull. If work is bottlenecking in any one process step, the Kanban board should make the bottleneck visible and give the team an opportunity to address it.

1 Place Development Items In Queue

Product owners, agile customers, or those upstream responsible for selecting and prioritizing work should add items into the queue on the left side of the board. They'll manage priority of those items in the queue. As an item is added into the queue, note the date it is added directly on the sticky. This is the first date we'll use later to measure cycle time. Knowing this date lets us measure wait time in the queue.

Posting current product goals helps to avoid "thrash" – constant reprioritization that comes from unknown or changing goals.

2 Pull Work From Column to Column

Work is pulled into work-in-progress columns from buffer columns to the left of the Kanban board. A team member ready to work can pull work so long as placing the work into progress won't break work-in-progress limits.

When work is pulled from the left-most waiting queue, record the date pulled on the sticky. This will help us measure cycle time from the point the item enters work-in-progress.

When items are placed into the final column on the right indicating they're completely done, record the date completed there. This marks the date the item is completed and allows us measure cycle time for this item.

If work isn't available to pull, it's a sign that there may be a bottleneck upstream that needs resolving. See what you can do to help get things moving.

3 Standup Daily

As with other agile processes, keep a daily standup meeting, or daily Scrum. Meet in front of the Kanban board where it's easy to point out what you're working on and move stickies into progress.

Team members responsible for downstream work often speak first in the standup pulling work they're starting on today which may free up capacity for others upstream to pull work.

4 Measure and Report

On a regular basis record items' start and finish dates in a record-keeping tool such as a spreadsheet. Use the spreadsheet to calculate average cycle time for work in progress, and if you've recorded the date items entered the waiting queue, the average wait time. Post average cycle times.

If development items were estimated, you can calculate average cycle times for items of similar size as well as overall average cycle time.

If development items are part of an overall release plan, it's easy to track the number of items finished within the release. If you're using an agile approach where you maintain a burn-down, or burn-up chart, update the chart as you would normally. You'll still be able to determine if you're on track for your release date.

Although it takes a bit more effort, on a daily basis record the amount of work currently in progress in each process step in order to create a cumulative flow diagram. While the Kanban board shows work in progress on any given day, the cumulative flow diagram shows flow and cycle time as it changes over time. Use this chart to spot trends or correlate build-ups of work-in-progress with other events within the organization.

In addition to metrics you might normally gather, Kanban metrics emphasize throughput, and help visualize flow.

5 Reflect and Adapt on a Regular Cycle

If you've been using a typical agile process, you may be accustomed to stopping every 1-3 weeks for a routing product demonstration, and retrospective. Continue this practice demonstrating features complete, discussing the work completed and rate of completion, and then reflecting on your process and making sensible process changes. If you've stopped using iteration or sprint planning, it still works to count the number of stories or development items completed to use as a measurement of velocity over that time period.

Routine process reflection and change is critical. Adopting a Kanban approach doesn't mean you'll stop that practice.

MVP, MMF, and User Stories

In Lean thinking we're striving to increase the flow of value out to customers and users. However, we're also striving to build small amounts of software that allow us to iteratively design and improve, and incrementally grow the software.

- **User Stories** are commonly loaded onto the left edge of a Kanban board. While each story results in some valuable working software, by itself, it may not be releasable.
- **Minimal Marketable Features** (MMFs) are single features that can be added to an existing product and when released will be considered by users to be a valuable new addition to the product.
- **Minimal Viable Products** (MVP) are what we're striving to get to as soon as possible if we don't already have a product in the hands of users we can add to.

The size of the development item loaded on the left is a function of how your organization plans and constructs software.