TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

Project management

Viet-Trung Tran

trungtv.github.io

School of Information and Communication Technology

Outline

Building an information system

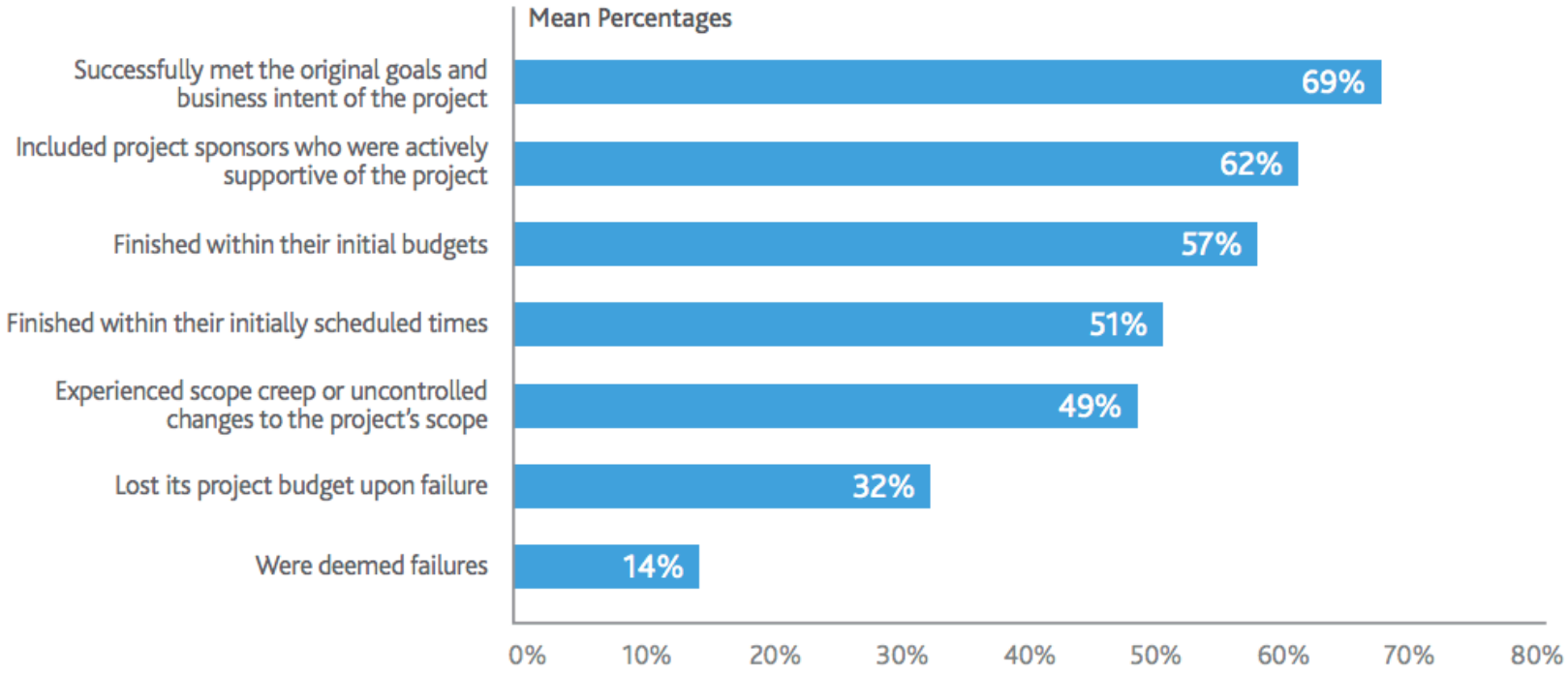
- A project is a planned undertaking with a start and finish, that produces an end result
- Should include all the activities needed to design, develop, and launch a new system
  - Identify
  - Plan
  - Organize
  - monitor

Classification of projects

- Successful
  - completed on time
  - within budget
  - meets users' requirements for functionality
- Challenged
  - either late or over budget
  - reduced in scope
- Failed
  - cancelled, or
  - system never used

Some facts

Q: In your estimation, what percentage of the projects completed within your organization in the past 12 months…?



- Most IT projects are less than successful as measured by
  - finishing on time
  - finishing within budget
  - meeting the need based on the original problem definition

Project success rates

Development paradigm	Successful	Challenged	Failed
Ad hoc	50%	35%	15%
Traditional	49%	32%	18%
Lean	72%	21%	7%
Agile	64%	30%	6%
Iterative	65%	28%	7%

(c) http://www.drdobbs.com/architecture-and-design/the-non-existent-software-crisis-debunki/240165910

Project success factors?

- Some primary reasons that projects fail
  - Undefined project management practices
  - Poor IT management and procedures
  - **Inadequate executive support for the project**
  - **Inexperienced project managers**
  - Unclear business needs and project objectives
  - **Inadequate user involvement**
- Some reasons that projects succeed:
  - Clear system requirement definitions
  - Substantial user involvement
  - Support from upper management
  - Thorough and detailed project plans
  - Realistic work schedules and milestones

Project Management

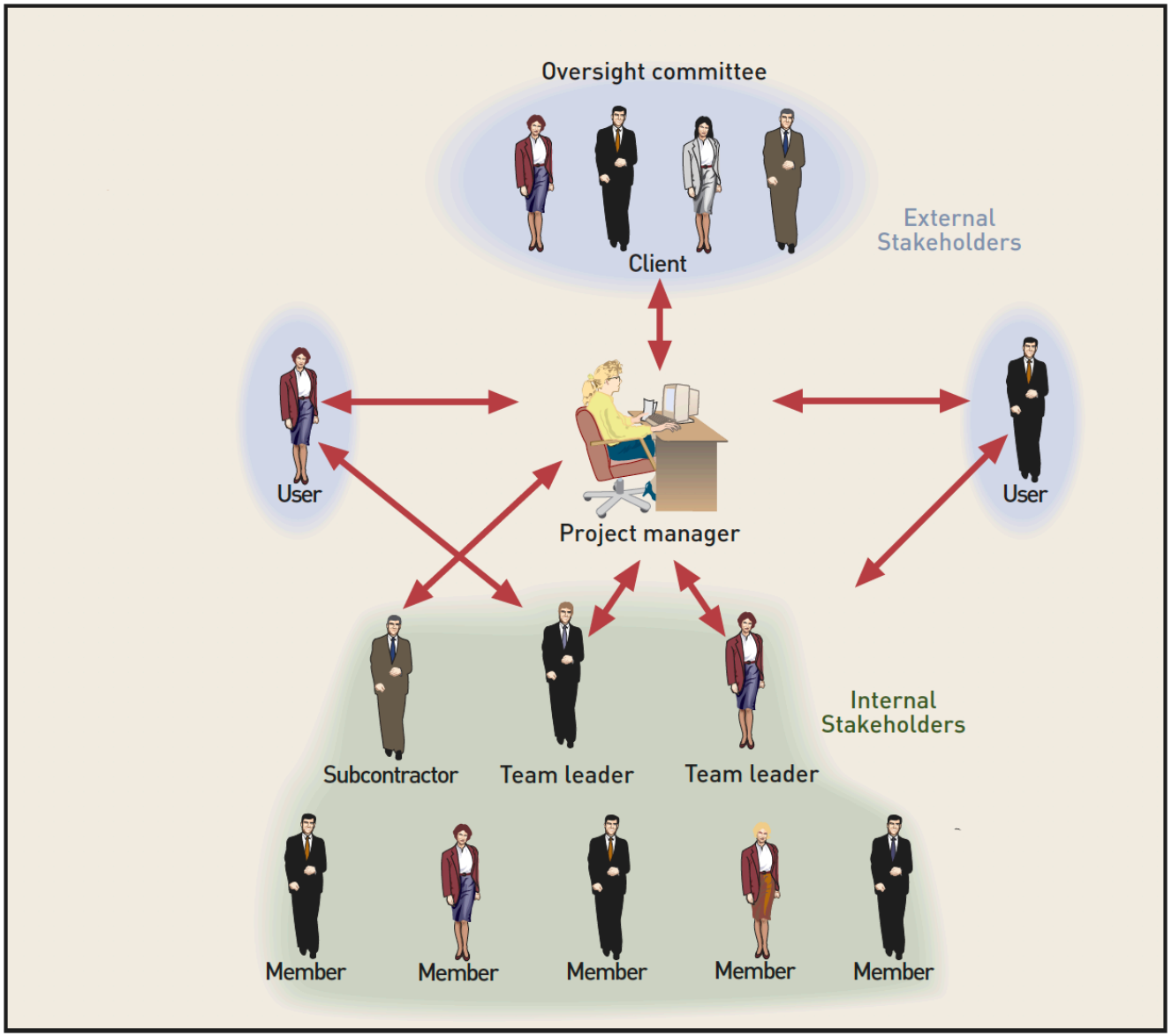
- Project management is organizing and directing other people to achieve a planned result within a predetermined schedule and budget
  - Small projects are hard to manage
  - This challenge is multiplied for large, complex projects

Project Manager - Internal roles

- manage people and resources
- serves as centre-point of the team
  - develop project schedule
  - recruiting and training
  - assigning tasks to teams and team members
  - assess project risks
  - monitory and control project deliverables

Project Manager - External roles

- Conduct public relations
- serves as relay to outside
  - report project's status and progress
  - work directly with client and other stakeholders
  - identifies resources needs
  - obtains resources


Oversight committee
Client
External Stakeholders
User
User
Project manager
Subcontractor
Team leader
Team leader
Internal Stakeholders
Member
Member
Member
Member
Member

Roles

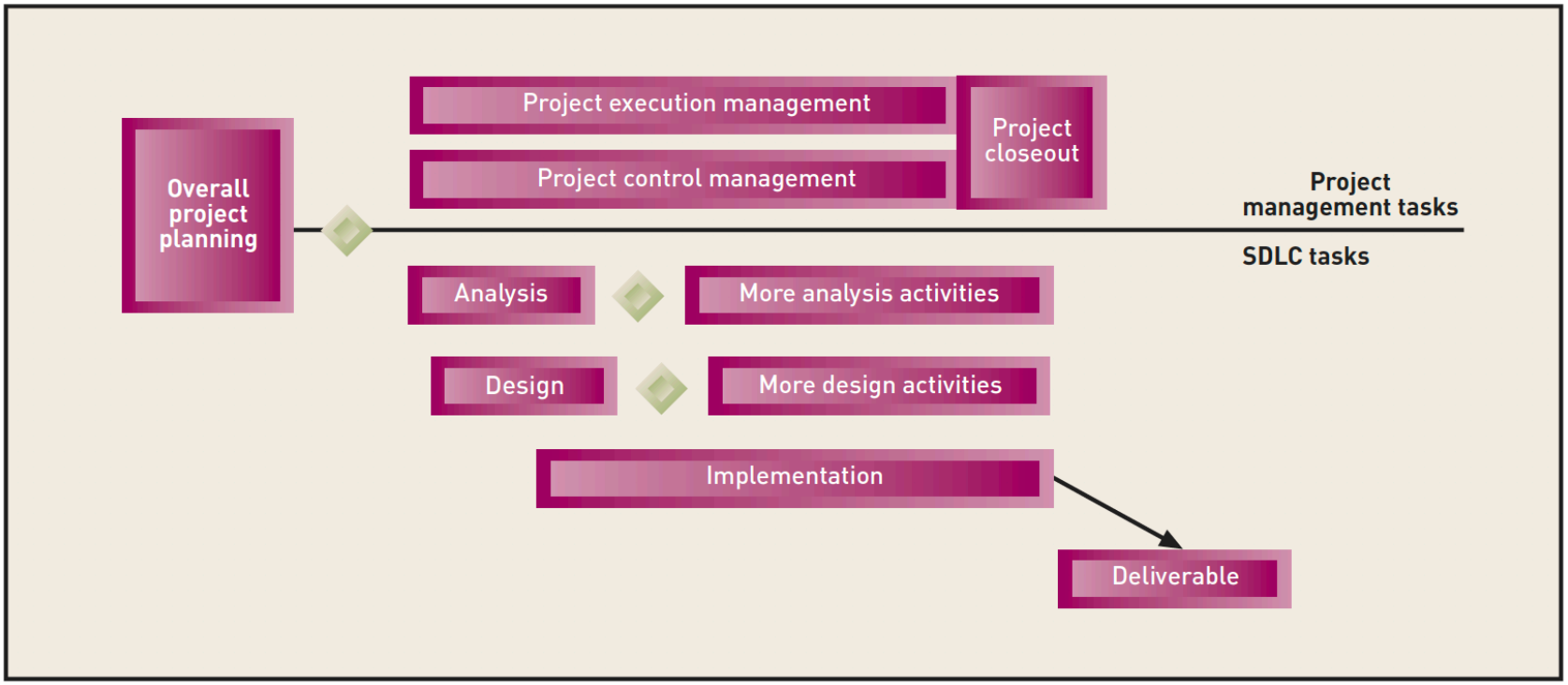
- Project Manager
  - develops the project schedule
  - recruits and trains team members
  - assigns works to teams
  - assesses project risks
  - monitors and controls project deliverables and milestones
- Team Leaders
  - facilitates members of their team
  - looking forward for road blocks
  - access to resources
  - facilitate communication

Roles [2]

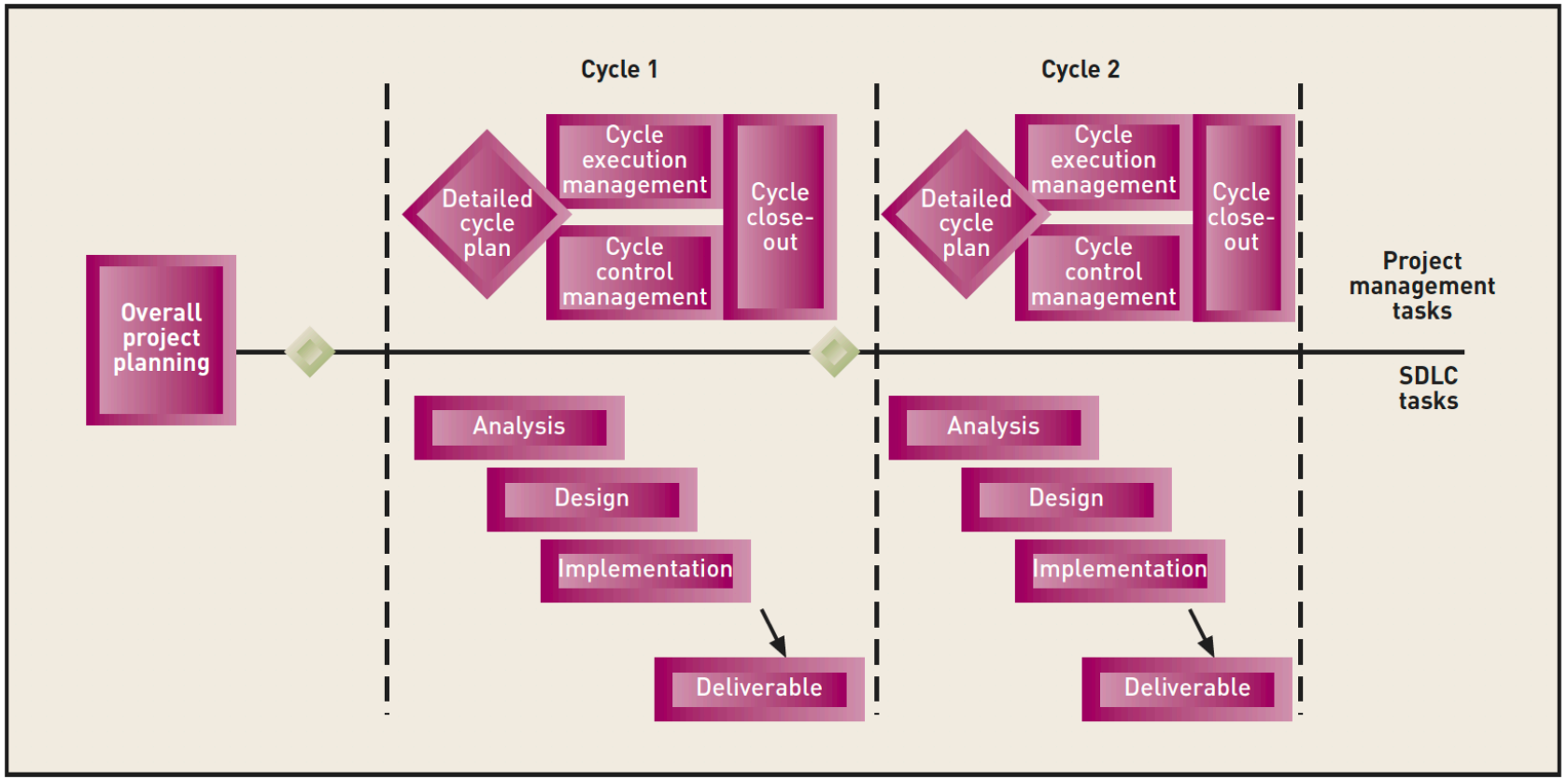
- Developers
  - responsible for delivering the product, based on the specifications
  - often involved in developing the specification too
  - prototyping
- QA
  - verifies the system meets the specs.
  - should be actively involved in development
- Documentation
  - creates instructional materials
  - documents changes to the spec (from the user's perspective)

Project management and SDLC tasks for a predictive project

- project planning involves both project management and SDLC tasks, because planning for a project requires participation by both the key team members and the project manager

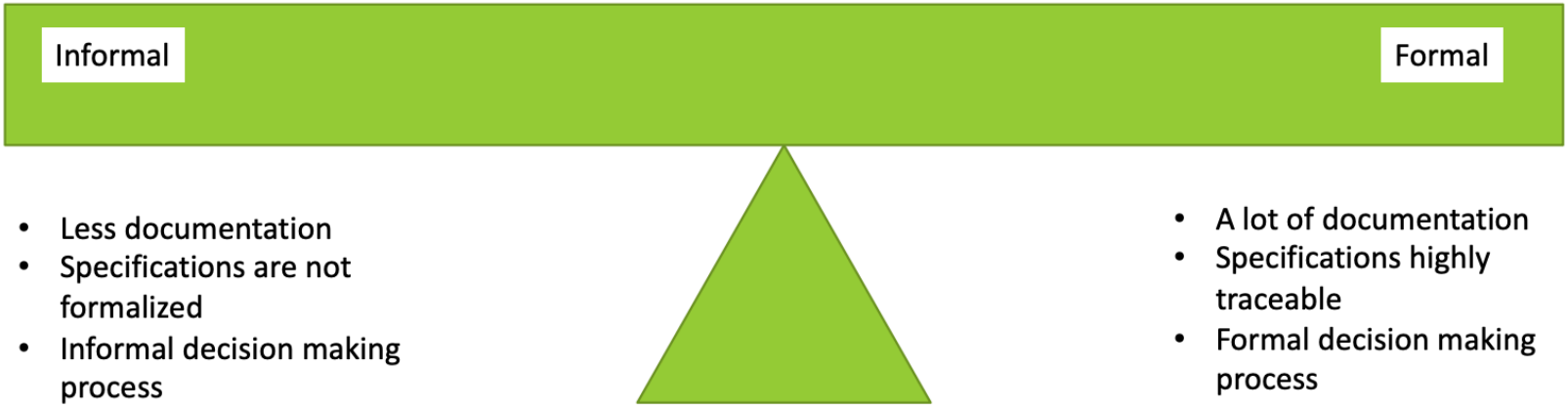


Project management and SDLC tasks for an adaptive project



Formality and Ceremony of Projects

- What level of formality is right?
  - how much documentation?
  - how traceable are the specifications?
  - how formal is the decision-making process?



Agile Software Development

- Agile Software Development is a philosophy of software development that embraces flexibility and agility
- Traditional Software Development
  - Processes and tools
  - Comprehensive documentation
  - Contract negotiation
  - Following a plan
- Agile Development
  - Individuals and Interactions
  - Working Software
  - Customer Collaboration
  - Responding to change

Agile's 12 principles

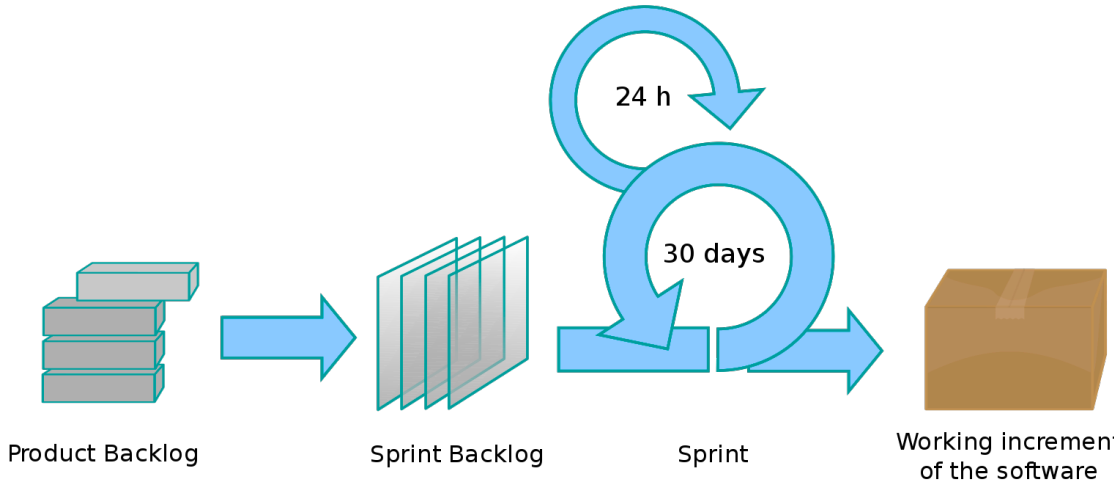
1. Customer satisfaction by early and continuous delivery of valuable software
2. Welcome changing requirements, even in late development
3. Working software is delivered frequently (weeks rather than months)
4. Close, daily cooperation between business people and developers
5. Projects are built around motivated individuals, who should be trusted
6. Face-to-face conversation is the best form of communication (co-location)
7. Working software is the primary measure of progress
8. Sustainable development, able to maintain a constant pace
9. Continuous attention to technical excellence and good design
10. Simplicity—the art of maximizing the amount of work not done—is essential
11. Best architectures, requirements, and designs emerge from self-organizing teams
12. Regularly, the team reflects on how to become more effective, and adjusts accordingly

Agile Development

- Small, quick increments
  - 1-4 weeks
- Minimal up-front planning and design
- All functions take place at each iteration
  - planning, analysis, design, coding, unit testing, acceptance testing
- All teams include a customer representative Daily stand-up (or scrum)
- At end of each iteration product is demo'd to stakeholders

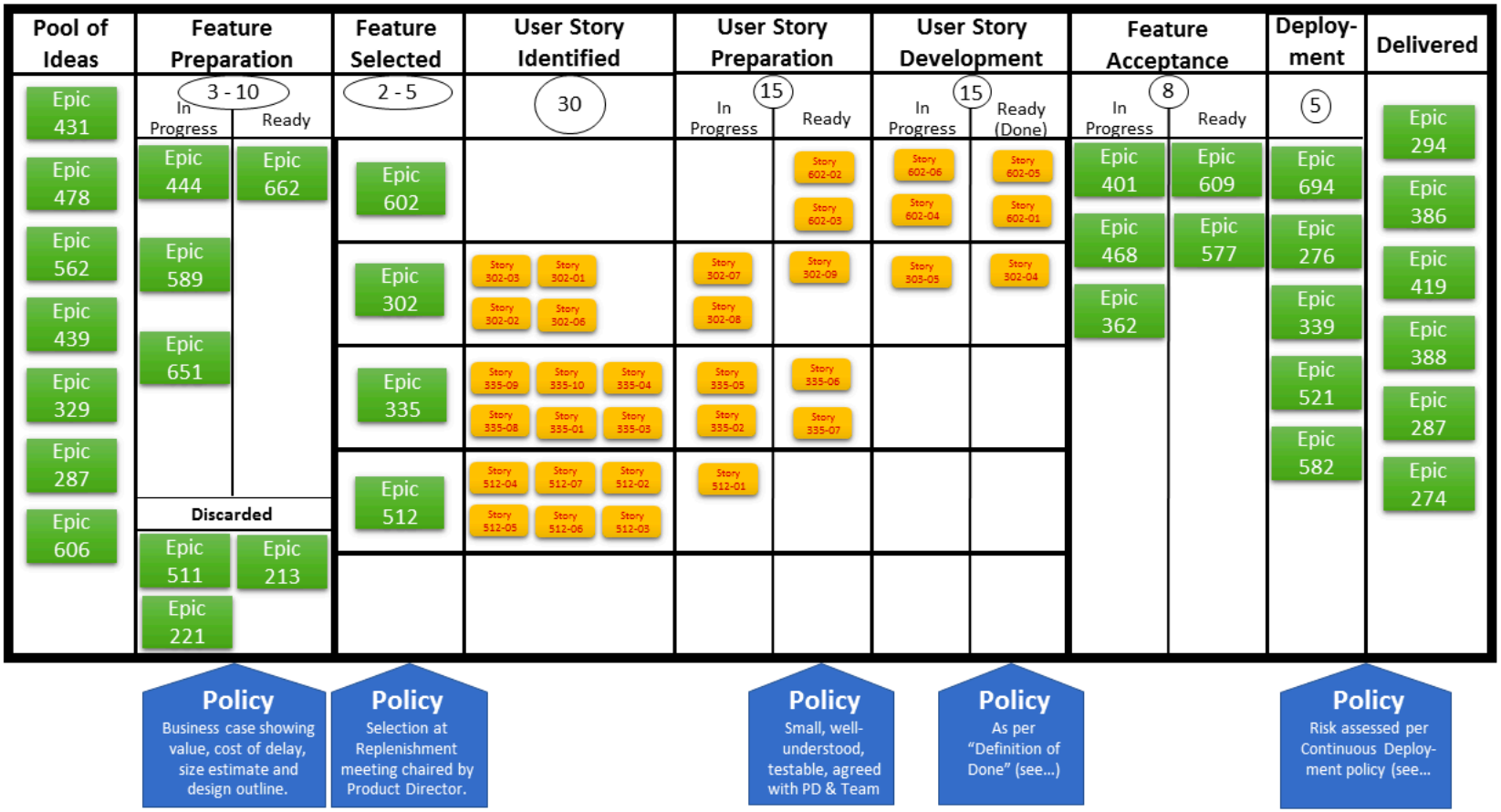
Scrum

- for teams of 3-9 developers
- 2 week cycles (sprints)
- daily meeting
- deliverable software at end of each cycle



Kanban

- visualization technique
- helps spot bottlenecks
- members pull work when they have capacity



Manifesto for Half-Arsed Agile Software Development

We have heard about new ways of developing software by paying consultants and reading Gartner reports. Through this we have been told to value:

Individuals and interactions over processes and tools
and we have mandatory processes and tools to control how those individuals (we prefer the term 'resources') interact

Working software over comprehensive documentation
as long as that software is comprehensively documented

Customer collaboration over contract negotiation
within the boundaries of strict contracts, of course, and subject to rigorous change control

Responding to change over following a plan
provided a detailed plan is in place to respond to the change, and it is followed precisely

That is, while the items on the left sound nice in theory, we're an enterprise company, and there's no way we're letting go of the items on the right.

Cobbled together one Saturday morning before breakfast by Kerry Buckley (@kerryb), following an article by Ron Jeffries and this suggestion from Eastmad.

Thank you for your attention!
Q&A