CSCI375
Project Management I

Outline

Project management

- The need for Project Management
- Meetings
- Team Members and roles

Project Management

Based on your experience, list the 3 main factors (each) that lead to a project's:

- success?
- failure?

Project Management

Small projects are hard to manage (we already know this!!)

This challenge is multiplied for large, complex projects

Kinds of Project

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

Phoenix payroll system doomed from the

Liberal government commissioned a report to take a 'deep dive' into Phoenix

By Julie Ireton, CBC News Posted: Oct 05, 2017 12:36 PM ET | Last Updated: Oct 05, 2017 4:32 PM ET

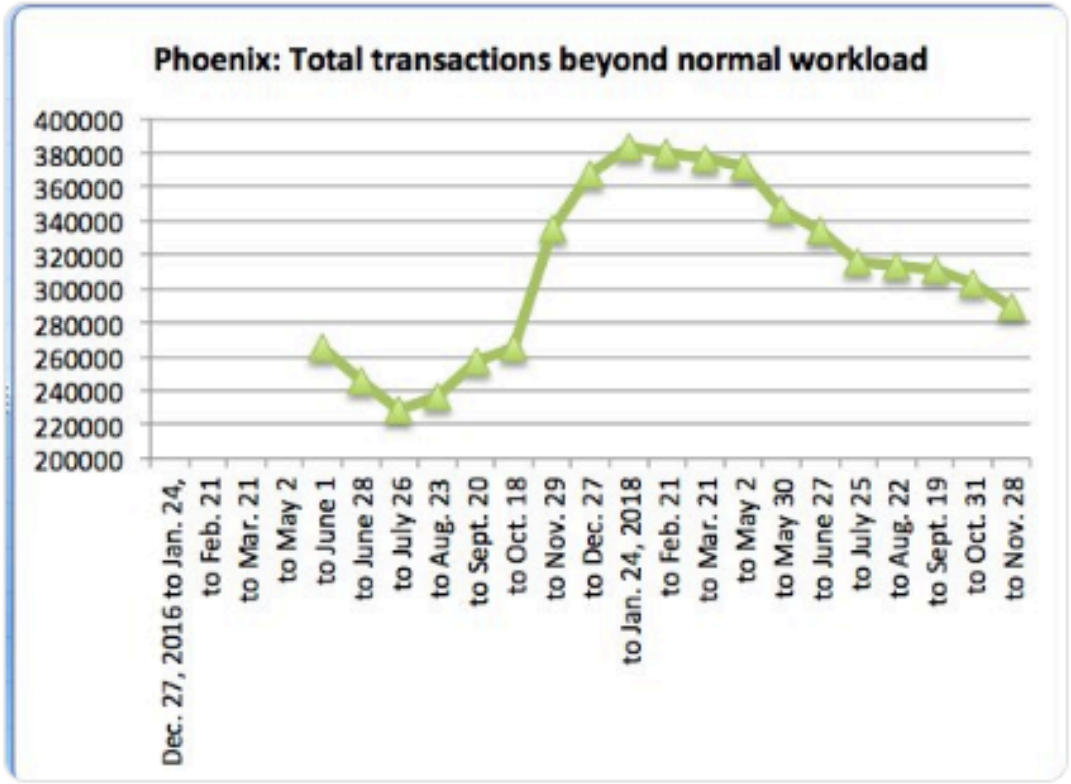
In August 2016, federal workers protested outside a hotel where the prime minister and federal cabinet ministers were meeting to 'remind' them to fix the payroll system. (Jeremy Eaton/CBC)

A new report published Thursday says a lack of proper definition, oversight and accountability plagued the government's failed Phoenix payroll system from the start.

James Bagnall
@JamesBagnall1

Follow

Well, it seems the federal government is finally digging into that massive PhoenixPay backlog, but progress is still painfully slow considering the hundreds of extra hires. Here's the latest snapshot of total pay transactions still to be dealt with beyond normal workload.



11:00 AM · 5 Dec 2018

Project management

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

Watch: https://www.youtube.com/watch?v=xh7P1WNV0ME

Primary reasons for project failure

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

Primary reasons for project failure

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

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%

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

Project Management

What does a project manager do?

- organizes and directs
- helps the team stay on schedule
- keeps project on track

Project Manager

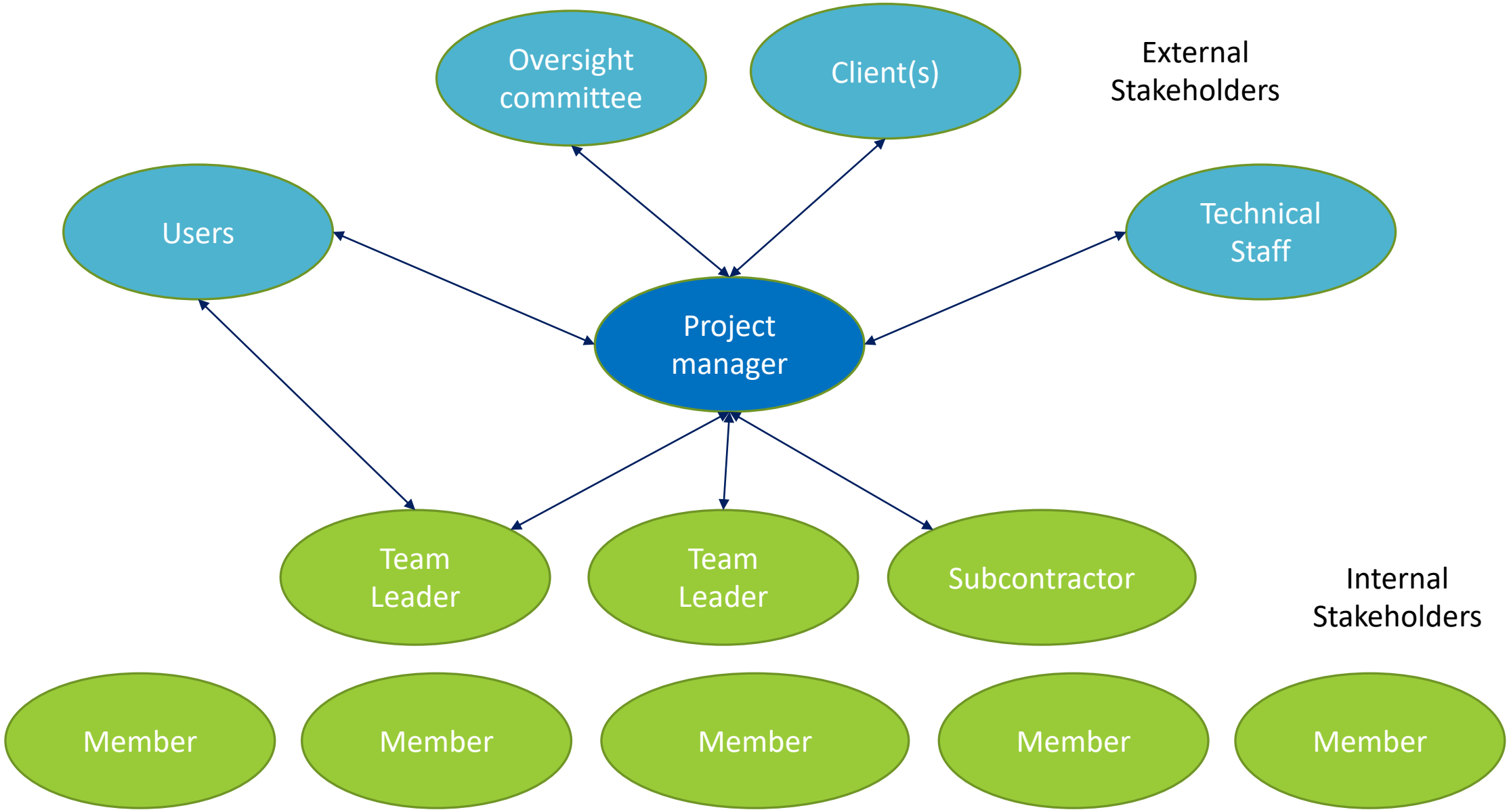
Internal roles

- 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

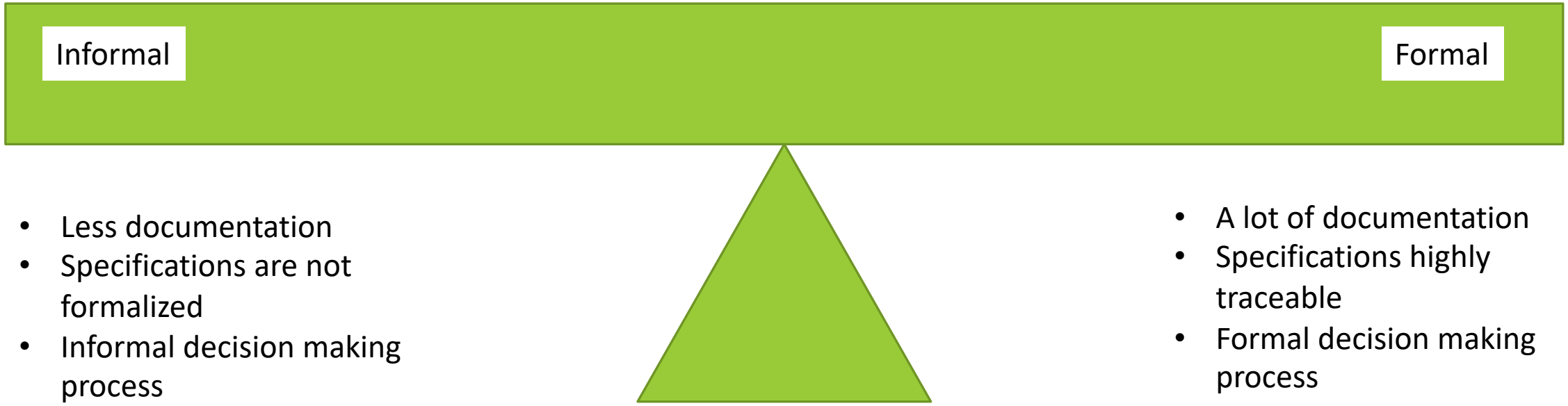
Project Manager

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(s)
External Stakeholders
Users
Technical Staff
Project manager
Team Leader
Team Leader
Subcontractor
Internal Stakeholders
Member
Member
Member
Member
Member

Formality and Ceremony of Projects



- Less documentation
- Specifications are not formalized
- Informal decision making process

- A lot of documentation
- Specifications highly traceable
- Formal decision making process

Formality and Ceremony

What level of formality is right?

- how much documentation?
- how traceable are the specifications?
- how formal is the decision-making process?

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

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)

Software Development Cycles

What to do, when?

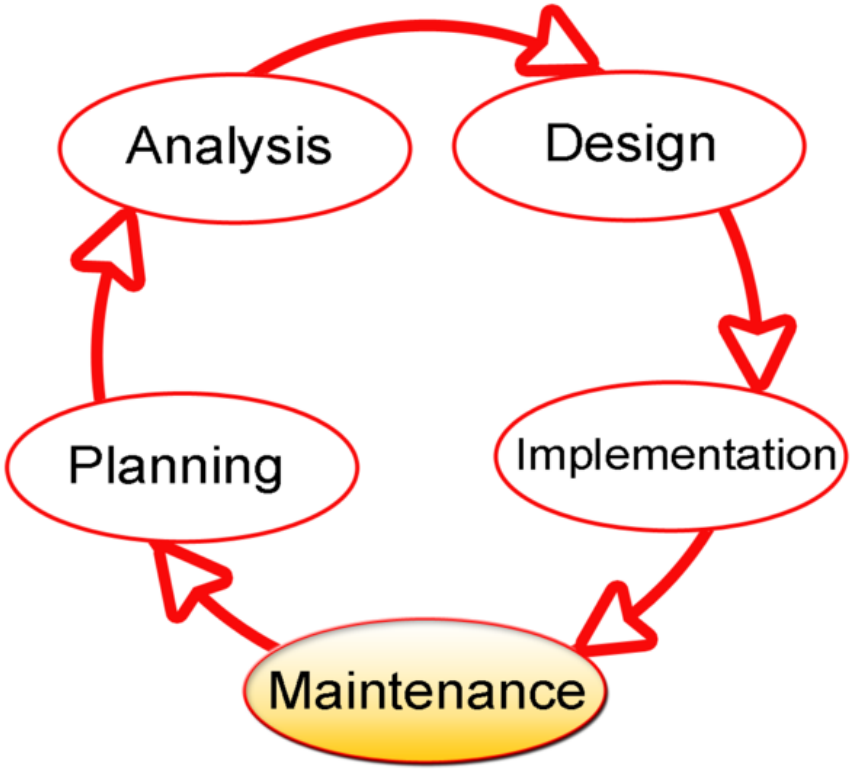
Who should do what?

Systems Development Life Cycle (SDLC)

When developing an information system:

- plan
- create
- test
- deploy

Applies to hardware and software or combo of both



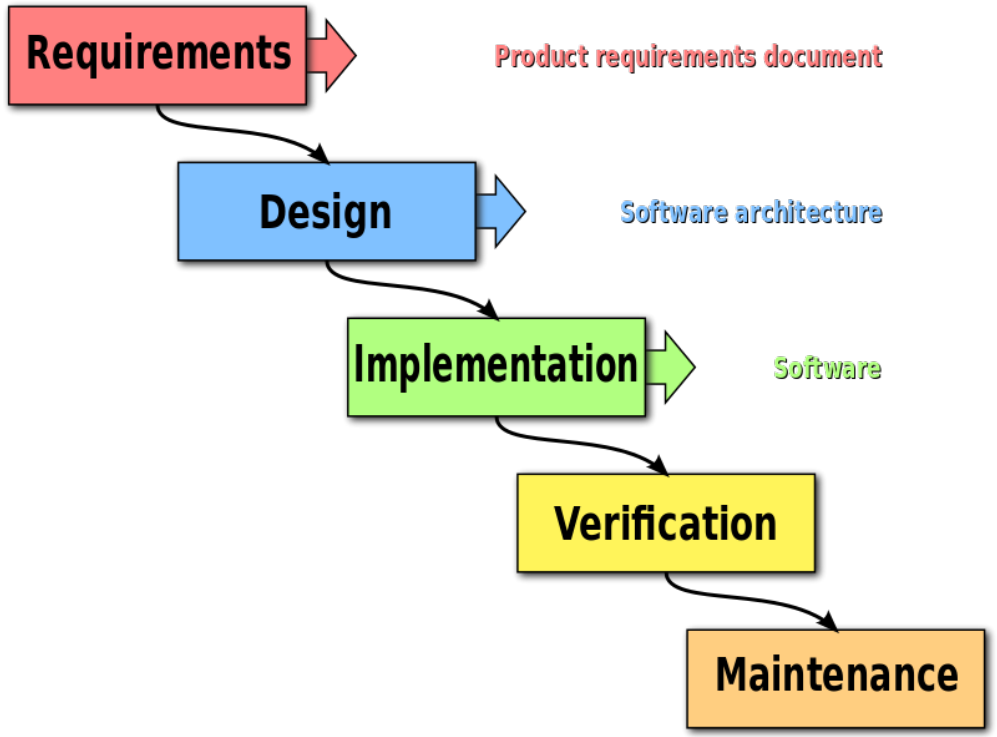
https://en.wikipedia.org/wiki/File:SDLC-Maintenance-Highlighted.png

Waterfall

Sequential, non-iterative design process

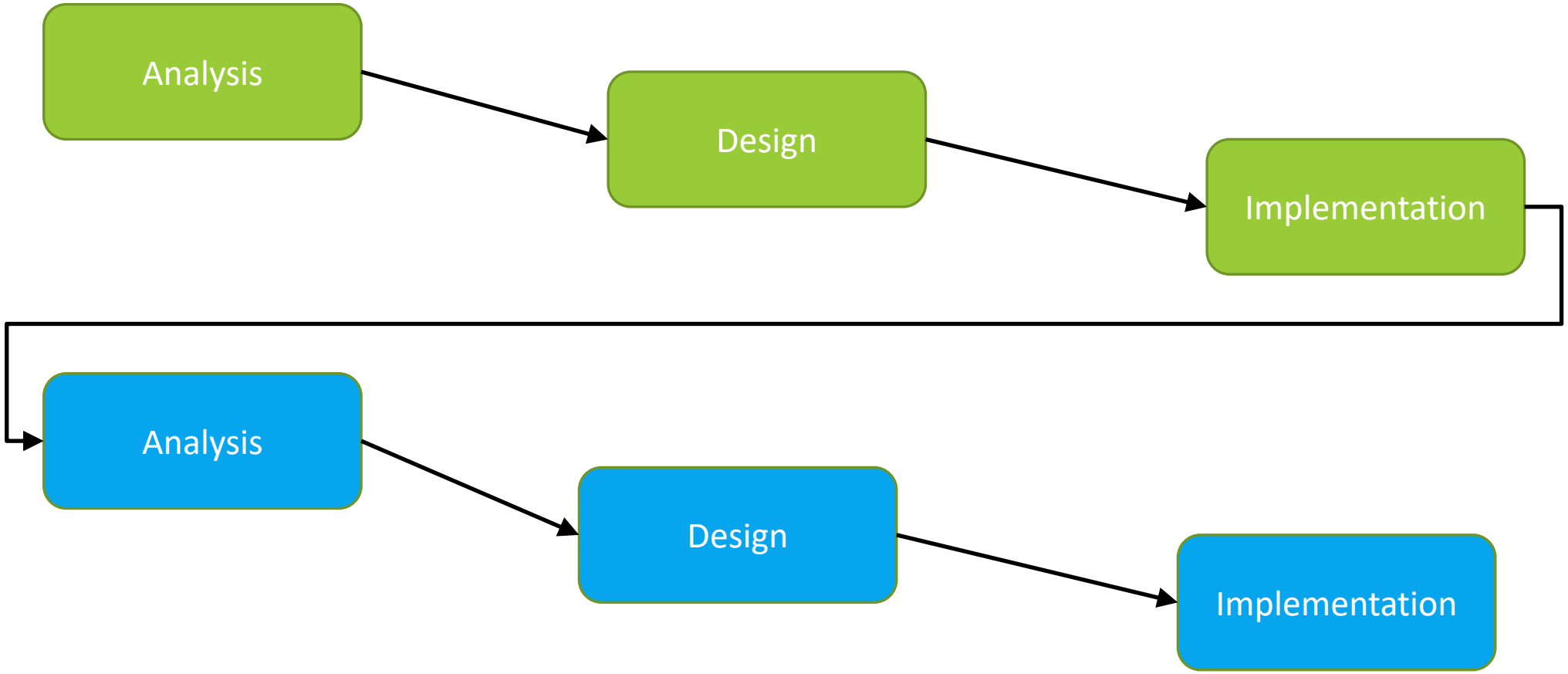
Originated in manufacturing and construction

Highly structured environments where most decisions need to be made ahead of time



https://en.wikipedia.org/wiki/File:Waterfall_model.svg

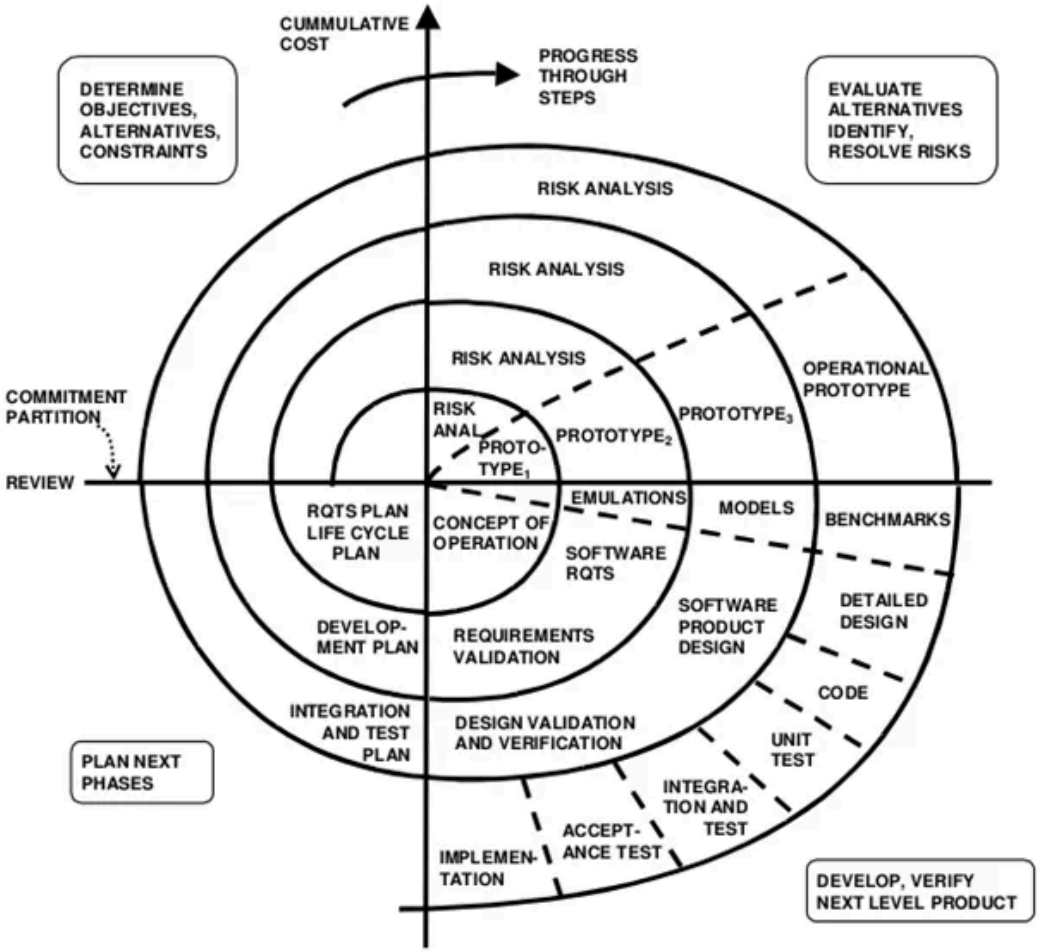
Incremental



Spiral

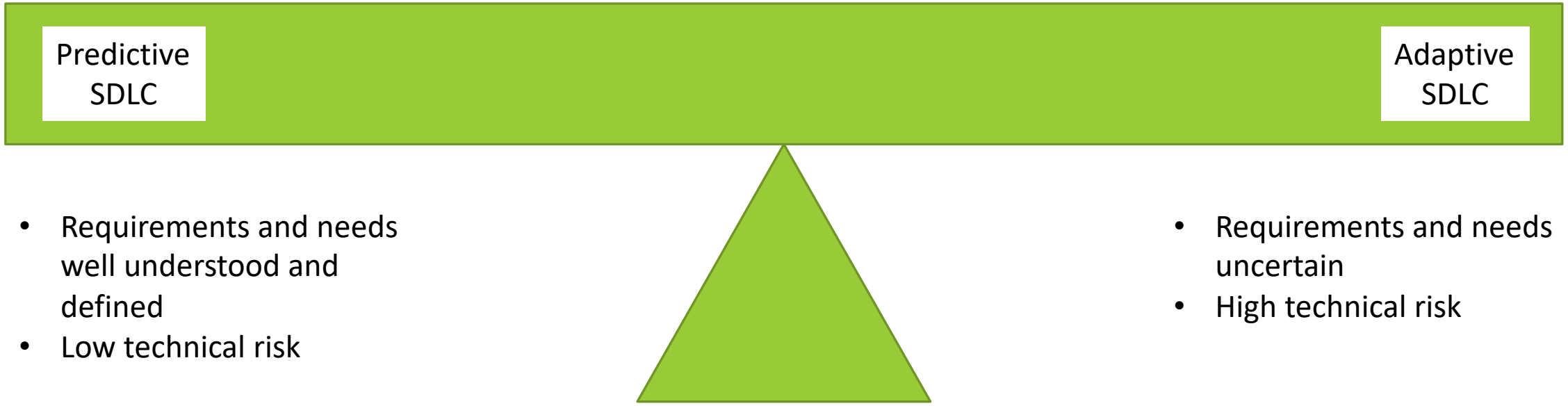
https://en.m.wikipedia.org/wiki/File:Software_Development_Spiral.svg

Spiral



https://melsatar.blog/2012/03/15/software-development-life-cycle-models-and-methodologies/

Choices of SDLC



- Requirements and needs well understood and defined
- Low technical risk

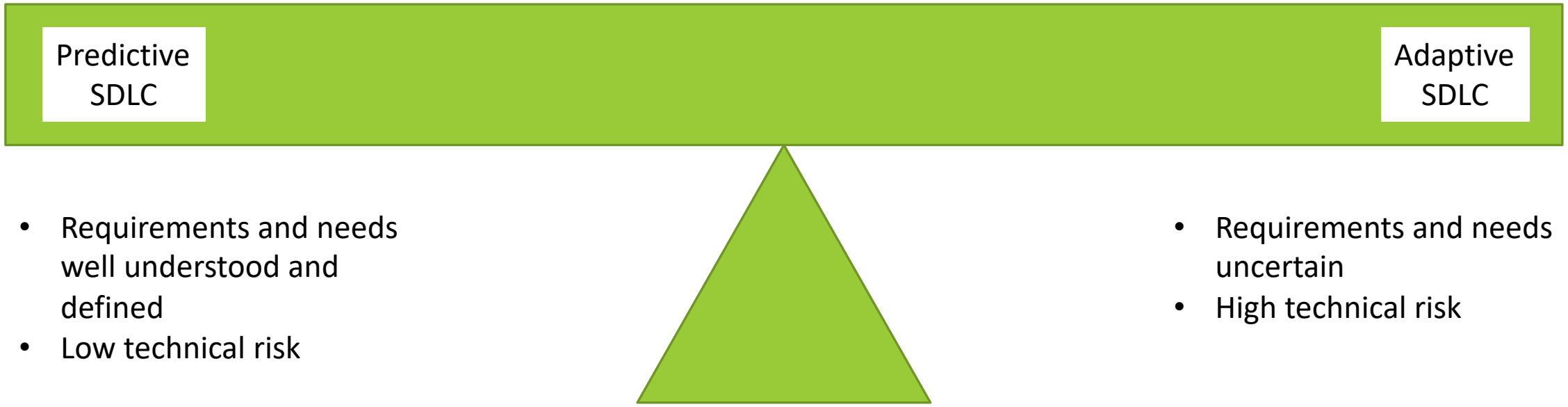
- Requirements and needs uncertain
- High technical risk

Choices of SDLC

Waterfall

Incremental

Spiral



- Requirements and needs well understood and defined
- Low technical risk

- Requirements and needs uncertain
- High technical risk

SDLC Activity

Traditional Software Development

Values:

- Processes and tools
- Comprehensive documentation
- Contract negotiation
- Following a plan

Agile Development

Values:

- Individuals and Interactions
- Working Software
- Customer Collaboration
- Responding to change

Agile Development

An adaptive software development philosophy

- requirements and solutions evolve
- fast iterations

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

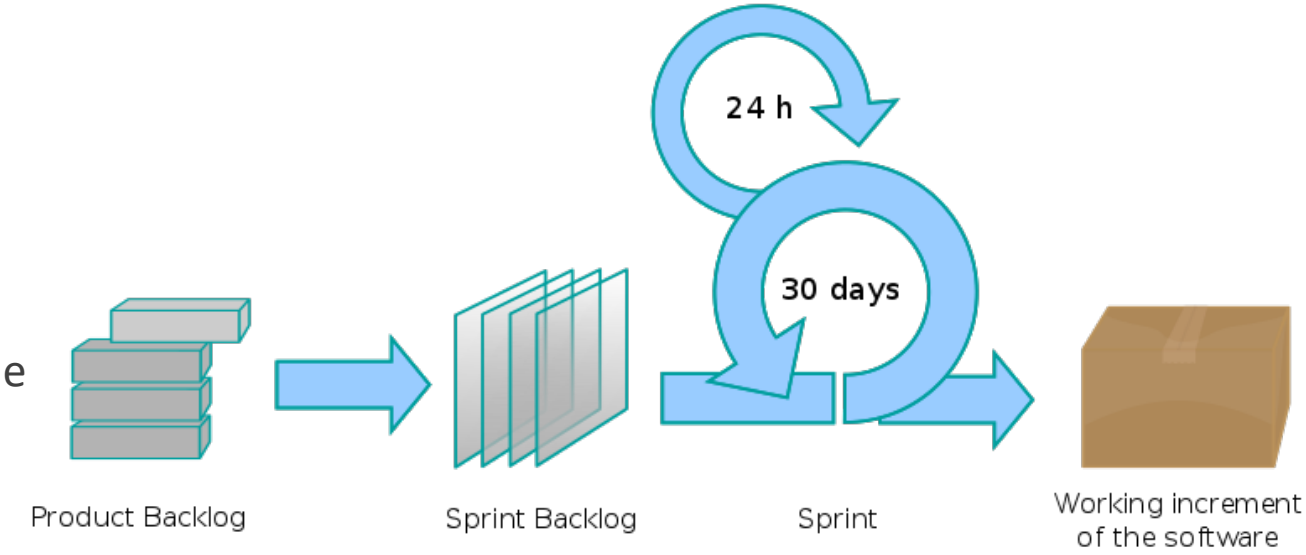
Daily stand-up (or scrum)

At end of each iteration product is **demo'd to stakeholders**

Agile Development

Most common variants:

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

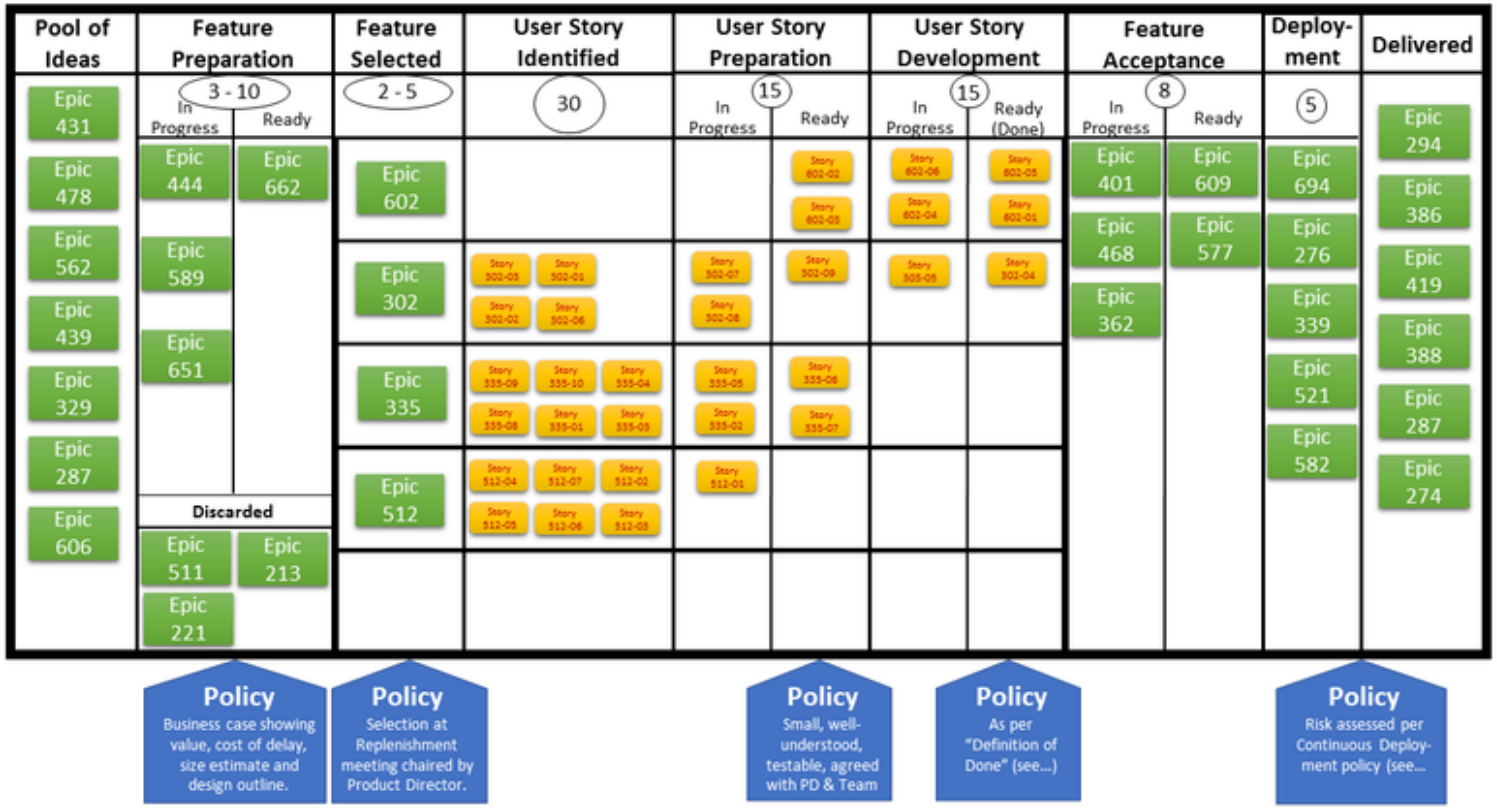


https://en.wikipedia.org/wiki/File:Scrum_process.svg

Agile Development

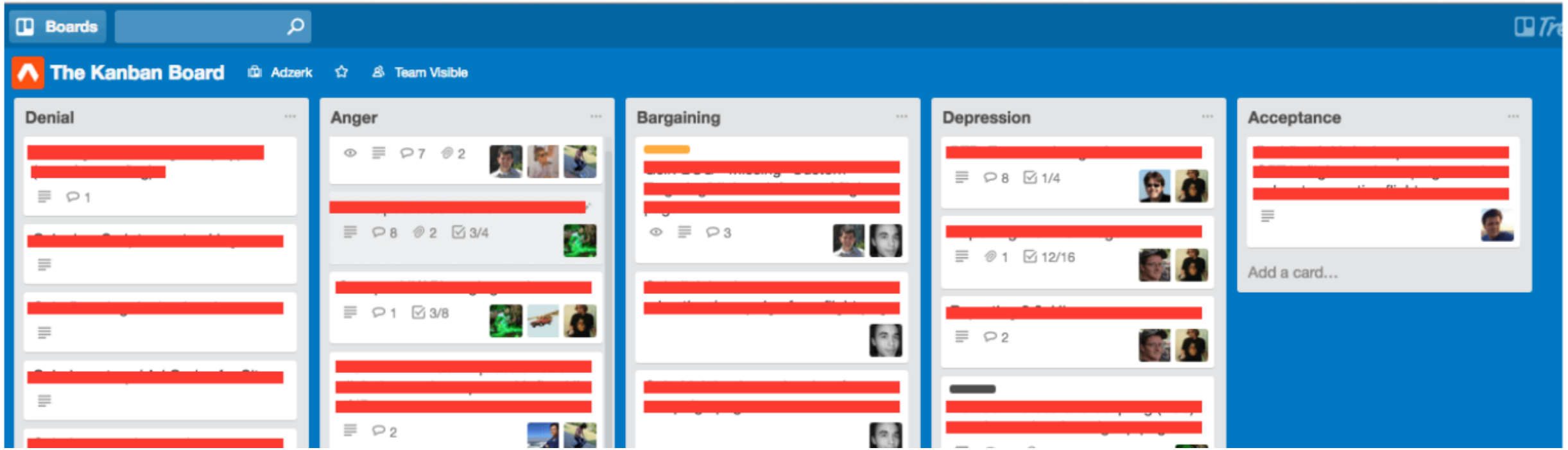
Most common variants:

- Scrum
- Kanban
  - visualization technique
  - helps spot bottlenecks
  - members pull work when they hav capacity



https://en.wikipedia.org/wiki/File:Sample_Kanban_Board.png

Variants of Kanban: the Stages of Grief



It's better to maintain the view that all our code will eventually break or become obsolete than to assume that the code we write will last forever.

https://adzerk.com/blog/2016/04/kanban-and-the-stages-of-grief/

Changing is hard.

Pretending to change is much easier.

Detecting Agile BS

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.

Summary

Complex projects require some form of management

Project manager roles

How to make meetings useful

Software Development Life Cycles

Readings

Chapter 11 (9 in 6th): Project Planning and Project Management (from page 335 to end)