Deployment
Within a Traditional IT environment

Rob Allen

akrabat.com ~ @akrabat ~ October 2016

What?

Software deployment is all of the activities that make a software system available for use.

Wikipedia

Traditional IT environment

- Servers "owned" by client
    - Perceived control and compliance
    - IT departments control policy
- Variety of stack
    - Windows/IIS
    - IBMi
    - Solaris
    - Linux

What we're not going to cover

- Cloud deployments
- Provisioning of servers
- Continuous Integration
- Continuous Delivery
- Chat Ops

Fundamentals

Development processes

Fundamental development processes are still required

- Local development environments
- Source code control
- Database schema management
- Management of configuration data

Source code control

- Git
- Subversion
- Team Foundation Server

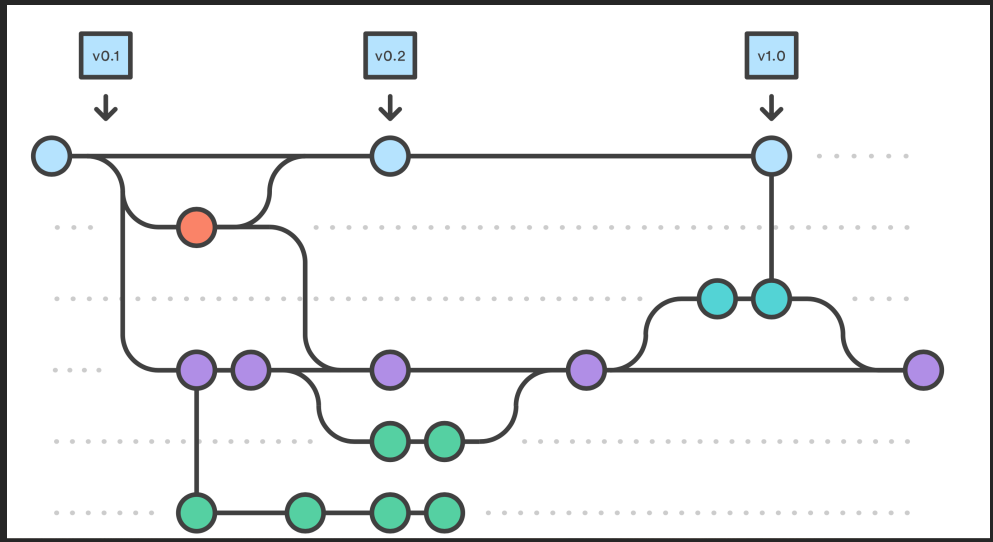
Code organisation

- Isolate work
- Branch!
    - One branch per logical work unit
    - Merge once approved
    - Always have a way to create a hot fix from what's on live

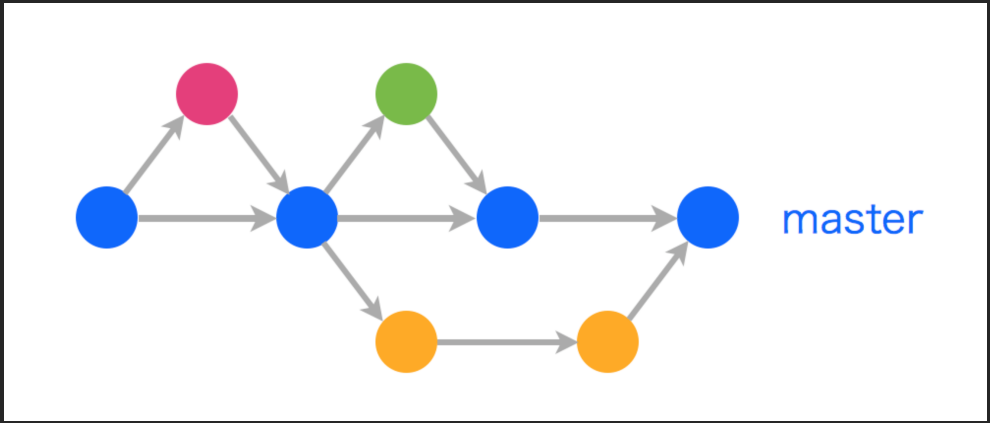
Branching methodology

- ~~Branch per team~~
- ~~Branch per sprint~~
- Branch per feature
    - Git flow
    - Github flow

Git flow



Github Flow



Continuous integration

- Automatically test every PR
- Types of tests:
    - Coding style
    - Unit tests
    - Functional tests
    - Acceptance tests

Database schema management

aka: Migrations

- Schema changes stored in individual files
- *delta* files with UP and DOWN functionality
- Version controlled
- Destructive changes possible… (be careful!)

Migration tools

It really *doesn't matter* which tool you use

- Simple SQL files
- Your framework's system
- Liquibase
- DBDeploy
- Flyway
- DBMaestro
- Redgate SQL Compare

Management of configuration data

Each environment needs different configuration:

- Database credentials
- Debug / logging levels
- Links to other servers

Context awareness

- Automatic
  - Automatic detection based on URL
  - Environment variables (12factor.net/config)
- Local configuration file

Deployment

Manual deployment

Simply copy the files to the right place

Problems:

- Slow
- Inconsistent between each deploy
- Failures in production
- No records. Who deployed what, when (or where)?

Simple deployment systems

- FTP / SFTP
- rsync
- Remote Desktop

It's more than just copying code

Other things to think about

- File permissions
- Preserve user uploaded files
- Steps after upload
    - Stale cache?
    - Cache priming?
- Keeping records

People aren't machines!

Automation

- People: good at creative judgement
- Machines: Good at doing the same thing over and over

Advantages of automation

- Deferred deployments
- Self-service deployments
- Automatically triggered deployments
- Smoke tests (automated tests after deploying)
- Automatic rollbacks (restore last successful deployment if this one fails)

Scripts

- Provide consistency
- Provide a platform for additional abilities
    - continuous deployment
    - smoke tests
- No need to wait for other people
- Faster and less hassle.

What needs to be done?

Write a deployment checklist. Then script it.

- Copy files & folders
- database schema & content updates
- configuration updates
- r/w permissions on files & folders
- SSL certs?
- Clear caches

Tooling

- Simple script
- Make-like tool
- Specialist tool

Simple script

- Bash
- Powershell
- Usually runs on the server

Make-like tool

- Abstracted script
- Easier to make cross platform
- Make/Phing/Ant/Rake
- More likely to push to server

Specialist tools

- Fabric/Capistrano
- Ansible/Chef/Puppet/Salt
- Deploybot
- More likely to be automated

Challenges

- Technical
- Political

Technical

Infrastructure challenges

- VPN
- Firewall
- Operating systems

Political

- What is acceptable?
- What are you allowed to do?
- Does IT change things for you?
- Does IT change things unexpectedly?
- Change request documentation?

Case studies

Case study 1: Internal

- Self-hosted by client
- Within company network
- Windows/IIS/SQL Server

Case study 1: Internal

- Multi-script solution
    - 1. Tag code & build tarball
    - 2. Ship tarball to UAT server via VPN
    - 3. IT copy tarball from UAT to Live
    - 4. Script to untar & run deploy steps

Case study 2: Minicomputer

- Self-hosted by client
- Some access to Internet
- IBM i-series/DB2

Case study 2: Minicomputer

- CI tooling
- Checklist
  - Manual tag creation for release
  - Automatic script to deploy

To sum up

Thank you!

Feedback: https://joind.in/talk/38e78

Rob Allen ~ akrabat.com ~ @akrabat