Web Scaling Software Development into the Cloud

Philip Haynes

Principal Consultant

The Shrinking Time to Deliver Projects

- Dimensions of project complexity continue to increase
  - Size, global distribution and user volumes
- Yet the time to deliver projects decreases
- Multi-level REST architectures can increase project delivery velocity by 5-10x

The Shrinking Time to Deliver Projects

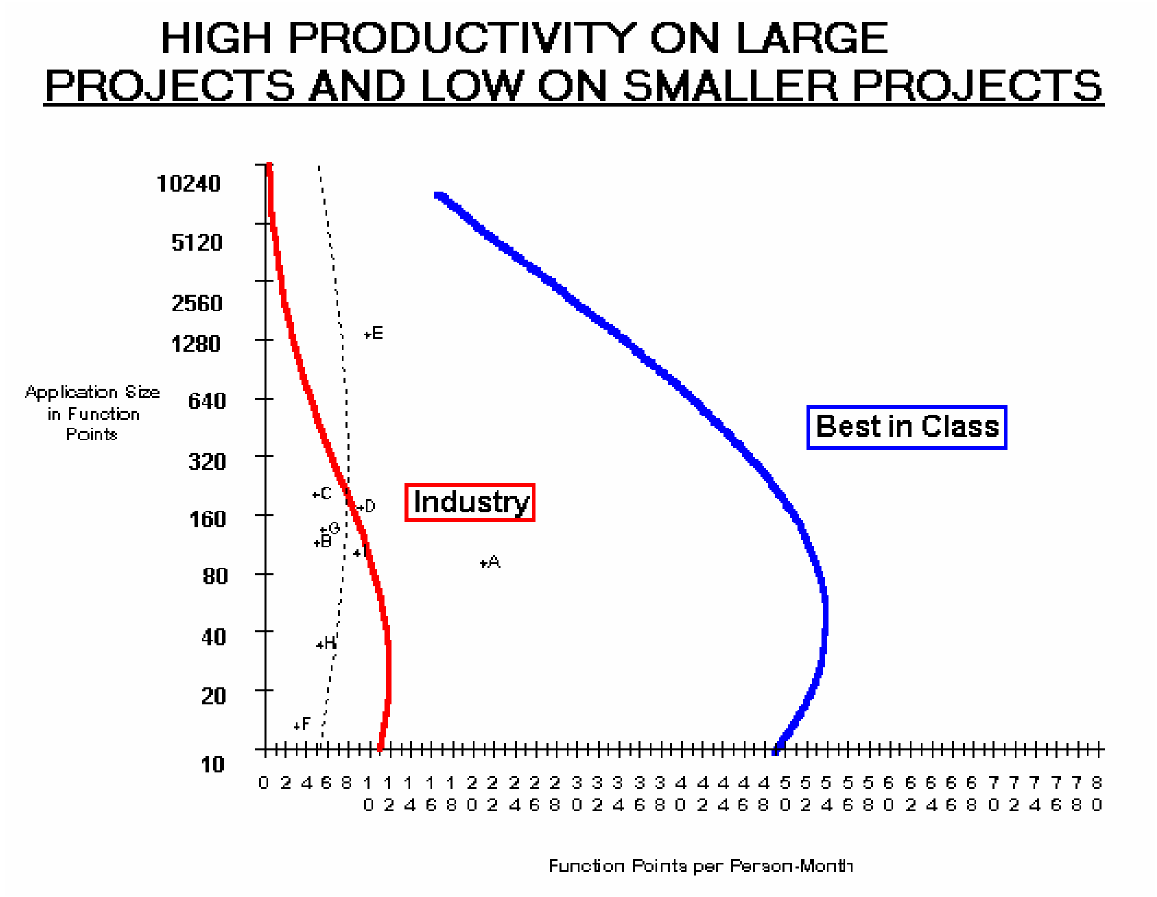
- The Management Model implicit in Open Source Development Wins
  - Adjust open source models for a commercial environment

Q: HTTP server project biggest impact on the world?
A: Its method of collaborative decision making

Roy Fielding – Apache 3.0 (a tall tale) ApacheCon Europe '08

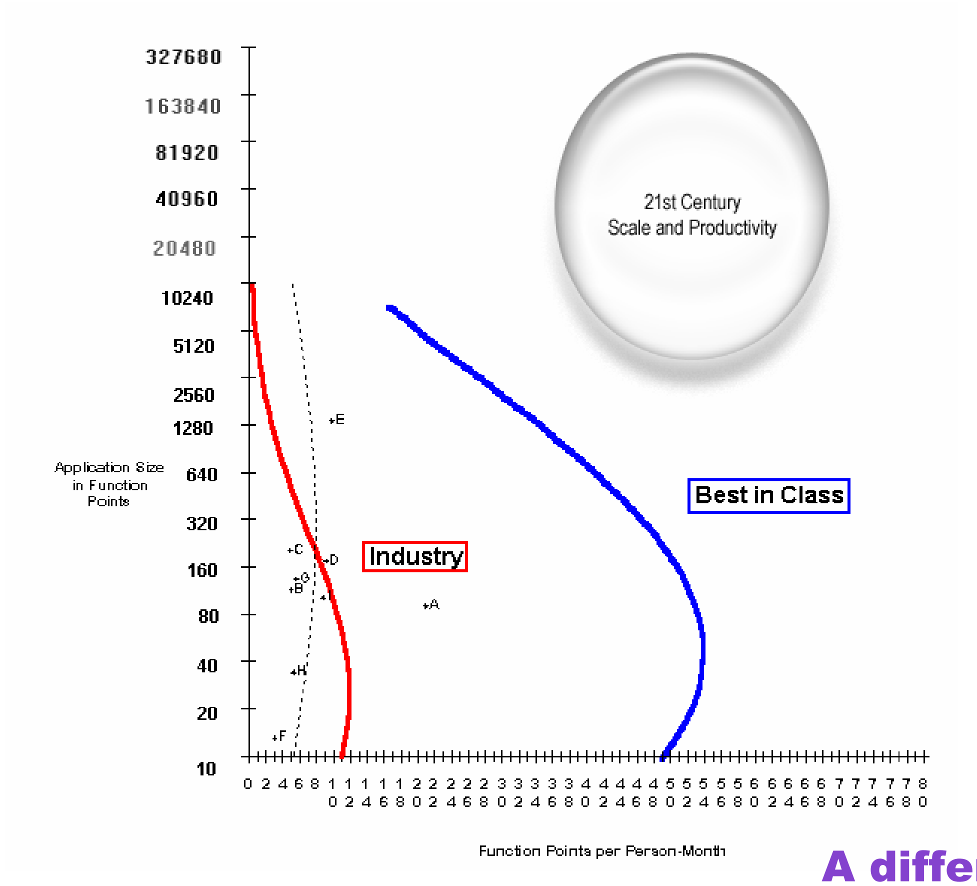
.JAOO

Last Century Productivity



Object Consulting

21st Century Productivity



A different sport..

Software Engineering in the Large

- Significant Functionality > 10,000 FP / 1 million SLOC
- Consumer rather than enterprise focused (>1M users)
- Globally deployed
- Transact large amounts of money (> $1B)
- Multi-jurisdictional / highly regulated
- Significant media component
- Multiple system portals be it RFID, Mobile Phone, PC etc.
- Deal with hard technical issues such as Real Time
- > $10-$100M budget

Issues of Scale

- Great place to for software engineering challenges!
  - (and BIG toys)

Issues of Scale

- Time
- High Stakes / Politics
- Funding
- Requirements
- Communication and Control
- Resourcing – enough of the right people fast enough
  - Multi-disciplinary approach
  - Numbers of geographically dispersed people
- Quality, availability and ensuring the system works
- Integration / concurrent development
- Operations
- Cost

The Architecture of the Web - REST

- As a "Proof of Concept" for the construction of large systems the Web is a good starting point.
- So:
  - Why does the Web work;
  - Why did it grow so fast; and
  - How might insights be applied to software engineering in the large?

Architecture of The Web
Removing friction to scale

- Simple. Anyone can do it.
- Free. Everyone can afford it.
- Open Networking. Everyone can join in.
- Inherently Hyperlinked.
- Exploit “Small World” effects.
- Robustly accepts failure.
- Always on.
- Loosely coupled.
- No central point of control.
- Sharing, communities and people.
  - Enables a multi-disciplinary approach

Architecture of the Web - Technology

- Text for file & configuration stores
- URI’s - allows identification of anything and everything
- DNS – turning computer conventions to English
- HTTP – Not just a file download mechanism
  - A complete and sufficient protocol for wide-area distributed programming.
  - Not just GET’s but POST, PUT and DELETE operations and error codes.
- HTML – simple code for dealing with hyperlinked data
- Javascript – Downloadable programs on the client
- Stateless
- End-to-end
- Build real working examples first

Architecture of the Web - Technology

- Keep it simple
  - Remove accidental complexity from architectures

Software development is an inherently "hyperlinked"

Don't fight it

Software Engineering In The Large – Enablement

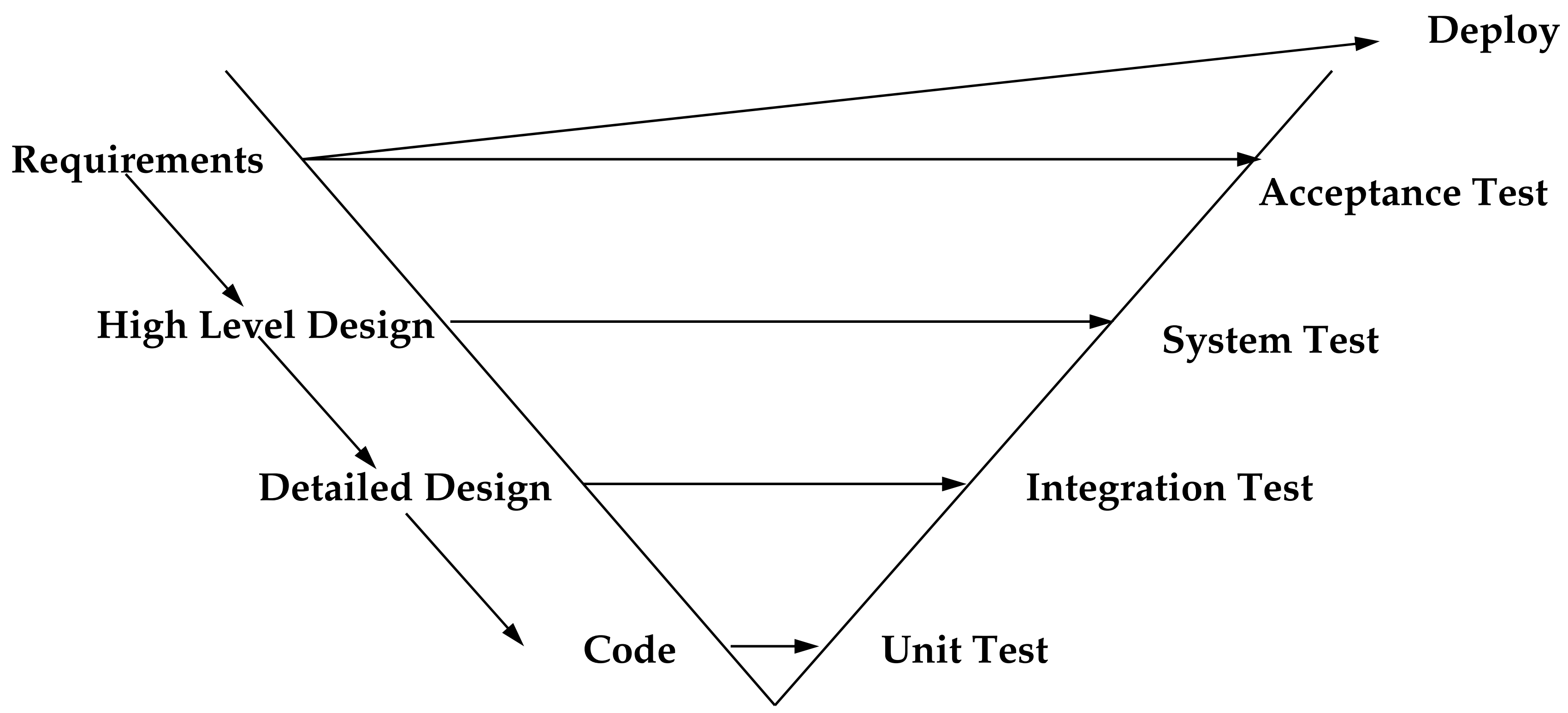
- Give everything a URI
  - Code
  - Runtime & Build time view of Components
  - Requirements
  - People
  - Contracts
  - *Absolutely everything*

- URI's give us:
  - Executable contracts
  - Place to stub out architecture
  - Mechanism for configuration control and traceability
  - A consistent multi-platform interface for on demand programming
  - Mechanism to change representations for example XML to HTML
  - An ability to get something working early

Software Engineering in the Large - Enablement

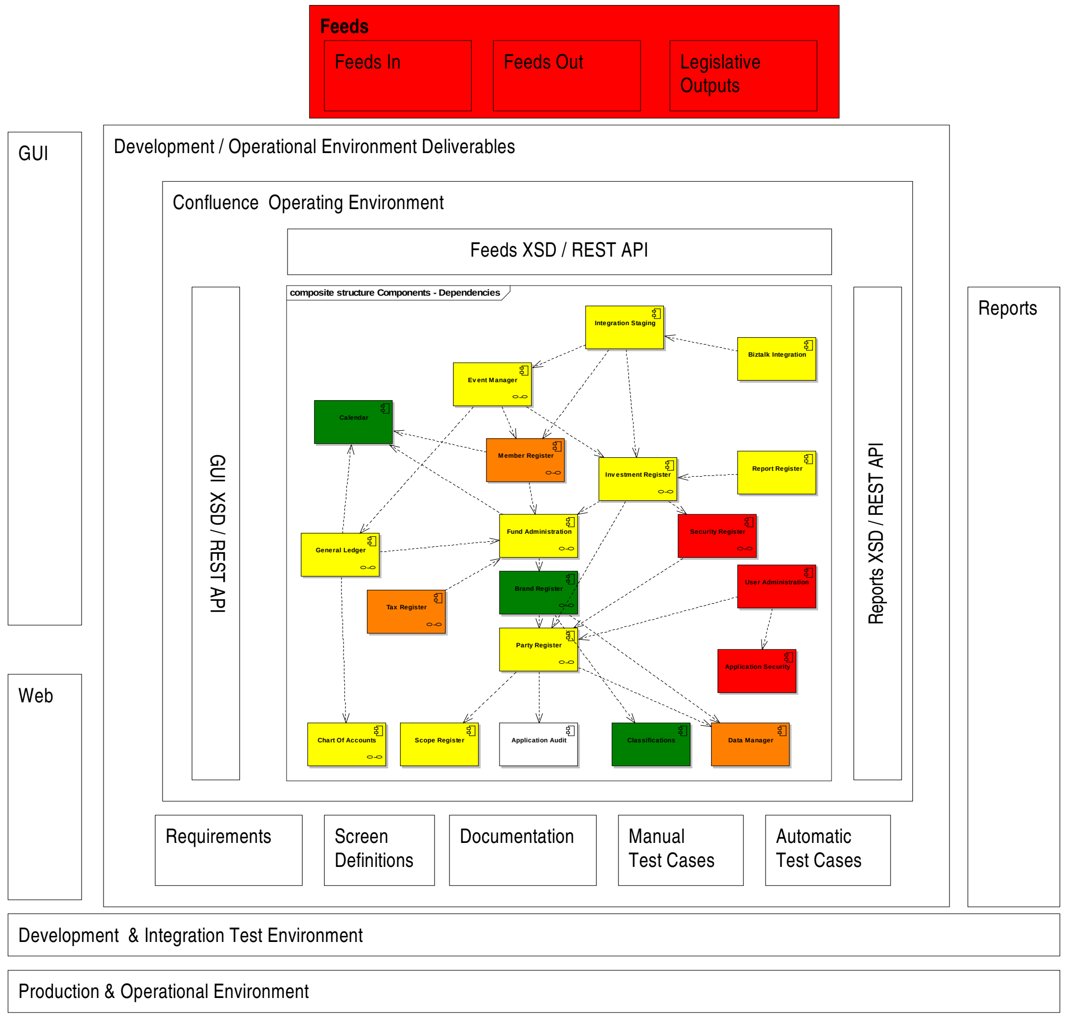
- Hyperlink Everything
  - Software Development is inherently hyperlinked
  - Hyperlinking provides a structured mechanism for accessing large amounts of system
  - View all system elements and click traceably through them
- Bind software components as late as possible
  - Design for mash ups from day one.
  - Smart URI’s rather than direct in memory bindings
- Enable independent evolution of system elements
- Enable reuse

Concurrent Development



JAOO

Component Dependency Analysis



Object Consulting

Programme Level Requirements

- Specification takes to long for it all be done in advance of development
- Instead identify core areas of scope and ensure that these are partitioned
- Detailed specification performed concurrently with build
- A “crash-hot” requirements team is key

Software Engineering in the Large - Continuous Rather Than Batch Process

- Continuous Build
  - Goes with out saying but Bamboo is a “Good Thing”
- Continuous Publishing
  - Wiki / Blogs provide a consistent mechanism of project state
  - Integration point of all system elements
- Continuous Running System
  - Systems should be designed to always run
- Continuous Visibility
  - Constant battle against the “Fog of War”
  - Significant visualization work required here

Software Engineering in the Large - Continuous Rather Than Batch Process

- Testing is 20-40% of project budget
- Manual regression testing is infeasible when programming in the large. It is too:
  - Slow
  - Expensive
  - Error prone
  - Can’t find faults only evident on scale
  - Loses intellectual property
- Continuous Automated Testing
- Automated test Via REST Interfaces
  - From a Thermodynamic / theoretical perspective this reduces entropy
- Automated testing makes testing a partition able task
  - Industrialize Development

Software Engineering in the Large
Industrial Development / Manufacture

- Identify repetitive, large volume software development tasks. These include:
  - Integration, Automated Test Creation, Page Creation
- Simple approaches to create function
  - Command line programmes to integrate systems together
  - Standalone HTML pages
- Web mechanisms to link and integrate components
- Low cost international facilities to implement function
- 40-70% of systems development effort in many project is amenable to this approach

Software Engineering in the Large - Deploy to a Cloud Computer

- Remove friction between development and operations
  - Releasing early and often is a “Good Thing”
- Clouds can be both large and small
- From a practical perspective ensure infinite computing capacity
  - $14K = 8 core / 32 GB ram / 16 TB reliable data store.
  - PSP – massive CPU capability
  - Reduces operational costs by 100x
- Virtualizes the entire computing environment
- Remove operational capital expenditure requirements
- Enable business people
- Maximize IT project
  - Minimize IT carbon footprint

Software Engineering in the Large - Applying Resource

- Web Technology is ubiquitous.
  - Staff can be sourced and virtually exported across the globe
  - Training of staff in correct use of REST technology is critical
- URI enable complete separation of different system aspects.
  - Web development, software engineering, package integration, testing, system integration etc.
  - Mechanism to enabling outsourcing by concurrent development teams

Software Engineering in the Large - Applying Resource

- IT Business Community based development.
  - Staff can be sourced and virtually exported across the globe.
  - Access to multi-disciplinary skill sets
  - Greatly simplify Project Management challenge
  - Resource scale and flexibility
  - Speed
- Increasing use of Open Source style models in commercial development
- Significant constraint is “management bandwidth”
  - Must invert the “load” to the network

Where is the Speed Improvement

- Simplification of tasks using Web architecture
- Productivity optimization through task simplification
- URI partitioning enables parallel development
- URI and hyperlinks support integration and developer learning
- Minimise the “management bottleneck” removed
- High-end engineering focused upon high-end problems
- Manufacturing techniques on repetitive tasks
- Multi-organizational development shortening team build-up and disbursement.
- Concurrent testing
- Deployment time minimized

Where is the Speed Improvement

- Multiple time zones
- REST enabling end-user programming

Questions