Agile Deployment: The View from the Inside

John Birtley

buildMonkey

john@buildmonkey.com

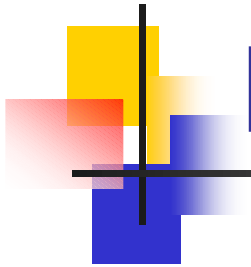
Hubert Matthews Oxyware

hubert@oxyware.com

XPDay, London, Nov 2004

Classic deployment scenario

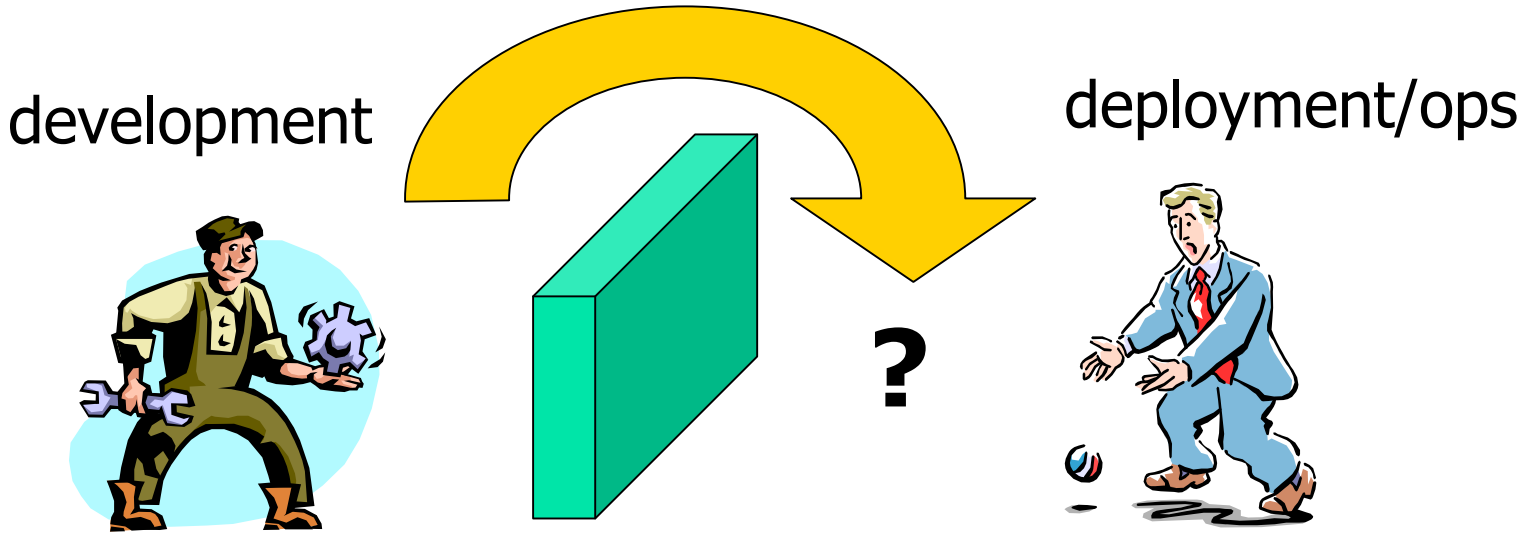
- Q: How many programmers does it take to change a lightbulb?
- A: “What’s the problem? The one on my desk works fine!”



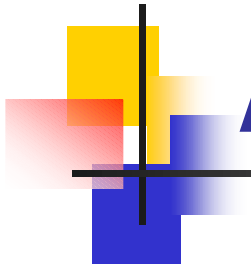
More serious answers/points

- There is one set of artifacts (code, etc)
- They must run in many environments
- But the environment is a refactorable component
- Q: How do you know these two environments are the same?
- Code is worth nothing until it’s deployed

“We’ve built it; now what?”



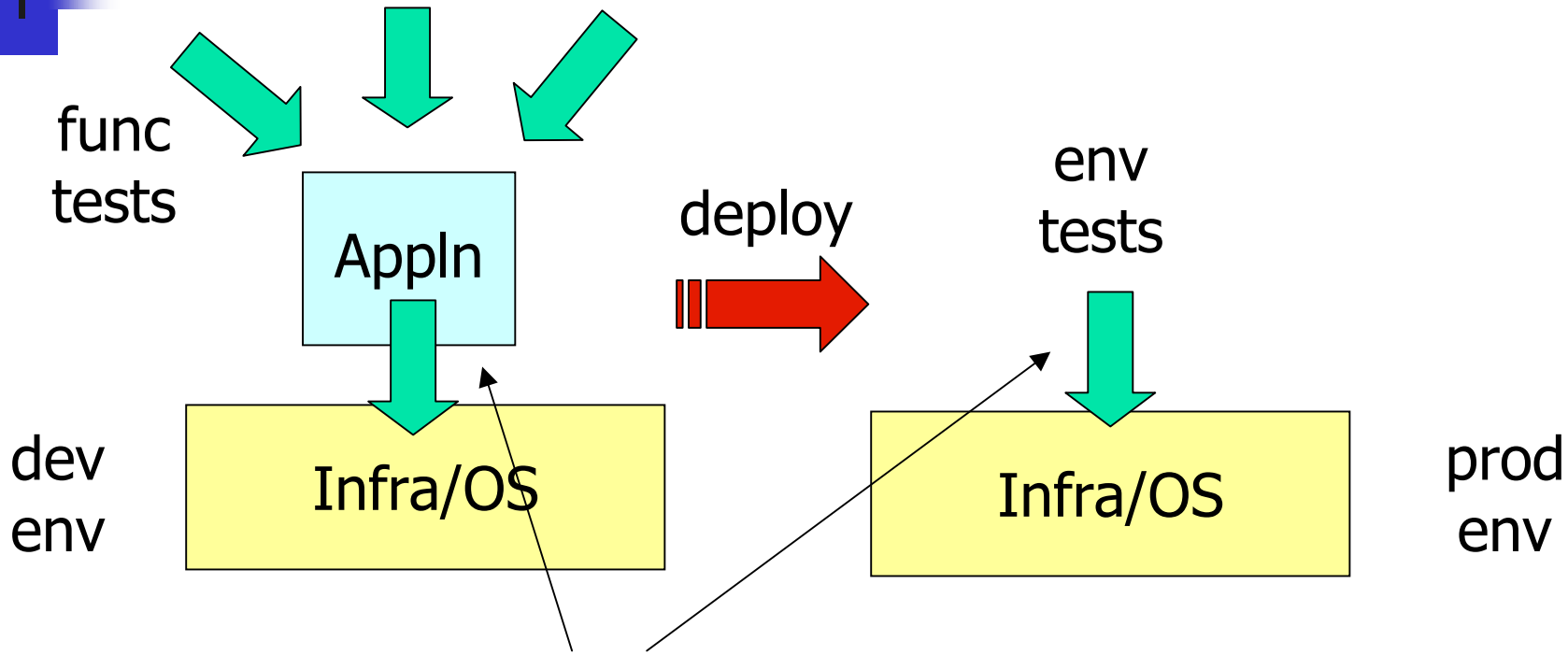
- How much emphasis is placed on building and how much on deployment in your projects?



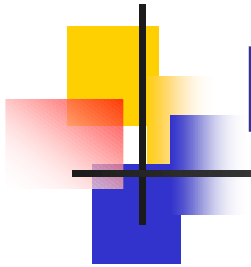
Agile deployment

- Deployment is on the critical path
  - At the end of the project from the developers' point of view
  - At the start of the project from the business's and the users' point of view
- Agile methods have improved development times and responsiveness
  - But what about deployment?

Environment testing

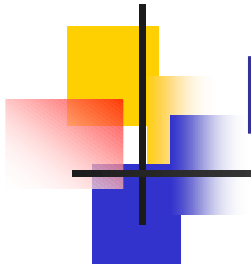


- External dependencies - you rely on them
- User perception of the quality of your application can suffer, even though the environment is out of your control



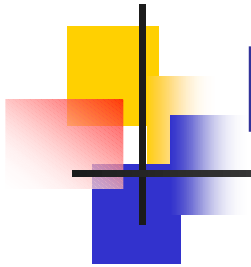
Deployment and contracts

- Dev environment is a stub for the production environment
  - Implements the whole contract for the whole i/f
  - Usually differs in non-functional aspects
- Test-driven deployment
  - Tests to show that dev stub implements the contract - primarily functional tests
  - Makes the contract explicit
  - Acceptance/user tests for the environment
- "Deploy by Contract"
  - Useful for handover and outsourcing/hosting



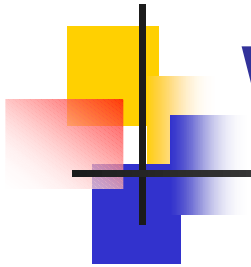
Exercise - web discussion forum

- Standard web application
- Discussion forums and news
  - c.f. ServerSide.com
- Tomcat, Linux, Oracle, LDAP
  - Oracle and LDAP on separate machines



Exercise 1 - dependencies

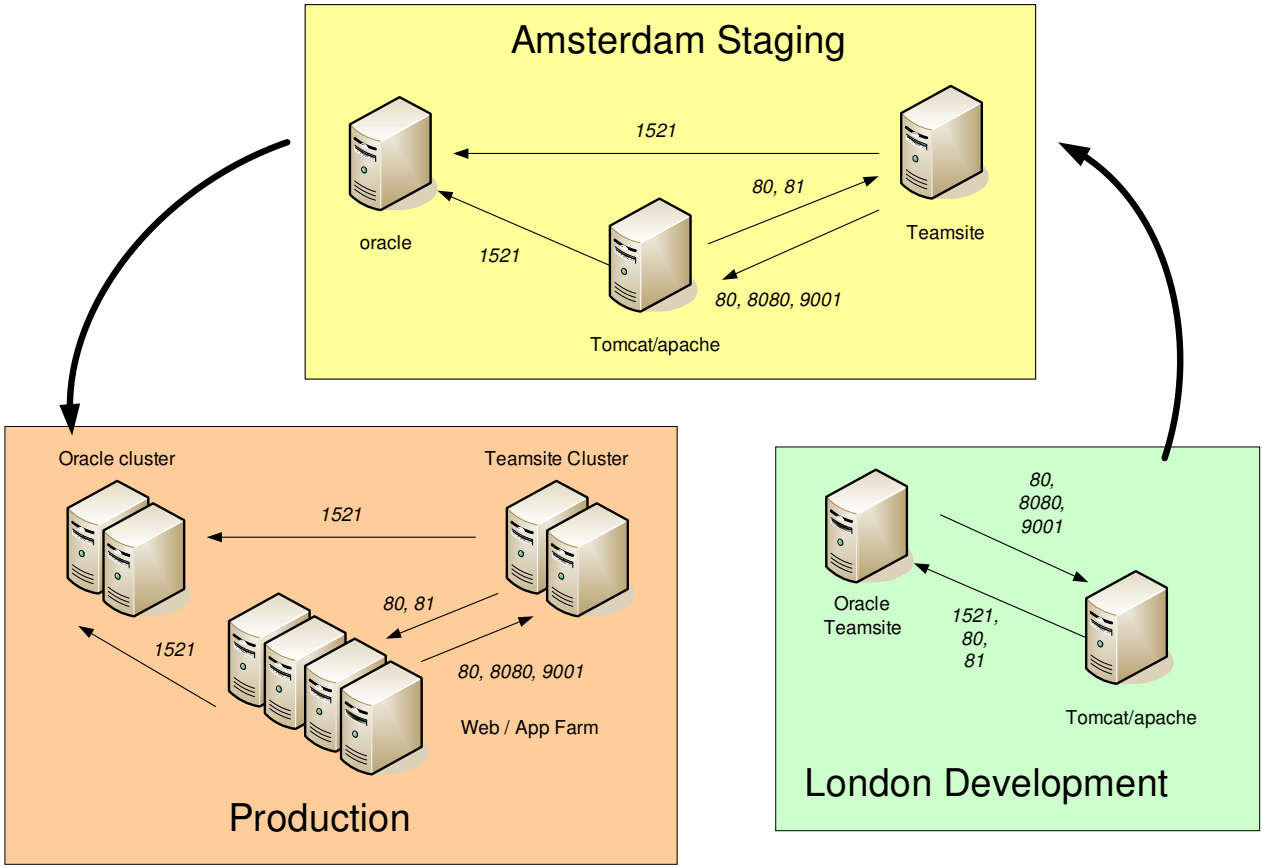
- In pairs, write down:
- What external dependencies does the application have?
  - TCP, DNS, files, directories, users/groups, environment variables, etc
  - Everything needed to deploy the app
  - Both “must haves” and “must nots”
- (10-15 minutes)

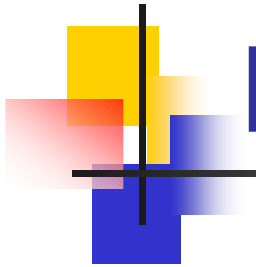


War Story Number 1

- Deployment, Dutch style
  - (take it away, John…)

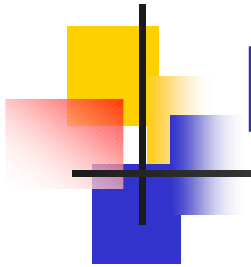
Are these the same?





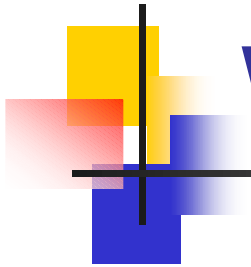
Exercise 2 - infrastructure

- In different pairs:
  - What needs to be done before you can install Tomcat, Oracle and LDAP
  - Is the platform “application ready”?
  - Network level
  - OS level
  - How could you test all this?
- (maximum 5 minutes)



Network and OS mock objects

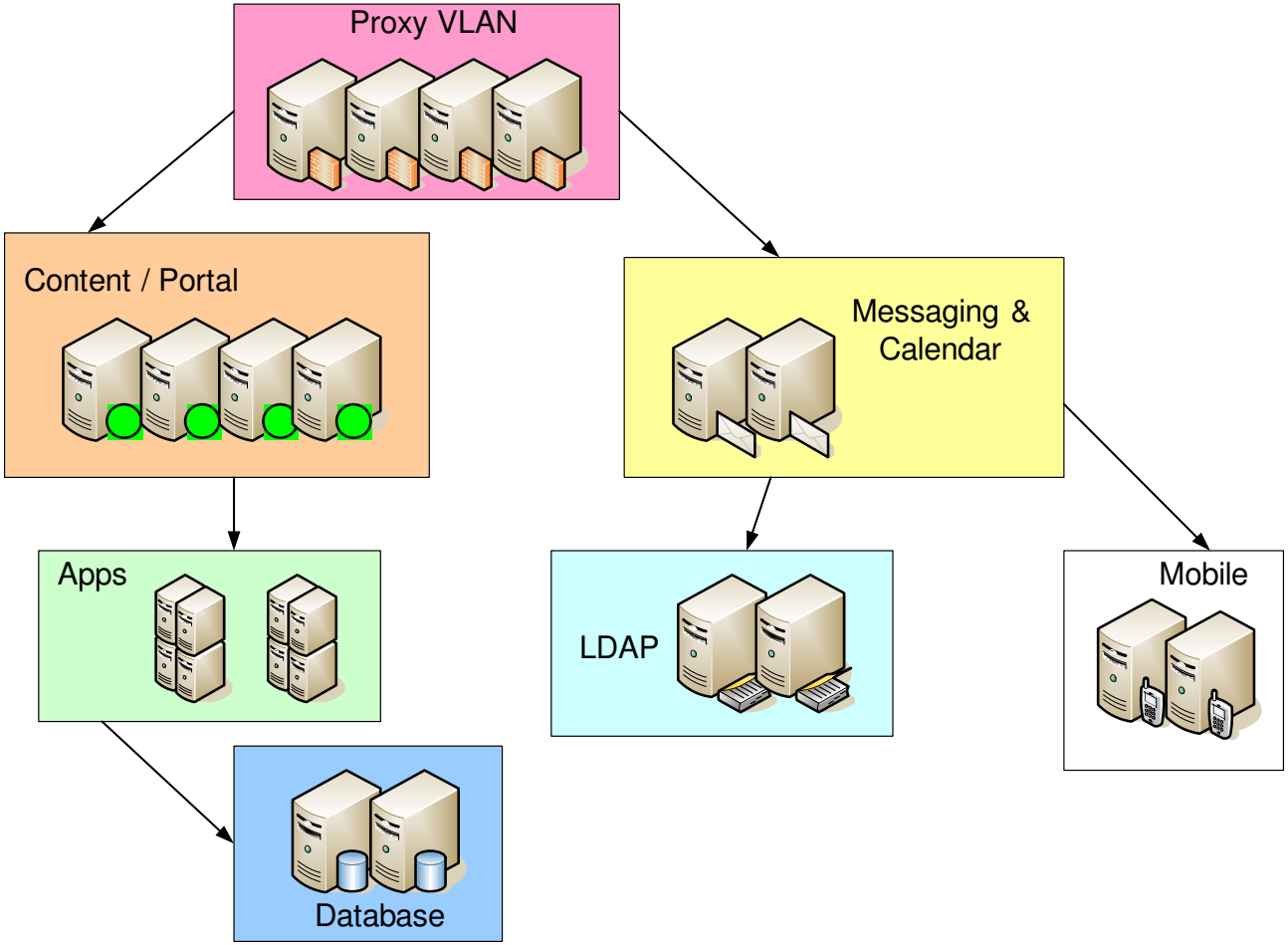
- In large systems the network and OS levels are often set up by different people or vendors. How do you test their work?
- H.F.H. - hassle-free-handover
- Tools to simulate network daemons and services for infrastructure signoff
- It’s not “Who’s Wrong”, it’s “What’s Wrong”

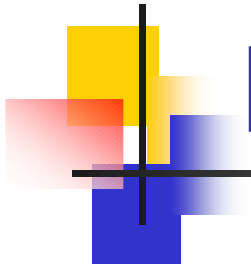


War Story Number 2

- Mobile ISP
- £3 million of infrastructure
- How it was tested
- How the development environment was set up
  - (Take it away again, John…)

£3 million infrastructure, 90 servers





Division of responsibility

- Having an automated Deployment Verification System (DVS) makes responsibilities clear
- Reduces fingerpointing
  - “the app is broken” v. “the network sucks”
- Major time saver on large contracts
  - Liability issues, signoff

System admin as refactoring

- System administration changes the deployment environment
- DVS acts as tests for sys admins
  - How to make changes without breaking something somewhere?
- Remember the pain of trying to refactor without tests?

Summary

- Deployment is the forgotten child of the software business
- It is a thief. It steals lots of time, especially late in a project, right on the critical path
- “Early and often” is the motto
- Tools such as DVS help to reduce stress levels
- Automate everything that doesn’t require human judgement