Months to minutes

Setting the scene

I am

- Dan North
  - dnorth@drw.com
  - http://blog.dannorth.net
  - @tastapod
- 20 years in IT
  - 8 years with ThoughtWorks
    - "the BDD guy"
    - "agile troublemaker"
    - developer
    - build primate
    - coach
      - agile process
      - development
    - consultant
    - organisational change
  - 1 year (so far!) with DRW Trading
    - programmer

we do

- machine-speed trading
  - HFT
- people-speed trading
  - "heads-up"

we need

- low latency
- high throughput
- awesome UX
  - correct
  - obvious
- data integrity
  - compliance
  - correctness

Agile says...

process

- you need a prioritised backlog
  - have themes
    - "reset the board" every week
    - even they are only guidelines
  - review daily
    - stuff happens
  - macro-level estimation
    - really is only a guideline
    - build trust through delivery
- you need stories & acceptance criteria
  - common sense, communication, trust
- analysis paralysis is bad
  - useful as long as you are still learning
  - and the learning is SMART
- behaviour should be driven from the UI
  - behaviour driven from purpose
  - what's the difference between this and ready?
- you can't release rubbish
  - getting to production exposes flaws
  - drive journey right through
  - bold, but not reckless
- best kind of user testing
  - if your users trust you
  - and you are responsive to feedback
  - and you can roll back instantly

technology

- don't rewrite where you can revive
  - write good, short-lived code
    - waves of diminishing value
    - permanent scaffolding
    - so has to be good quality
    - painting the Forth bridge
  - have narrow, collaborating services
    - integrate at UI
- automated tests are better than manual tests
  - manual tests uncover subtle bugs
  - also randomized tests
    - cf QuickCheck
    - concurrency bugs
  - nothing beats putting it live!
- mocks are useful
  - mostly test inputs and outputs
  - hand-roll simple recorders
  - don't fear system tests at the edges
    - e.g. sockets
- production code should be test-driven
  - unit tests should be test-driven!
    - expect to refactor for unit testing
  - unit test the risky areas
    - make the rest "obvious"
  - pure TDD => deliberate ignorance!
    - solve the right problem
    - make system small enough
  - leave TODOs in the code
    - make the comment stand alone
    - use the word "should"
- copy & paste is bad
  - quickly adapt/reuse existing code
  - Ginger Cake
- spikes should be discarded
  - Spike And Stabilise
  - spike to explore alternatives
  - add tests when it stabilizes

people

- rotate pairs, avoid silos
  - pairs in "shallow" silos
  - make design decisions with the team
  - sync up regularly
- you need analysts/testers/DBAs/etc.
  - you need the skills covered
  - have generalizing specialists
  - have experts on hand
    - call when you need them
- put yourself in the user's head
  - user knows what they want, not how
  - "I'll know it when I see it"
  - put yourself in a UX designer's head

What survived?

- Planning
  - weekly themes
  - "natural" planning
- Pairing
  - in (shallow) silos
- Stand-ups
  - 1 x recap and planning
  - 1 x status with sponsor
- Co-location
  - Programmers & traders on same desk
  - Dev-ops are critical
  - Own your build
  - Own your testing
  - Own your analysis
- Refactoring
  - mercilessly!
- Prove it!
  - habitability
- Continuous delivery
  - build, deploy, rollback
  - sometimes 20-30 times/day!
- Automated testing
  - but not -> 100% coverage
- Intention-revealing test names
  - eventually!
  - "blah"
  - sketching
    - Kevlin Henney

What emerged?

- Deliberate discovery
  - Focus on reducing ignorance
  - Short Software Half-Life
    - continuous rewrite
    - three phases
  - Get comfortable with uncertainty
    - I'm not!
  - Product ownership
    - make it yours
    - look with your feet

Technology choices

- fast dev/test cycle
  - java, javascript
  - rake, fig
- fast deploy/undeploy
  - side-by-side
- fast SCM
  - svn -> git

Principles

- Anything is better than nothing
- What does the user want to see?
- Assume code won't be there in 2 months
- *"Onward!"*

Conclusions

- Not your father's agile
  - slow and steady
  - what happened to "people over process"?
- You can go insanely fast
  - even in "the enterprise"
  - but it requires discipline
- Trust and collaboration trumps process
  - but wait, that's agile!
- Shoot for the stars
  - you might surprise yourself