Software Process Models

INSTRUCTOR: MEHROZE KHAN

What is Agile ?

- An iterative approach to project management that helps teams be responsive and deliver value to their customers faster.

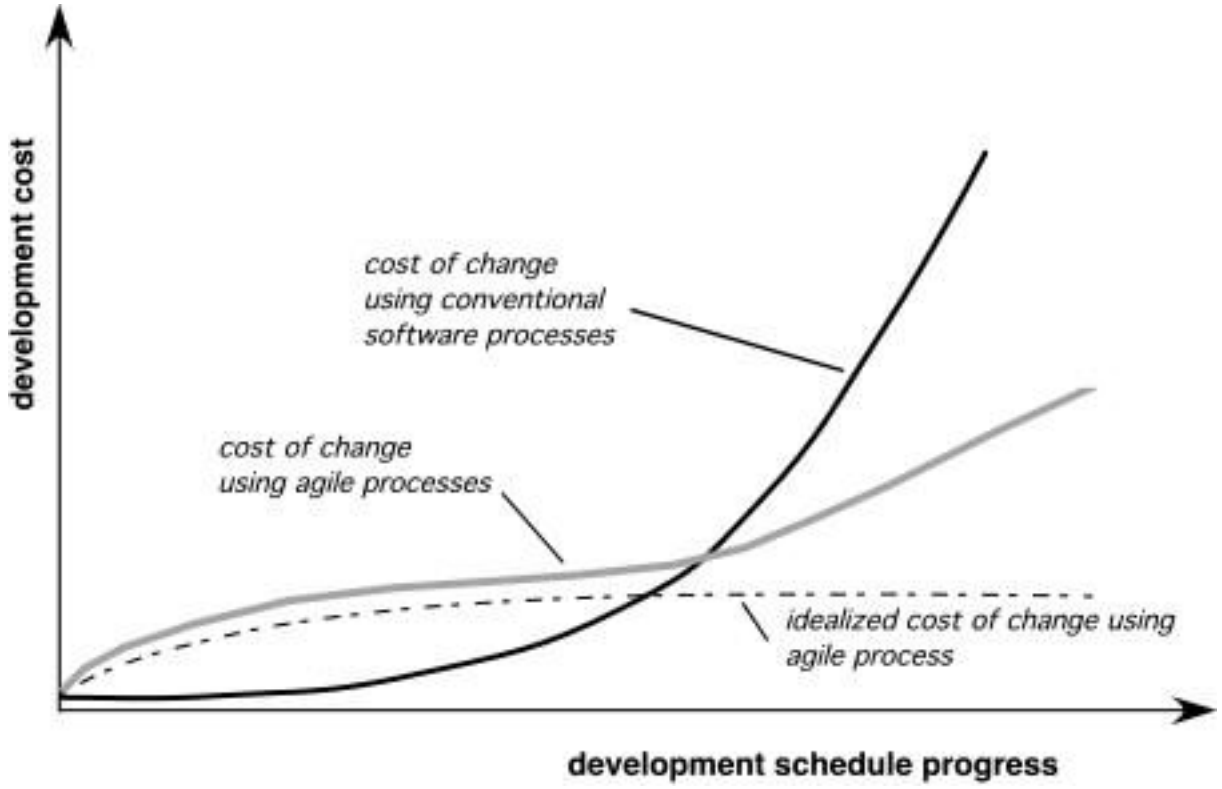
Key Characteristics of Agile Development

- Adaptive Planning
- Evolutionary Development
- Early Delivery
- Continual Improvement
- Responsiveness to change

The Manifesto for Agile Software Development

- Agile manifesto
  - Concentrate on responding to change rather than on creating a plan and then following it.
  - Value individuals and interactions over process and tools
  - Prefer to invest time in producing working software rather than in producing comprehensive documentation
  - Focus on customer collaboration rather than contract negotiation

Agility and the Cost of Change



Human Factors

- Key traits must exist among the people on an agile team and the team itself:
  - Competence.
  - Common focus.
  - Collaboration.
  - Decision-making ability.
  - Fuzzy problem-solving ability.
  - Mutual trust and respect.
  - Self-organization.

Extreme Programming (XP)

- The most widely used agile process, originally proposed by Kent Beck

1. **XP Planning**
   - Begins with the creation of **"user stories"**
   - Each *user story* is written by the customer and is placed on an index card. The customer assigns a ***value*** (i.e., a priority) to the story based on the overall business value of the feature or function.
   - Agile team assesses each story and assigns a **cost**(measured in development weeks)
   - Stories are grouped to form a **deliverable increment**
   - A **commitment** is made on delivery date
   - After the first increment **"project velocity"** (number of customer stories implemented during the first release) is used to help define subsequent delivery dates for other increments

Extreme Programming (XP)

2. **XP Design**
   - Follows the **KIS principle**
   - Encourage the use of **CRC cards**. CRC (class-responsibility collaborator) cards; identify and organize the object-oriented classes that are relevant to the current software increment.
   - For difficult design problems, suggests the creation of **"spike solutions"**—a design prototype.
   - **Refactoring**—modifying/optimizing the code in a way that does not change the external behavior of the software

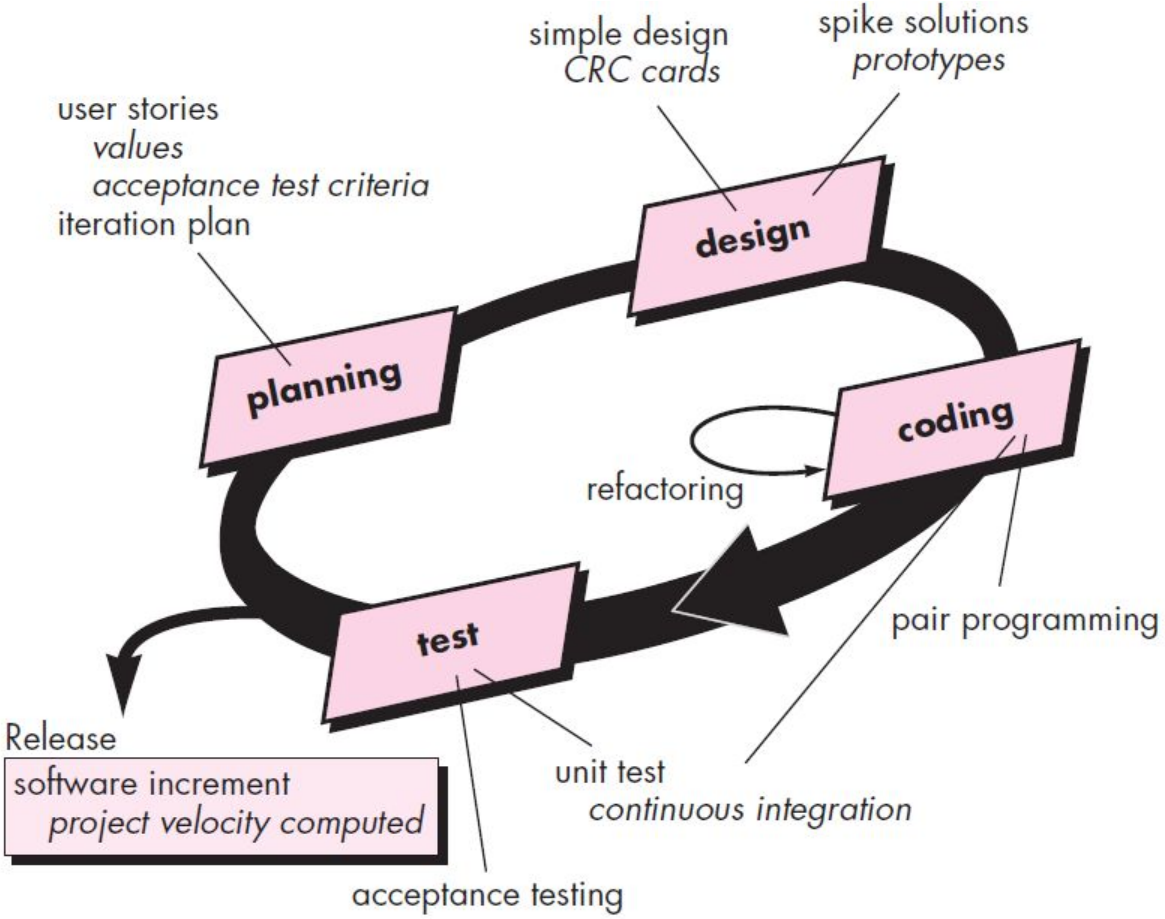
Extreme Programming (XP)

3. **XP Coding**
   - Recommends the construction of **unit tests** for each user story before coding commences.
   - Once the code is complete, it can be unit-tested immediately, thereby providing instantaneous feedback to the developers.
   - Encourages **"pair programming"**; two people work together at one computer workstation to create code for a story. This provides a mechanism for real-time problem solving and real-time quality assurance.
   - **Continuous Integration**: As pair programmers complete their work, the code they develop is integrated with the work of others.

Extreme Programming (XP)

4. **XP Testing**
   - All unit tests are executed daily
   - XP ***acceptance tests***, also called *customer tests,* are specified by the customer and focus on overall system features and functionality that are visible and reviewable by the customer.

Extreme Programming (XP)



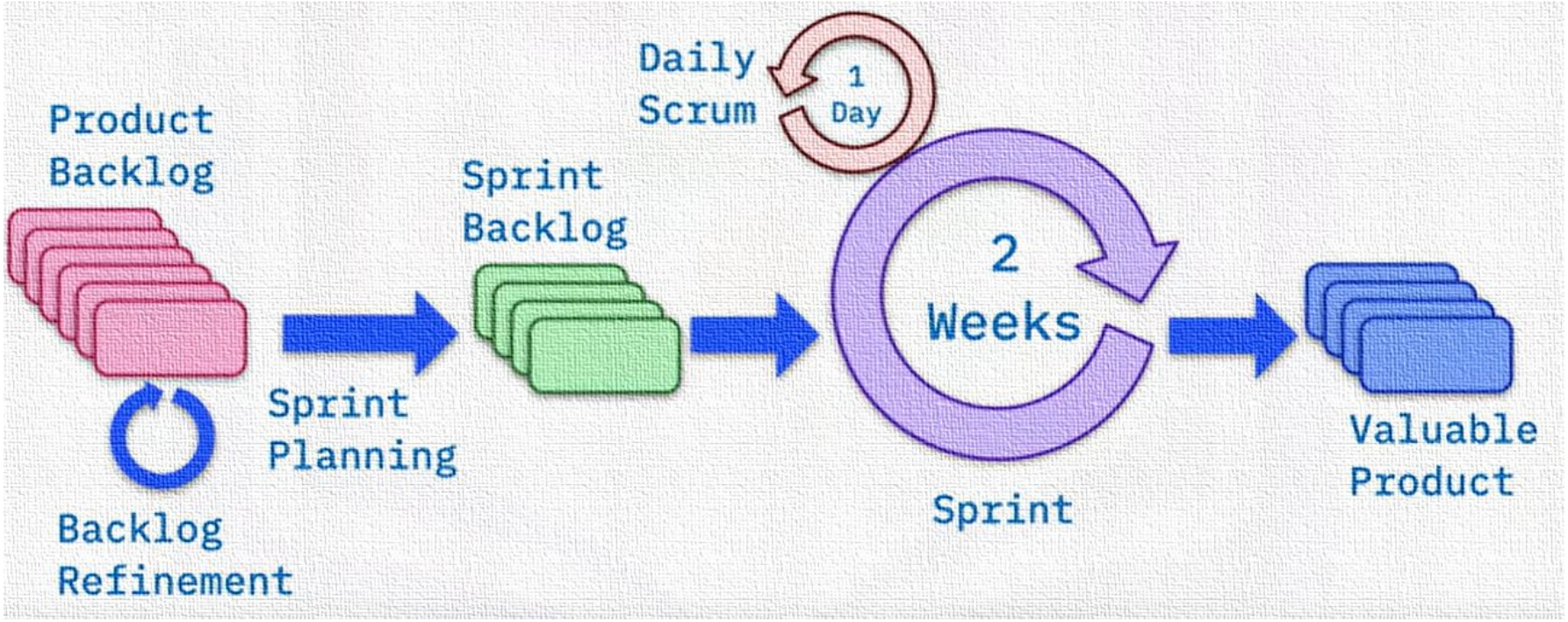
Scrum

- It is a management framework for incremental product development.
- Prescribes small, cross functional, self organizing teams.
- Provides a structure of roles, meetings, rules and artifacts.
- Uses fixed length iterations called sprints.
- Has a goal to build a potentially shippable product increment with every iteration.

Sprint

- A sprint is one iteration through design, code, test and deploy cycle.
- Every sprint should have a goal.
- Sprints are usually 2 to 4 weeks in duration.

Scrum


Daily
Scrum
1
Day
Product
Backlog
Sprint
Backlog
2
Weeks
Sprint
Planning
Valuable
Product
Backlog
Refinement
Sprint

Scrum Team

- The Scrum team is a self-organizing interdisciplinary team consisting of a:
  1. Product Owner
  2. Scrum Master
  3. Development Team

Product Owner

- Represents the stakeholder interests.
- Articulates the product vision.
- Is the final arbiter of requirements questions.
- Constantly re-prioritizes the product backlog, adjusting any expectations.
- Accepts or rejects each product increment.
- Decides whether to ship.
- Decides whether to continue development.

Scrum Master

- Facilitates the scrum process.
- Coaches the team.
- Creates an environment to allow the team to be self organizing.
- Shields the team from external interference to keep it “in the zone”.
- Helps remove obstacles identified by team members during the meeting.
- Enforces sprint timeboxes.
- Has no management authority over team.

Development Team

- It is a cross-functional team consisting of:
  - Developers
  - Testers
  - Business Analysts
  - Domain Experts
  - Others

Development Team

- Self Organizing
  - There are no externally assigned roles.
- Self Managing
  - They self assign their own work
- Negotiates commitments with the product owner
  - One sprint at a time

Scrum Artifacts

- The principle Scrum artifacts are:
  1. Product Backlog
  2. Sprint Backlog
  3. Code Increment

Scrum Artifacts

- **Product Backlog:**

  The product backlog is a prioritized list of product requirements or features that provide business value for the customer.

- **Sprint Backlog:**

  The sprint backlog is the subset of product backlog items selected by the product team to be completed as the code increment during the current active sprint.

Scrum Artifacts

- **Code Increment:**

  The increment is the union of all product backlog items completed in previous sprints and all backlog items to be completed in the current sprints.

Scrum Events

- Sprint Planning Meeting
- Daily Scrum Meeting (Daily Stand-up)
- Sprint Review
- Sprint Retrospective

Sprint Planning Meeting

- Prior to beginning, development team works with the product owner and all other stakeholders to **develop the items in the product backlog**.
- The product owner and the development team **rank the items in the product backlog** by the importance of the owner's business needs and the complexity of the software engineering tasks required to complete each of them.
- Prior to starting each sprint, the product owner states her **development goal for the increment** to be completed in the upcoming sprint.
- The Scrum master and the development team **select the items to move from product backlog to the sprint backlog**.
- The development team **determines what can be delivered** in the increment within the constraints of the time-box allocated for the sprint and, with the Scrum master, **what work will be needed to deliver** the increment.

Daily Scrum Meeting

- The daily Scrum meeting is a 15-minute event scheduled at the start of each workday to allow team members to synchronize their activities and make plans for the next 24 hours.
- Three key questions are asked and answered by all team members:
  - What did you do since the last team meeting ?
  - What obstacles are you encountering ?
  - What do you plan to accomplish by the next team meeting ?

Sprint Review Meeting

- The sprint review occurs at the end of the sprint when the development team has judged the increment complete.
- The sprint review is often time-boxed as a 4-hour meeting for a 4-week sprint.
- The Scrum master, the development team, the product owner, and selected stakeholders attend this review.
- The primary activity is a ***demo*** of the software increment completed during the sprint.
- The product owner may accept the increment as complete or not.

Sprint Retrospective

- Ideally, before beginning another sprint planning meeting, the Scrum master will schedule a 3-hour meeting (for a 4-week sprint) with the development team called a ***sprint retrospective*.**
- During this meeting, the team discusses:
  - What went well in the sprint ?
  - What could be improved ?
  - What the team will commit to improving in the next sprint ?

Project/Sprint Velocity

- By looking at the amount of work the team has completed in previous sprints, you should be able to estimate how much work they can do in future sprints.
- In Agile development, this estimate is known as **sprint velocity**.
- Sprint velocity estimate gives senior management and other stakeholders a better idea of when to expect delivery of the product.

Calculating Sprint Velocity

- In order to estimate what work can be completed in the future, you need to measure the work that has previously been done.
- To get a good average measurement of work that has been done, plan to review the previous three sprints.
- In the next example, story points are used to measure the amount of work completed in each sprint.
- A **story point** is a measurement used by Agile development teams to estimate how much effort and time it will take to complete a user story.

Calculating Sprint Velocity

Step 1: Count how many user story points are completed in each sprint

At the end of a sprint, add up how many story points the team completed.

- **For example, assume that in sprint 1:**
  - The team committed to completing five user stories.
  - Each user story had eight story points for a total of 40 story points.
  - The team completed three of the five user stories.
- **In sprint 2:**
  - The team committed to seven user stories (including the two that were not completed in sprint 1).
  - Each user story had eight story points for a total of 56 story points.
  - The team completed four of the seven user stories.

Calculating Sprint Velocity

- **In sprint 3:**
- The team committed to nine user stories.
- Each user story had eight story points for a total of 72 story points.
- The team completed five of the nine user stories.

Calculating Sprint Velocity

- **Step 2: Calculate the average of completed story points**
- Simply add up the total of story points completed from each sprint, then divide by the number of sprints.
- **Sprint 1**: 3 user stories x 8 story points = 24
  Sprint 2: 4 user stories x 8 story points = 32
  Sprint 3: 5 user stories x 8 story points = 40
  Total = 96 story points
- So, your **average sprint velocity** is 96 ÷ 3 = 32.
- You can now base the amount of work to be done in future sprints on the average of 32 story points.
- If you have 160 story points remaining to be completed in the project, you can assume that your team will need another five sprints to complete the project.

Kanban

- **Kanban** is a very popular framework for development in the agile software development methodology. It provides a transparent way of **visualizing the tasks and work capacity of a team**.
- It mainly **uses physical and digital boards** to allow the team members to visualize the current state of the project they are working on.
- Kanban originated in **Toyota** in the **1940s**. Kanban's meaning in Japanese is "**billboards**".
- The **Kanban board** has columns and story cards. The columns are nothing, but workflow states and cards are nothing but a demonstration of the actual task a team member is performing.

Kanban Card

- The Kanban cards are essential pieces on the Kanban board as it represents the work that the team is working on. These cards will have
  1. Priority
  2. Owner
  3. Type
  4. Due date

Work In Progress Limit(WIP Limit)

- A **column** in Kanban board represents the work stage, and you can place a WIP (Work in Progress) limit on the column.
- The **WIP limit** means the maximum number of cards that can stay on that column.
- Since Kanban project management uses a **pull-based** system, as and when a developer is free, he/she can pull a card from the to-do column to the dev column.

Kanban Board

- **Kanban Board** is an agile project management tool that helps implement Kanban to manage projects for personal and business purposes.
- It is a physical or digital (JIRA) board designed to **help teams visualize their work** at different stages and processes.
- It also helps **represent the stages of work** with columns using cards.
- It has columns that represent the status of the work like
  1. To-do,
  2. Dev
  3. Testing
  4. Done.

Pull Based System

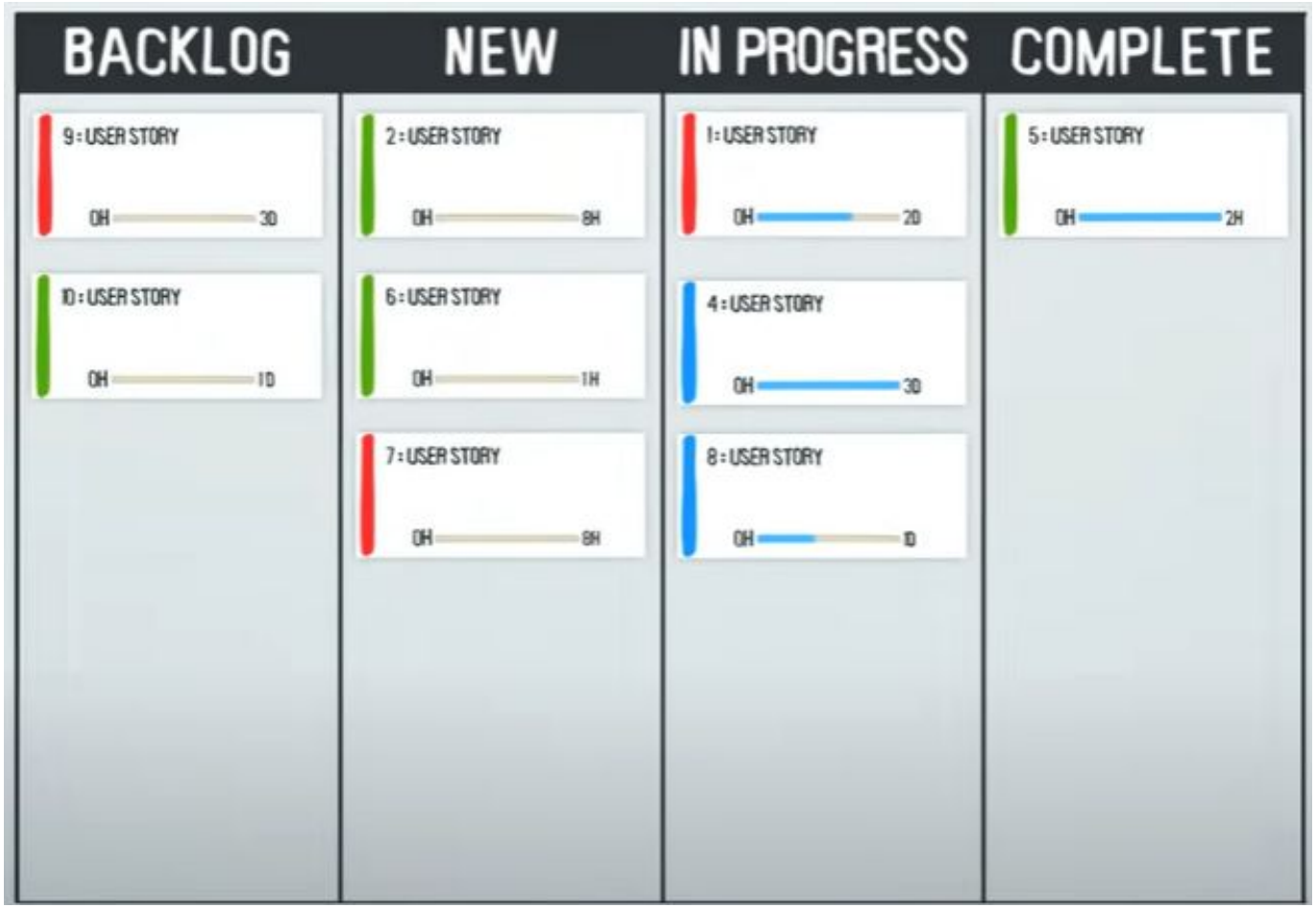
- Kanban is a pull-based method where tasks are being pulled rather being pushed. As soon as you have completed your current card, you can pull a new card from the previous column of the Kanban board.

Kanban

1. **Lead Time**: It is the duration between a new card's arrival in your workflow and its final departure from the workflow.
2. **Cycle Time**: It is a duration between the card's arrival in the working state and when the card is ready for release.
3. **WIP**: Work in progress (WIP) limits the maximum amount of work items in the different stages of the workflow.
4. **Throughput**: It is the actual performance, and it tells the actual number of cards delivered in a given timeframe.

Throughput = WIP/Cycle Time

Kanban Board



Kanban Board

Kanban Board

Kanban Board