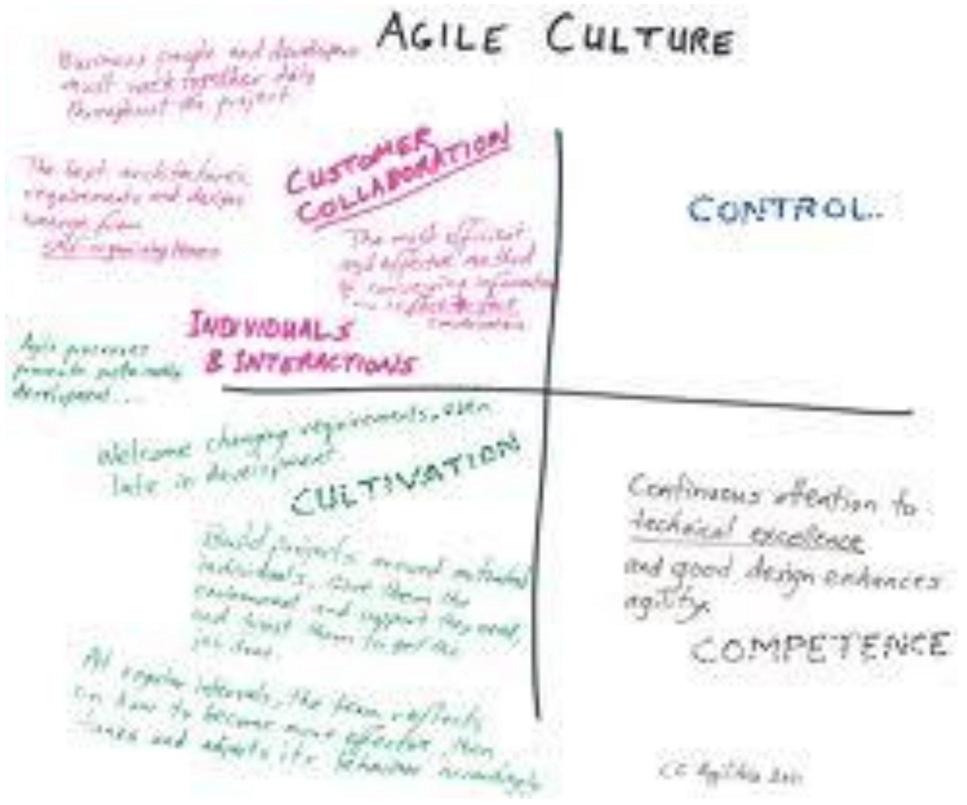
Enterprise Agile Initiative

Being Agile – Agile for Delivery

Course Audience & Lesson Objectives

- Course Audience:
  - This course is intended for beginners in Agile
- Lesson Objectives:
  - Existing Process Models and their observations
  - Basic understanding of Agile
  - Terms used within a Scrum Framework
  - Implementing LEAN
  - Using Kanban
  - Getting Agile L1 Certified

What is Agile?



AGILE CULTURE
Business people and developers must work together daily throughout the project
The best architectures, requirements and designs emerge from self-organizing teams
CUSTOMER COLLABORATION
The most efficient and effective method of conveying information ... is face-to-face conversation
CONTROL..
INDIVIDUALS & INTERACTIONS
Agile processes promote sustainable development...
Welcome changing requirements, even late in development
CULTIVATION
Build projects around motivated individuals. Give them the environment and support they need and trust them to get the job done.
At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly
Continuous attention to technical excellence and good design enhances agility.
COMPETENCE

Before we start…. An IT Industry Update

- One good news
  - More and more projects are moving to AGILE.
- One not-so good news
  - Many AGILE projects do not end up in getting the benefits, that is expected by following AGILE
- One question
  - What makes AGILE project success?

Process Models

- **Agile is NOT….**
  - A <u>new</u> set of practices

- **Agile is…**
  - A set of best practices derived from various process models
  - 8 of the 12 Agile principles are derived from known process models

- Observations of the existing process models derive agile practices

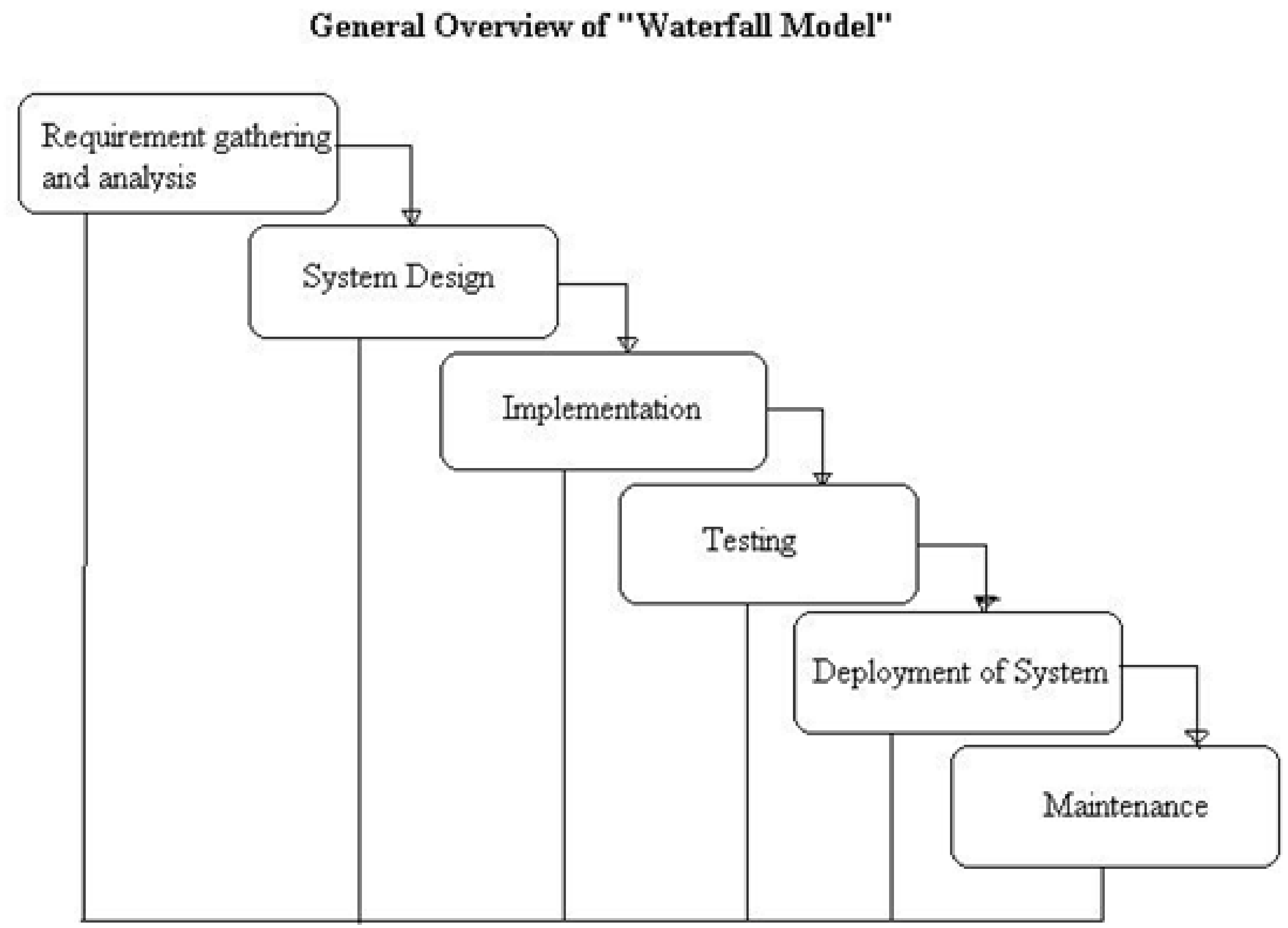
What are current Process models?

- Waterfall Model
- Prototyping Model
- V Model
- Incremental Model

Waterfall Method

- Traditionally software development has been carried out using the classical waterfall methodology (Also called SDLC)
- Key features of this method are
  - Sequential process in which progress is seen as flowing steadily downwards (like a waterfall)
  - Model was derived from manufacturing and construction industries
  - Logical and easy to implement model.. If requirements are frozen at start

Waterfall Method


General Overview of "Waterfall Model"
Requirement gathering and analysis
System Design
Implementation
Testing
Deployment of System
Maintenance

Waterfall Model - Observations

- Late Visibility
- Longer waiting time for the Customer
- No Customer Involvement
- No prioritization of tasks
- Late Customer Feedback
- Enhanced Rework
- No Sustainable pace of Development
- Not flexible to change
- Limited Team Ownership
- Measure of Progress is Process Oriented
- Larger Scope
- Estimation challenge
- Idle Time

Prototyping Model

- Used when the customer is not clear on his requirements and hence generates ideas out of working software

- Rapid Prototype
  - Create an initial prototype for the customer
  - Customer generates further requirements after working with the prototype

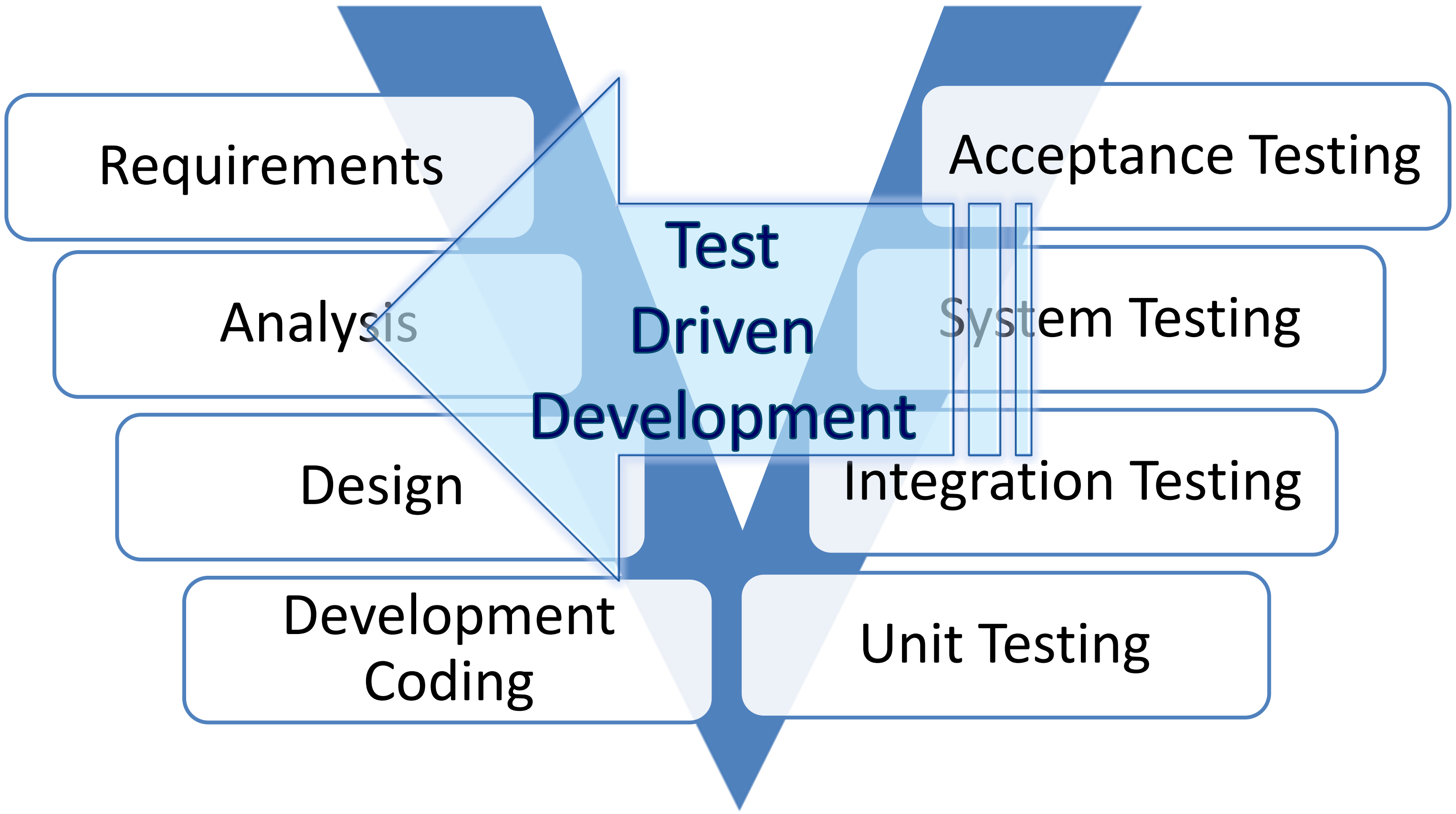
- Throwaway prototype
  - Create an initial prototype for the customer
  - Dispose the prototype and start afresh

V-Shape Model

- In Waterfall testing is late in the development environment
- This model plans for testing earlier in the delivery life cycle rather than just at the end
- Test Driven Development (TDD)
  - Just have test cases ready at the start of the development cycle
  - The confirmation Criteria of the requirement is defined first/derived form TDD

Agile Principle

V-Shape Model



Incremental Model

- Its an iterative waterfall. Each iteration repeats the entire SDLC steps over again.
- Agile is derived from this model
  - Attempt to create Repetitive Waterfall
- Eg.: Word Processing Software
  - Deliver incremental features in each iteration rather than have the customer wait for a longer durations.

Agile Principle

Why AGILE ?

- 80%* of all software projects fail to deliver working software on time and within budget

- Top Reasons:
  - Lack of customer involvement
  - Poor or vague requirements
  - Unrealistic schedules
  - Lack of change management in Waterfall Model
  - Lack of sufficient testing
  - Inflexible processes

* IEEE estimate

How does AGILE address the top reasons?

- Lack of customer involvement
  - Agile makes the customer a member of the delivery team.

- Poor or vague requirements
  - Requirements are written as acceptance tests just before any code is written. (TDD)
  - A requirement in agile is a story. For each story there is an acceptance criteria

- Unrealistic schedules
  - Agile makes estimating & scheduling a collaborative process between the customer & the development team.

How AGILE addresses top reasons?

- Lack of change management
  - Change is the crux of Agile. Anything can change except the delivery date. Everyone has to be realistic abo
- Lack of testin
  - of the code.
- ble processes
  - Agile integrates Retrospection where the current delivery processes can undergo continuous improvement
  - Agile is also referred to as "Process Lite" – Lightweight and flexible in terms of delivery processes followed.

Projects with rapidly changing requirements will derive more benefits of Agile

Mastek Agile

Mastek™

History of Agile

- The Agile Manifesto introduced the term in 2001
- In February 2001, 17 software developers met at a ski resort in Snowbird, Utah, to discuss lightweight development methods
- They published the “**Manifesto for Agile Software Development**” to define the approach now known as agile software development

What is AGILE?

Agile: Dictionary meaning

adj.

- Characterized by **quickness**, **lightness**, and ea
  movement; nimble.
- Mentally quick

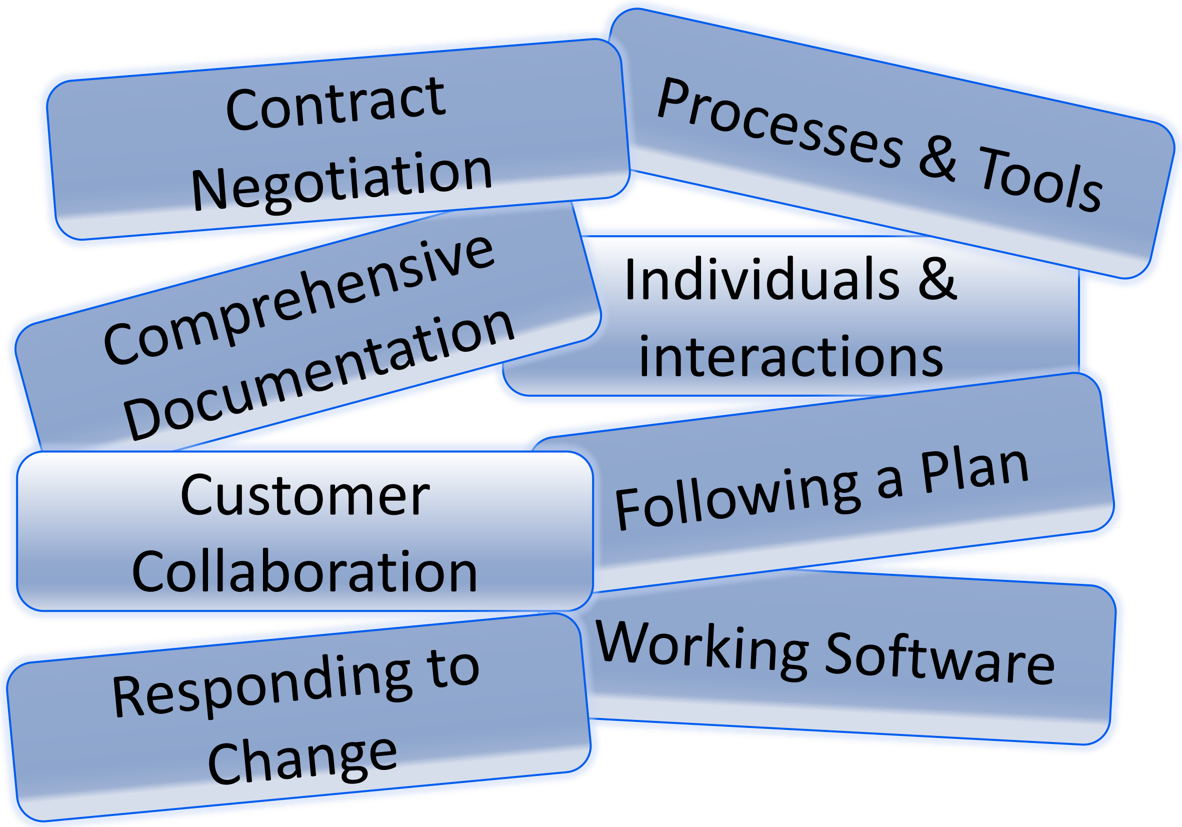
itis, from agere, to drive, do.]

ag'ile·ly *adv*.

- agileness ag'ile·ness *n*.

Agile is a light-weight methodology that enables teams to develop software in the face of vague and rapidly changing requirements

Activity: What is more important?



Agile Manifesto

A statement of values

Individuals & interactions	over	Processes & Tools
Working Software	over	Comprehensive Documentation
Customer Collaboration	over	Contract Negotiation
Responding to Change	over	Following a Plan

Key Agile Principles (1-4 of 12)

1. Customer satisfaction by rapid delivery of useful software
   - Quickly deliver working software
2. Welcome changing requirements, even late in development
   - Iterative & Incremental development
   - Customer Involvement
3. Working software is delivered frequently (weeks rather than months)
   - Deliver in short increments
4. Working software is the principal measure of progress
   - Not Artifacts but functionality delivered

Key Agile Principles (5-8 of 12)

5. Sustainable development, able to maintain a constant pace
   - Not Back or Front Loaded Systems
6. Close, daily cooperation between business people and developers
   - Team Collaboration & Customer Involvement
7. Face-to-face conversation is the best form of communication (co-location)
   - Regular Team Meetings & collaboration via tools
8. Simplicity
   - KISS (Keep it Simple and Sober)

Key Agile Principles (9-12 of 12)

9. Continuous attention to technical excellence and good design
   - Product excellence due to small incremental releases and regular customer feedback
   - Process excellence due to regular reviews.
10. Projects are built around motivated individuals, who should be trusted
    - The team and members are self motivated
    - Team involvement and hence ownership
11. Self-organizing teams
    - Due to Ownership and Roles Defines (Scrum Master etc…)
12. Regular adaptation to changing circumstances

Mastek Agile Manifesto (1-3 of 6)

1. Customer/Business outcomes over Internal Objectives
   - Deliver to customers objectives rather that the projects objectives
   - Reduce too much focus on internal meetings and milestones

2. Continuous improvement over the status Quo
   - Improve rather than sticking to the same way of doings things
   - Do no continue what we did in the past if it did not work

3. Responsiveness over Business as Usual
   - Reduce waiting between teams for providing services
   - LEAN & KanBan help improve this

Mastek Agile Manifesto (4-6 of 6)

4. Flexibility / Adaptability over following a Plan/Process
   - Do not blindly continue following a plan
   - Relook at the plan from business objectives perspective
   - Be responsive to change

5. Dialogue/Teamwork over roles Structures
   - Resolve issues yourself with direct dialog with that team member rather than escalation

6. Proactivity over waiting to be asked
   - Irradiate waiting for receiving services, requested for.

Key Terminology: Summary

- User Story
- Time-boxed (Sprint)
- Definition of Done
- Product Backlog
- Velocity
- Burn down/up Charts

Key Agile Terminology

- User Story
  - A short description of the functionality told from the perspective of the end user or customer.
  - (eg: User wants to book/ search for flights)
  - User stories are typically smaller in nature achievable within a defined timeline
  - A very large user story is referred to as an epic
  - The BA / customer or both write the User Story

Key Agile Terminology

- User Story
  - Attempt to keep user stories independent of each although it would be part of a flow
  - Should be able to attempt any part of the plan without affecting the rest of the plan
  - If customer requests for a billing module before booking , we should be able to accommodate that.
  - A user story can be further divided into tasks
    - Book Tickets
      - ►Login
      - ►Check availability
      - ►Book

User Stories are Not

- Requirements documents
- Use Cases
- Scenarios

User Story

- How are user stories different from a Use Case / Requirement?
- Made up of 3C’s
  - Card
  - Conversation
  - Confirmation

User Story Format

- **Card**
  - Typically documented on an index card and written in just 2-3 lines

As a <Role>

I want to <Use Case>

So that <Benefits>

  - **Role** - User
  - **Use Case** - What to do
  - **Benefits** - Outcome

User Story Format

- **Benefits of using this format?**
  - Focus is always on the **value** the customer will get

As a <Role>

I want to <Use Case>

So that <Benefits>

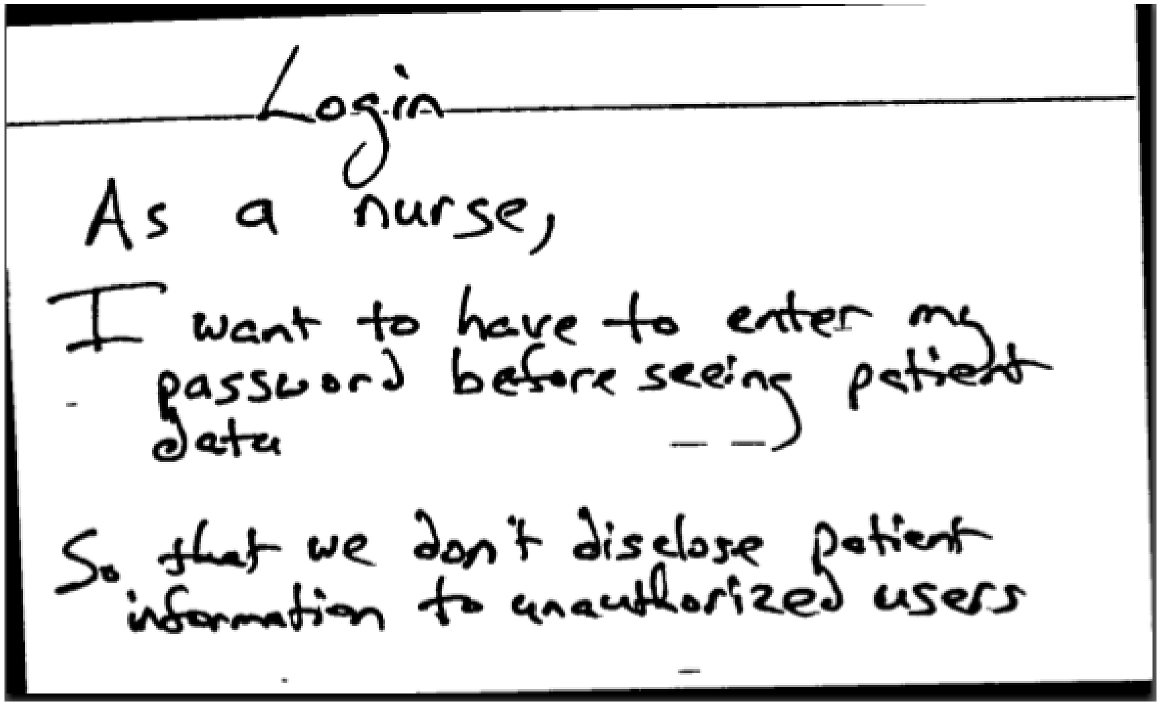
  - Role helps identify the actor
  - Use Case states what to do
  - Benefits decides the value the user will get

User Story Format

As a nurse, I want to have to enter my password before seeing patient data, so that we don't disclose patient information to unauthorized users.

As a <role>
I want <feature>
So that <business value>

- Card
- Conversation
- Confirmation



Login

As a nurse,
I want to have to enter my password before seeing patient data
So that we don't disclose patient information to unauthorized users

Traditional Requirements Anaysis

Guidelines for Good Stories

- Start with goal stories
- Write closed stories (stories that have a definite end point)
  - "A recruiter can review resumes from applicants to one of her ads" instead of "A recruiter can manage the ads she has placed"

How to Write Good User Stories

- Independent of each other
  - Try to reduce the dependency
- Negotiable
  - The story was created post discussion with the customer
- Valuable
  - What value will it add to the customer
- Estimable
  - If you get different estimates even after a team discussion, its not negotiable and hence not estimable
- Small
  - Crisp and to the point
- Testable
  - Always should have a confirmation mechanism

Examples of specifying value to users

- **Good:**
  - "A user can search for jobs"
  - "A company can post new jobs"
- **Bad:**
  - "The software will be written in C++"
  - "The program will connect to the database through a connection pool"

User Story Contents

- Card
  - Describes the intent/content of the story
- Conversation
  - Describes the conversation between various stakeholders
- Confirmation
  - Describes the acceptance test criteria for the story

User Story - Activity

- **Identify the top 6 user stories for a Training Calendar application in terms of the highest value derived by the user**

As a <Role>

I want to <Use Case>

So that <Benefits>

User Story – 3C's

- Conversation
  - Sizing of the User story
    - Discussion of a user story within the team to decide on its complexity
  - Based on its complexity the team sizes the user story
    - This is NOT Effort Estimation
    - This is NOT the time required to solution the user story
  - Sizing is NOT estimation
    - The complexity based sizing exercise, will assist in arriving at the effort required later on.

User Story – 3C’s

- Confirmation
  - Every user story will have an acceptance test case / Confirmation criteria (TDD)
  - Defines when the user story will be complete

User Story – Sizing

- Sizing (of User stories)
  - Each user story will differ from another
  - Every user story has Story Points assigned to it based on its complexity.
  - Sizing is always a relative process
    - Fibonacci Series (1,2,3,5,8)
    - Exponential (1,2,4,8,16)
    - T-Shirt Sizing (S,M,L,XL)
  - (**Brings more clarity on what size to assign)
  - Team along with the Customer will be involved in sizing

User Story – Sizing Exercise

- How much time would we need to build a 4 floors building??
- Building 2 floors took 6 months……
- A base reference point makes sizing easier
- Select a base reasonably
  - Can be changed after discussion with team if the sizing was identified to be incorrect

User Story – Sizing

- How much time would we need to paint these cars?

- Nano
- WagonR
- Swift
- City 7
- XUV 500
- Audi Q7

- Each team member gives different points based on his role (Tester, Developer….)
  - In case of major difference, a discussion should resolve it.

User Story – Sizing

- This Sizing does not involve Technology.
  - That’s part of the effort estimation based on the sizing
- Sizing typically starts in Pre-Sales
  - FP can be used to validate sizing
- Outcome of the sizing is to decide how many user stories to take on in a single release

Key Agile Terminology

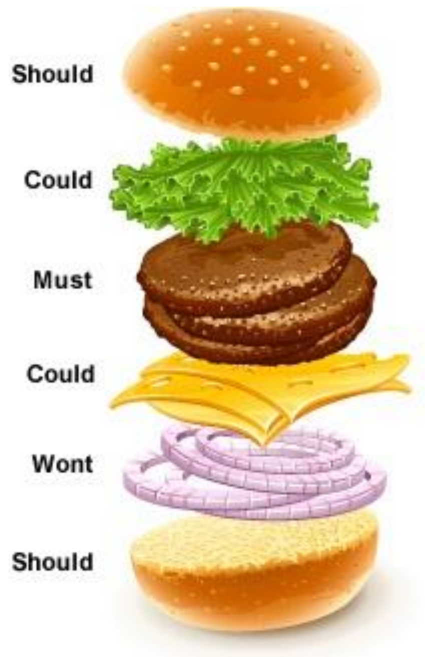
- Product Backlog
  - Scrum term for the prioritized list of all the functionality desired in the product
  - List of prioritized User Stories
  - The Product Owner decides/maintains the product backlog
    - Product Owner = Customer
  - Gets updated after every sprint depending on what was completed/pending in that sprint

Product Backlog

- Product Owner decides the priority of items based on their..
    - Business Value
    - Cost
    - Risk (High Risk first or last)

Product Backlog

- MoSCoW
  - Used to prioritize items in the backlog
- MUST:
  - Describes a requirement that must be satisfied
- SHOULD:
  - Represents a high-priority item
- COULD:
  - A requirement which is considered desirable.
- WONT:
  - Represents a requirement that will not be implemented in current release



Key Agile Terminology

- DOD (Definition of Done)
  - Decides when a user story is done/completed
    - Tested
    - Bug Fixed
    - Delivered
    - Deployed
  - May vary for each team
  - Decided post discussion within the team
  - Also called a sprint goal

Key Agile Terminology

- Time-boxed
  - Planning technique used in agile projects where the schedule is divided into a number of separate time periods
  - Each time period is of a fixed duration and referred to as a "Sprint"
  - A Sprint ends once the time period elapses (typically 2 to 4 weeks as per Scrum)
  - Large projects can have multiple sprints for multiple modules
  - Team size should be 5 to 9 (as per Scrum).

Key Agile Terminology

- Sprint
  - Sprints are typically of the same length
  - Initially for the first sprint we may go with experience to get the duration right, after which the duration should stabilize
  - We have the option of modifying the length in the next sprint onwards, but it should be constant then on.
  - Any work remaining in the sprint has to stop when the Sprint duration completes
  - The sprint has either succeeded/completed or failed.

Sprint

- New or Ad-hoc change requests cannot be accommodated in between a sprint
  - Should plan and accommodate in the next sprint

- If a Sprint is not delivering value to the customer….?
  - Team discusses with the Customer to identify a way to deliver value to the customer on sprint completion

Key Agile Terminology

- Velocity
  - It is the amount of value delivered in each iteration, measured in either story points, days or hours.
  - Only completed stories are counted for calculating velocity.
  - Velocity is useful for planning for future features and releases
  - It is important for the sprint duration to be constant else it would impact the velocity.
  - The no. of resources per sprint should not vary largely.
  - Plan Sprint wise and not for the entire product

Velocity

Sprint - 1	Sprint - 2
10 Story Points	5 Story Points

- Velocity is the average of the two (7 Story Points)
- If you attempt 10 stories in the first sprint, take a number close to that for sprint-2
- Use experience for the duration of first few sprints, then use Velocity to plan sprints.
  - Velocity will be constant as you proceed

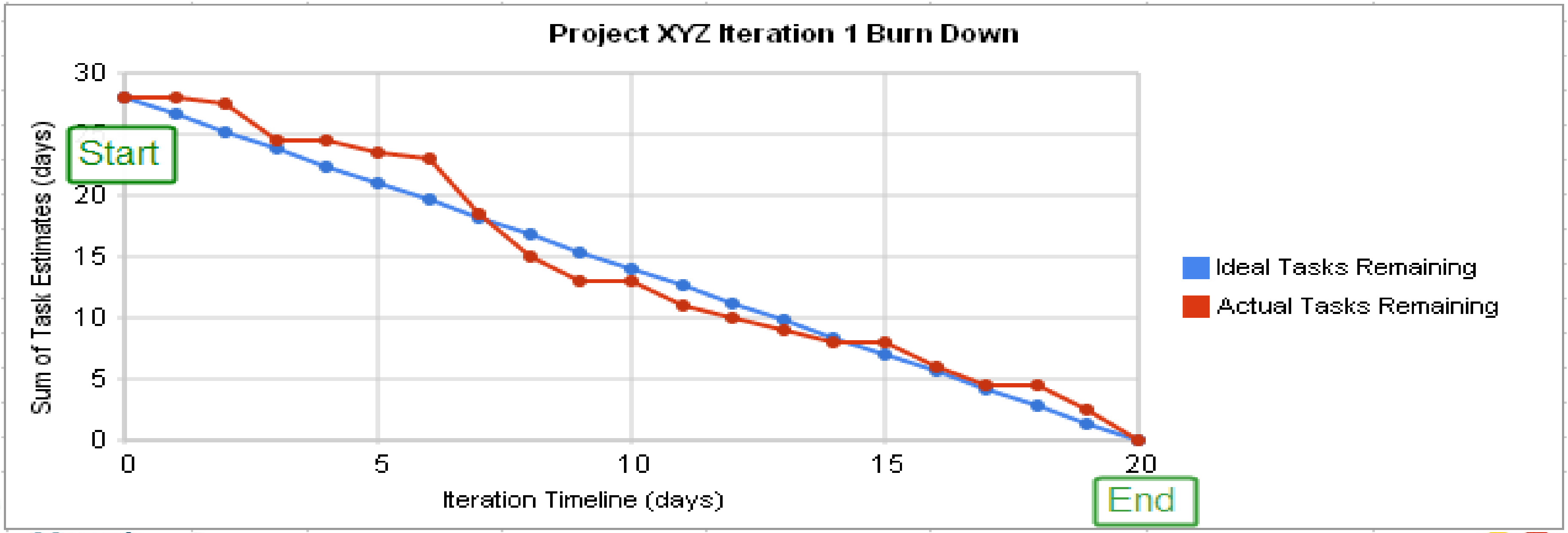
Velocity

- Initially the velocity may vary and would stabilize later on.

- Monitors progress and allows you to plan
  - Total Story points divided by the velocity will give you number of sprints required.

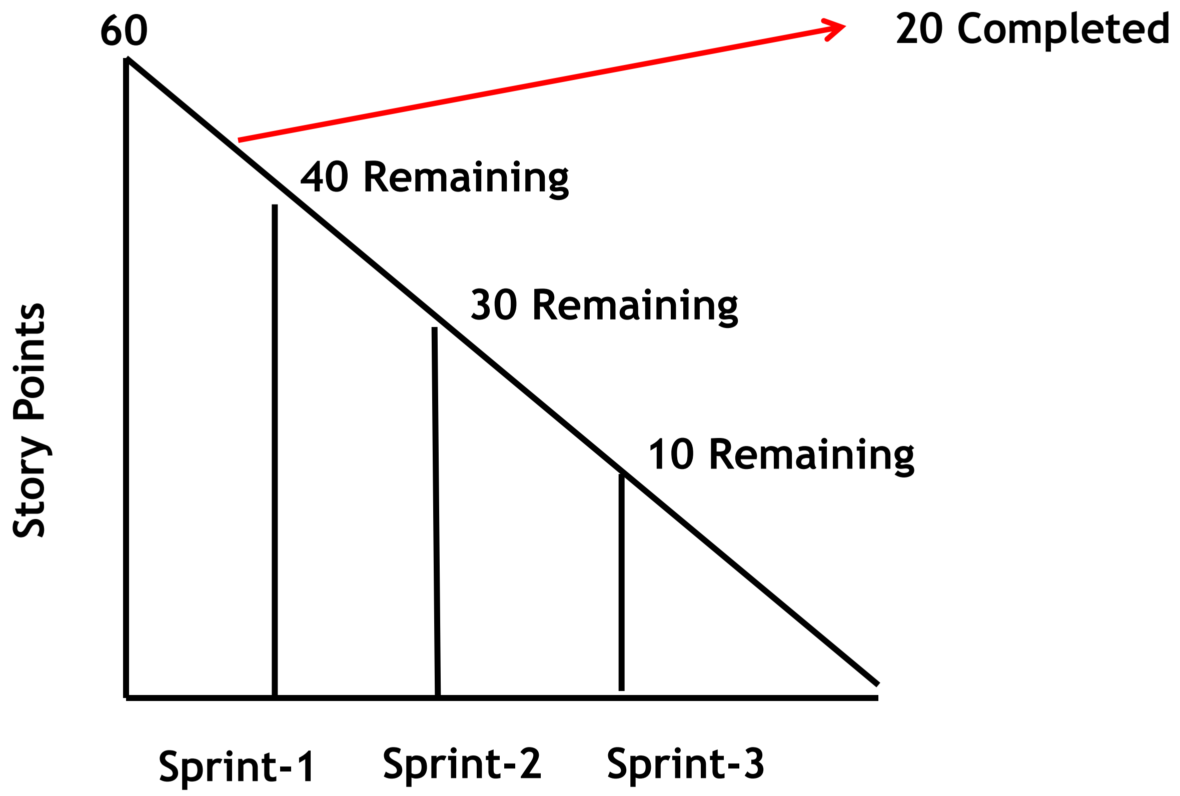
Key Agile Terminology

- Release Burn Down Chart
  - A burn down chart is a graphical representation of work left to do versus time.
  - Shows how many story points are completed in a sprint
  - The outstanding work (or backlog) is on the vertical axis, with time along the horizontal.

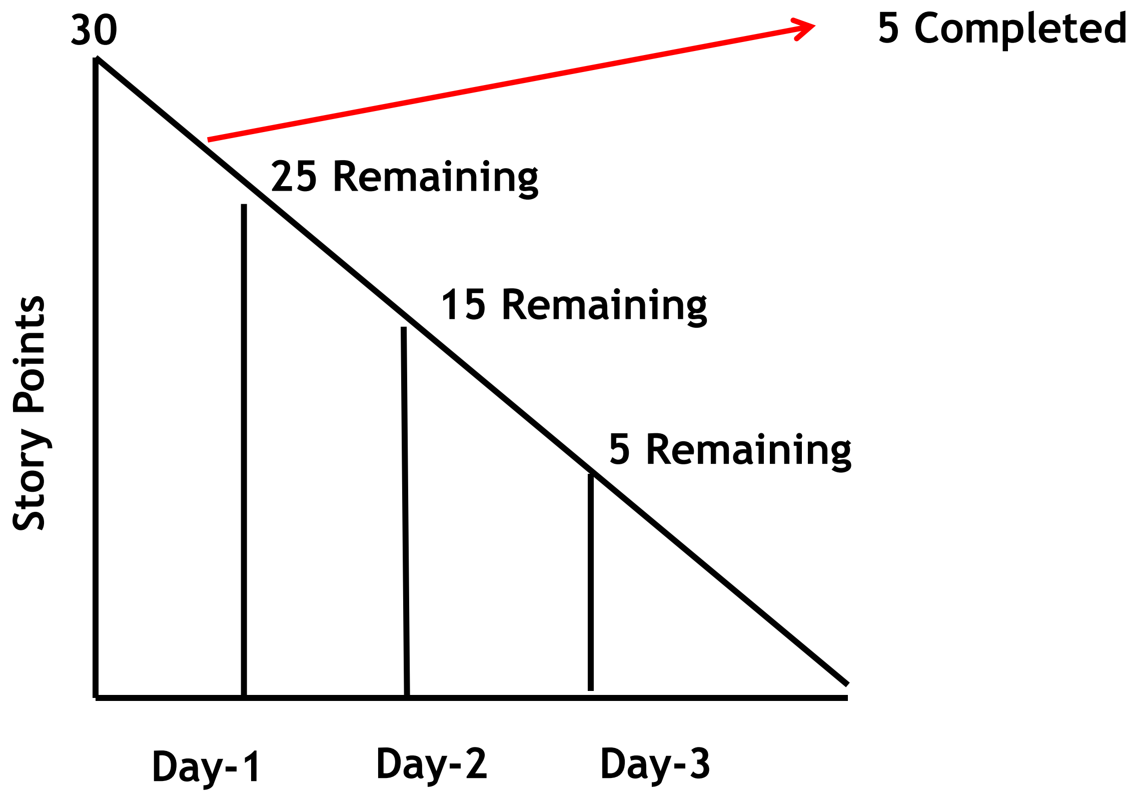


Release Burn Down Chart

- Total of all story points is what has to be completed in a release
- Targets what is remaining
- Helps to calculate the velocity (the slope of the line)



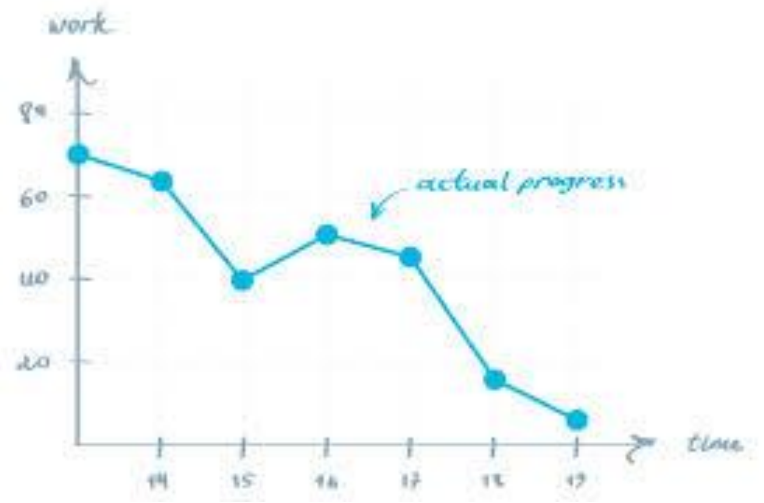
Sprint Burn Down Chart



Velocity: Scope Creep

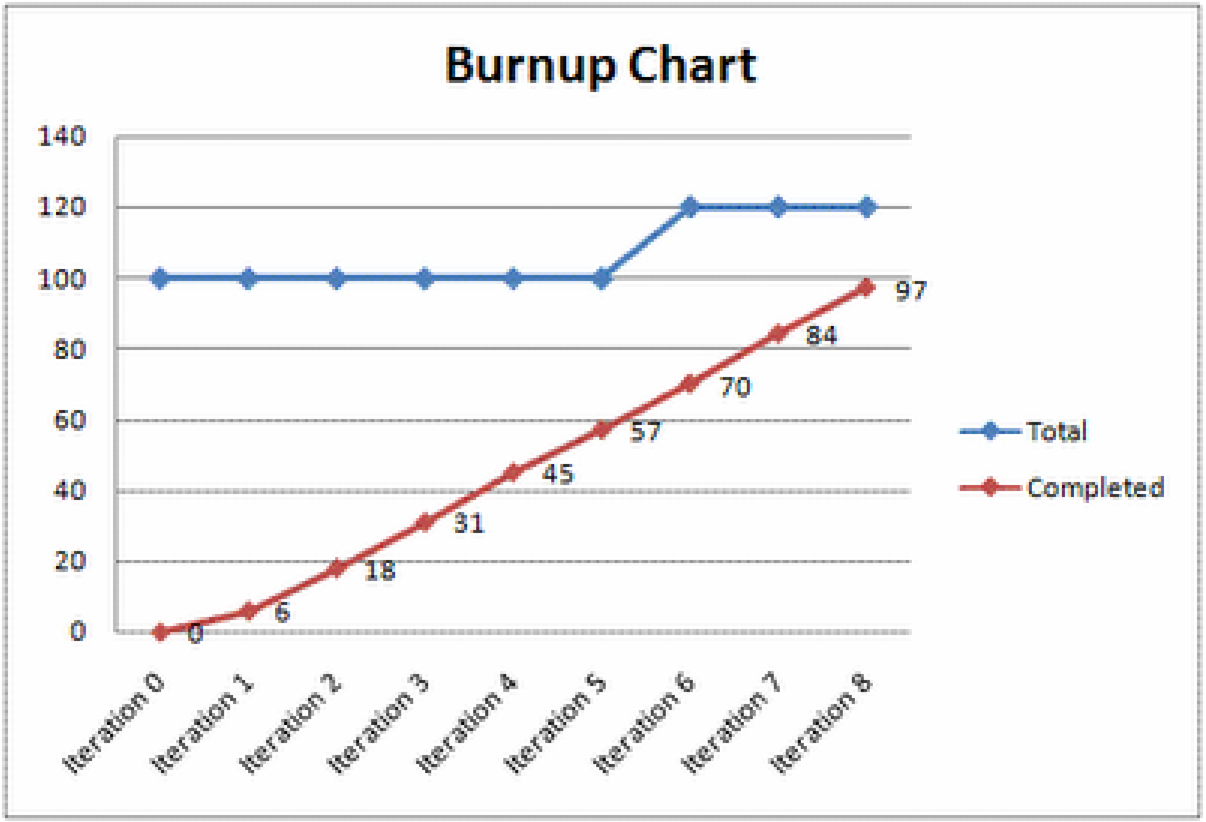
- Scope Creep
  - More user stories added in between by the customer
  - Changes and defects may also get added

(**The graph can show a spike in such cases in between)



Key Agile Terminology

- Burn Up Chart
  - Tracks what is completed



Key Terminology: Summary

- User Story
- Time-boxed (Sprint)
- Definition of Done
- Product Backlog
- Velocity
- Burn down/up Charts

AGILE implementation methodologies

- Well-known agile software development methods:
  - Agile Modeling
  - Agile Unified Process (AUP)
  - Dynamic Systems Development Method (DSDM)
  - Essential Unified Process (EssUP)
  - Extreme Programming (XP)
  - Feature Driven Development (FDD)
  - Scrum (Most popular of the lot)
  - Velocity tracking

AGILE implementation methodologies

- Dynamic Systems Development Method (DSDM)
  - When requirement constantly change
  - Similar to generic Agile principles
  - Key feature in DSDM is prioritization (Pareto Principle)

  - Pareto Principle:
    - 80% of defects occur due to 20% of issues
    - Select least number of user stories that give maximum Customer value

AGILE implementation methodologies

- Extreme Programming (XP)
  - Pair Programming
  - Code Refactoring
  - Test Driven Development (TDD)
  - Continuous Integration

All Agile methods have following things in common

- Iterative Development
- Responsive to Changing Requirements
- “Process Lite” Model
- Customer (Product Owner) Is a Part of the Team
- Focus is on delivering business value by frequent delivery
- Smaller Cross Functional team
- Test Driven Development / Early Testing
- Inspect And Adapt
- Regular Reviews

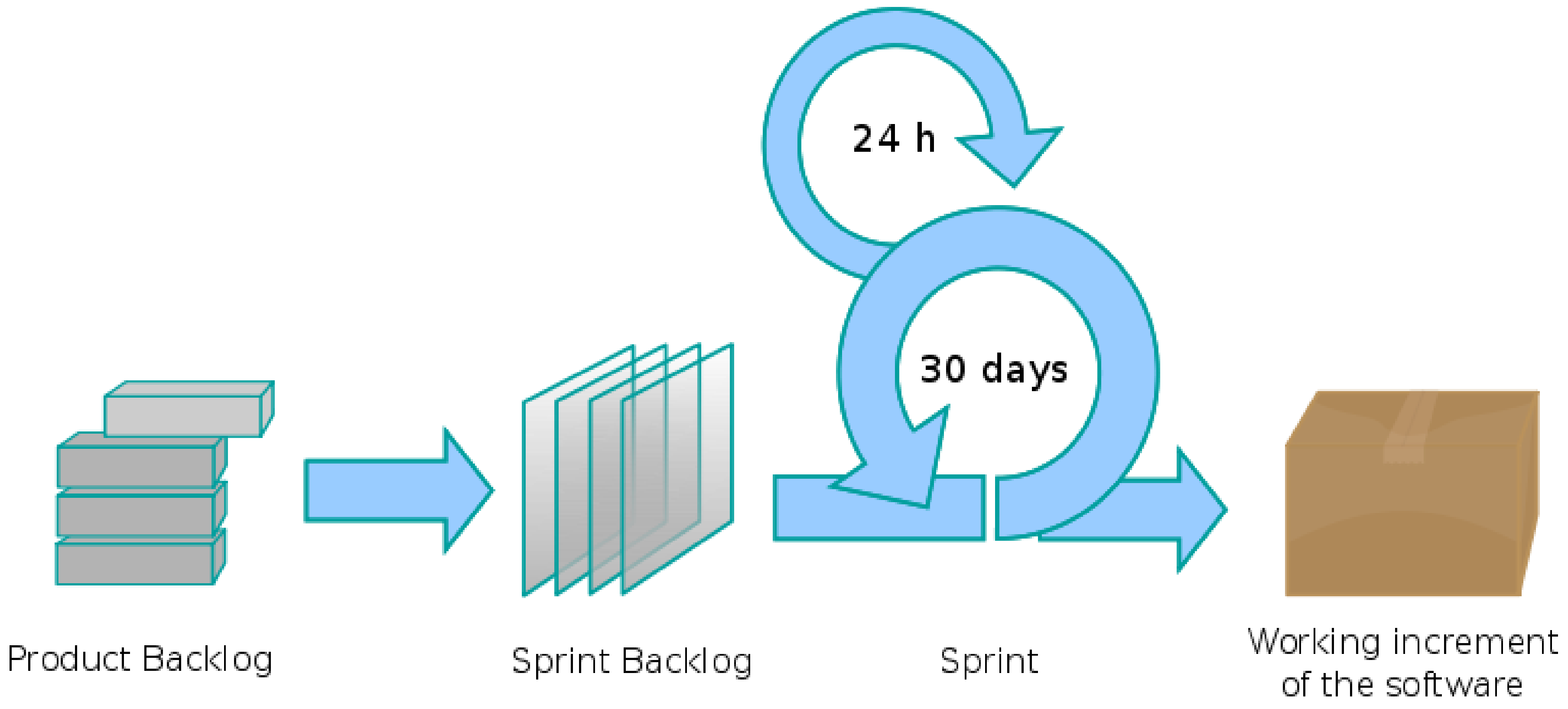
Scrum

What is SCRUM?

- term from rugby
- a process with a set of roles and practices for agile development
- iterative = timeboxed (sprints)
- incremental = features added incrementally
- continuous process improvements = retrospectives

AGILE Scrum

- Scrum is an iterative, incremental methodology for project management.



Why SCRUM?

- Frequent deliveries of completed functionality
- Small iterations = easier to adapt to change
- Customer involvement = customer satisfaction
- Deliver business value = Most important requirements are done first, prioritized frequently
- Visible progress = predictable progress
- Continuous improvement
- Helps focus and motivate team

Scrum Framework

Roles
- Product owner
- Team
- Scrum Master

Meetings
- Sprint Planning
- Daily Scrum Meeting
- Sprint Review
- Sprint Retrospective

Artifacts
- Product Backlog
- Sprint Backlog
- Burn down

Scrum Roles – Pigs and Chickens (1)

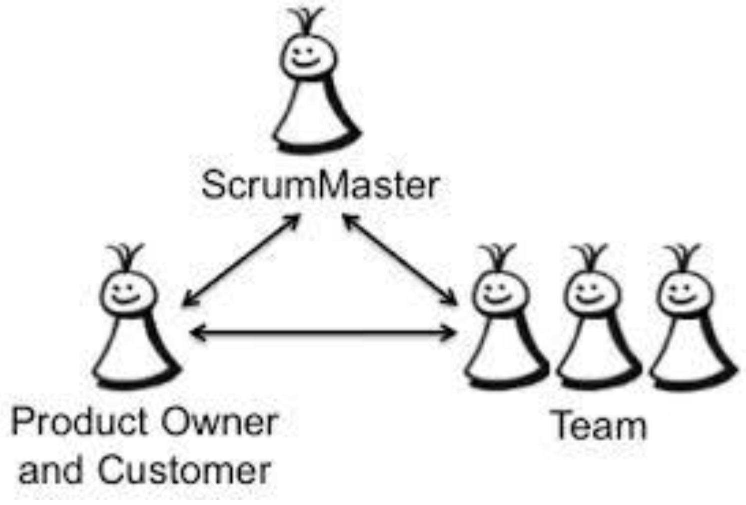
- A pig and a chicken are walking down a road. The chicken looks at the pig and says, "Hey, why don't we open a restaurant?" The pig looks back at the chicken and says, "Good idea, what do you want to call it?" The chicken thinks about it and says, "Why don't we call it 'Ham and Eggs'?" "I don't think so," says the pig, "I'd be committed but you'd only be involved.“

- Ham and Eggs - committed or just involved

Roles – Pigs and Chickens (2)

- Pigs
  - Product Owner - voice of the customer
  - Scrum Master - enforcer of Scrum process, facilitates (removing impediments) team to reach sprint goal
  - Team - cross-functional (design, developer, test), usually 5-9 people who does the work

- Chickens
  - Users
  - Stakeholders (Customers, Vendors)
  - Managers



Scrum Framework

Roles

- Product owner
- Team
- Scrum Master

Meetings

- Daily Scrum Meeting
- Sprint Planning
- Sprint Review
- Sprint Retrospective

Artifacts

- Product Backlog
- Sprint Backlog
- Burn down

Product Owner

- Voice of the customer
- Decide on release date and content
- Defines sprint goals (output of each sprint)
- Defines the features (user stories) of the product
- Prioritize features according to its value
- Adjust features and priority every iteration, as nee
- and update the Product Backlog
- Accept or reject work results

The Team

- Responsibility to deliver product
- Typically 5-9 people
- Cross-functional:
  - Programmers, testers, user experience designers, etc.
- Members should be full-time
  - May be exceptions (e.g., database administrator)
- Teams are self-organizing
- Membership should change only between sprints

The Scrum Master

- Removes impediments
- Responsible for enacting Scrum values and practices
- Ensure that the team is fully functional and productive
- Represents management to the project (though not a leader of the team)
- Servant Leader

- Is this a full time role?
- Is the PM?

Scrum Framework

Roles

- Scrum Master
- Product owner
- Team

Meetings

- Sprint Planning
- Daily Scrum Meeting
- Sprint Review
- Sprint Retrospective

Artifacts

- Product Backlog
- Sprint Backlog
- Burn down

Scrum Process

- Artifact-1: Product Backlog
- Inputs:
  - User stories identification & Sizing based on 3C's
  - MoSCoW & Customer discussion for prioritization
- Output:

Product Backlog	
Tasks	**Story Points**
User Story-1	7
User Story-2	9
User Story-3	5
User Story-4	8

Scrum Process

- Sprint Planning
  - Decide Length of sprint (2 to 4 weeks)
  - Decide Number of Sprints

- Why is the length between 2 to 4 weeks?
  - Cannot deliver value in a less than 2 weeks
  - More than 4 weeks will be back to Waterfall

Scrum Process

- Artifact-2: Sprint Planning Meeting
  - How many story points should the team take up for this sprint

- Inputs:
  - Product Backlog (proritized user stories)
  - Velocity of the team (How many story points can be completed)
  - Capcity of the team (No. of resources)
  - Business Considerations (Customer Inputs) & Technology

Scrum Process

- Artifact-2: Sprint Planning Meeting
  - Decides the number of story points the team takes up for the sprint

- Output: Sprint Backlog

Product Backlog	
Stories	**Story Points**
User Story-1	7
User Story-2	9
User Story-3	5
User Story-4	8

Sprint Backlog	
Tasks	Story Points
User Story-1	7
User Story-2	9

Task1

Task1

Task1

Sprint Planning Meeting

- Team selects items from the product backlog they can commit to completing
- Sprint backlog is created
  - Tasks are identified and each is estimated (1-16 hours)
  - Collaboratively, not done alone by the Scrum Master
- High-level design & Technology is considered

As a vacation planner, I want to see photos of the hotels.

→

1. Code the middle tier (8)
2. Code the user interface (4)
3. Write test fixtures (4)
4. Code the foo class (6)
5. Update performance tests (4)

Product Backlog

Sprint Backlog

Scrum Process

- Artifact-3: Daily Scrum Meeting
  - Each day we have a daily Scrum meeting

- Inputs: Just 15 minutes every day

Daily Scrum Meeting

- Daily standup meetings conducted by the Scrum Master
- Same time, same location
- All are welcome, but only pigs may speak
- Timeboxed at 15 min
- Just 3 Questions
    - Scrum Master to remove impediments
- Not to be addressed to scrum master, but to inform each other
- Helps avoid other unnecessary meetings

The Daily Scrum Meeting – cont.

- **Everyone answers just 3 questions :**

1. **What did you do yesterday?**
2. **What will you do today?**
3. **Is anything in your way?**

- These are *not* status for the Scrum Master
- They are commitments in front of peers

Sprint Review Meeting

- Product Owner Reviews the completed & non-completed work
- Present completed work during the sprint to Customer
  - A demo of new features or underlying architecture
- In-complete work may or may not be carried over to next sprint. (Put back into the Product /Backlog)
- Informal
  - Max 4 hour
  - 2-hour prep time
  - No slides
- Whole team participates

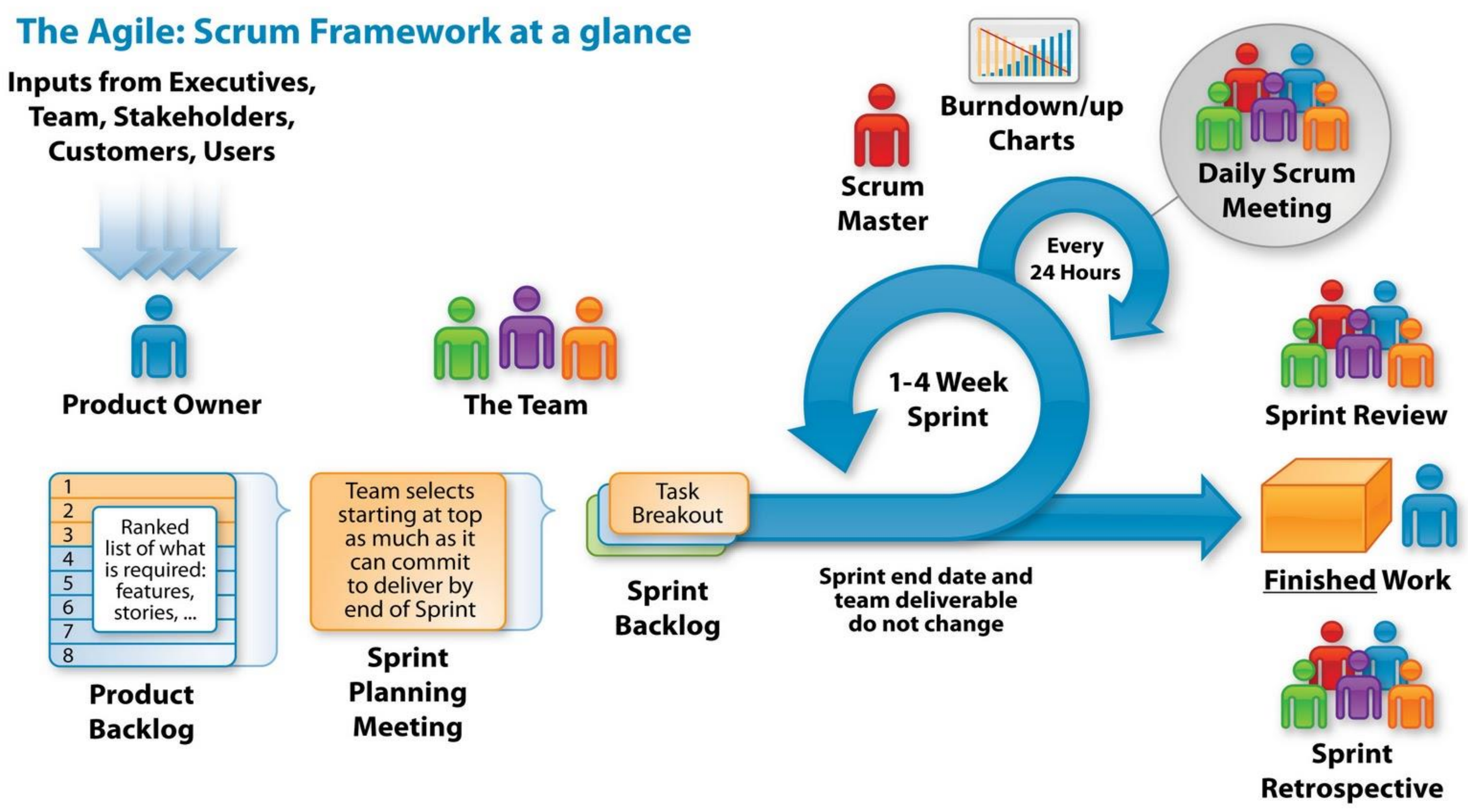
Sprint Retrospective Meeting

- Done after every sprint (after sprint review meeting)
- All team members reflect on past sprint
- Make continuous process improvements and not product improvements
- Whole team, Scrum Master, Product Owner participates
- Other stakeholders may attend

– Max 3 hours

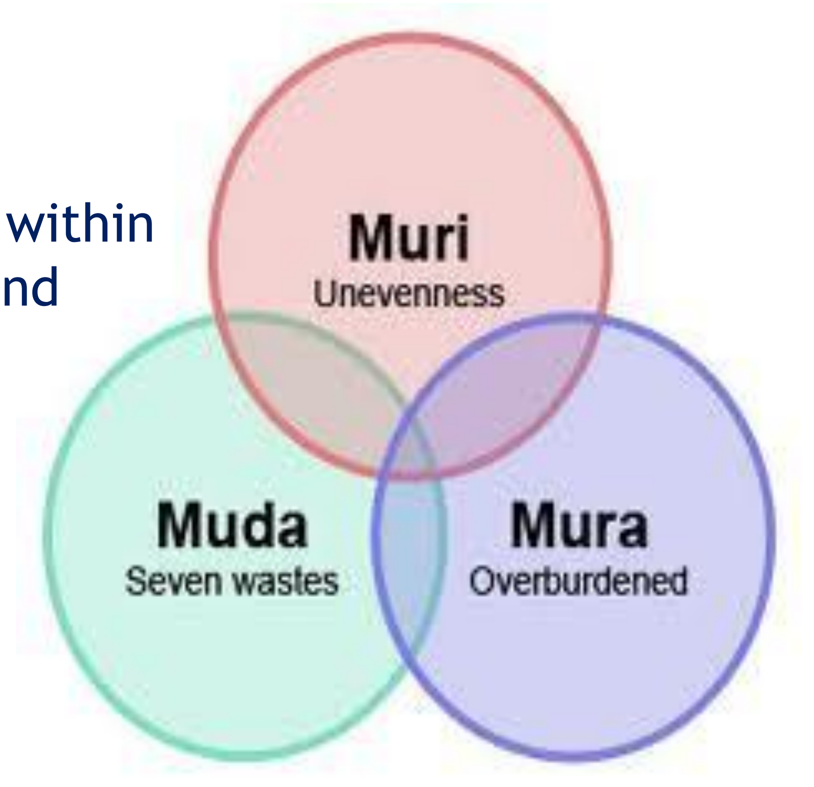
- Questions asked:
  - Start What can be improved in the next sprint?
  - Stop What did not go well during sprint?
  - Continue What went well during sprint?

Scrum as a whole


The Agile: Scrum Framework at a glance
Inputs from Executives, Team, Stakeholders, Customers, Users
Product Owner
The Team
Scrum Master
Burndown/up Charts
Daily Scrum Meeting
Every 24 Hours
1-4 Week Sprint
Sprint Review
1
2
3
4
5
6
7
8
Ranked list of what is required: features, stories, ...
Product Backlog
Team selects starting at top as much as it can commit to deliver by end of Sprint
Sprint Planning Meeting
Task Breakout
Sprint Backlog
Sprint end date and team deliverable do not change
Finished Work
Sprint Retrospective

LEAN

Lean is a philosophy that focuses on continuous elimination of **waste, inconsistency and Unreasonableness** within our organization, using a set of tools and guidelines.

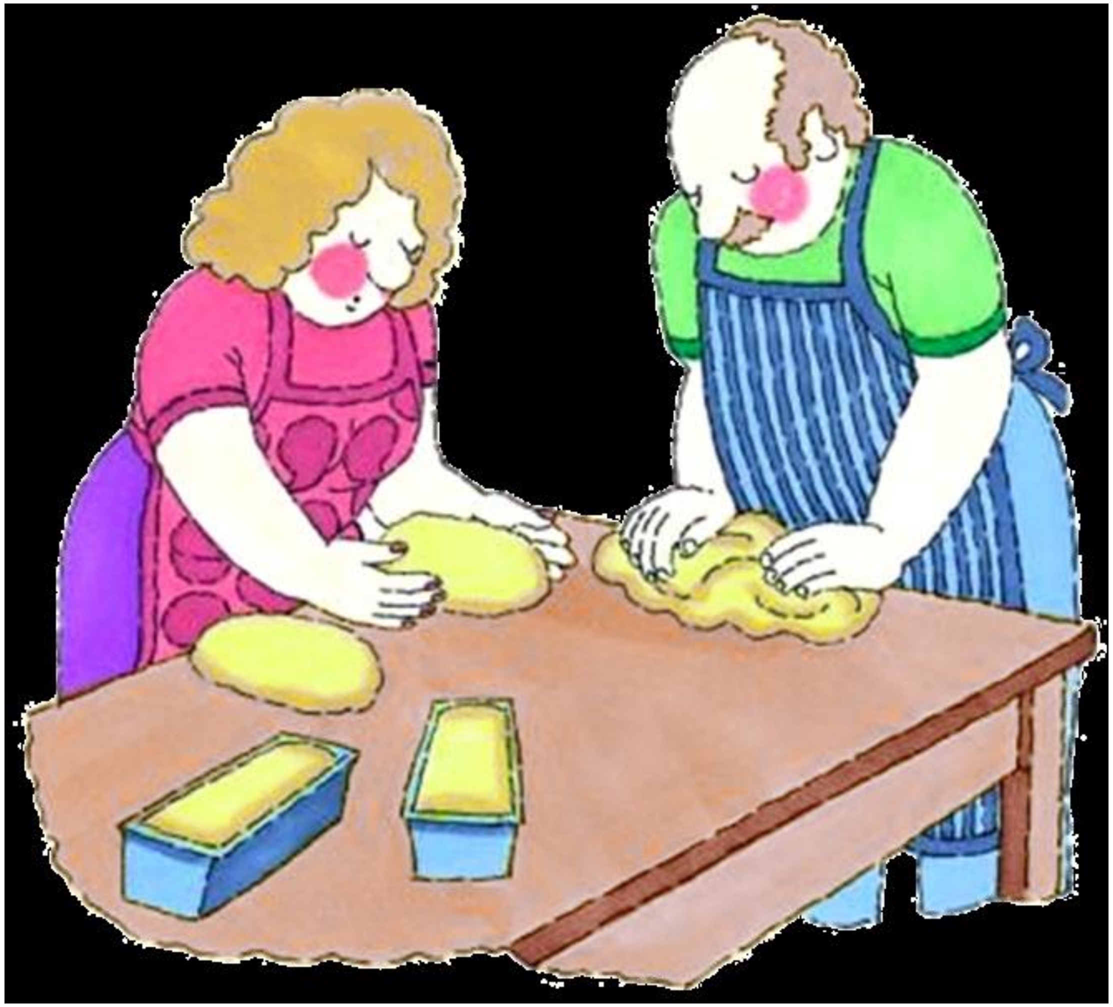


Lean

- Japanese technique
- From the Manufacturing Domain
- Originated via Toyota Production System (TPS)
- Customer (internal/external) decides what is waste
  - Antonym of waste is Value
- Value=No of features/Quality
- Maps to Mastek's Agile Principle
  - Continuous Improvement over status Quo

A Chicken Story

A newly wed couple decided to share house hold responsibilities



A Chicken Story

Their first conflict came about when the wife cut both ends off a chicken before she cooked..

Why do you cut off both ends of a chicken before cooking it?
That's How you cook chicken. That's how my mamma cooks it

A Chicken Story

Next time they were together with a mother and grandmother, the husband asked..

He then asked the grandmother…

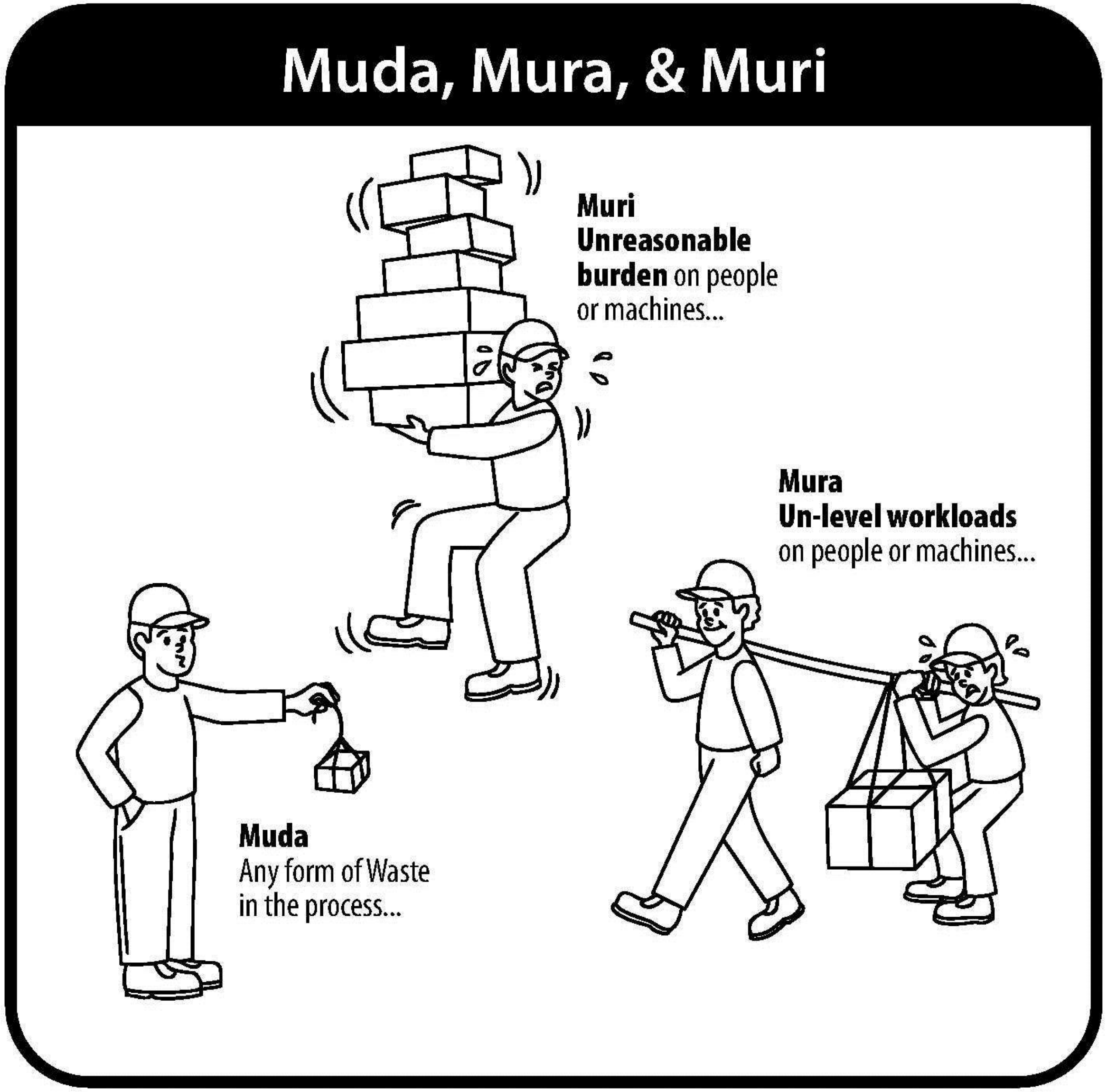
I cut both ends because I never had a pan big enough!
That's how my mother taught me to cook chicken
?
Grandma ,Why do you cut off both ends of a chicken before cooking it?

Moral of the Story

So, the moral of the story is:

Understand the logic and rationale behind every process step rather than just accepting it as it comes.

The 3 Ms Of Lean


Muda, Mura, & Muri
Muri
Unreasonable burden on people or machines...
Mura
Un-level workloads on people or machines...
Muda
Any form of Waste in the process...

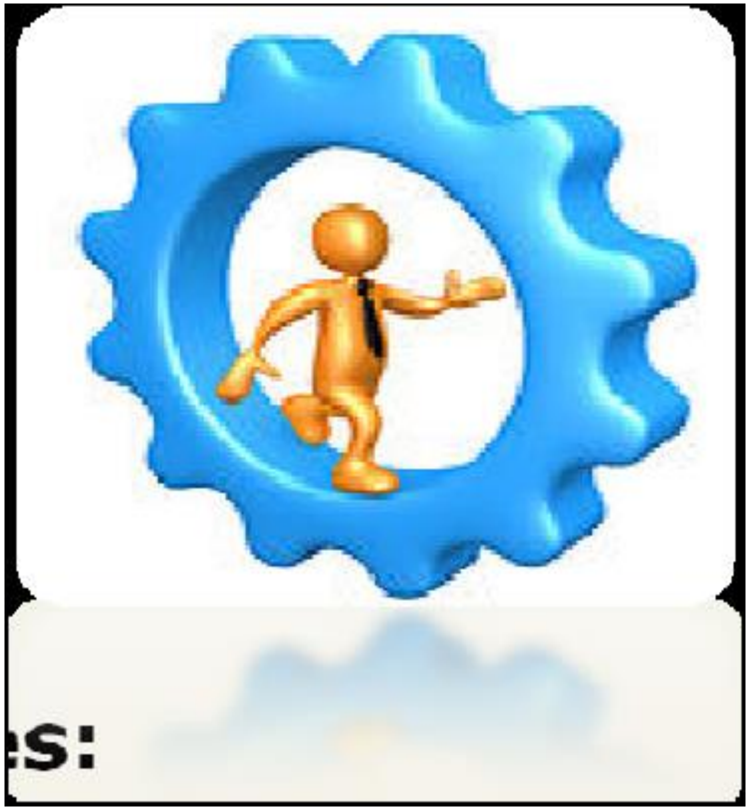
Muda - 8 Wastes

D - Defects
O - Over production
W- Waiting
N - Not utilizing Talent
T - Transportation
I - Inventory
M- Motion
E - Excess Processing

Muda – 8 Wastes

Defects
Anything that is not fit for use /deficiencies in service.

Motion
Any people movement that does not add value to the customer.



Muda – 8 Wastes

Waiting

Waiting for anything –people, material, machine or information

Not utilizing talent

Not using the existing knowledge Leads to duplication of efforts

Muda – 8 Wastes

Transportation

Temporarily relocating and moving products/data.

Inventory

Inventory takes up space. It becomes obsolete if work requirements change.

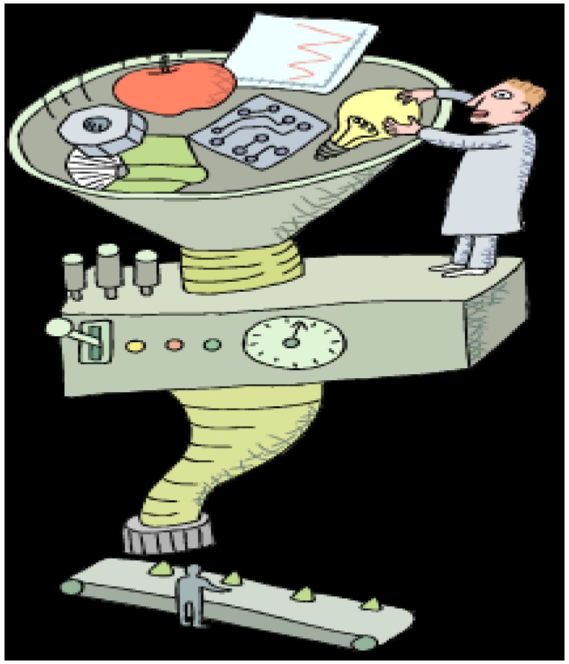
Muda – 8 Wastes

Over Production

Producing more than what is required to meet customer demand.

Excess Processing

Putting more work than required by the customer.



Identify the types of Waste

1. Missed schedules
2. Changing from one task to another or looking for information etc.
3. Developers waiting for requirements
4. Endless refinements to reports
5. Multiple forms with same information produced
6. Cross skilled resources not used
7. Documents moving from one department to another for approval
8. Software coded but not tested, lying in the system

Value Added Activities

- Lean focuses on improving the proportion of Value Added activities and reducing the Non Value Added activities
- Value Added (VA)
  - Product or service is transformed into a state required by the customer;
  - The customer is willing to pay for it
- Non Value Added (NVA)
  - Activities which consume resources but do not create value;
  - Customer is not willing to pay for these
- Non Value Added but Needed (NVAN)
  - Activities causing no value add but cannot be eliminated based on the current state of technology or thinking

Value Stream Map

- Value Stream Mapping
  - an exercise to identify VA, NVA and NVAN steps in a process using well defined symbols

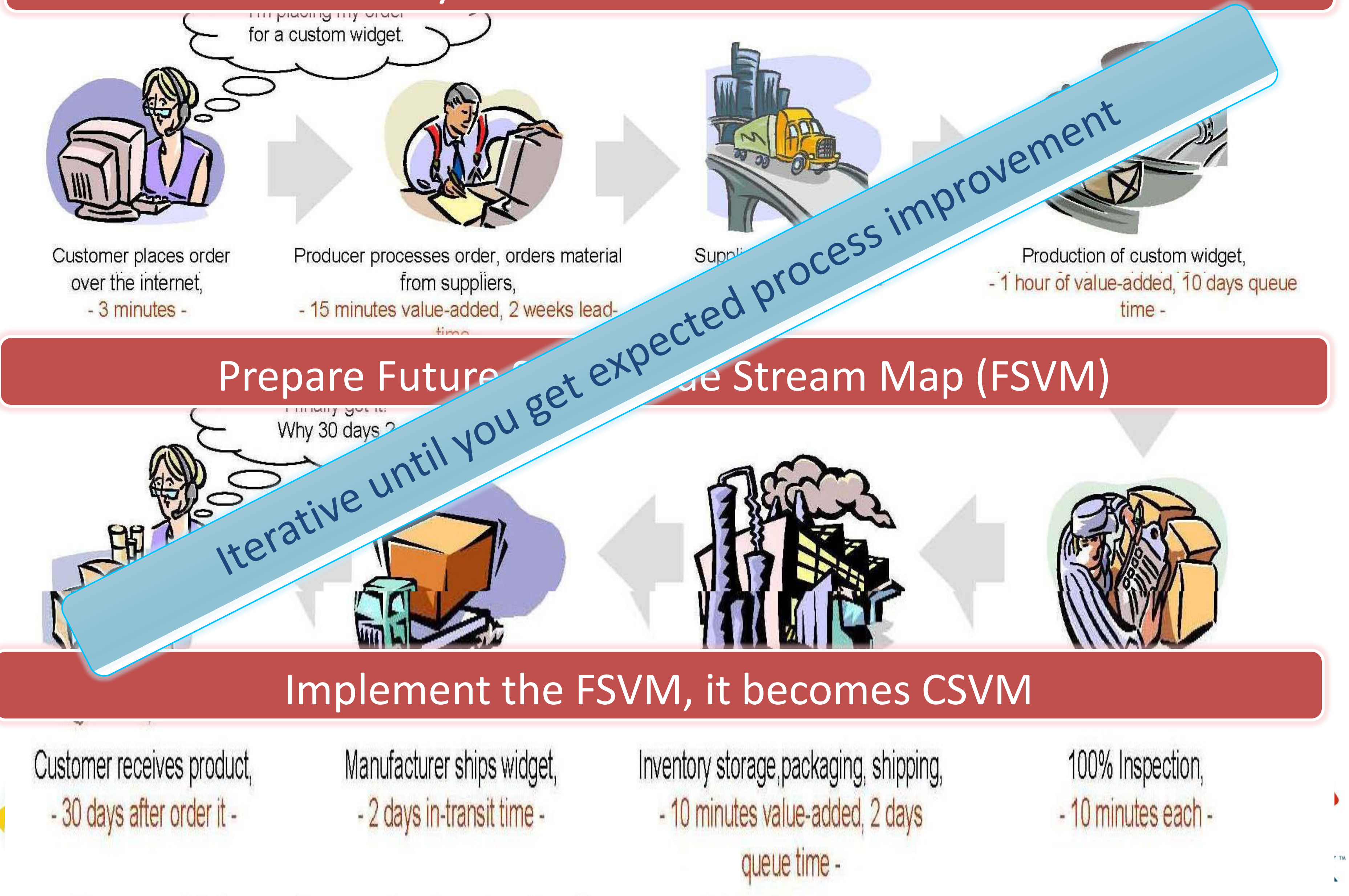
POS: Current State Value Stream Map (CSVM)

Identify all Non Value Added Activities

Prepare Future State Value Stream Map (FSVM)

Iterative until you get expected process improvement

Implement the FSVM, it becomes CSVM



for a custom widget.
Customer places order over the internet, - 3 minutes -
Producer processes order, orders material from suppliers, - 15 minutes value-added, 2 weeks lead-
Suppl
Production of custom widget, - 1 hour of value-added, 10 days queue time -
Why 30 days ?
Customer receives product, - 30 days after order it -
Manufacturer ships widget, - 2 days in-transit time -
Inventory storage,packaging, shipping, - 10 minutes value-added, 2 days queue time -
100% Inspection, - 10 minutes each -

LEAN Applied

- **Value Add**
  - How long does it take to get a reimbursement?
  - How long does it take for generating a Purchase Order?
  - How long does it take for a new employee to get billable
  - How long it takes for my external vendors to get the payment for his service provided?
- Lean drives Elimination of waste
- Is Goal Oriented Rather then tool oriented

Kanban

To Do
In Progress
Done
USE KANBAN
SUBSCRIBE TO KanbanTool
LEARN ABOUT KANBAN
GET SOME STICKY NOTES

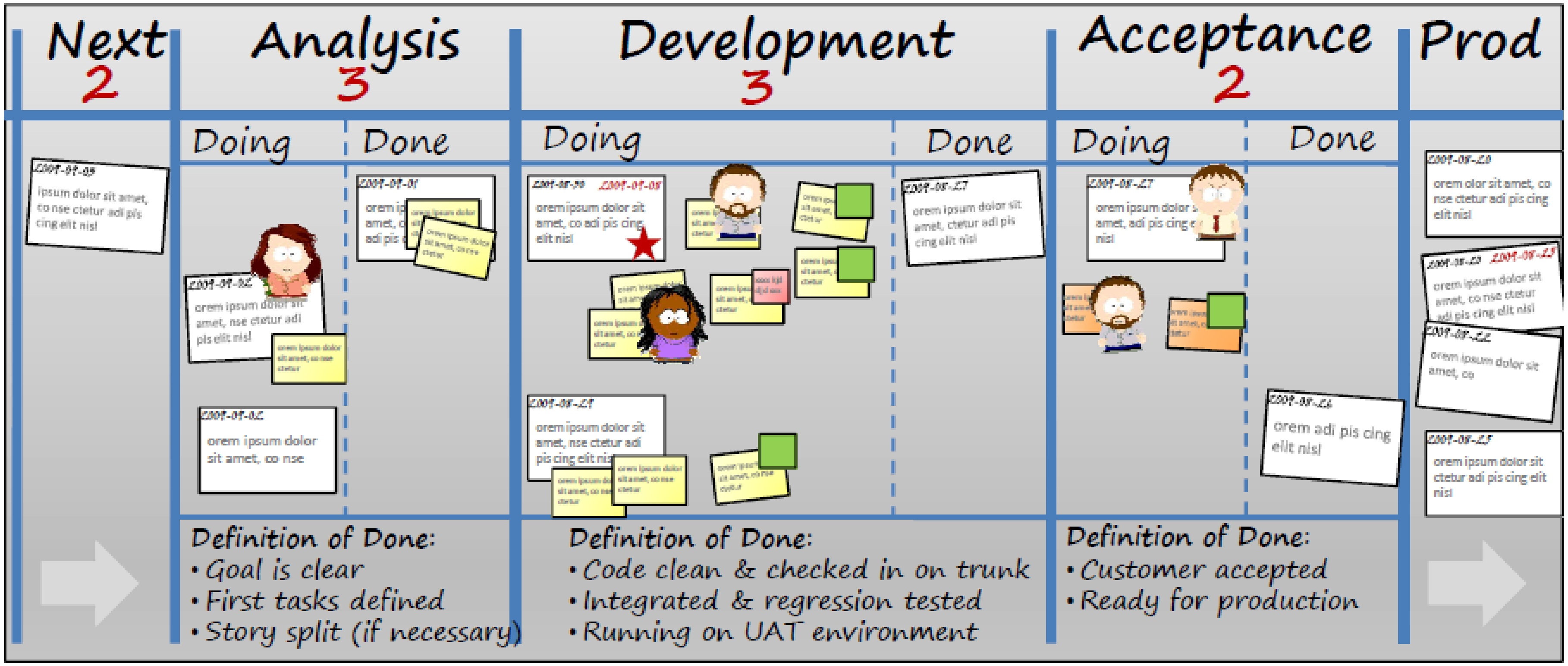
Kanban

- Kanban:
  - The name 'Kanban' originates from Japanese, and translates roughly as "signboard".
  - Kanban traces back to the early days of the Toyota production system.
- Kanban development adapted by David J. Anderson.
- It is an approach to incremental, evolutionary process and systems change for organizations.

Kanban in a Nutshell

- Visualize the Workflow
- Split the work in to pieces, write each item on a card and put on the wall.
- Use named columns to illustrate where each item is in the workflow.
- Limit Work in progress
  - Assign explicit limits to how many items may be in progress at each workflow state.
- Measure the lead times
- Optimise to reduce lead times.

Kanban: Day to Day

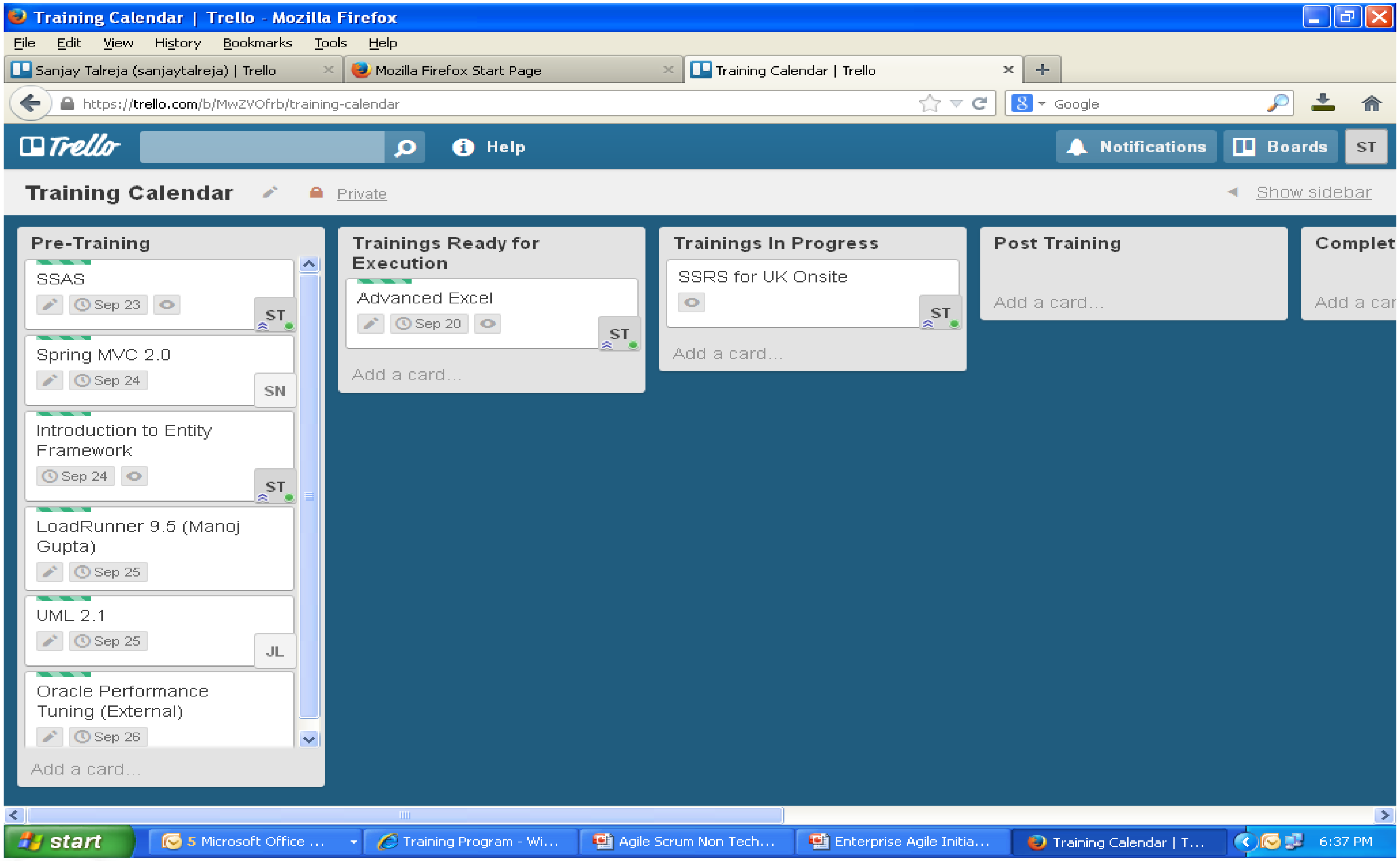


How do I go Agile?

I'm OK using SCRUM to manage my parents' visit.
But, why is there a user story for 'Fighting with Mother-in-law' ?!
To Do
In Process
Done
PJ
WWW.ITWORLD.COM

Tools for Agile: Manual Task Board

Tools for Agile: Trello



Training Calendar | Trello - Mozilla Firefox

https://trello.com/b/MwZVOfrb/training-calendar

Training Calendar · Private

Pre-Training
- SSAS — Sep 23 — ST
- Spring MVC 2.0 — Sep 24 — SN
- Introduction to Entity Framework — Sep 24 — ST
- LoadRunner 9.5 (Manoj Gupta) — Sep 25
- UML 2.1 — Sep 25 — JL
- Oracle Performance Tuning (External) — Sep 26

Add a card...

Trainings Ready for Execution
- Advanced Excel — Sep 20 — ST

Add a card...

Trainings In Progress
- SSRS for UK Onsite — ST

Add a card...

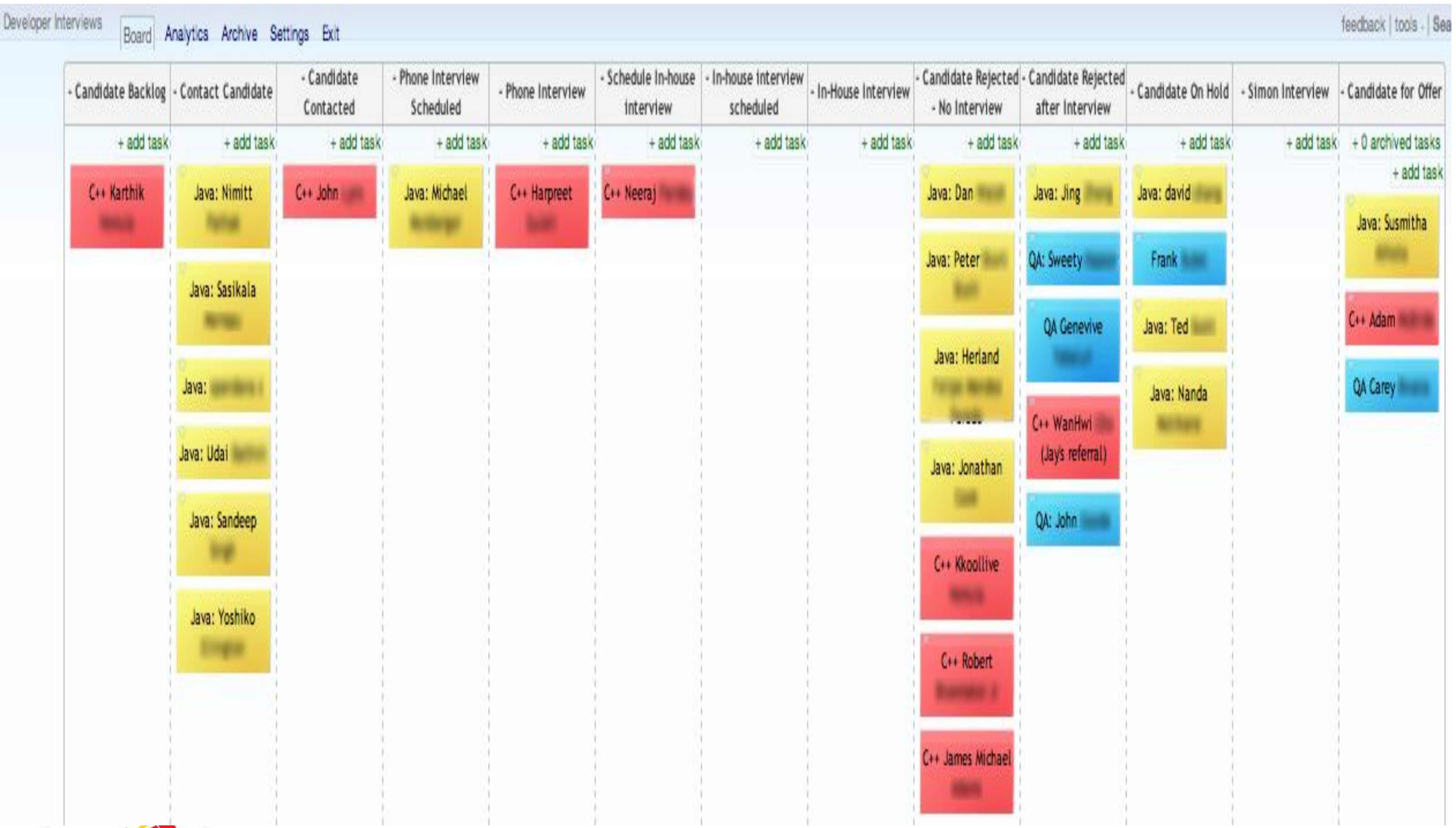
Post Training

Add a card...

Complet

Add a car

Kanban Board for a Recruitment Process



Developer Interviews   Board   Analytics   Archive   Settings   Exit   feedback | tools | Sea

- Candidate Backlog	- Contact Candidate	- Candidate Contacted	- Phone Interview Scheduled	- Phone Interview	- Schedule In-house interview	- In-house interview scheduled	- In-House Interview	- Candidate Rejected - No Interview	- Candidate Rejected after Interview	- Candidate On Hold	- Simon Interview	- Candidate for Offer
+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ add task	+ 0 archived tasks + add task
C++ Karthik [redacted]	Java: Nimitt [redacted]	C++ John [redacted]	Java: Michael [redacted]	C++ Harpreet [redacted]	C++ Neeraj [redacted]			Java: Dan [redacted]	Java: Jing [redacted]	Java: david [redacted]		Java: Susmitha [redacted]
	Java: Sasikala [redacted]							Java: Peter [redacted]	QA: Sweety [redacted]	Frank [redacted]		C++ Adam [redacted]
	Java: [redacted]							Java: Herland [redacted]	QA Genevive [redacted]	Java: Ted [redacted]		QA Carey [redacted]
	Java: Udai [redacted]							Java: Jonathan [redacted]	C++ WanHwi [redacted] (Jays referral)	Java: Nanda [redacted]		
	Java: Sandeep [redacted]							C++ Kkoollive [redacted]	QA: John [redacted]			
	Java: Yoshiko [redacted]							C++ Robert [redacted]				
								C++ James Michael [redacted]				

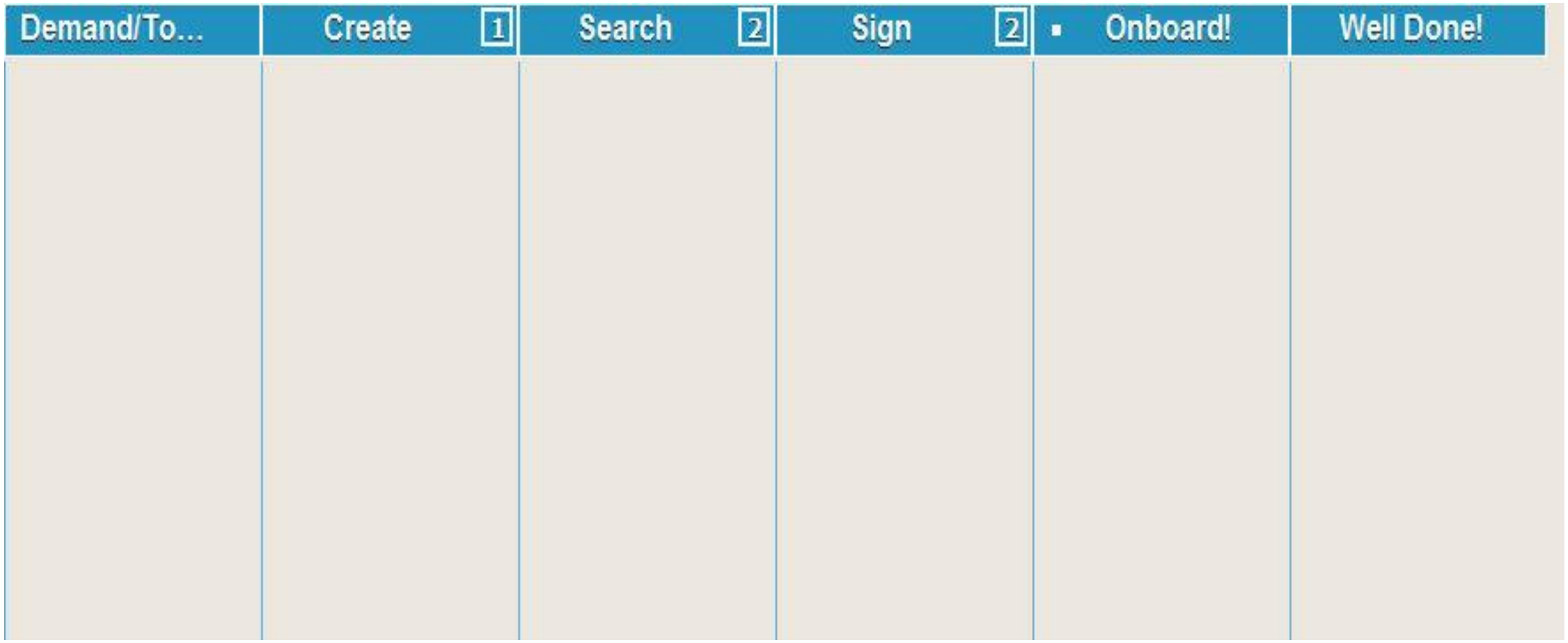
Kanban Board for a Recruitment Process

Current Workflow for Hiring:

- raise need for position ->
- create position description ->
- publish >
- filter candidates >
- best and final between a few top candidates >
- prepare offer >
- send offer >
- signed offer >
- prep for onboarding >
- onboard >
- 3 month – successful hire

Kanban Board for a Recruitment Process

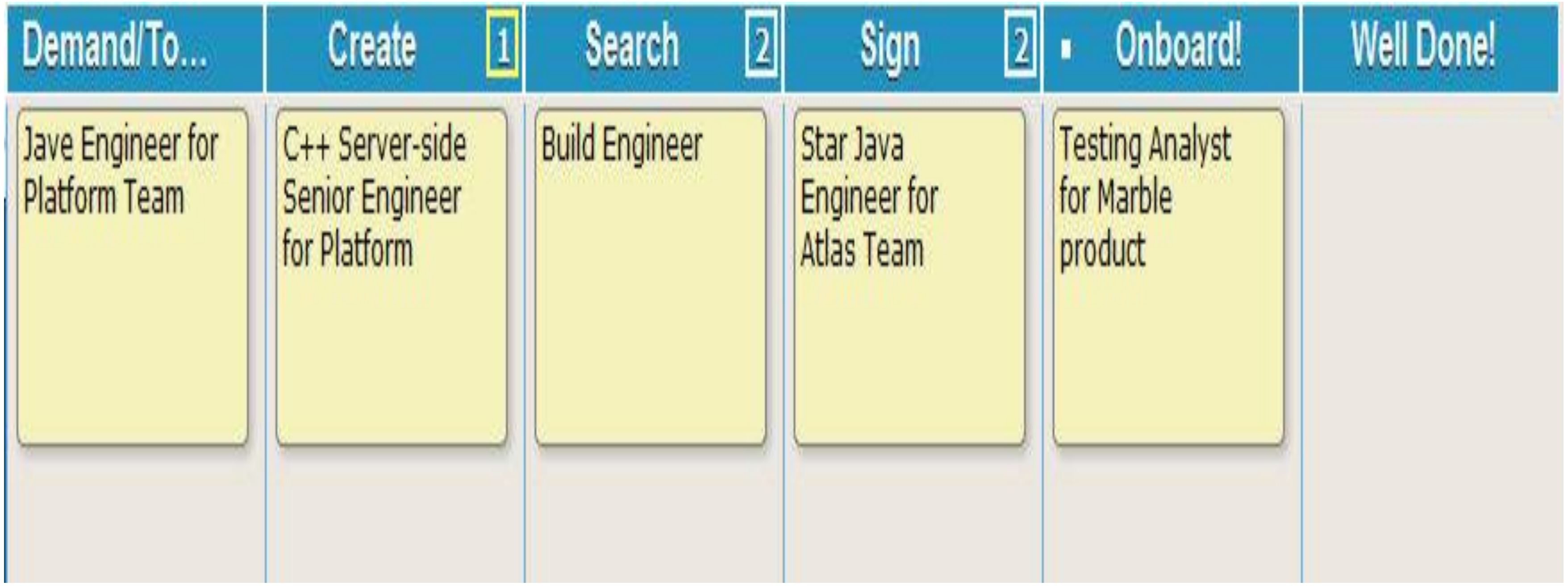
Step 1: Create a Simple Task Board with distinct phases



Demand/To...	Create 1	Search 2	Sign 2	▪ Onboard!	Well Done!

Kanban Board for a Recruitment Process

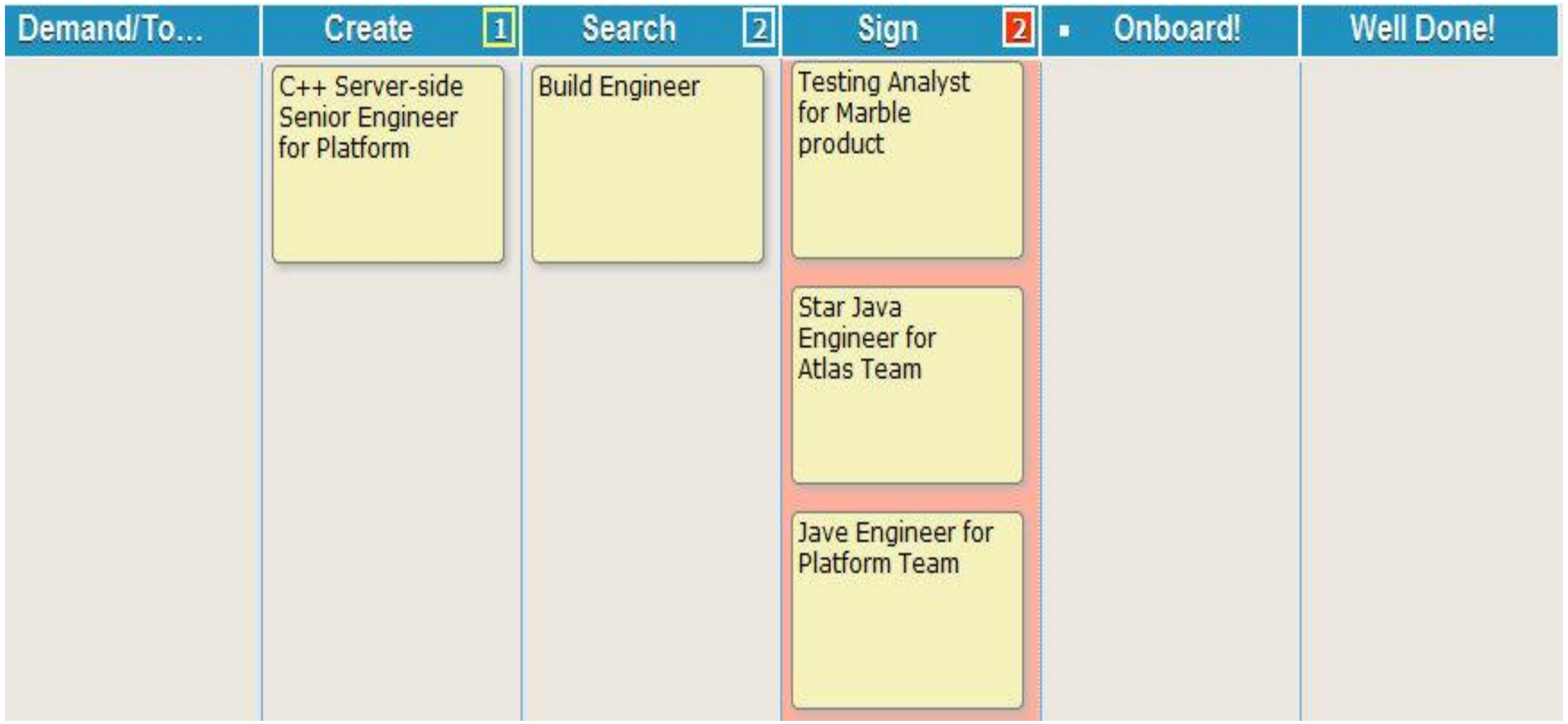
Step 2: Populate the board with the current work:



Demand/To...	Create 1	Search 2	Sign 2	Onboard!	Well Done!
Jave Engineer for Platform Team	C++ Server-side Senior Engineer for Platform	Build Engineer	Star Java Engineer for Atlas Team	Testing Analyst for Marble product	

Kanban Board for a Recruitment Process

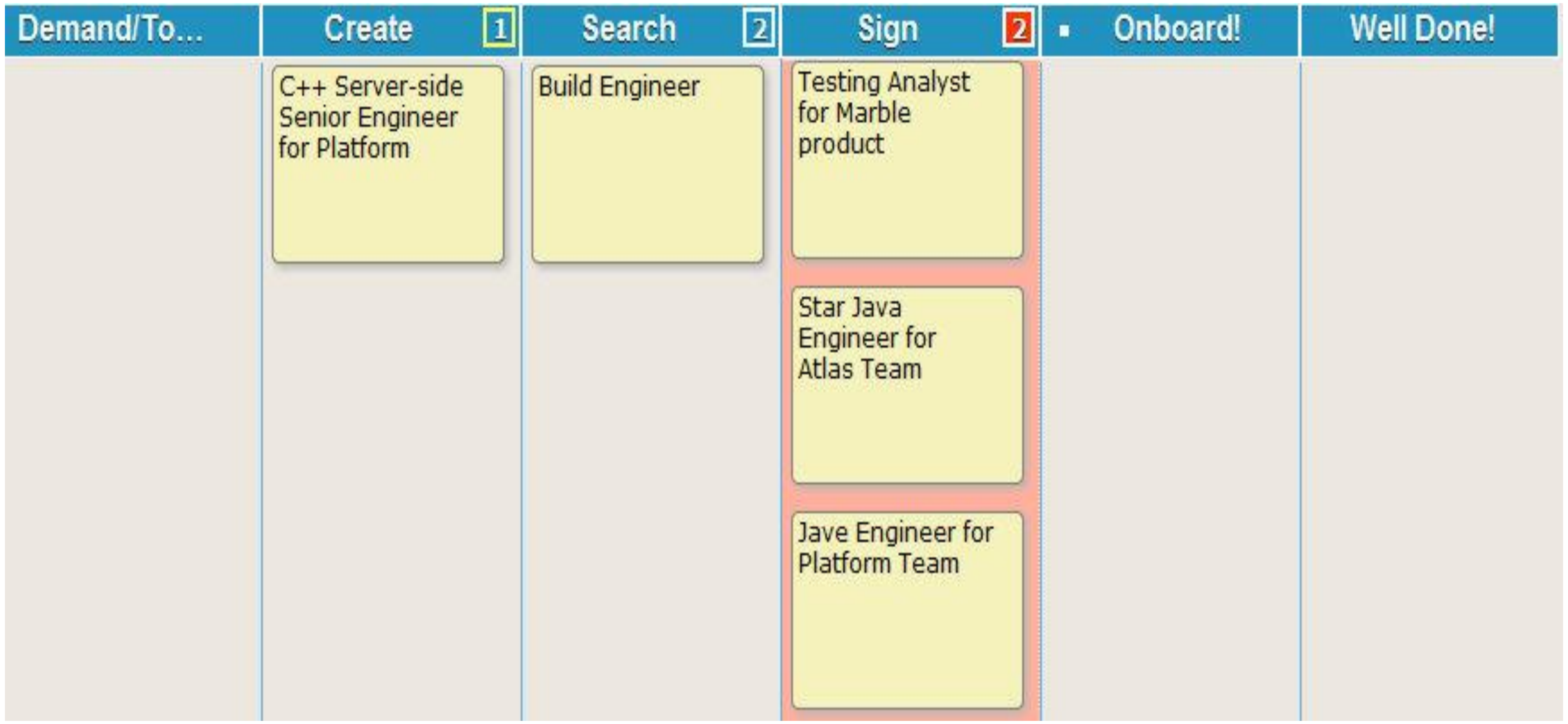
Step 3: Limit the work in Progress



Demand/To...	Create 1	Search 2	Sign 2	Onboard!	Well Done!
	C++ Server-side Senior Engineer for Platform	Build Engineer	Testing Analyst for Marble product		
			Star Java Engineer for Atlas Team		
			Jave Engineer for Platform Team		

Kanban Board for a Recruitment Process

Step 4: Manage the Flow



Demand/To...	Create 1	Search 2	Sign 2	▪ Onboard!	Well Done!
	C++ Server-side Senior Engineer for Platform	Build Engineer	Testing Analyst for Marble product		
			Star Java Engineer for Atlas Team		
			Jave Engineer for Platform Team		

Thank You

BE AGILE

FAQ’s For AGILE

FAQ’s

- Is agile doable in a fix bid project (cost is fixed, you decide resources and timeline)
  - Agile is a process model for delivering projects and not a costing model
  - Something should be variable in a project either the duration or team size etc..
  - Helps to plan better and meet the duration decided
  - T&M projects will see more benefits of Agile than Fixed bid

- Is agile doable in a fix bid project (cost is fixed, you decide resources and timeline)
  - Agile is a process model for delivering projects and not a costing model

FAQ’s

- Is Incremental and Iterative same
  - Iterative is doing tome more he same thing again and again
  - Incremental doing some more each time
  - They are both the same thing
- Does Agile bring down project cost
  - Agile is a delivery model and not a costing model
  - Agile may be more expensive
  - Both costing and agile are not related
- Does Agile proposal impact the customer if it is fixed cost/bid
  - If existing contract is in waterfall, besides implementing Scrum we cannot do much.
  - It does allow you to plan better with small sprints delivery

FAQ’s

- Are we informing customer about agile way of working
  - There is some thought over this and a team working on it
  - Some POC’s form Pre sales team have gone out mentioning this
  - If existing contract is in waterfall, besides implementing Scrum we cannot do much.
- Is there some standardization for assigning story point in terms of complexity to use stories across Mastek
  - There is some work of standardization in progress . Nothing concrete yet

FAQ’s

- If we move pending work to the next sprint, do we have the modify all the sprints. Will it impact release date
  - We do no plan that far for the entire project
  - We plan for a release
  - In the sprint meetings we can keep modifying the duration , amount of user stories to be taken up in that sprint
  - Hence the entire process is flexible
- Product Owner does not attend Scrum meetings
  - Can attend if he wishes