CS 428
The Essence of Software Development

WINTER 2020

BRUCE F. WEBSTER

Truisms of software development

- Software development is hard and unpredictable (Armour)
- As a result, software specification and estimation are likewise difficult
  - Glass (2002): Estimation is usually done at the wrong time (start of project) and by the wrong people (managers)
  - Spinrad (1998): In creating a new software program, all the important mistakes are made the first day.
- Rate of overruns (schedule, cost) and outright failure is high
  - The first 90% of the project takes 90% of the schedule. The remaining 10% of the project takes the other 90% of the schedule.
  - Requirements explode when you move into design and implementation
- People matter most.
  - Process/methodology is not a panacea or a crutch or a silver bullet
- Ignore or minimize software quality assurance (SQA) at your own peril
- Be on the lookout for organizational dysfunctions
- Observations and experiences?

Why these three books?

- ***The Mythical Man-Month*** (Brooks): first published in 1975; added to in 1995; remains a classic because it set forth most of the fundamental issues and causes of delays and failures in software projects
- ***Peopleware*** (DeMarco & Lister): first published in 1987; currently in 3rd edition (2013); focuses on individual, team, and organizational impacts on software engineering
- ***Accelerate*** (Forsgren et al.): first published in 2018; presents four years of research into what practices separate high-performing IT organizations from the rest.

- Concept: levels of complexity in types of software
  - Individual program for personal use
  - Commercial product for distribution and sale (word processor, game, app)
  - “Programming system” (custom operating system, large-scale integrated system) for in-house use
  - Commercial “programming system” (OS, ERP, etc.) for distribution and sale
- What are some other types of added software complexity?
- What can make software difficult to maintain and update?

MMM: Chapter 1: the tar pit

The Tar Pit (cont.)

- The Joys of the Craft of Programming
  - The sheer joy of making things
  - The pleasure of making things that are useful to other people
  - The fascination of building complex systems
  - The joy [heh] of always learning
  - The delight of working in such a tractable medium “only slightly removed from pure thought-stuff…yet…is real in the sense that it move and works, producing visible outputs separate from the construct itself”
- Why do you enjoy software engineering (assuming you do)?

The Tar pit (cont.)

- The Woes of the Craft
  - You must perform perfectly
  - Other people set your objectives, provide your resources, and furnish your information
    - Usually your authority is not sufficient for your responsibility
  - You often depend upon other people's programs, which are less than perfect
    - The upper bound of quality of a complex system is determined by the lowest quality of any of its essential components
  - Designing grand concepts is fun; finding nitty little bugs is just work
  - Debugging has *at best* linear convergence
  - The product is often obsolete before it is completed
- What are other painful things you've discovered about software engineering?

Peopleware Ch 1: Somewhere today, a Project is Failing

- "For the overwhelming majority of the [failed] projects we studied, *there was not a single technological issues to explain the failure*."
- "The major problems of our work [e.g., IT development and deployment] are not so much *technological* as they are *sociological* in nature."
- "Most managers are willing to concede…that they've got more people worries than technical worries. *But they seldom manage that way*."
- "The main reason we tend to focus on the technical rather than the human side of the work is not because it's more crucial, but because it's easier to do."
- Observations and feedback?

Accelerate: Chapter 1

- Five categories of capabilities to achieve high performance
  - Continuous delivery, architecture, product & process, lean management & monitoring, cultural
- “High-performing organizations” must accelerate:
  - Delivery of goods and services “to delight their customers”
  - Engagement with market to detect and understand customer demand
  - Anticipation of compliance and regulatory changes
  - Responses to potential risks (security, market changes, etc.)
- Focus on key transformations based on actual results (evidence)
- Adopt DevOps (not a silver bullet, though)
  - Merging of the development and operations sides of an organization
  - Improving communication, visualizing work, eliminating bottlenecks, and continually automating and improving procedures
  - Culture that an engineering group creates which promotes the outcomes mentioned above

To do for next week (01/13)

- **CRITICAL**: if you don't have a GitHub idea already, got to GitHub.com and register for one. Then send me (bwebster@bfwa.com) your GitHub ID or your email associated with your GitHub ID
- Log into the class Github (https://github.com/cs428TAs/w2020/wiki)
  - Sometime this week: create a proposed project and/or endorse an existing one
  - By start of class next week (01/13): vote on at least three (3) projects
  - We will finalize projects and teams next week in class
- Join CS428-W20.slack.com if you haven't already
- Readings for next week (01/13):
  - "The Five Orders of Ignorance", Philip Armour (PDF on the website)
  - *Mythical Man-Month*, chapter 1
  - *Peopleware*, chapter 1
  - *Accelerate*, chapters 1-2