Software Engineering

Reagan Shirk

September 14, 2020

Project Management Structures

Waterfall

- Advantages:
  - conceptually simple
  - cleanly splits the problem into phases that can be performed independently of each other
  - natural approach to problem solving
  - easy to administer
  - each phase is a milestone
- Distadvantages:
  - assumes that requirements can be specified early and never change
  - may decide on technologies too early
  - follows the "big bang" approach
    - all or nothing delivery
    - can be risky
  - document orienteted
    - docs are required at the end of each phase
- Usage:
  - very widely used
  - good for projects where requirements can be understood easily and tech decisions are easy
    - good for familiar type of projects

Iterative Development

- Counters the "all or nothing" drawback of waterfall because you're constantly prototyping, developing, testing, adapting
- Combines benefit of prototyping and waterfall
- Develop and deliver software in increments
- Multpile rounds of waterfall according to Rafal
- Design $\rightarrow$ implement $\rightarrow$ analysis
  - wash, rinse, repeat
- Products almost always follow iterative development
- Used commonly in development, businesses want quick responses for software related issues and can't afford the all-or-nothing approach
- Newer approaches like XP and Agile rely on iterative development
- Problems with iterative approach
  - people want to get shit done at once
  - more time consuming than the ideal waterfall
  - rewriting code
  - can cost more
- benefits
  - get as you pay
  - constant feedback for improvement
- applicability

  - good for projects where response time is important
  - can't take the risk of a long projects
  - not all of the requirements are known

Timeboxing

- So far, time wasn't being considered as an issue
- Thinking of iterations as a linear sequence of iterations (What?)
  - Oh, setting a given amount of time for you to complete a thing I think
- Each iteration is a small waterfall where you decide the specs and plan the iteration
- The timeboxing is a fixed duration for the iteration, then you determine the next set of specs
- Divide the iteration into a handful of equal stages
- Use pipelining concepts to perform the iterations in parallel

Time boxed iterations

- General iterative development but the amount of time spent on the iteration is fixed
- Useful in many situations
- Predictable delivery times
- Product release and marketing are easier to plan
- Makes time a **non-negotiable parameter** and helps put the attention on schedule
- Prevents bloating the requirements
- Overall dev time is still unchanged

Model Basics

- The development is done in fixed duration stages, as I've typed a million times
- Each time box is divided into an amount of fixed stages
- Each stage has a specific task to be performed, they can be done independently
- Each stage is close to the same duration
- There is a **dedicated team for each stage**
- When one stage team finishes, it hands over the project to the next team

Prototyping

- addresses the requirements limitation of waterfall
- instead of solidifying requirements after a discussion, you can build a prototype to help understand the requirements
  - this helps alleviate the risk
- would you build a prototype for requirements where you're certain or requirements where you're a little fuzzy?
  - definitely more beneficial to prototype if your requirements are fuzzy
- Developing a prototype
  - starts with initial requirements
  - only key features which aren't well understood need to be included
    - no sense in including features you understand well
  - feedback from users is taken to improve the understanding

Phases in a Project

- Inception
  - ends with *Lifecycle Objectives milestone*
  - vision and high level capabilities are defined
- Elaboration
  - *Lifecycle Architecture Milestone*
  - most requirements are defined and the architecture is designed
- Construction
  - *Initial Operational Capbility*
- Transition
  - *Product Release*

RUP

- I stopped paying attention for a second oops
- No idea what he was talking about, I'll google/check the slides/watch the recording

Agile

- Developed in the 90s as something to do other than document driven approaches
- Kind of like a timeboxed model
- Most agile approaches have common principles (There are so many fucking types of agile)
  - Working software is the measure of success/progress
  - he changed the slide again dang
    - I should start paying attention

XP

- A type of agile
- User stories → release planning → iteration → acceptance test → small release
  - Acceptance testing can loop you back to between release planning and iteration, user stories can go straight to acceptance testing