Agile Project Management
Why Agile?
What is Agile Project Management?

• Agile project management is an approach based on:
  • delivering requirements iteratively and
  • incrementally throughout the project life cycle.

• At the core of agile is the requirement to exhibit central values and behaviours of
  • trust
  • Flexibility
  • empowerment and collaboration.
Characteristics of Agile

• Agile project's defining characteristic is that it produces and delivers work in short bursts (or sprints) of anything up to a few weeks.

• These are repeated to refine the working deliverable until it meets the client's requirements.

• Unlike the traditional waterfall project management Agile starts work with a rough idea of what is required and by delivering something in a short period of time, clarifies the requirements as the project progresses.

• Collaborative relationships are established between stakeholders and the team members delivering the work.
• Agile projects need **documentation, reviews and processes** just as traditional projects do to meet requirements, manage costs and schedules, deliver benefits and avoid scope creep;

• Agile **does not expect to fully understand the requirements** before work can begin.

• Instead it **emphasises the importance of delivering a working product as something tangible** for the client that can then be refined until it fulfils the client's needs.

• The key **measure of project progress is this series of working deliverables**.
Pareto Principle

• 20% of the User Stories (functional work) probably contain 80% of the customer value. So find them and do those first

• Find the 20 percent that delights customers, deliver them, and repeat.
Prioritisation - MoSoCoW

- **Must** - Cannot deliver/go live without this.
- **Should** - Important but not vital
- **Could** - Wanted or desirable but less important. ”nice to haves”
- **Won’t** - Team has agreed it will not deliver
Principles of Agile

- Customer collaboration over contract negotiation
- Individuals and interaction over process and tools
- Responding to change over following a structured plan
- Prototyping/working solutions over comprehensive documentation
Benefits to Business/Clients

- Increased flexibility
- Increased transparency
- Speed up delivery
- Improve quality
- Satisfy stakeholders and realise efficiencies
- Decreased risk of missed objectives
Benefits to Project Management

- Reduced waste through minimization of resources
- Increased flexibility and adaptability to change
- Faster turnaround times
- Faster detection of issues and defects
- A Optimal project control
- Increased focus on specific customer needs
- Increased frequency of collaboration and feedback
- Builds accountability
- Encourages diversity of ideas
Agile Methodologies

• Scrum
• Lean
• Kanban
• DSDM
Agile Project Management with SCRUM

• Jeff Sutherland created the scrum process in 1993, he borrowed the term "scrum" from an analogy put forth in a 1986 study by Takeuchi and Nonaka, published in the Harvard Business Review.

• In that study, Takeuchi and Nonaka compare high-performing, cross-functional teams to the scrum formation used by Rugby teams.

• Scrum is the leading agile development methodology, used by Fortune 500 companies around the world.
SCRUM Values

COURAGE
Scrum Team members have courage to do the right thing and work on tough problems.

FOCUS
Everyone focuses on the work of the Sprint and the goals of the Scrum Team.

COMMITMENT
People personally commit to achieving the goals of the Scrum Team.

RESPECT
Scrum Team members respect each other to be capable, independent people.

OPENNESS
The Scrum Team and its stakeholders agree to be open about all the work and the challenges with performing the work.

Scrum Values © 2017 Scrum.org
Detailed Framework

- **Vision**
- **Product Backlog**
  - Prioritized Features desired by Customer

- **Sprint Backlog**
  - Features assigned to Sprint
  - Estimated by team

- **Sprint Planning Meeting**
  - Review Product Backlog
  - Sprint Goal
  - Estimate Sprint Backlog
  - Commit to the Sprint

- **Daily Scrum**
  - Done since last meeting
  - Plan for today
  - Obstacles?

- **Sprint 1-4 weeks**

- **Sprint Review**
  - Demo features to all
  - Discuss what's done and what wasn't done

- **Sprint Retrospective**
  - Inspect and Adapt

- **Potentially Shippable Product Increment**
SCRUM Events

- Sprint
- Sprint Planning
- Daily Scrum
- Sprint Review
- Sprint Retrospective
SCRUM Roles

- **SCRUM Development Team** - consists of seven plus or minus two people who are jointly responsible for the delivery of the product.

- **Product Owner** - represents the voice of the customer and has the authority to make decisions about the product.

- **SCRUM Master** - is the keeper of the process, the advocate for the team, and the protector of the team.
SCRUM Development Team

• Cross-functional (e.g., includes members with testing skills, and others not traditionally called developers: business analysts, designers, domain experts, etc.)
• Self-organizing / self-managing, without externally assigned roles
• Plans one Sprint at a time with the Product Owner
• Has autonomy regarding how to develop the increment
• Intensely collaborative
• Most successful when located in one team room, particularly for the first few Sprints
• Most successful with long-term, full-time membership. Scrum moves work to a flexible learning team and avoids moving people or splitting them between teams.
• 6 ± 3 members
• Has a leadership role
Product Owner

• Single person responsible for maximizing the return on investment (ROI) of the development effort
• Responsible for product vision
• Constantly re-prioritizes the Product Backlog, adjusting any long term expectations such as release plans
• Final arbiter of requirements questions
• Decides whether to release
• Decides whether to continue development
• Considers stakeholder interests
• May contribute as a team member
• Has a leadership role
SCRUM Master

• Works with the organization to make Scrum possible
• Ensures Scrum is understood and enacted
• Creates an environment conducive to team self-organization
• Shields the team from external interference and distractions to keep it in group flow (a.k.a. the zone)
• Promotes improved engineering practices
• Has no management authority over the team
• Helps resolve impediments
• Has a leadership role
SCRUM Roles integrated with SCRUM Framework
SCRUM Roles integrated with SCRUM Framework
Tracking Progress - SCRUM Burndown Chart

- Effort (pts)
- Work remaining
- Total amount of work done
- New features
- Iterations
- Time

- Team velocity
- Work done that iteration
- Work remaining
- 160 pts
- Effort (pts)
- Time
- You are here

- Iterations
- I1 I2 I3
## Implication of Agile to Project Management

<table>
<thead>
<tr>
<th>Project Management Function</th>
<th>Implication</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning</td>
<td>Less formal, based on sprints</td>
</tr>
<tr>
<td>Scope</td>
<td>Collaborative and interactive approach to requirements as they are not fully known. Change is welcomed, scope creep is expected</td>
</tr>
<tr>
<td>Cost</td>
<td>Based on number of sprints and effort, iterative, bottom up</td>
</tr>
<tr>
<td>Quality</td>
<td>Early testing, continuous improvement</td>
</tr>
<tr>
<td>Project Team</td>
<td>Greater communication and collaboration</td>
</tr>
<tr>
<td>Knowledge Areas</td>
<td>Activities</td>
</tr>
<tr>
<td>------------------------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>Project Integration Management</td>
<td>Direct &amp; Manage Project Execution</td>
</tr>
<tr>
<td></td>
<td>Monitor &amp; Control Project Work</td>
</tr>
<tr>
<td></td>
<td>Perform Integrated Change Control</td>
</tr>
<tr>
<td>Project Scope Management</td>
<td>Collect Requirements</td>
</tr>
<tr>
<td></td>
<td>Define Scope</td>
</tr>
<tr>
<td></td>
<td>Create WBS</td>
</tr>
<tr>
<td></td>
<td>Verify Scope</td>
</tr>
<tr>
<td></td>
<td>Control Scope</td>
</tr>
<tr>
<td>Project Time Management</td>
<td>Define Activities</td>
</tr>
<tr>
<td></td>
<td>Sequence Activities</td>
</tr>
<tr>
<td></td>
<td>Estimate Activity Duration</td>
</tr>
<tr>
<td></td>
<td>Develop Schedule</td>
</tr>
<tr>
<td></td>
<td>Control Schedule</td>
</tr>
<tr>
<td>Project Quality Management</td>
<td>Plan Quality</td>
</tr>
<tr>
<td></td>
<td>Perform Quality Control</td>
</tr>
<tr>
<td>Project Cost Management</td>
<td>Estimate Costs</td>
</tr>
<tr>
<td>Project Human Resource</td>
<td>Develop Project Team</td>
</tr>
<tr>
<td>Management</td>
<td>Manage Project Team</td>
</tr>
<tr>
<td>Project Communication</td>
<td>Plan Communications</td>
</tr>
<tr>
<td>Management</td>
<td>Distribute Information</td>
</tr>
<tr>
<td></td>
<td>Report Performance</td>
</tr>
<tr>
<td>Project Risk Management</td>
<td>Monitor &amp; Control Risks</td>
</tr>
</tbody>
</table>
Comparison of Agile and Waterfall
Waterfall Principles

- Sequential steps - creating stage gates
- Strong project documentation
- Low customer involvement
Waterfall challenges

**Poor quality**

Gamut: Analysis, Design, Code, Test

Out of time? Cut here.

**Poor visibility**

Gamut: Analysis, Design, Code, Test

1/2 way?
Too risky

Houston we have a problem

Can't handle change

And finally, most importantly, it's just not a great way for handling change.

'I know what I really want!'
Differences between Agile and Waterfall

• Traditional ‘waterfall’ approaches will tend to treat scope as the driver and calculate the consequential time and cost;

• Whereas ‘agile’ commits set resources over limited periods to deliver products that are developed over successive cycles.
<table>
<thead>
<tr>
<th>WATERFALL</th>
<th>AGILE</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Detailed, long-term project plans with single timeline</td>
<td>• Shorter planning based on iterations and multiple deliveries</td>
</tr>
<tr>
<td>• Definitive and rigid project management and team roles</td>
<td>• Flexible, cross-functional team composition</td>
</tr>
<tr>
<td>• Changes in deliverables are discouraged and costly</td>
<td>• Changes in deliverables are expected and less impactfulful</td>
</tr>
<tr>
<td>• Fully completed product delivered at the end of the timeline</td>
<td>• Product delivered in functional stages</td>
</tr>
<tr>
<td>• Contract-based approach to scope and requirements</td>
<td>• Collaborative and interactive approach to requirements</td>
</tr>
<tr>
<td>• Customer is involved only at the beginning and end of a project</td>
<td>• Customer is involved throughout the sprint</td>
</tr>
<tr>
<td>• Linear-phased approach creates dependencies</td>
<td>• Concurrent approach seeks to reduce dependencies</td>
</tr>
</tbody>
</table>
Similarities of Agile and Waterfall

• They both have the same goal i.e. deliver a quality product in an effective and responsive manner

• Same Principles:
  • plan the work out completely before beginning
  • lock down requirements early
  • institute multiple reviews
  • move forward in a step-by-step, sequential manner
  • move forward only when all parts of the previous steps were complete
  • capture all details with extensive documentation
• They both work on the same Project Management functions:
  • Cost
  • Scope
  • Time/Schedule
  • Quality
• They are both based on the same development stages:
  • Analyze- the requirement
  • Design- a capability to satisfy the requirement
  • Build- the capability
  • Test- the capability to ensure the requirement is met
  • Deploy- the capability
Hybrid Methodologies

• Opportunity exists to combine agile with other methodologies such as waterfall to create a hybrid solution.

• **Waterfall**- waterfall can be used for **Planning**, where rapid or repetitive steps are not required

• **Agile**- Once a project enters the **development phase**, rapid and repetitive changes require a different approach and this is where Agile kicks in to deliver the best results in the shortest amount of time.
Using Waterfall and Agile
Benefits of Hybrid
References


