CSC 171 – Module 01

Why Agile and Lean Approaches Work

Traditional Development Data

- Standish Group Chaos Report (1995)
  - Successful projects: 16.2%
    - The project is completed on-time and on-budget, with all features and functions as initially specified.
  - Challenged projects: 52.7%
    - The project is completed and operational but over-budget, over the time estimate, and offers fewer features and functions than originally specified.
  - Impaired projects: 31.1%
    - The project is canceled at some point during the development cycle.

Standish Group Chaos Report FY2011–2015

Size	Method	Successful	Challenged	Failed
All Size Projects	Agile	39%	52%	9%
	Waterfall	11%	60%	29%
Large Size Projects	Agile	18%	59%	23%
	Waterfall	3%	55%	42%
Medium Size Projects	Agile	27%	62%	11%
	Waterfall	7%	68%	25%
Small Size Projects	Agile	58%	38%	4%
	Waterfall	44%	45%	11%

Traditional definition of success: OnTime, OnBudget, and OnTarget
Modern definition of success: OnTime, OnBudget, with a satisfactory result

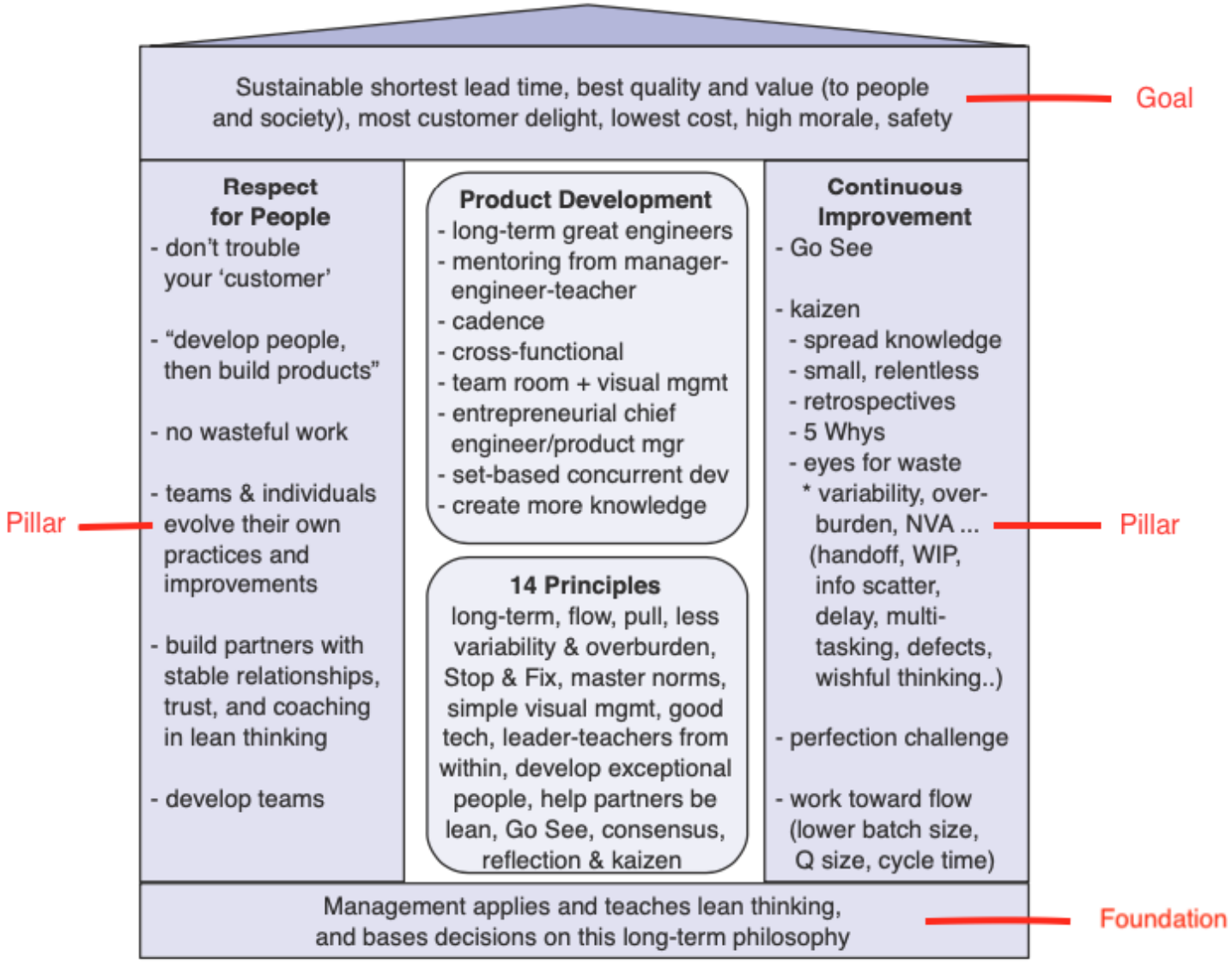
Traditional Development Approaches

- Sequential development
  - Assumes that everything can be predicted
  - Work can be handed off from one person to another and it would all "just work"
  - Only encourages one delivery, at the end
- Realities of software projects
  - We cannot predict everything about the project, we need to adapt quickly to changing realities
  - There are no one-person projects and there is need for interaction and collaboration
  - Software development is an "empirical process", which means assessment and learning is needed as development proceeds
    - Sequential development approaches do not provide enough learning opportunities

Lean Thinking

- Lean (or lean thinking) is the English name—popularized by MIT researchers—to describe the system now known as the Toyota Way inside the company that created it. [2]
- Two Pillars of Lean
  1. Respect for people (including employees, supply partners, customers, …)
  2. Continuous improvement
- Lean principles [2]
  1. Base management decisions on a long-term philosophy, even at the expense of short-term financial goals.
  2. Move toward flow; move to ever-smaller batch sizes and cycle times to deliver value fast & expose weakness.
  3. Use pull systems; decide as late as possible.
  4. Level the work—reduce variability and over- burden to remove unevenness.
  5. Build a culture of stopping and fixing problems; teach everyone to methodically study problems.
  6. Master norms (practices) to enable kaizen and employee empowerment.
  7. Use simple visual management to reveal problems and coordinate.
  8. Use only well-tested technology that serves your people and process.
  9. Grow leaders from within who thoroughly understand the work, live the philosophy, and teach it to others.
  10. Develop exceptional people and teams who follow your company's philosophy.
  11. Respect your extended network of partners by challenging them to grow and helping them improve.
  12. Go see for yourself at the real place of work to really understand the situation and help.
  13. Make decisions slowly by consensus, thoroughly considering options; implement rapidly.
  14. Become and sustain a learning organization through relentless reflection and kaizen.

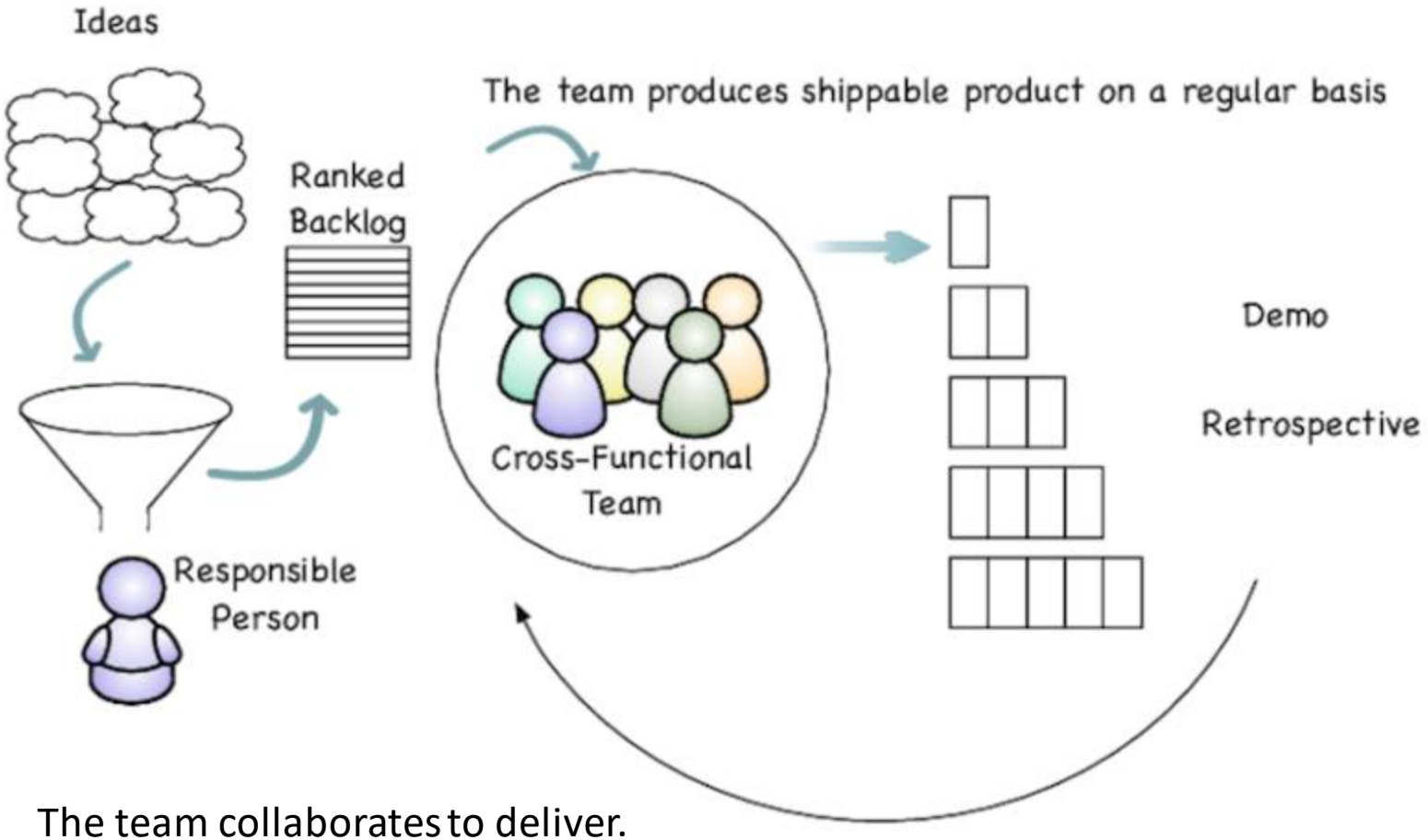
The Lean Thinking House [2]



Agile Software Development

- Manifesto
  - Individuals and interactions over processes and tools
  - Working software over comprehensive documentation
  - Customer collaboration over contract negotiation
  - Responding to change over following a plan
- Principles
- Agile software development methods
  - Scrum, Kanban, Scrumban, lean software development, extreme programming (XP), feature-driven development (FDD), dynamic systems development method (DSDM), …
- Agile software development practices
  - Backlogs, continuous integration, cross-functional team, daily stand-up, iterative and incremental development, pair programming, planning poker, refactoring, retrospective, test-driven development (TDD), timeboxing, user story, …

Agile Development Approaches



- The team collaborates to deliver.
- The team limits its work in progress (WIP).
- The team delivers releasable product often.
- The team reviews its work product and process on a regular basis.

Scrum

- Sprint
  - Time-boxed iteration, usually 2 to 4 weeks
  - Each sprint includes planning, analysis, design, coding, testing, and getting feedback
- Artifacts
  - Product backlog
    - Ordered list of ideas for the product, typically begins short and vague, and becomes longer and more defined as time goes on
  - Sprint backlog
    - The list of product backlog items chosen for development in the current sprint and plans for accomplishing the work
  - Product increment
    - Be of high enough quality to be given to users, be acceptable to the product owner, meets the team's definition of done
- The scrum team
  - Product owner
    - Collaborates with stakeholders and team to clarify what to build
  - Development team
    - Delivers product increment, self-organizing, cross-functional, normally 3 to 9 people
  - Scrum master
    - Coaches the scrum team and organization, helps team to identify and remove impediments, …
- Activities
  - Product backlog refinement (grooming), sprint planning, daily scrum, sprint review, sprint retrospective

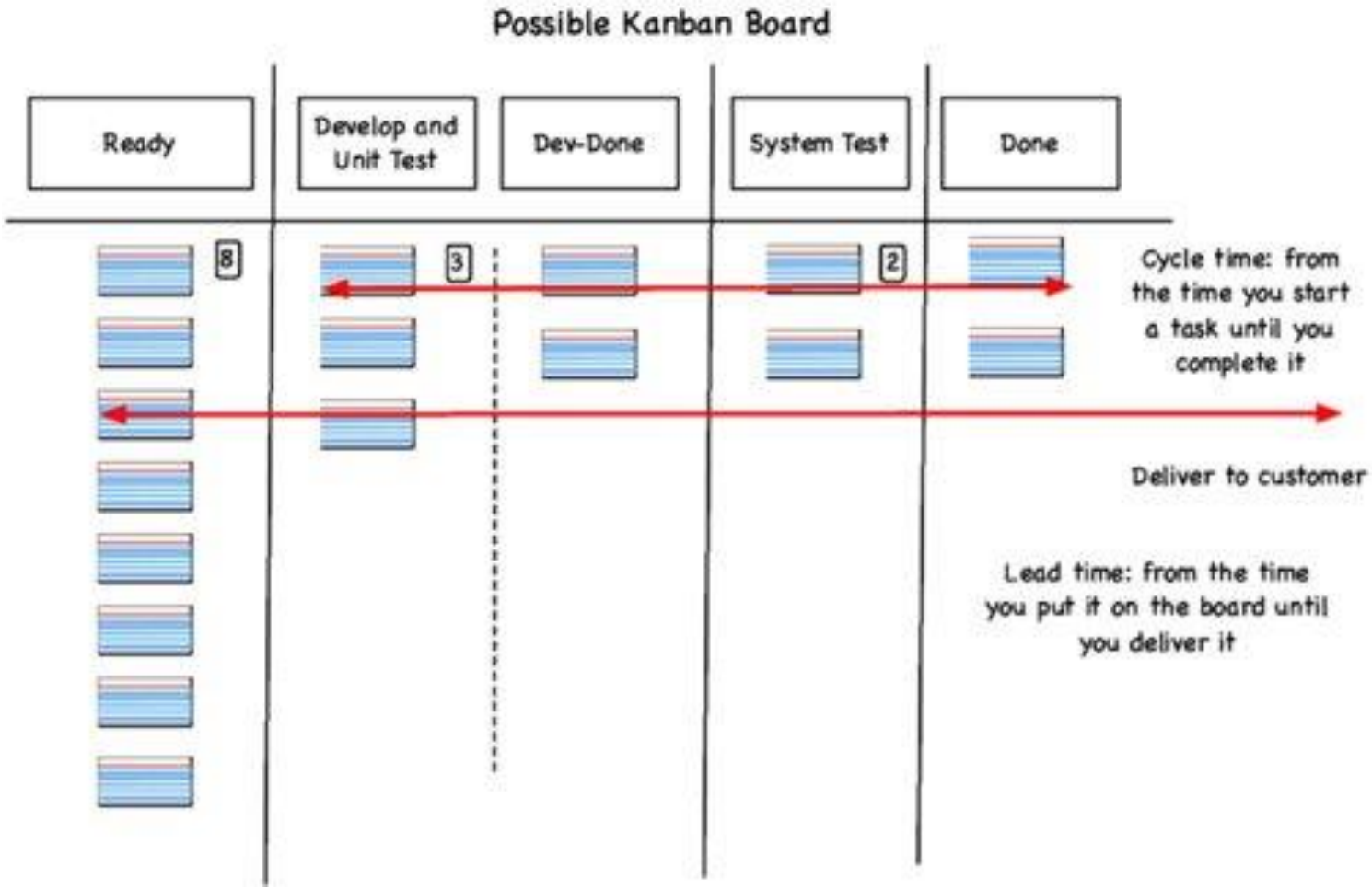
Scrum Activities

Activity	Who	What	When
Sprint planning	Whole scrum team	Select and understand the work to be done in the next sprint	At the beginning of each sprint
Daily scrum	Whole scrum team	Ensure team is on track for that sprint	Every day
Sprint review	Whole scrum team and interested stakeholders	Review the resulting product increment	End of each sprint
Sprint retrospective	Whole scrum team	Review and improve the process	End of each sprint
Product backlog refinement (grooming)	Whole scrum team	Prioritize and estimate product backlog items, split large items into smaller ones, merge smaller items, …	Ongoing

Kanban

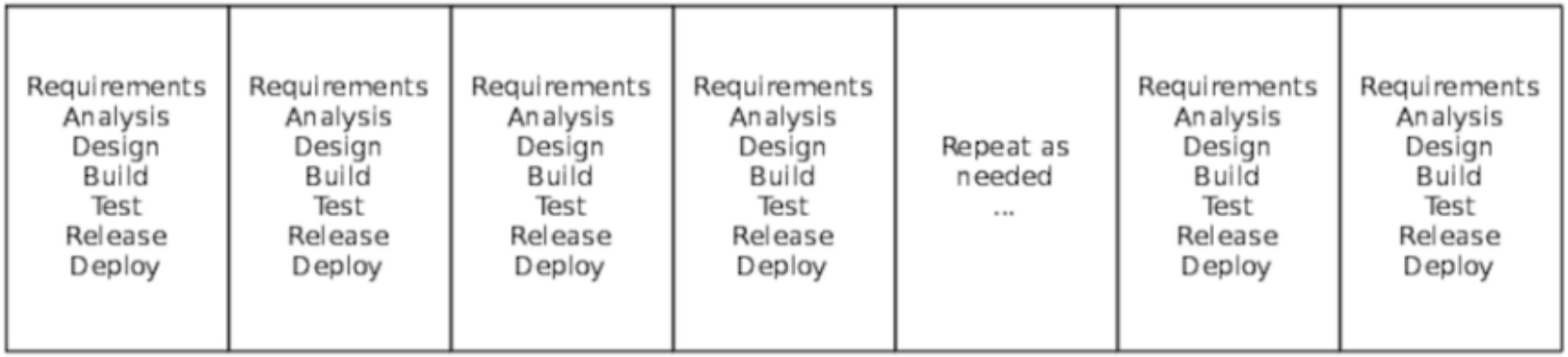
- Is focused on process improvement, and is based on Lean values and Lean thinking
- Core practices
  - Visualize work
  - Limit work in progress (WIP)
  - Make policies explicit
  - Manage flow
  - Implement feedback loops
  - Improve collaboratively, evolve experimentally
- Kanban board
  - Kanban card, a visual representation of a work item, can include name, description, due date, assignee, tasks, ...
  - Kanban column, represent a workflow stage, workflow can be mapped to meet the unique process of any team
  - WIP limit, the maximum amount of cards (work items) can be put in each column (workflow stage)
  - An example

Kanban – Cont.


Possible Kanban Board
Ready
Develop and Unit Test
Dev-Done
System Test
Done
8
3
2
Cycle time: from the time you start a task until you complete it
Deliver to customer
Lead time: from the time you put it on the board until you deliver it

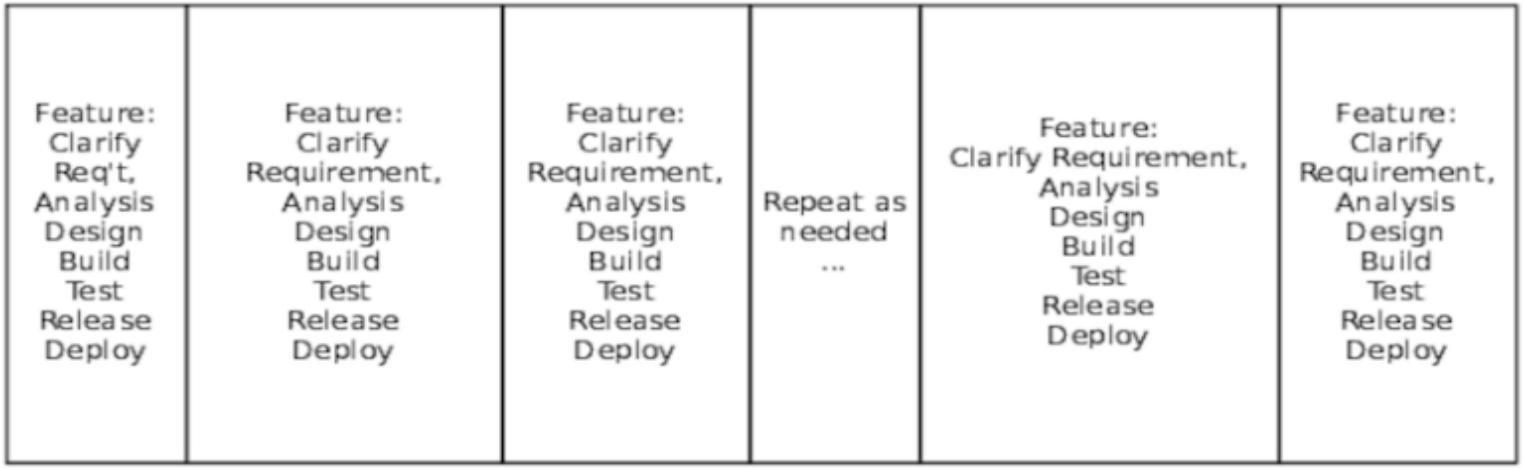
Iteration-Based Agile Approach

- A team works in timeboxes of the same size for every iteration
- Each timebox
  - Is the same size
  - Results in running tested features
  - Includes these activities: requirements, analysis, design, build, test, release, and deploy.
- The product owner and the team manage the work in progress by estimating the number of stories (and other work) the team can commit to in a timebox
- Key metrics
  - Velocity: the amount of work a team can complete in an iteration
- Example
  - Scrum



Flow-Based Agile Approach

- Focuses on the continual pulling of work, there is no timebox
- The team limits the number of features active at any time with WIP limits for each activity
- Key metrics
  - Cycle time: how long each feature take on average
- After finishing some work, the team delivers and learns, retrospectives, and reviews what it wants to improve
- Example
  - Kanban



References

- [1] Create Your Successful Agile Project, Johanna Rothman, Pragmatic Programmers LLC, 2017. ISBN:9781680502602
- [2] Lean Primer, https://www.leanprimer.com/downloads/lean_primer.pdf