CSC 171 – Module 03

Start Your Agile Project Right

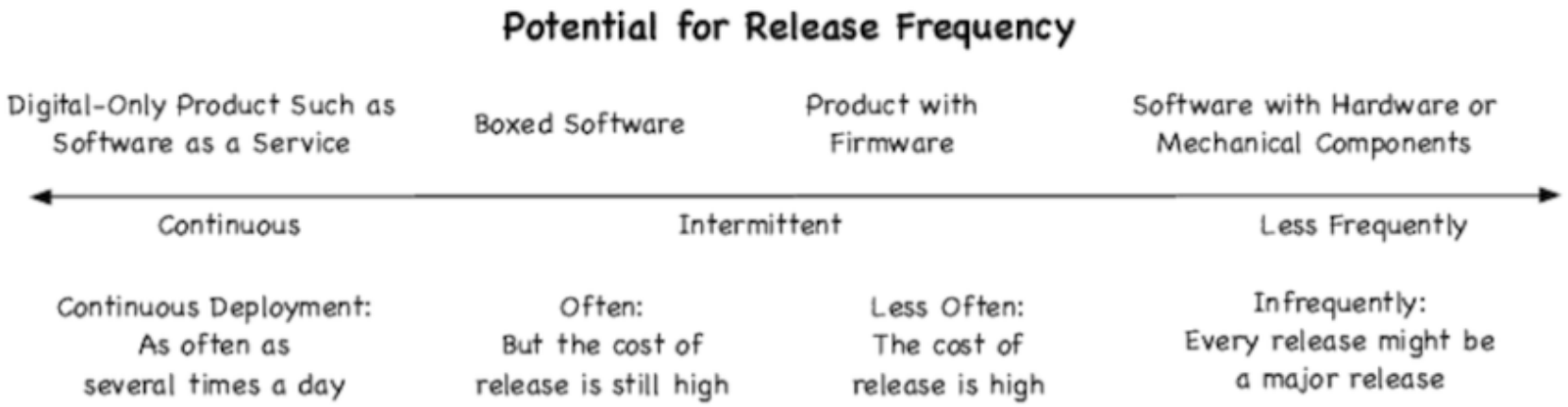
Project Charter

- Typical components of a traditional project charter
  - 3Ws
    - What (scope overview)
    - Why (business case)
    - When (milestone schedule with acceptance criteria)
  - 3Rs
    - Risks, contingency plans
    - Resources (people, money, others)
    - Routines (how the team plans to work)
  - 3Cs
    - Communication needs (stakeholders and their wants and hopes)
    - Collection of knowledge (lessons learned)
    - Commitment (signatures)
- Agile project charter
  - High-level summary of the project's key success factors, includes at least the major objectives of the project, scope boundaries, and reciprocal agreements between the project's implementation team and external stakeholders. [2]

Charter the Project

- Charter the project as a team
  - The team needs to own the project charter or plan and any experiments to understand the design and architecture.
- Project vision
  - A one-to-three-line statement of what the product manager wants the project to fulfill with this release.
  - It addresses the following
    - Who is the main recipient of the project's outcome?
    - What can the main recipient do with that outcome?
    - What is the value to the main recipient?
  - Example
    - This project can help all the retired people move their money by Sept. 30, so they can save money from federal, state, and local taxes.
- Release criteria
  - Release criteria tell what "done" means for the entire project.
  - Example
    - All code must compile and build for all platforms
    - Zero high priority bugs
    - For all open bugs, documentation in release notes with workarounds
    - All planned QA tests run, at least 90% pass
    - Number of open defects decreasing for last three weeks.
    - Feature x and y unit tested by developers, system tested by QA, verified with customers A, B before release
    - Ready to release by June 1
- Product backlog

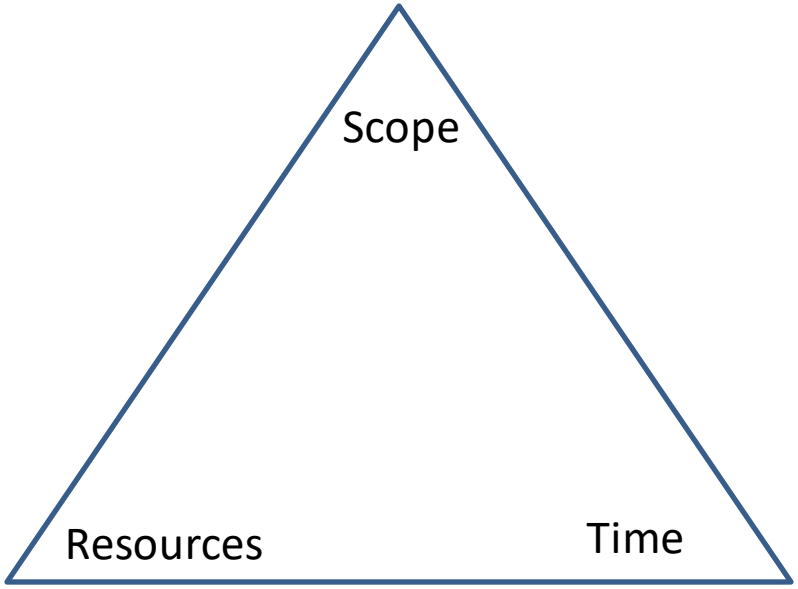
Product Type & Release Frequency



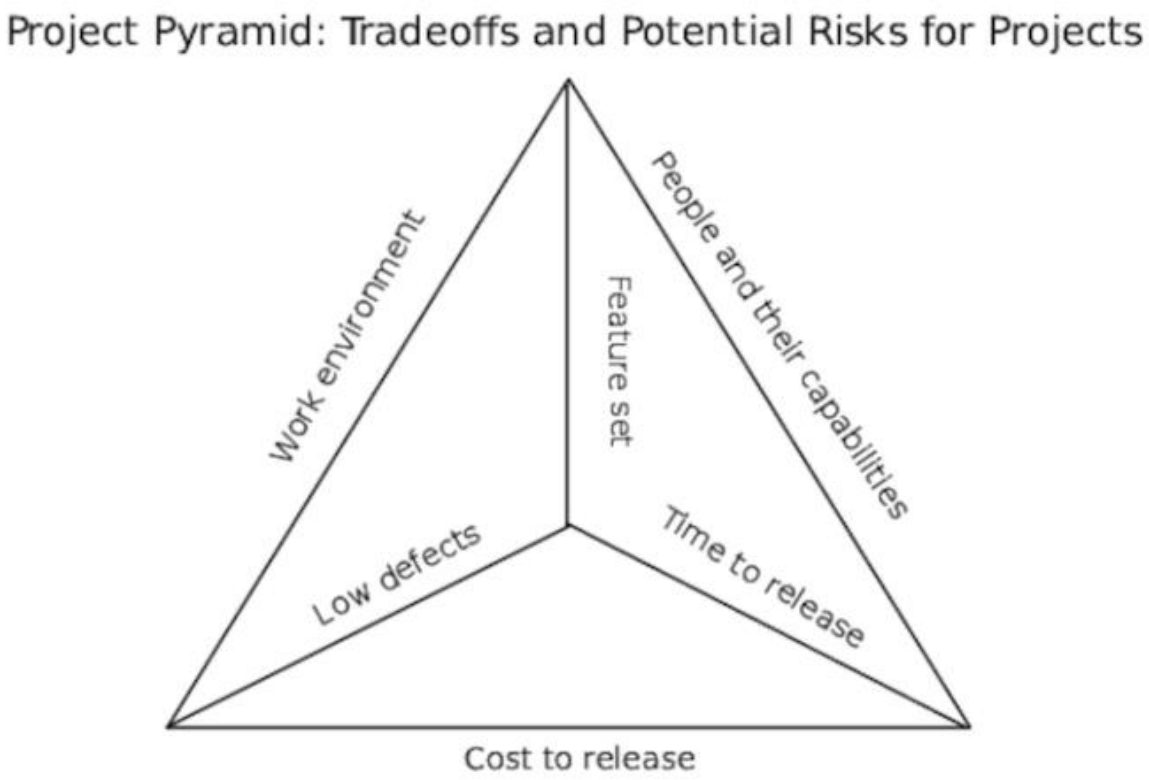
Product can be released internally often if cannot be released externally often.

The Iron Triangle

- Constrains of project management
  - Scope
    - Work to be done
  - Resources
    - Include budget and development team members
  - Time
    - Development duration
- The waterfall approach
  - Scope is fixed
  - Resources and time are variable
- Agile approaches
  - Resources and time are fixed
  - Scope is variable



Assess Project's Risks



- Feature set is expected to change with agile approaches.
- Conduct a retrospective for recently completed work to assess potential risks.
- Ask the team how it wants to manage risks.
- Managing risk with Scrum: https://www.scrum.org/resources/blog/managing-risk

Risk Management

Traditional Methods

- A large amount of up-front planning occurs for identifying, evaluating, and developing risk response plans for project risks
- When a new risk arises during project execution, it is added to the risk register and a risk owner is assigned to address it

Agile Methods

- Team members identify risks
- The Product Owner uses risks to adjust the product backlog
- Risks are constantly identified
- Risks are mitigated as a result of product increment being completed in short durations

Start Architecture Thinking

- In traditional approaches
  - Big Design Up Front (BDUF)
- In agile approaches
  - Evolve the architecture as the product proceeds
  - Product architecture should be created and evolved by the team, not handed down by an architect outside the team
  - Having architectural expertise on the team helps
- Recommendations
  - Create architectural spikes to experiment
    - A Spike is a solution-specific experiment, aimed at answering a question or gathering information.

Designing Agile Architecture

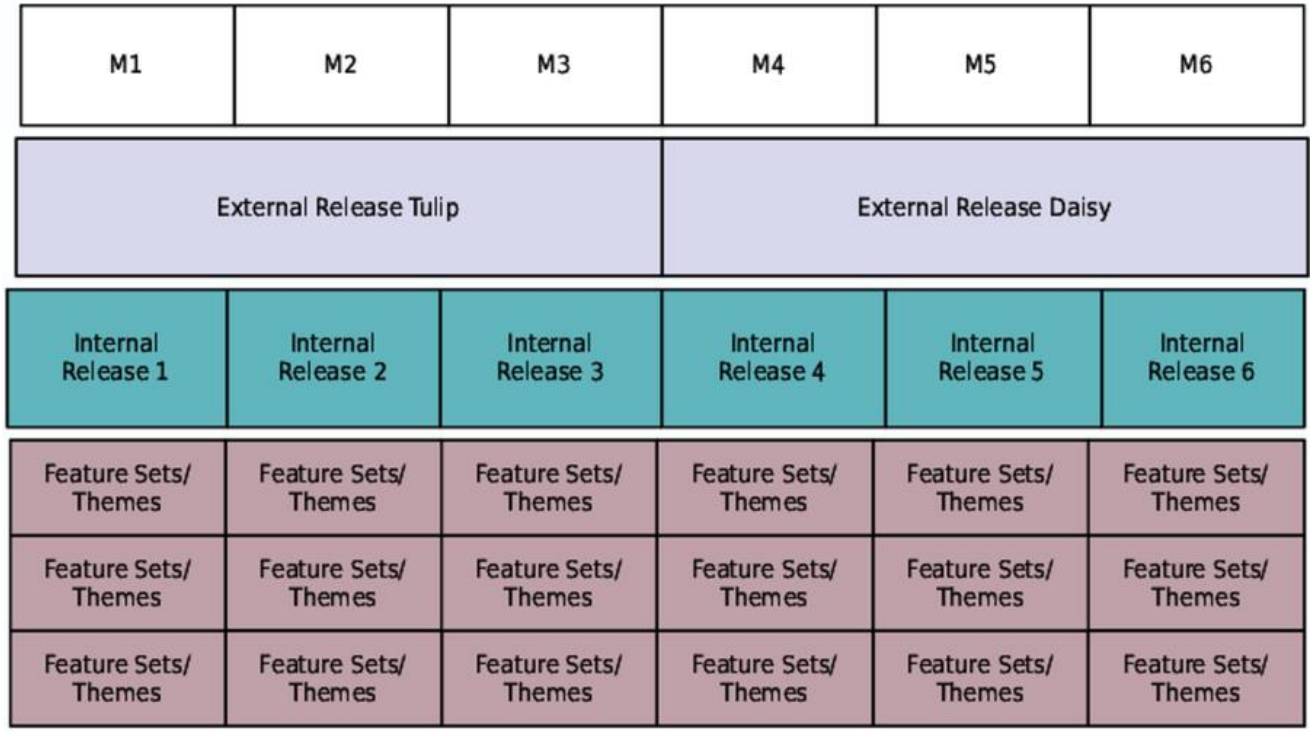
- Issues with too much up-front design
  - Early decisions could be wrong or subject to change
  - Excessive effort will delay development and the opportunity for early customer feedback
- Issues with too little up-front design
  - Architecture significant requirements might not be well supported
  - More effort might be needed to fix architecture issues during development
- The right amount of up-front design depends on the team's context
  - Requirement stability
  - Technical risk
    - Risk is the product of the probability of failure and the impact of that failure. [3]
    - Uncertainty of technologies and design or system itself
    - The team’s and customer’s risk tolerance
  - How soon the customer wants to start using the system?
  - Team's agility, environment, and architecture and technical experience

Tactics for Designing Agile Architecture [3]

Tactic	Impact on responsiveness to change	Reduces up-front design effort?
Keep designs simple	Increases modifiability	Yes
Prove the architecture with code iteratively	Increases modifiability	Yes
Use good design practices	Increases modifiability	No
Delay decision making	Increases tolerance to change	Yes
Plan for options	Increases tolerance to change	No

Project Plans

- In traditional approaches
  - Require detailed project plans (e.g., communication plans and Gantt charts) at the beginning
- In agile approaches
  - No need for detailed project plans
  - Need initial product roadmap
  - Need initial product backlog



References

- [1] Create Your Successful Agile Project, Johanna Rothman, Pragmatic Programmers LLC, 2017. ISBN:9781680502602
- [2] Project Chartering, https://www.agilealliance.org/glossary/project-chartering/
- [3] Agility, Risk, and Uncertainty, Michael Waterman, March/April 2018, IEEE Software