The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

csc444h: software engineering I

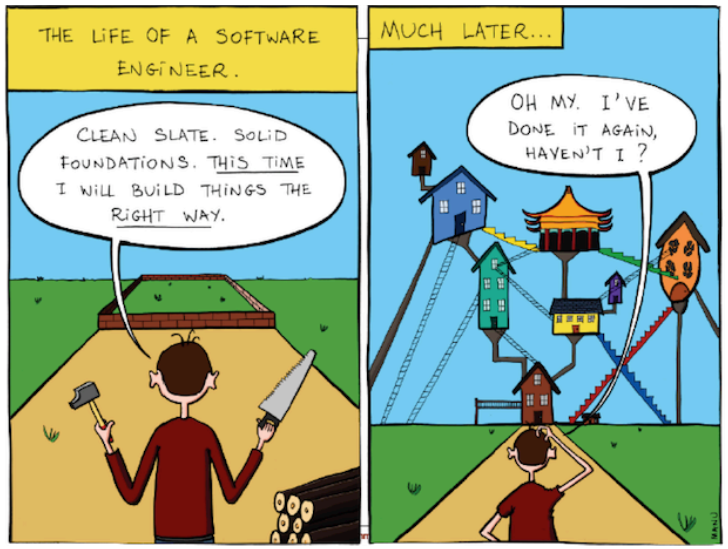
matt medland
matt@cs.utoronto.ca
http://www.cs.utoronto.ca/~matt/csc444

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

why do we need a course on engineering (large) software systems?

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

motivation



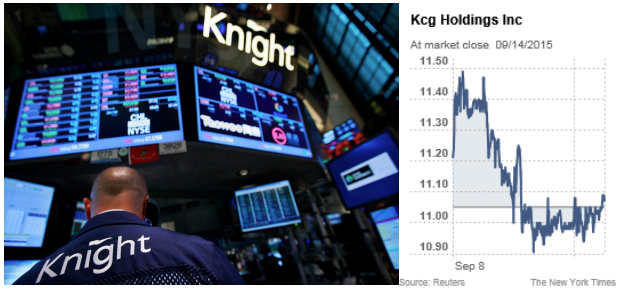
The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

motivation (2)

- historically, humans are not very good at engineering large software systems
  - see "why software fails" under readings
- how bad can we be? software is everywhere! some of us must be ok at it…right?
- ZDNet reported in 2012; *"Worldwide cost of IT failure: $3 trillion"*
  - lots of room for improvement!

motivation (3)

- Knight Capital trading "glitch" cost $440m



- software accidently bought high, sold low
- $10m per minute

motivation (4)

- national programme for IT in the NHS (UK national health service) NPfIT:
  - cancelled project took 9 years and cost £12bn
  - from computerworlduk.com, 11/09/2011:

"The [UK] government will formally announce the scrapping of the National programme for IT in the NHS…to "urgently dismantle" the health service IT scheme comes after a series of damming reports…The final nail in the £12 billion scheme will be announced this morning…set up in 2002, is not fit to provide services to the NHS. 'There can be no confidence that the programme has delivered or can be delivered as originally conceived,'"

motivation (5)

- is it because they didn't use agile?
- agile projects can fail too! and just as bad
  - universal credit is Britain's plan to consolidate all welfare payments into one.
  - touted as the world's biggest agile software project, now close to total failure!
  - original budget = £2.2bn, cost so far = £12.8bn
  - article:
    http://news.slashdot.org/story/13/05/25/139218/worlds-biggest-agile-software-project-close-to-failure
- I bet the welfare recipients could have used that £12.8bn!

motivation (6)

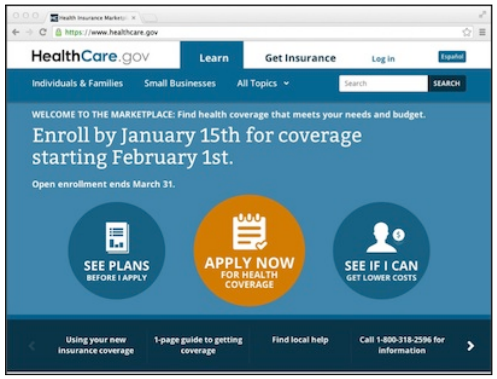
2003 blackout

race condition on computer system in Ohio caused stalling of audio/visual alerts

primary system failed, then shortly after, the backup also failed – same bug!

motivation (7)

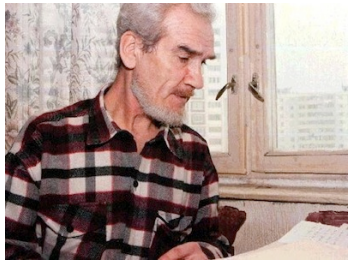
healthcare.gov



$500m for a website? are you serious?!?!

motivation (8)

- the hero you've never heard of:
- Sept 26, 1983, at height of cold war
- console reported 5 minuteman intcntl. ballistic missiles launched



Soviet air defense officer, Stanislav Petrov

- was actually static (noisy signal)
- reported *"I had a funny feeling in my gut"* that it wasn't real
  - decided against WWIII – thanks Stan!

motivation (9)

- Michigan dept. of corrections grants prisoners early release

- Calif. paroles 450 violent offenders (without supervision)

motivation (10)

- author of *Why Software Fails* states that:

"Studies indicate that large-scale projects fail three to five times more often than small ones."

- article:
  http://spectrum.ieee.org/computing/software/why-software-fails

what is large?

- lots of talk about "large" systems
- what do we mean when we say a software system is ***"large?"***
  - class discussion
  - some examples
  - largest software you have ever worked with?

what is large? (2)

- what makes a software system ***"large?"***
  - kloc?, "what about comments?", ok, fine, executable statements when compiled?
  - number of bytes?
  - person-hours, or some other effort metric?
  - number of developers?
  - number of features?
  - number of processors running the code?
  - number of users of the software?
  - number of bugs?

what is large? (3)

  - size of the box? number of floppy disks, and hours it takes to install it? ☺

what is large? (4)

- answer (well at least my answer):
  - something that is not a *"spike"* and is intended for users other than developer – released
  - can benefit from, and is not hindered by, proper practices (modeling, source control, automated unit testing, continuous integration…)
  - standard development tools are not too cumbersome (ex. making an eclipse project is overkill => not "large" for our purposes)
  - so, basically anything non-trivial

summary

- this course deals with the challenges of major software projects
  - working with legacy code/systems
  - analyzing problems
  - deciding what is feasible, with the given team, and the amount of time available
  - managing next release development
  - delivering quality software in a professional software environment

The Edward S. Rogers Sr. Department
of Electrical & Computer Engineering
UNIVERSITY OF TORONTO

summary (2)

- this course is different from most
  - you will work on a much larger project with code written by (many) people you've never met
  - you will use your own judgment in deciding what is feasible
  - you will manage your own risks and plan
- your mileage will vary
  - there are no right or wrong answers
  - credit will be given for good judgment