CS 428 Week #7 Readings

WINTER 2020, WEEK #7

BRUCE F. WEBSTER

- Interactive discipline for the architect
  - The architecture is valuable input into estimating the implementation and testing
  - If the schedule is unacceptably long, the architect can look for ways to simplify
  - Big challenge: features that may seem simple to the customer may actually be very difficult to design and implement
- The second-system effect
  - Brooks notes later that true iterative development can diminish this problem, but…
  - The first shipping version usually has many deferred features; there is a strong temptation to throw in "everything but the kitchen sink" into version 1.1 or 2.0
- Real-world issue: incurring 'technical debt' and not handling it
- Observations and experiences?

MMM CH 5: The Second-System Effect

- As with “second system effect”, Brooks feels his comments here are superseded by use of iterative/incremental software development
- Still, far too often, “pilot” or “prototype” systems are forced to evolve into production systems
- Only after your first cut do you often see how you should have done it in the first place
- What has been your observation/experience?

MMM Ch 11: Plan to throw one away

MMM Ch 11: Continued

- Plan the organization for change
  - Still a very real issue: lack of technical advancement track in most organizations
  - Instead, developers are pushed into management if they want to be promoted
- Two steps forward and one step back
  - Most ‘maintenance’ work involved adding new features
  - Introduces software entropy (or, if you prefer, software rot)
  - Production systems that are modified become less stable/reliable over time
  - “Less effort is spent on fixing original design flaws; more is spent on fixing flaws introduced by earlier fixes”
- Your observations/experience?

PW Ch 4: Quality – if time permits

- We tend to tie our self-esteem not to the quantity of what we produce but the quality, and yet management often emphasizes just the opposite
- "Quality, far beyond that required by the end user, is a means to higher productivity"
  - Remember Brooks: we spend 50% of a project schedule in testing & QA, whether we plan for that or not
- High quality can result in cost reduction
- "Quality is free, but only to those who are willing to pay heavily for it."
- Observations and feedback?

- The Seven False Hopes of Software Management
  - There is some new trick that could send productivity soaring (cf. “No Silver Bullet”)
  - Other managers are getting gains of 100-200% or more!
  - Technology is moving so swiftly that you’re being passed by.
  - Changing languages [or methodologies] will give you huge gains.
  - Because of the backlog, you need to double productivity immediately.
  - You automated everything else; isn’t it time to automate away your developers?
  - Your people will work better if you put them under a lot of pressure.
- Observations and feedback?

PW Ch 6: Laetrile

- Westrum organizational culture models (see table next slide)
  - Pathological (power-oriented) – based on fear & threat – information withholding
  - Bureaucratic (rule-oriented) – based on group protection and turf battles
  - Generative (performance-oriented) – everything subordinated to good performance & goals
  - Impacts both software delivery performance and organizational performance
- Elements of "good information"
  - Provides needed answers
  - Timely (i.e., can be acted upon to make a difference)
  - Presented so that it can be understood and used
- Continuous delivery and lean management both impact (for good) organizational culture

ACC CH 3: Measuring and Changing Culture

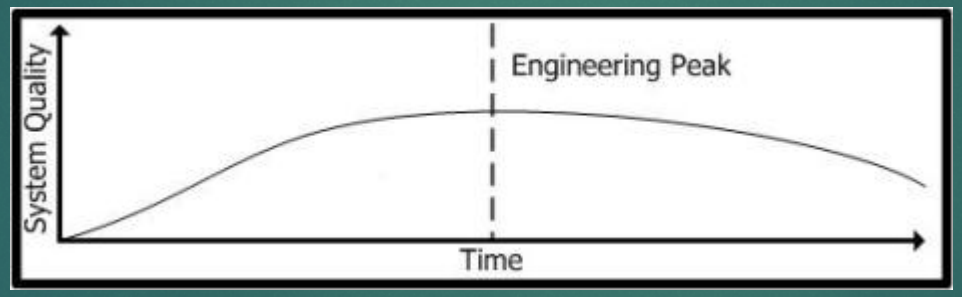
Pathological (power-oriented)	Bureaucratic (rule-oriented)	Generative (performance)
Low cooperation	Modest cooperation	High cooperation
Messengers “shot”	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure leads to scapegoating	Failure leads to justice	Failure leads to inquiry
Novetly crushed	Novelty leads to problems	Novelty implemented

ACC CH 3 (cont.)

- Certain cycle mismatches for organizations looking to adopt new technologies – keep your eyes open for these:
- **Firefly**: initial release is inadequate; you wait for the next; the next release is not quite functional enough, either; rinse and repeat until you finally give up.
- **Underdone**: new release of established technology that's not quite ready for production use
- **Conveyor Belt**: limited lifespan – challenge to make use of it and then migrate off before it goes away.
- **Landfill**: great promise (hype), but dead on (or soon after) arrival.

Getting Technology Lifecycles in Sync (Baseline, 2009) [Link]

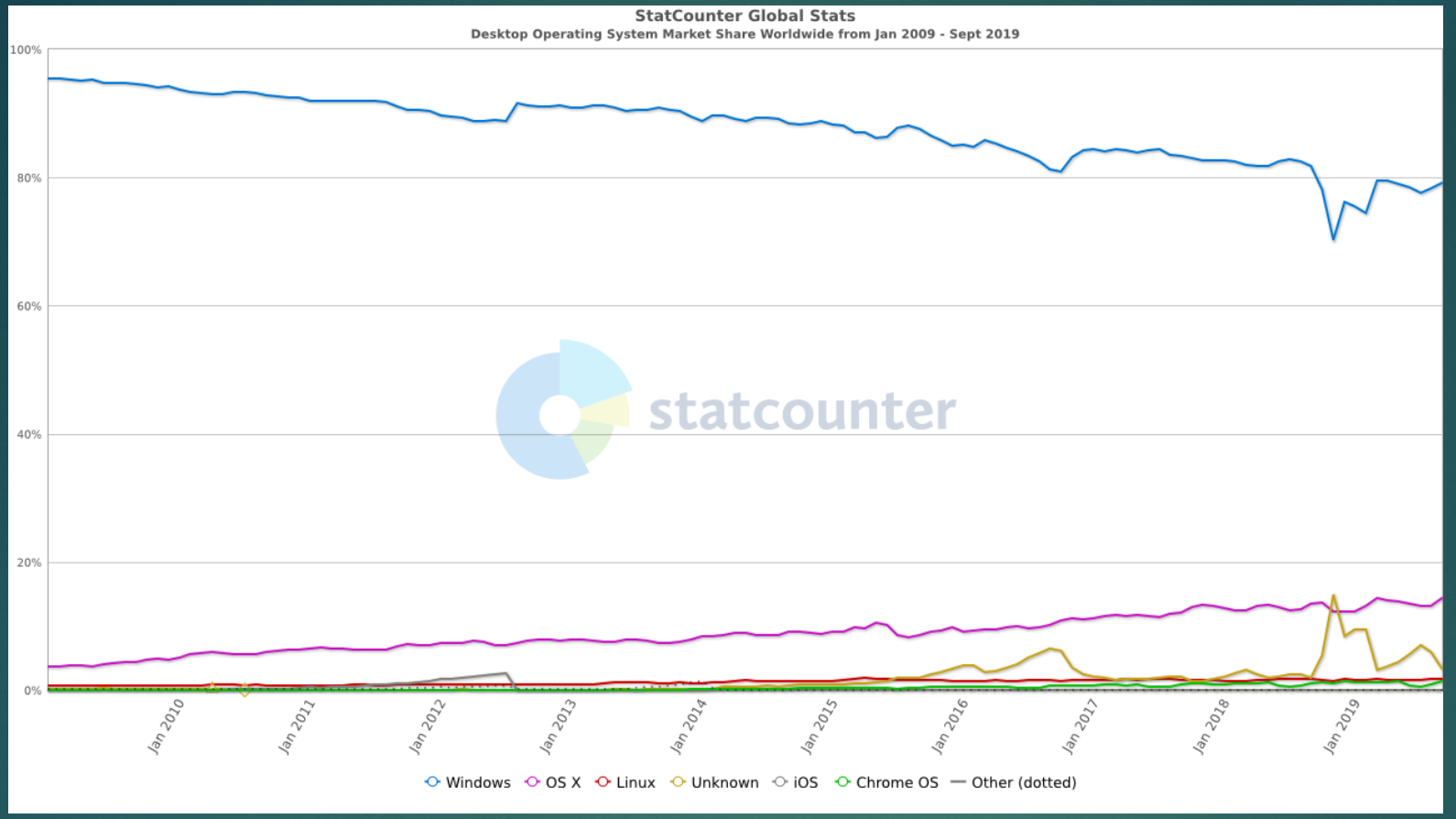
- System/product quality rises over time to a peak value and then starts to decline. Why?
  - The developer loses conceptual control of the system (bloat, loss of personnel, organizational changes)
  - Software rot sets in (piecemeal changes, growing incompatibilities w/external interfaces)
  - The enhanced system finally outgrows its original foundation
  - Market or business needs shift beyond the product's fundamental design
  - The developer beings to add "blue sky"/"kitchen sink" enhancements
  - Backward compatibility is maintained at all costs



The Arc of Engineering (Baseline, 2008) [Link]

Microsoft Windows Forever and Ever? (Windows Magazine, 1996) [Link]

- Me, in 1996: "As impossible as it might seem, Windows may still be dominant in 2025, nearly 30 years from now. It may look and work a bit differently, just as phones, TVs, and cars from 30 years ago do, but the principles will be the same. Our grandchildren will wonder about the quaint relics of terminology and work flow (when was the last time you actually put gloves in a glove compartment?), but they'll be able to clearly see the inheritance from MS-DOS/Windows to whatever they use."
- As of September 2019 – see next slide.



Windows Forever and Ever? (cont.)

Windows Forever and Ever? (cont.)

- It seems that the first sufficiently adequate technology in a given sphere usually gains broad acceptance and entrenches itself. There may be some contention for standards early on, but that eventually settles out. And once the solution gains acceptance, the rest of civilization reinforces it–through market forces, legal standards, competitive pressures, and economic incentives.
- Once the technology is entrenched, the focus is then on refinement and slow upgrading of the existing technology, not on radical innovations and wholesale replacements. This has been shown time and again in major areas of applied technology: communication (phones), transportation (cars), and so on. And the same thing is happening now with information technology.
- The sheer scale of adoption of Microsoft technology will provide tremendous momentum to keep thing moving in the same direction. The investment in hardware, software, market standards, training, business process, development expertise, custom applications, and deployed environments all argue against any broad changes, even those introduced by Microsoft. That investment grows year by year and will dominate more, not less, as time goes on.
- Thoughts and observations?