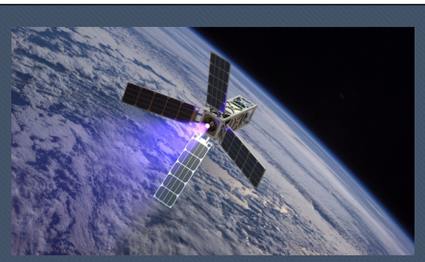
Managing **Technical Debt** in **Space**.

~~Managing~~ Preventing **Technical Debt** in **Space**.



Writing Deep Space Exploration **Software** for **Raspberry PI** CAT **CubeSats**

Inspired by Dr. Sandy Antunes Project Calliope

pico-satellite to capture earth’s music

{breathe}

What?

Managing Technical Debt in Space

- Brief History of Space Software Development
- Technical Debt
- Old vs. Modern DevOps
- Future Quantum Solutions

/ Tables Text

Who?

Managing Technical Debt in Space

- Those building CubSat and Pico Satellite Software
- Helping ensure your CubSat doesn't go b00m because of software
- College Under/Graduates looking to get into the Software & DevOps game
- Anyone sending Arduino, Raspberry PI, and Cylon.js to space

/ Tables Text

Legacy for the Next Generation

Get girls in STEM.

Sydney

Reader.

READING

80%

LIKE HER MOTHER

75%

DEAR GOD LIKE ME

25%

Rowan

Tinkerer.

TINKERING

50%

LIKE HER MOTHER

10%

DEAR GOD LIKE ME

90%

?

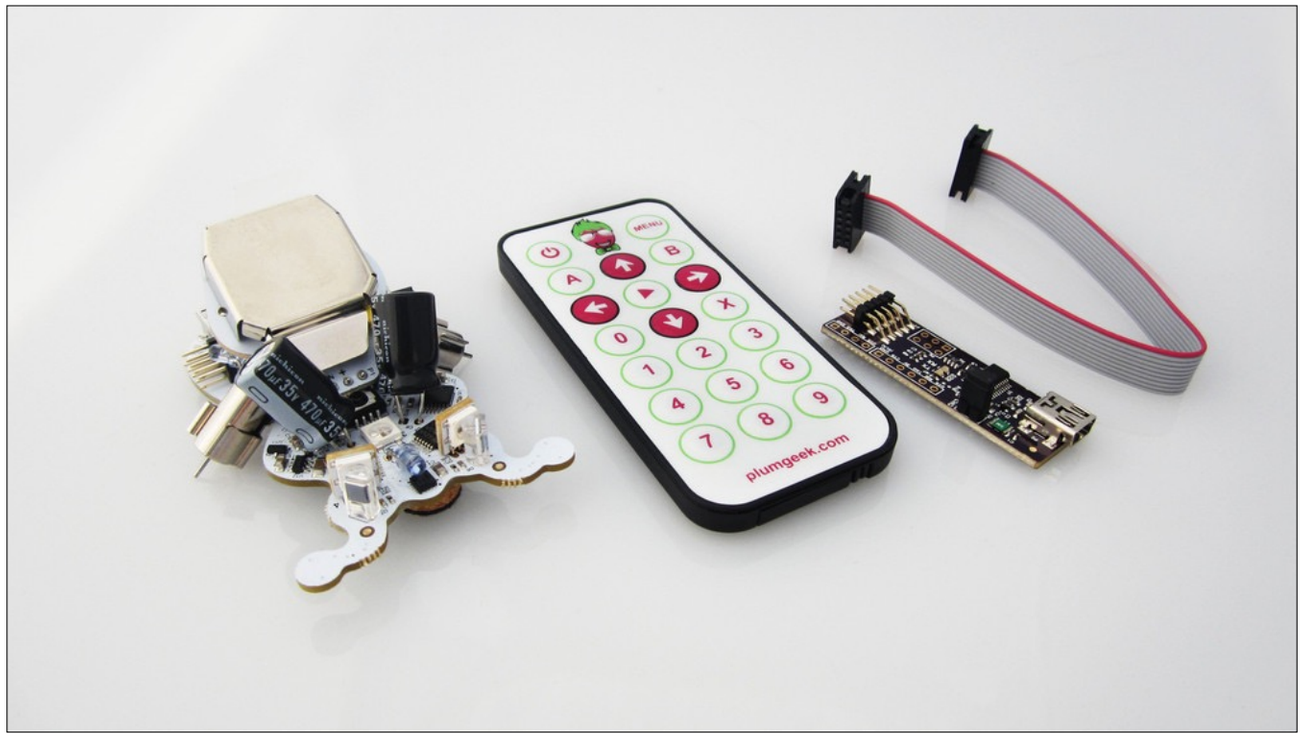
... work in progress...

???

LIKE HER MOTHER

DEAR GOD NOT LIKE ME

- girls in stem
- learn failure
- learn hard work

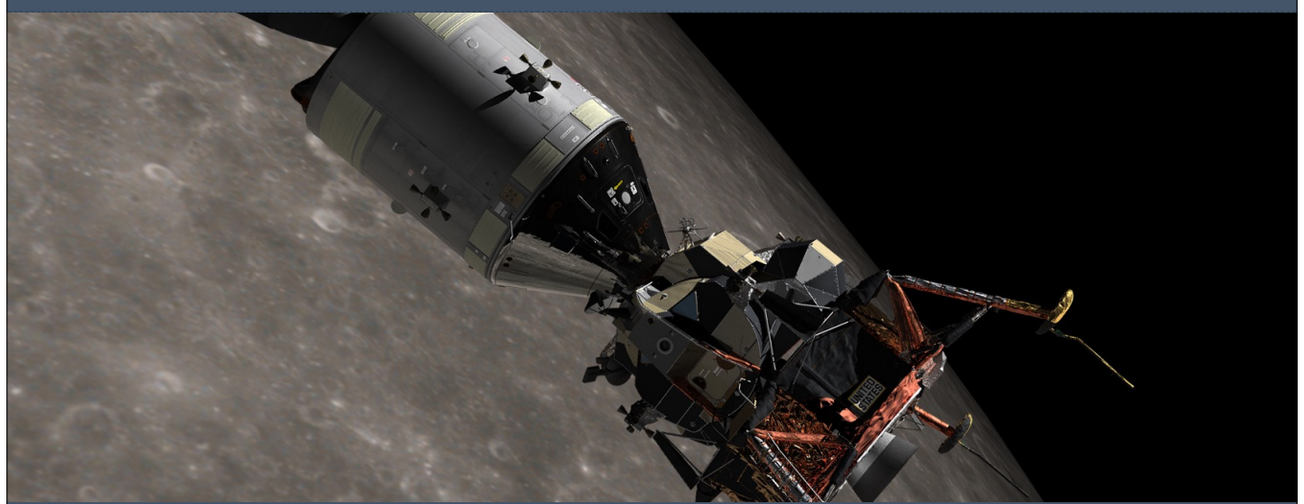


experiemnt

- Margaret Hamilton of Apollo 11
- Mars Climate Orbiter
- Saturn V
- Ariane 4 & 5
- Space Shuttle

Apollo 11

Put first humans on the moon.



Margaret Hamilton

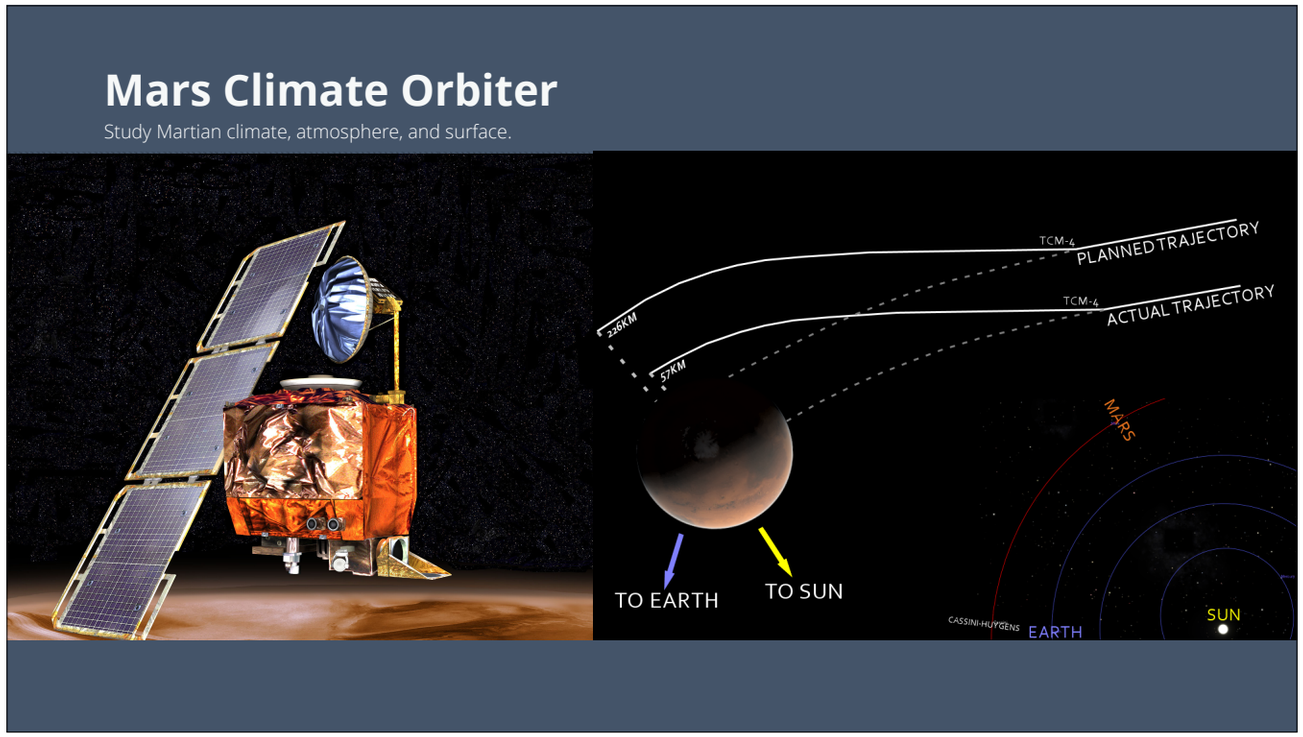
Director of software programming for Apollo & Skylab

software engineering
parallel computing
priority scheduling
end to end testing
Lesson: Leadership Support for Great Engineering, Brilliant Engineers can create fault tolerant systems

Apollo Guidance Computer

Used Assembly Code
Used Core Rope Memory (LOL Memory, Little Old Lady)
Used concurrency with priority in tasks
3 minutes before Lunar lander reached Moon’s surface, pahty alarm
rendezvous radar system switch wrong position, too much data
Lesson: Leadership Support for Great Engineering, Brilliant Engineers can create fault tolerant systems

{breathe}

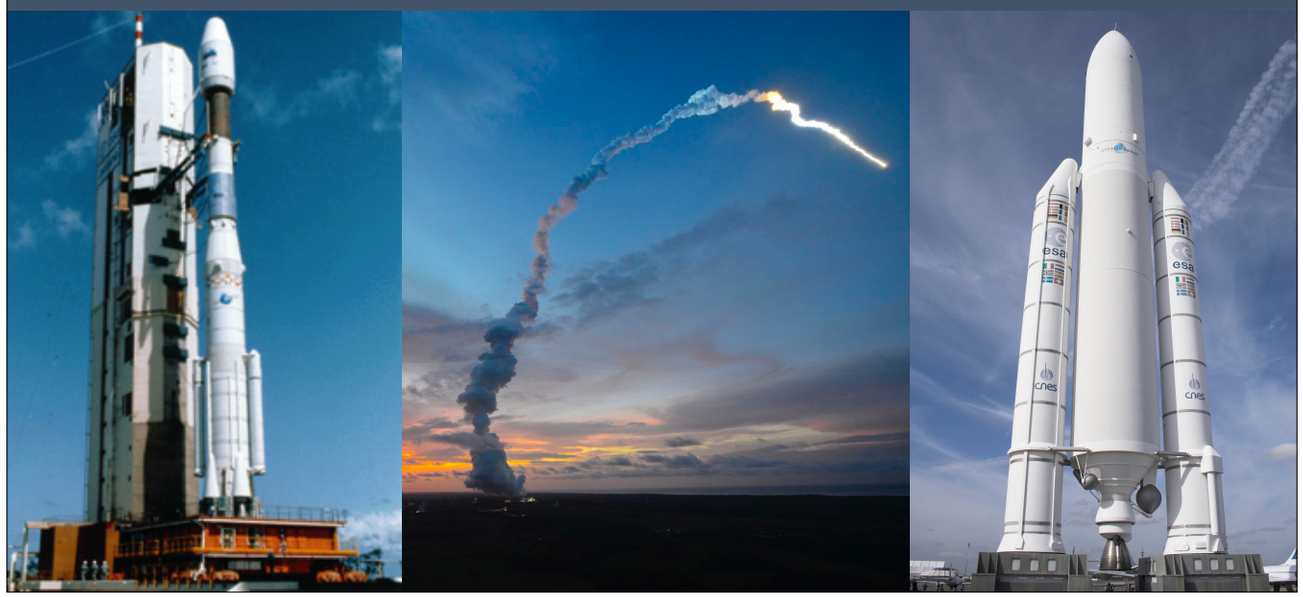


- went too close to Mars and disintegrated in the atmosphere
- NASA specified newton-seconds (N s), Lockheed used pound-second (lb s)
- Ground issued thruster firings, put it too close to Mars' surface
- 2 navigators noticed earlier, but their concerns were dismissed
- even had a meeting to fix it, but didn't end up doing it
- Mars Global Surveyor had conversion correctly, but algorithm was too complex (or not commented enough) to read, so new code didn't include conversion
- Lesson: MOAR UNIT TESTS!!!11oneone

Saturn V

- George Mueller, NASA Admin during Apollo
- “you don't want to be testing piece-wise in space. You want to test the entire system because who knows which one's going to fail, and you'd better have it all together so that whatever fails, you have a reasonable chance of finding the real failure mode, not just the one you were looking for.”
- Carried Astronauts around the moon on its 3rd flight vs. 10th or later as originally planned.
- Saved millions of dollars, years of development, and probably enabled us to win the space race
- Lesson: Test Entire System as Soon as Possible

Ariane 4 and 5



- Maiden flight of launcher ended in failure
- veered off course and went b00m
- unhandled exception: converted a 64-bit integer to a 16-bit signed integer
- caused primary & backup guidance systems to fail
- unguided & out of control
- code reused from Ariane 4
- Ariane 5’s velocity MUCH higher
- … velocity monitoring wasn’t needed.
- Lesson: “If it ain’t broke, don’t fix it.” But you SHOULD re-test it.

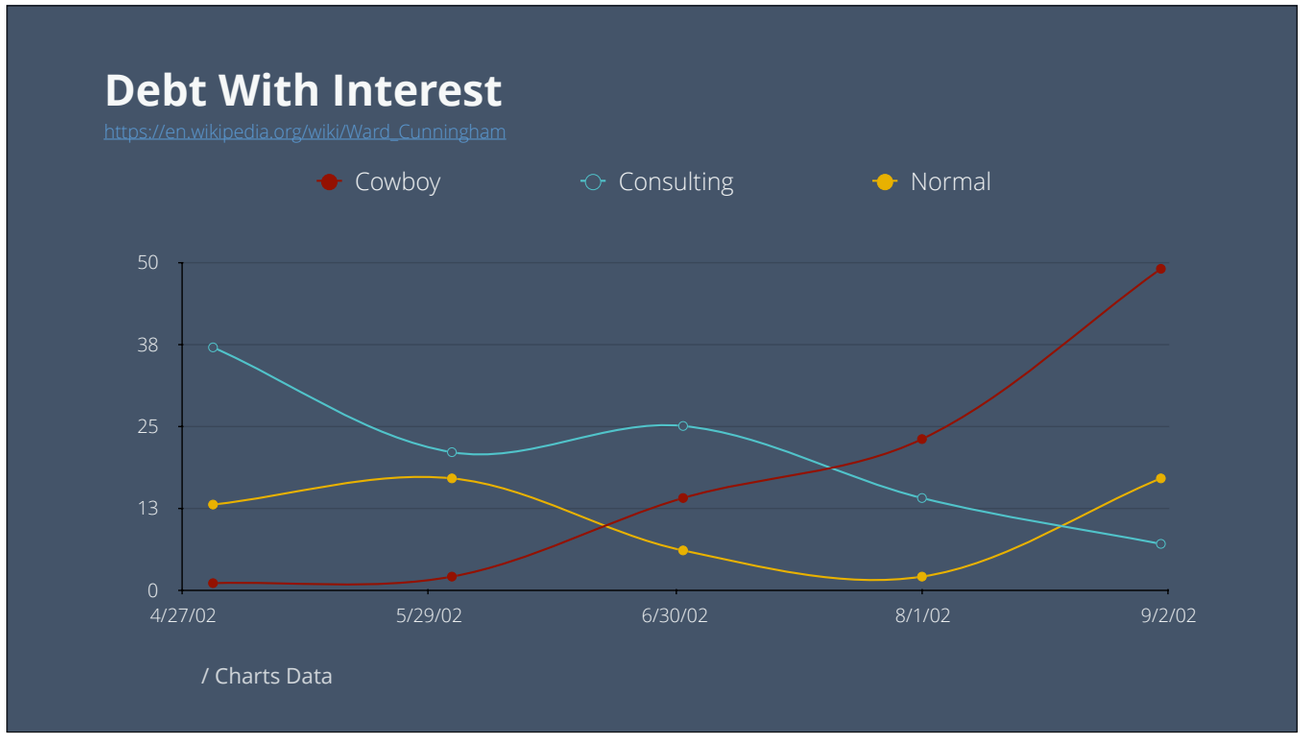
Space Shuttle

- before challenger: 2 weeks
- after challenger: 18 months
- 1,000 unit tests for every line of code
- code reviews to place certain lines of code next to each other in case of lightning strikes
- constantly would run out of system resources running so many unit and end to end tests & complex test simulations
- Lesson: After mission(s) over, you have EPIC fixture data to test with. Simulations are powerful testing tool.

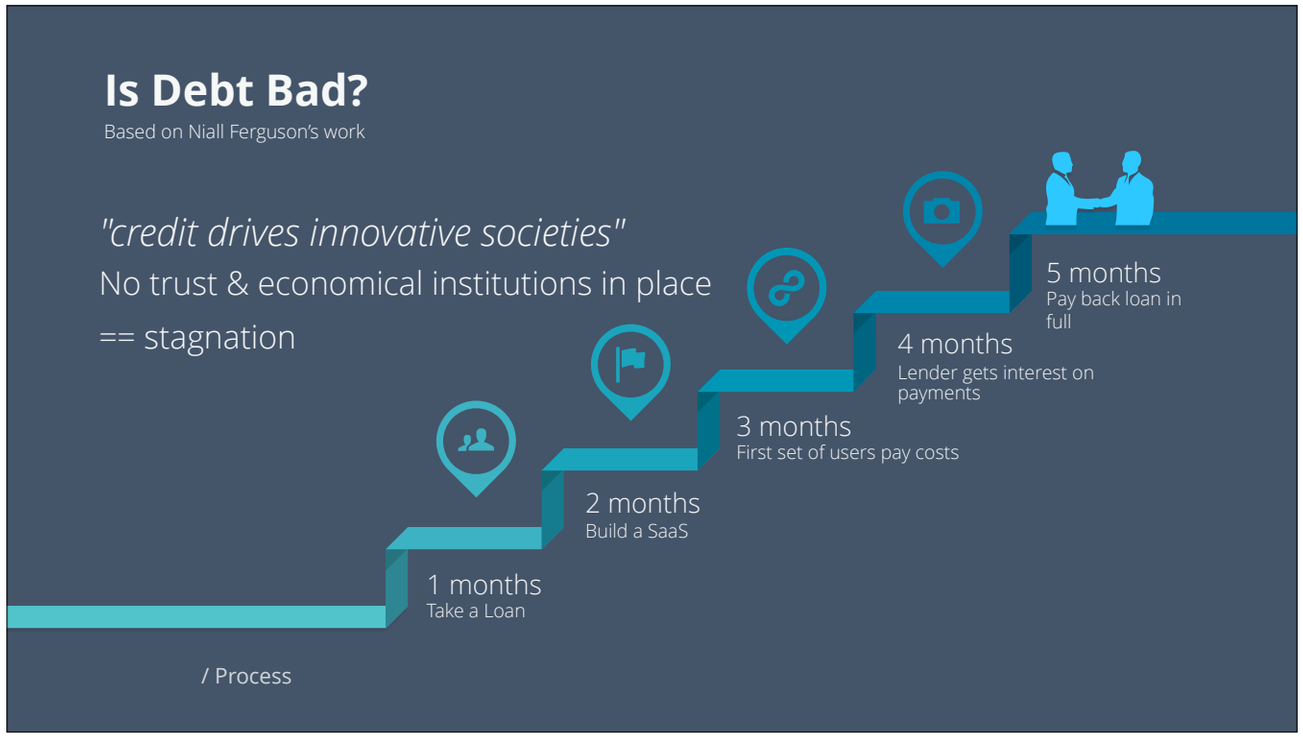
{breathe}

Technical Debt

https://medium.com/@joaomilho/festina-lente-e29070811b84#.iuhs8io61



Debt: Borrow or trade to get something now for a long term financial burden
Interest: Not just repaying what you borrow; incentive for the lender
Longer you wait to pay back, more interest
Ignore debt too long, you go bankrupt

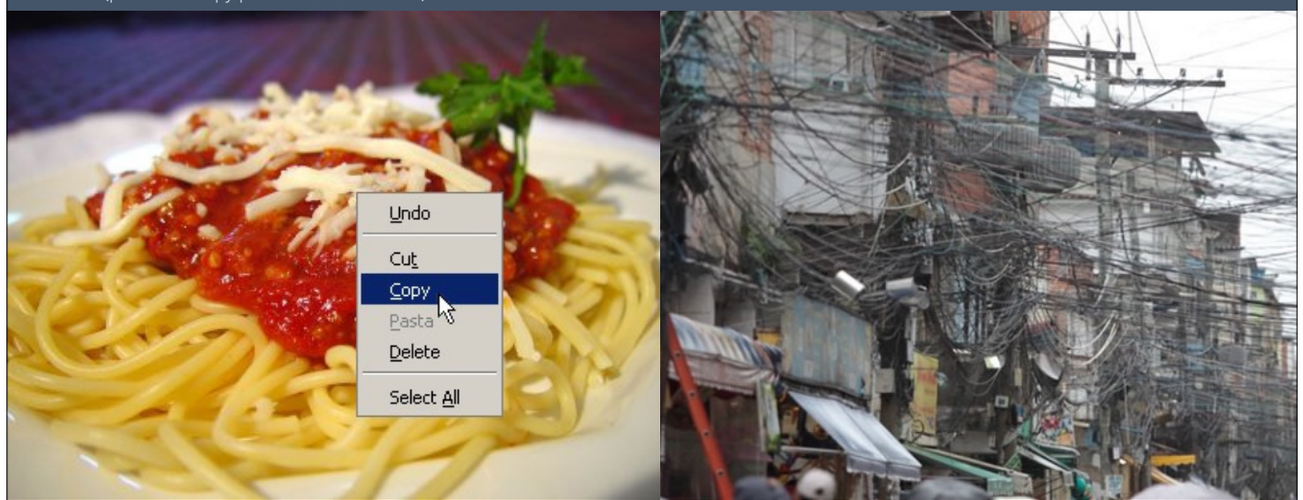


Niall Ferguson says credit drives innovative societies
Buy a machine, immediately create with it, use results to pay back debt
No trust & economical institutions in place == stagnation

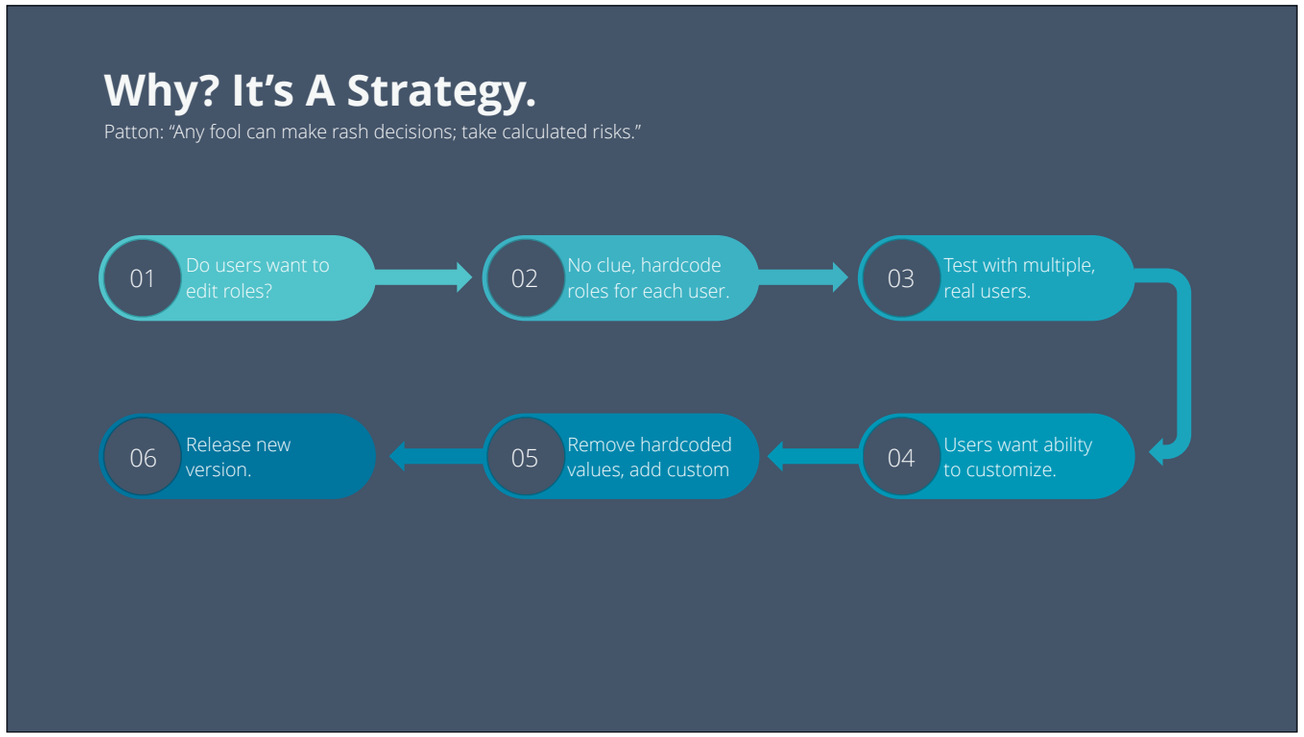
{breathe}

Metaphor for Non-Technical Managers

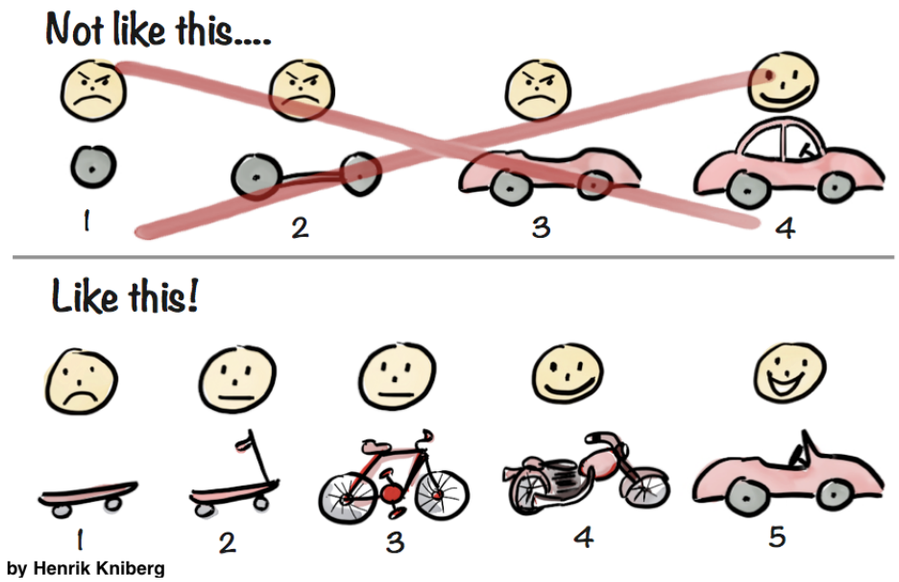
(picture of copy pasta & Brazil Favelas)



- If you don’t write software a good way w/biz understanding
- … you incur technical debt
- increases over time, like interest
- “Spaghetti Code” or Architecture
- Tightly Coupled
- Brazil Favelas w/power grid

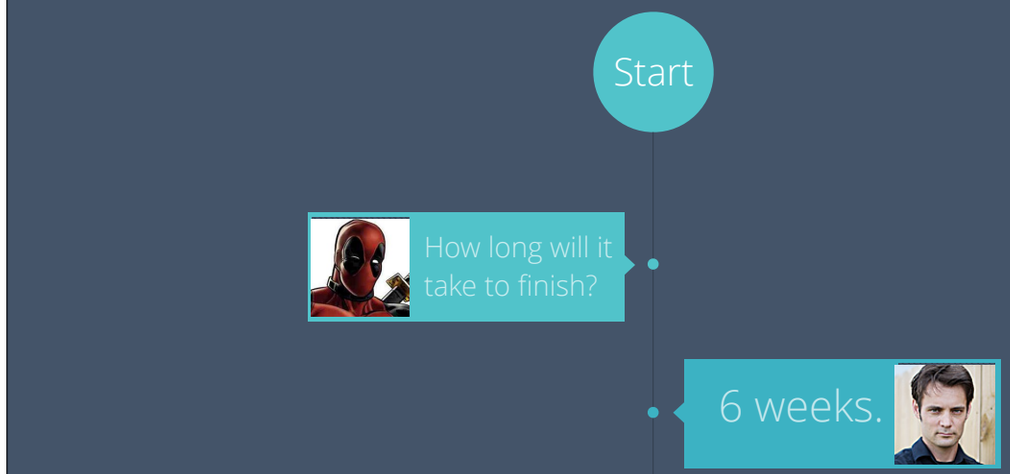


Defer decision until you know more later
Patton: “Any fool can make rash decisions; take calculated risks.”
Defer good implementation to meet deadline
Refactoring: Doing it right with hindsight
Takes more time than to code it originally
Managers are cool w/it because they never engage w/consequences

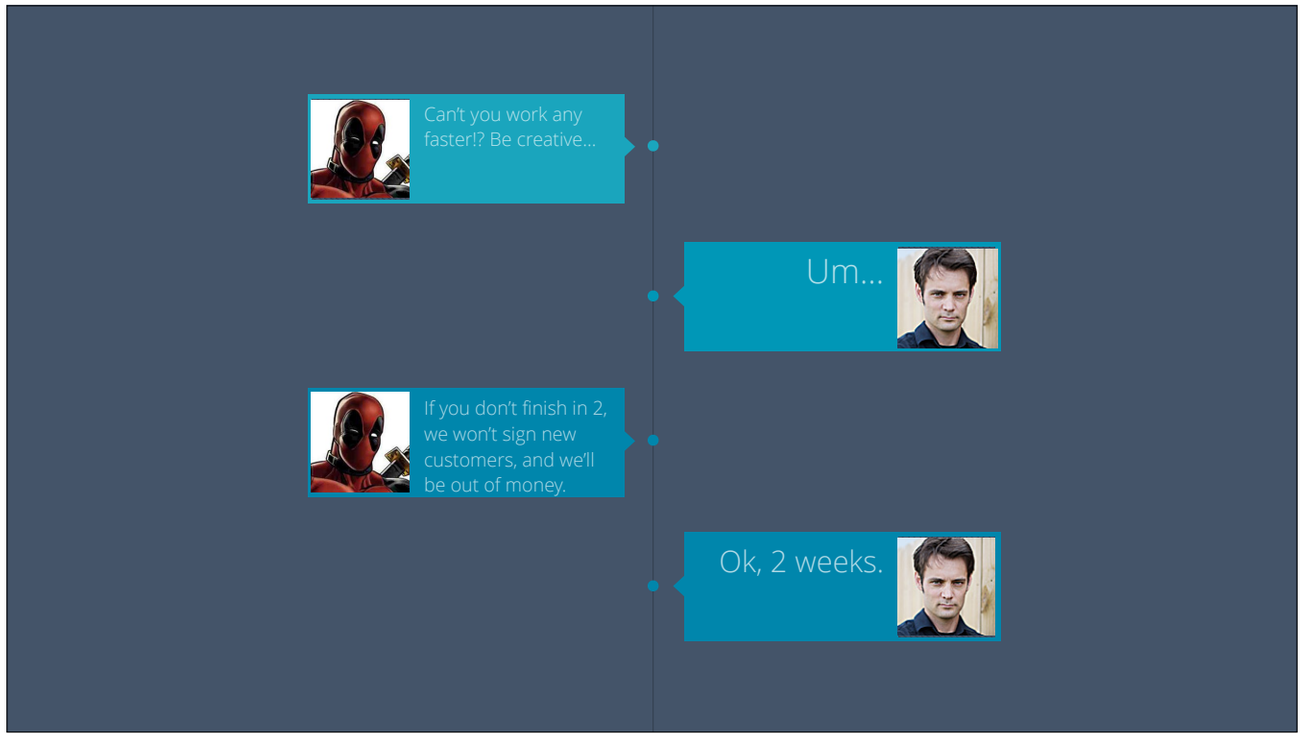


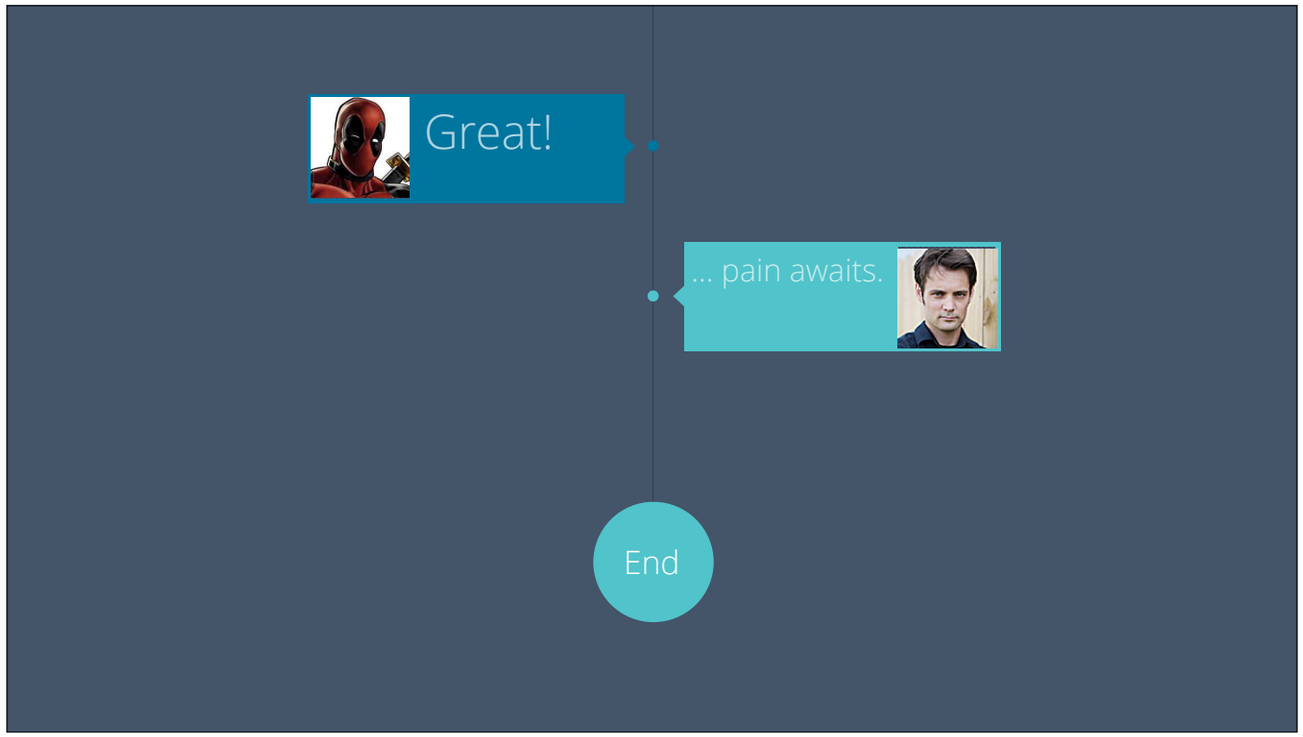
- LinkedIn’s founder Reid Hoffman “if you are not embarrassed by the first version of your product, you’ve launched too late”
- Release Early, Release Often
- MLP vs MVP: Sacrificing scope by ruthless prioritization
- Lean Startup (Book)
- Most Viable Product (Book)

The "Negotiation"



Bad code isn't technical debt, it’s bad code
Explain “The Negotiation”
Chad Fowler’s Crunch Mode Antipattern
Vet Devs make n00b mistakes
Kill passion, make best devs leave
no accountability, hurry == excuse for mistakes
erodes trust
it’s not an investment
The Clean Coder Book


Can't you work any faster!? Be creative...
Um...
If you don't finish in 2, we won't sign new customers, and we'll be out of money.
Ok, 2 weeks.


Great!
... pain awaits.
End

Rewrite

- False Dream
- Less Time to dev original product? No.
- Keep all features? No.
- Fix original behavior & knowledge lack that got us there? No.
- New Features Halted? No.

{breathe}

Modern DevOps & Lean Engineering

Prevent TD & Keep It In Check

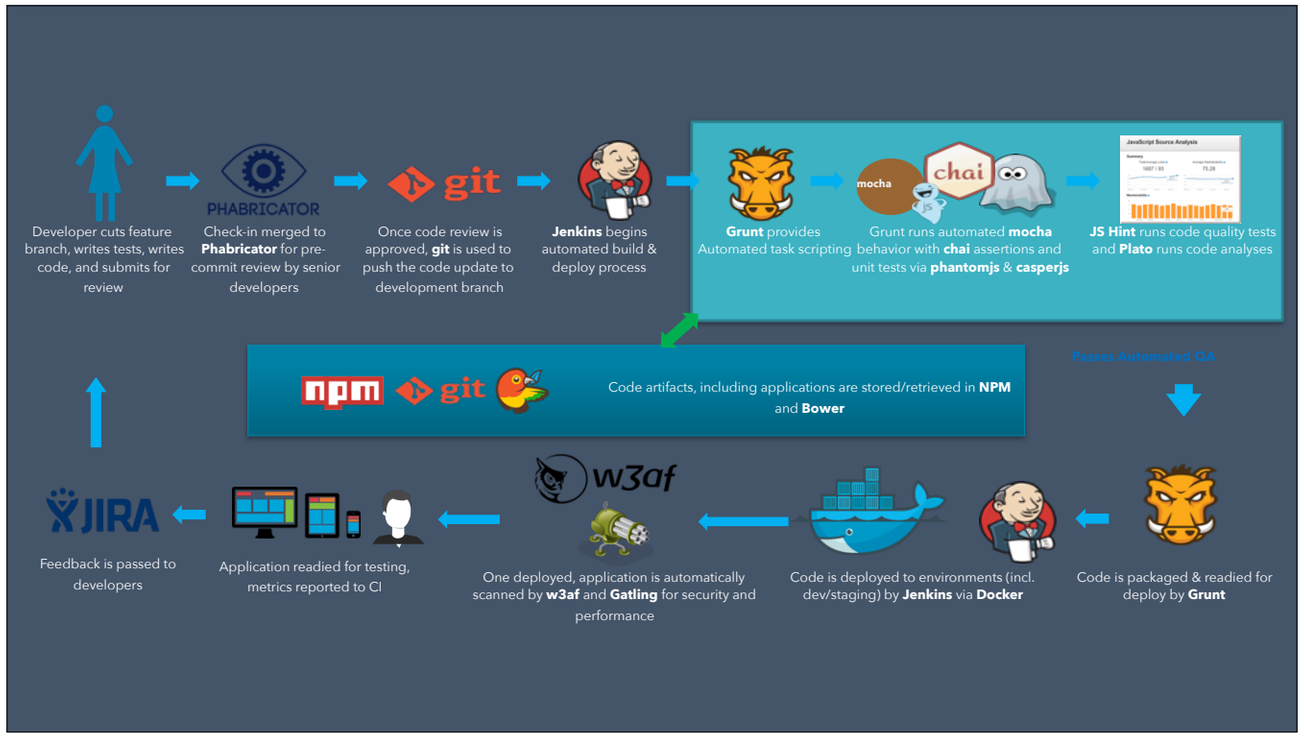
- Lean Engineering
- DevOps

DevOps

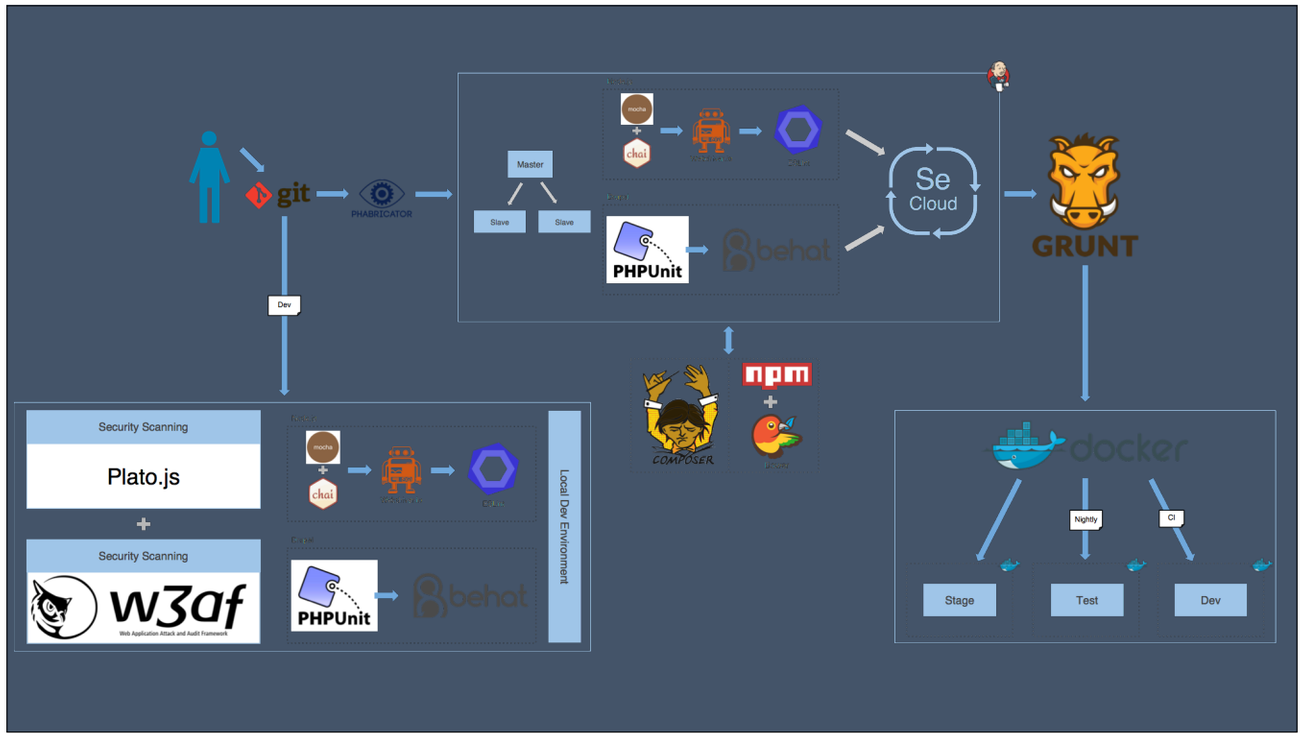
Prevent TD & Keep It In Check

- Merging IT Operations with Development
- Day 1 Activity
- Environment setup for development, testing, and staging as necessary
- Setup and configure DevOps tooling (SCM, CI, Code Review system, etc)
- Dry run of end to end process to ensure complete flow and control points

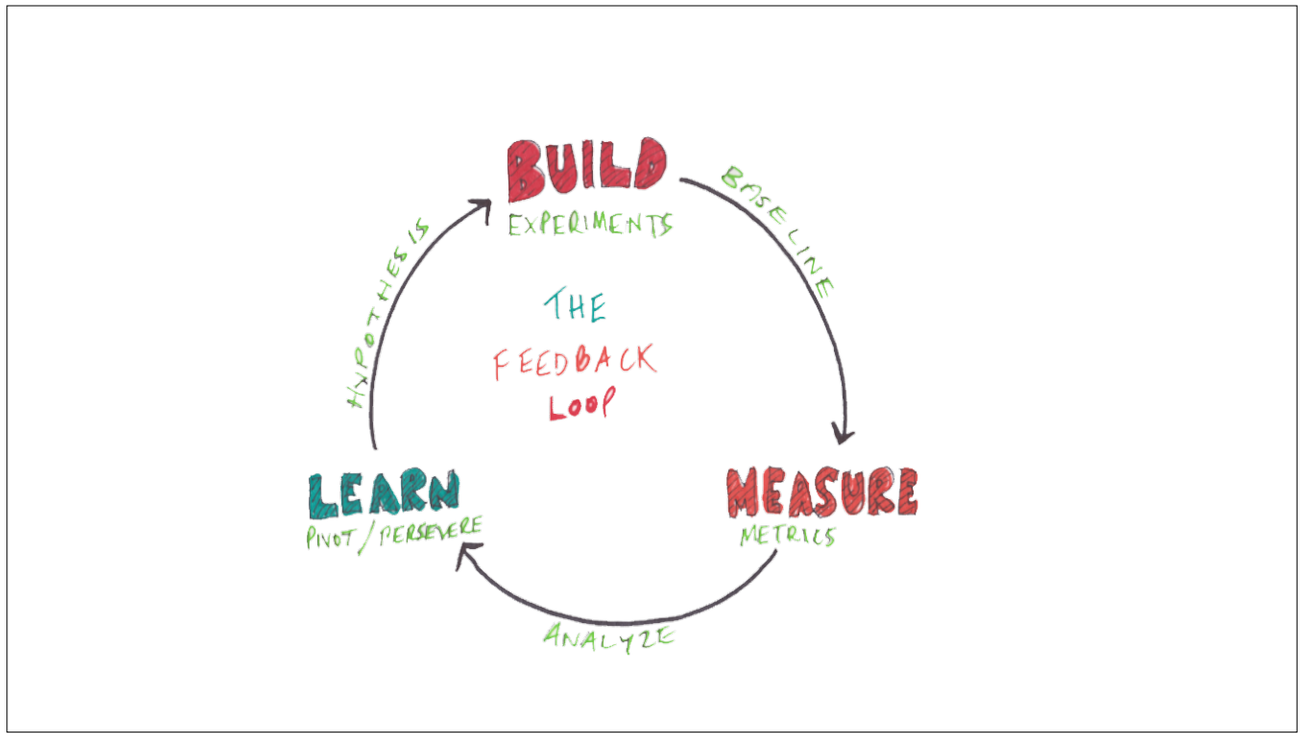
Parents, your job.
Kids get burnt a few times, THEN they'll put oven mitts on.



Close to nine in 10 (88%) now employ DevOps methodologies, and 42% strongly agree that DevOps is a top priority for their organizations. However, only 24% express strong satisfaction in the current state of their DevOps programs.



Automated Deployment


BUILD
EXPERIMENTS
BASE LINE
MEASURE
METRICS
ANALYZE
LEARN
PIVOT / PERSEVERE
HYPOTHESIS
THE
FEEDBACK
LOOP

{breathe}

Going Quantum

... not all the way.

- We want AI, but don't need it
- In the meantime...
- Deep Learning, Blue Brain
- Moaarrr Data (also NEW data)
- Quantum Processing (Faster Concurrency)
- Instant, Long Range Communication via entanglement (Lasers are cool, but...)


Still room for biology
Alex Nugent: "When algorithms get intertwined with architecture – they become pretty strict and specific,"
Low powered Arduino's that self-heal
memristors: based on neurons
http://knowm.org/

Thanks for your time! Questions?

- Jesse Warden
- @jesterxl
- jesse@jessewarden.com
- jesse.warden@accenture.com
- http://jessewarden.com
- Click Books Tab http://www.oreilly.com/pub/au/4997