Computer Science

UCL

10 OPEN CHALLENGES IN SOFTWARE ENGINEERING

Anthony Finkelstein

Brief outline of talk …

The Discipline of Software Engineering…

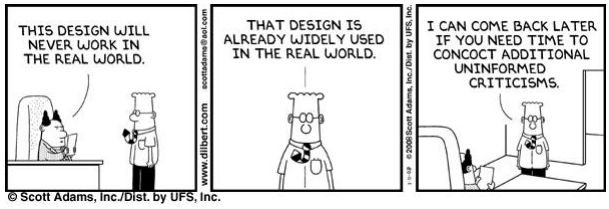
- Sustained relevance of ‘big agenda’
- Substantial scientific progress but (perhaps) receding impact on practice
- Significant advances in some areas
  - Testing
  - Automated verification (model-checking)
  - (largely outstripping capacity to absorb innovation)

The Discipline of Software Engineering…

- Uncertain directions in other areas
  - Software architecture
  - Software design
  - Software requirements
- Difficulties in making progress in some areas
  - Software development tools
  - 'Methodologies' (modelling and process combos)
  - Middleware
- Grounds for optimism

Challenge I – sketch

- Move to an 'evidence-based' practice
  - cf medicine
- Existing practice
  - Evidence-free
  - Anecdotal
  - Quasi-evidence-based



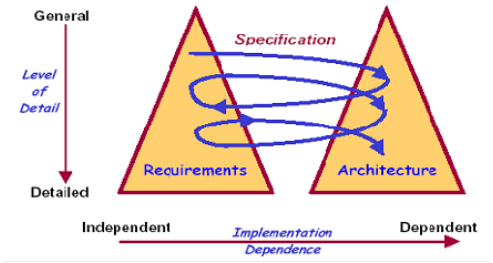
Challenge I – tentative approach

- Review 'classic' work
- Underpin work with clear hypotheses
- Openly encourage 'reproducibility' studies
- Reorganise research efforts around a 'translational pipeline'
- Restructure software engineering education to reflect an evidence-based approach
- Engage with the 'blogosphere'

http://www.sciencecartoonsplus.com/index.php

Challenge II – sketch

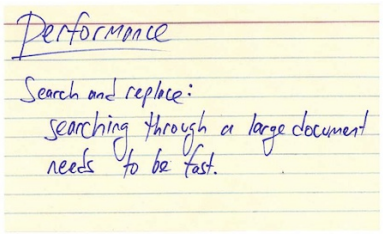
- Making 'twin peaks' more than a *picture*



Weaving together requirements and architectures
IEEE Computer, Vol. 34, No. 3. (2001), pp. 115-119.
by B. Nuseibeh

Challenge II – tentative approach

- 'Non-functional properties' drive architectures (perhaps)
- Map the relationships between these properties and architectural styles
- Insights from architectural evolution

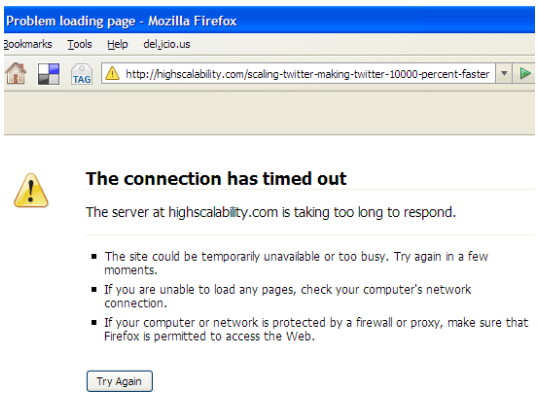


Challenge III – sketch

- Engineering scalability
  - 'Internet-scale' services
  - Handling large and rapid variations in the demand for resources
- Existing practice
  - Some high level patterns for limited classes of application
  - Resource profligacy
  - Suck it and see (dimension by dimension)

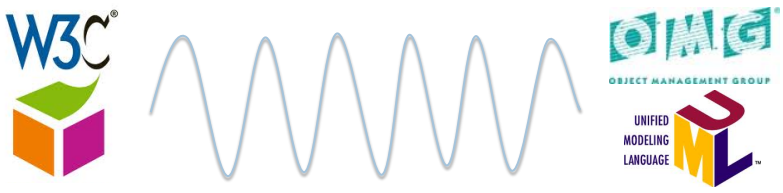
Challenge III – tentative approach

- Large-scale testbeds
- Scaling 'in the wild'
  - Surmounting the data challenge
- Architectural breakdowns
- Dynamic systems models



Challenge IV - sketch

- Convergence of web standards and software engineering standards
- Existing practice
  - Fundamentally separate worlds with OMG and W3C moving in different incompatible directions
  - Wasteful of effort and of technical opportunity



Challenge IV – tentative approach

- Stop playing at the periphery and pull back to fundamental requirements, a fudge probably will not work
- Devise and test shared schemes
- Identify quick wins
  - For example smart semantic tagging of software artefacts
- Start the 'hard grind' of engagement with standards bodies

Challenge V – sketch

- Resource estimation
- Existing practice
  - We are unable to reliably predict the cost/effort required to build a system. We may be fortunate and have built a very similar system before.
  - Function Points are precious little assistance. 'Jelly Beans' only work for small systems, relatively 'late' in the process.

Agile Montage

Challenge V – tentative approach

- Nothing even on the horizon here!
  - Perhaps machine learning has a part to play
- We are probably going to have to:
  - Rethink software economics
    - Making money a 'first class object' in software engineering
  - Get a much better handle on 'programmer productivity'
  - Provide an appropriate data-sharing infrastructure

Challenge VI – sketch

- New models around SaaS
- Existing practice
  - We know how to build SaaS (sort of, see III) but we don't know how to:
    - buy it
    - manage QoS
    - achieve interoperability

Challenge VI – tentative approach

- Stop 'wasting time' with fine grained software services (wake up and smell the cocoa)
- Enterprise mash-ups
- Requirements methods based on balancing mutability
- 'Security in the cloud'
- 'Walk away' methods

Challenge VII – sketch

- The apotheosis of 'apps'
- Existing practice
  - Channel delivery
  - Highly-tuned, device-specific interfaces across to services with 'sync' to clients
  - Because a viable payment model exists …

Challenge VII – tentative approach

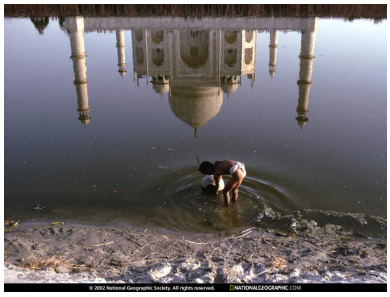
- Requirements engineering for mass-markets
- New types of 'product-family' engineering
- App Stores [SM]
- App management
- App assembly

Challenge VIII – sketch

- Development of emerging classes of 'adaptive' system
- Existing practice
  - Problems with systems that must adapt to context
  - Problems with systems embedding significant COTS/ Community Sourced independently evolving components
  - Problems with systems that involve user scripting and 'plug-ability'

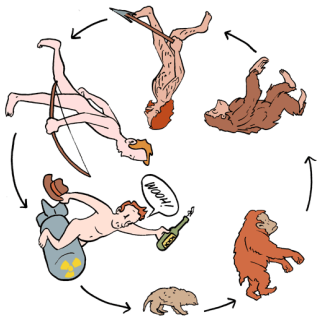
Challenge VIII – tentative approach

- Moving reflection from being a programming language level mechanism to software systems that can 'account for themselves' – models@runtime
  - Can reflect their requirements and (through monitoring) the extent to which those requirements are being satisfied



Challenge IX – sketch

- "History repeats itself, first as tragedy, second as farce" Karl Marx
- Existing practice
  - And third, and fourth, and …
  - See CHAOS reports passim

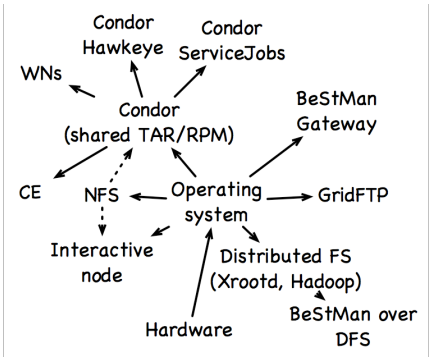


Challenge IX – tentative approach

- Mismatches at the boundaries between business and software engineering
  - Governance

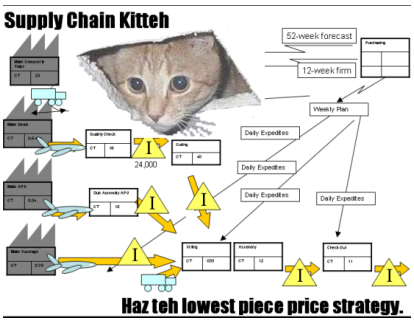
Challenge X – sketch

- Addressing complex inter-product and inter-supplier dependencies
- Existing practice
  - None to ad-hoc



Challenge X – tentative approach

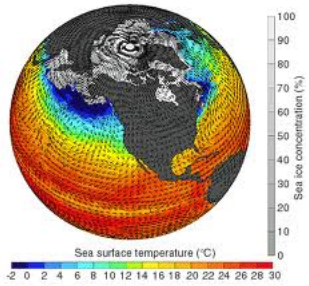
- Rethinking software production
- From garage 'design and make' to …
  - Supply chain
  - Software ecosystem



And by way of an inadequate conclusion

Two Free Challenges (for Oxford)

- Beyond … software engineering
  - Physiome, energy and sustainability models
    - require large composite heterogeneous models (& meta-models)
    - multiple stakeholders
    - subject to collaborative construction and rapid evolution
    - prone to error



And by way of an inadequate conclusion

Two Free Challenges (for Oxford)

- Bringing automated verification to software engineering practice
  - … and making the kind of breakthrough for theorem proving technology that has made model checking a practical reality