

Risk

Project Reminders

- Focus on good practices:
 - Maintain your backlog
 - Have standup meetings
 - Review and retrospective at the end
 - Testing, version control, code reviews

Risk Is Central to Software Engineering

- Every bug may cost more to fix later.
- But every change (even bug fixes) may introduce more bugs.
- There are lots of different kinds of risks in software projects!

Today

- Learning goals:
 - Understand how key risks threaten software project success
 - Three key principles that affect software risk: *second system effect*, *the mythical man month*, and *Conway's Law*.
 - (sorry — "man" is in the title of a book)

Technical Risks

- You chose to rely on a framework that was "almost done" — but it runs late.
- You rely on a platform, service, or framework that does not quite meet your needs
 - Or adds complexity without delivering enough value
- You underestimate the complexity of your own components

Financial Risks

- Running out of money (e.g., at a startup)
- Getting sued
- Need to give raises (for retention) but now can't hire needed staff

Requirements Risks

- Releasing software that does not meet user needs
 - (even if it is of high quality)
- Releasing software that frustrates users (poor UI)
- Releasing software too late

People-Related Risks

- People leaving
 - By choice (better job offer)
 - By circumstance or disaster (health problems)
 - By being fired (malfeasance)
 - By being stolen by Management to work on a higher-priority project

Management-Related Risks

- Management changes priorities
 - De-prioritizes a feature you invested in
 - Prioritizes a feature you *didn't* invest in
- Management turnover
 - Suddenly spending a lot of time "managing up"

Market Risks

- My summer interrr
- DVD authoring
- Does anyone auth
- Imagine working c
- right before broac



US Robotics V.92 USR5686E External Modem + Fax, 56K

Brand: [USRobotics](#)

4.0 (52) | [Search this page](#)

Amazon's Choice

\$65⁶⁶

Or **\$16.42** / 2 weeks (x4). [Select from 2 plans](#)

prime

FREE Returns

Apply for Prime Visa and pay ~~\$65.66~~ **\$0.00** for this order upon approval. [Find out how](#)

May be available at a lower price from [other sellers](#), potentially without free Prime shipping.

Brand	USRobotics
Internet service provider	Multiple wired internet service providers
Connectivity Technology	Wired
Compatible Devices	Personal Computer, Router
Color	red

About this item

- Connect to the Internet quickly and easily; installs in three simple steps
- Send e-mail and attachments up to 50-percent faster
- Send faxes from any Windows application and receive incoming



Safari

[Click to see full view](#)

Osborne I

- First sold in 1981
- 4 MHz CPU, 64 KB RAM
- 5" monochrome CRT
- Portable (24.5 lbs)



The Osborne Effect

- April 1983: Osborne Computer Corporation pre-announced several next-generation models
- Dealers canceled orders for Osborne I
- Osborne dramatically reduced prices for Osborne I
- September 1983: Osborne Computer Corporation bankrupt
- Note: Kaypro machine sales were starting to cut into Osborne sales, so this may have been a factor too!

Scenario #1: Old and Trusted, or New and Slick?

- You are starting a new web app development project.
- Worldwide math tutoring service. Connects tutors with students (who can afford to pay for the service). Vision: 24/7 tutoring. You can get help anytime, day or night, via worldwide staff.
- The year is 2030. React is old and stale (think of Ruby today). "Webby" is up and coming.
- Webby offers better performance, internationalization, and accessibility built-in.
- How will you decide? Changing later would be very expensive.

Scenario #2:

- You are three months from releasing the tutoring web app.
- New laws in 37 US states require tutors to hold *tutoring licenses*
- Addison-Wesley (textbook manufacturer) launches a new web-based tutoring service
- Now what?
- Discuss with a partner. **Submit your plan on Gradescope.**

The Mythical Man-Month

- (sorry; this is the title of a book from 1975 by Fred Brooks)
- Brooks's Law: adding more people to a late software project makes it *later*
 - New people consume resources getting up to speed ("hey, can you explain...?")
 - New people introduce more bugs
 - New people re-introduce old bugs
 - More people increase communication overhead (meetings...)

Communication Overhead

- Group intercommunication formula: $n(n - 1)/2$.
- Example: 50 developers give $50 \times (50 - 1)/2 = 1,225$ channels of communication.
- Moral: keep teams small (not 50!)

The Second System Effect

- The *first* time you design something, you *know* you don't know what you're doing.
- The *second* time, you think you know, and you fix all the things that were wrong the first time
- Therefore, the *second* system is the riskiest!
- I did this in my second system — even though I knew about the Second System Effect!

Incremental Slippage

- Q: How does a project get one year late?
- A: One day at a time.

Awareness–Understanding Matrix

	Aware	Not aware
Understand	Known knowns: Things we are aware of and understand	Unknown knowns: Things we are not aware of but do understand or know implicitly
Don't understand	Known unknowns: Things we are aware of but don't understand	Unknown unknowns: Things we are neither aware of nor understand

Inherent Vs. Accidental Complexity

- Some problems bring *inherent complexity*
 - Tax software is inherently complex because it has to be at least as complex as the tax code (law)
 - Automated driving software has to handle the complexities of physics *and* driving laws *and* human behavior
- But some software systems make problems *even harder*
 - You've seen these systems too

Conway's Law

- "[O]rganizations which design systems...are constrained to produce designs which are copies of the communication structures of these organizations."
- Therefore, organizational structure poses architectural risks!

Surfacing Risk

- Ask team members: what might go wrong?
 - A diverse team is more likely to identify more risks
- Then you can make mitigation plans.

Scenario #3: Cut or Press Forward?

- One month until you promised your investors the app would launch.
- Two key features have five weeks of estimated work left:
 - AI-based tutor screening (otherwise will have to interview prospective tutors; very expensive)
 - Algorithm-based tutor matching (e.g., need a calculus expert to do calculus tutoring)
- Ideas:
 - Move engineers from A to B (or vice versa); defer the other feature
 - Ask engineers to work evenings and weekends
 - Hire engineers from elsewhere
 - Something else?

Submit plan on Gradescope

Conclusion

- Surfacing risks in many categories enables you to mitigate them
- Mitigating risks often requires tradeoffs
- Know:
 - *Second system effect*
 - *Mythical man month: Adding new people to a late software project makes it later*