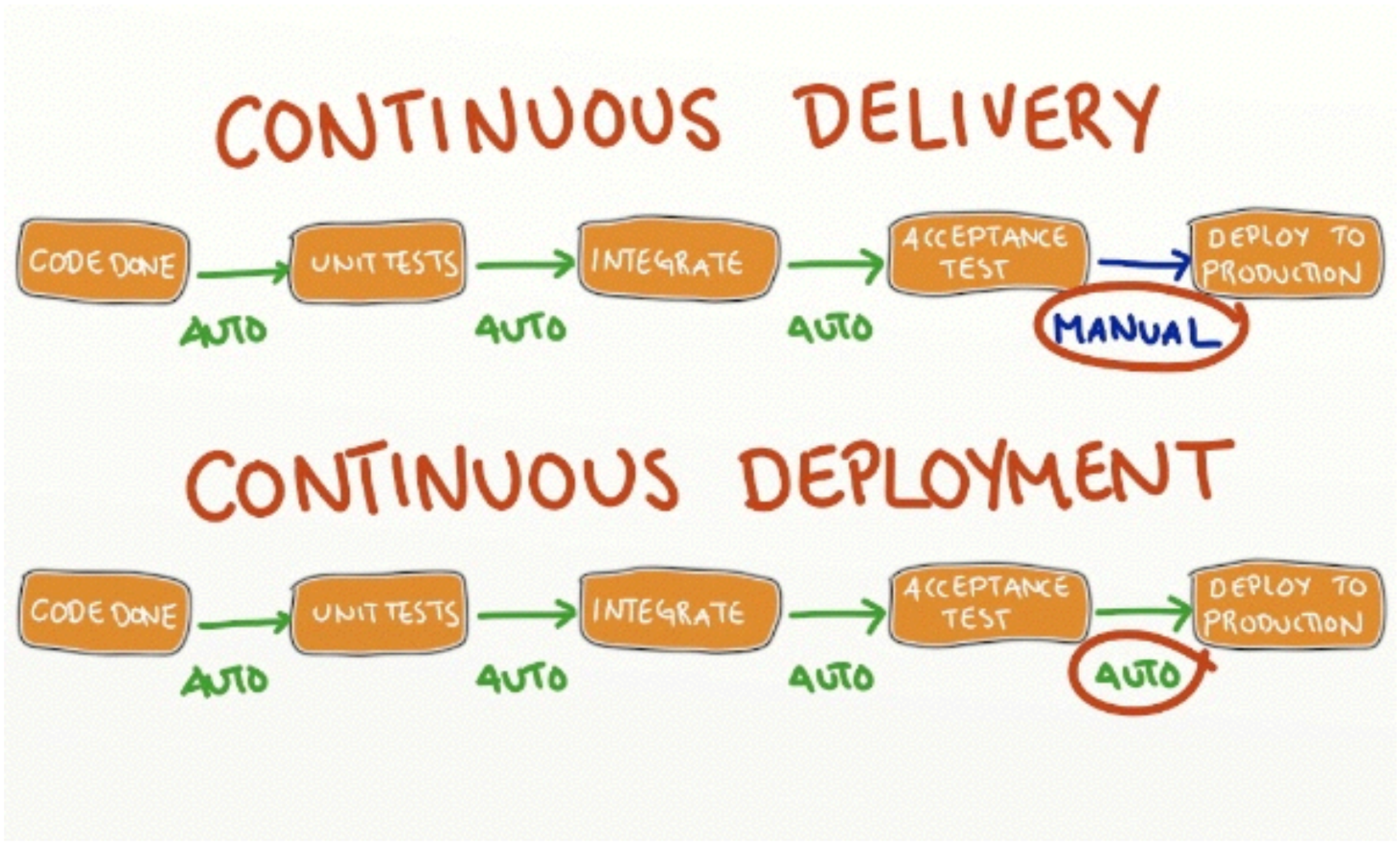
Continuous Integration

What is your Continuous Integration ?

What is your CI system ?

No Build Server	Nightly Build	Automated Build with Basic Automated Test	Use Metrics
More About Testing	Automated Acceptance Testing	Continuous Deployment	Continuous Delivery


CONTINUOUS DELIVERY
CODE DONE
UNIT TESTS
INTEGRATE
ACCEPTANCE TEST
DEPLOY TO PRODUCTION
AUTO
AUTO
AUTO
MANUAL
CONTINUOUS DEPLOYMENT
CODE DONE
UNIT TESTS
INTEGRATE
ACCEPTANCE TEST
DEPLOY TO PRODUCTION
AUTO
AUTO
AUTO
AUTO

Stressful Releases


LET'S PARTY!

What is CI ?

- Continuous Integration !!!
- Automated Everything
- Build as often as possible
  - Build early and often
  - e.g. On every commit/nightly build

What is CI ?

- Not just technical
- **Organization** and **culture**

What is CI ?

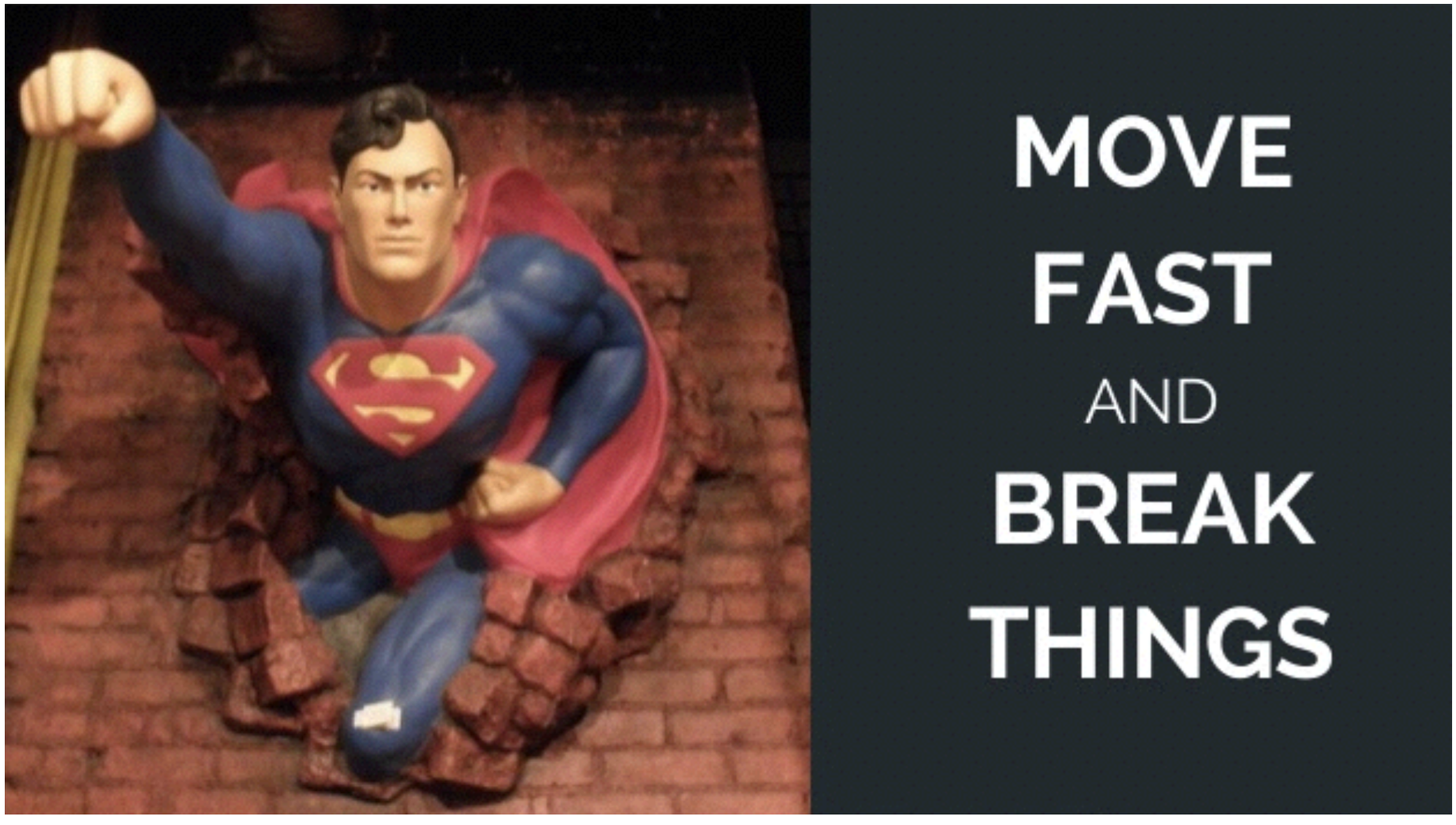
- Software development **practice**
- **What people do**, not about what tools
- Integrate frequently
- Fast feedback

Benefit of CI

- Fewer error/Reduce risks
- Fast feedback
- Much less manual testing
- Regression test without additional effort
- Generate deployable software
- Great confidence in the product

อะไรที่ไม่ใช่ CI

- Nightly build
- Developer branch
- Scheduling integration point
- Building via IDE


MOVE
FAST
AND
BREAK
THINGS

CI Practices

1. Maintain a single source repository
2. Automate the build
3. Make your build self-test
4. Everyone commit to the mainline everyday
5. Every commit should build the mainline on integration machine
6. Keep the build fact
7. Test in a clone of the production environment
8. Make it easy for everyone to get the latest executable
9. Everyone can see what is happen
10. Automate deployment

Single Source Repository

- Single point of truth
- Everyone’s code in the same place
- NOT a branch per developer
- Shared ownership

Automate Build

- IDE is not automating !!
- Use build tool
- Compile, package and test

Self-testing Build

- Direct from source to running build
  - No manual copy
  - No click
  - No edit config file

Self-testing Build

- Test with
  - Unit tests
  - Functional tests
  - Performance tests

Self-testing Build

- Responsible persons should be notified when anything fails
- Test web in more browser

Commit More Often

- At lease one per hour
- Need small unit of work
- To commit cleanly you need to update first

Every Commit Build

- It’s all about fast feedback
- Small changes
- Less to merge or fix

Publish Latest Distributable

- Make it easy to get final product
- Should only build once
- Configuration is separate

Test in Production Clone

- Detect multi-thread, cluster, load balance issue
- Test system architecture
- Real database

Keep Build Fast

- It’s all about feedback
- If thing break you find out about it, fresh in your mind
- Keep up with frequent commit

Everyone See What’s Happening

- Reduce time to fix
- No excuse to commit on broken build
- It’s not about blame, it about feedback

Automate Deployment

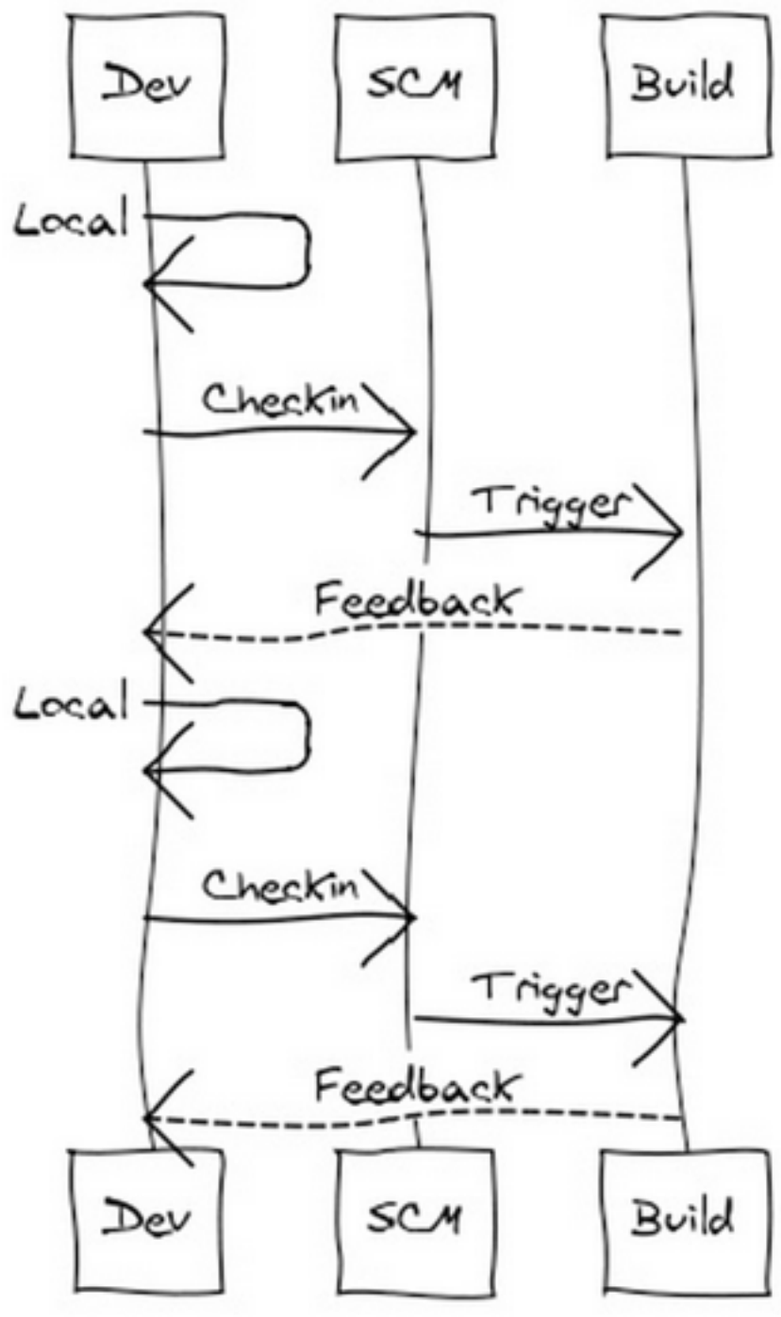
- Reduce human error
- Verify can run on somewhere other than “My Machine”
- Test not only on code, but deployment process too
- Don’t tie up System Administrators with boring stuff
- Don’t tie up Developers waiting for feedback

Make CI Work

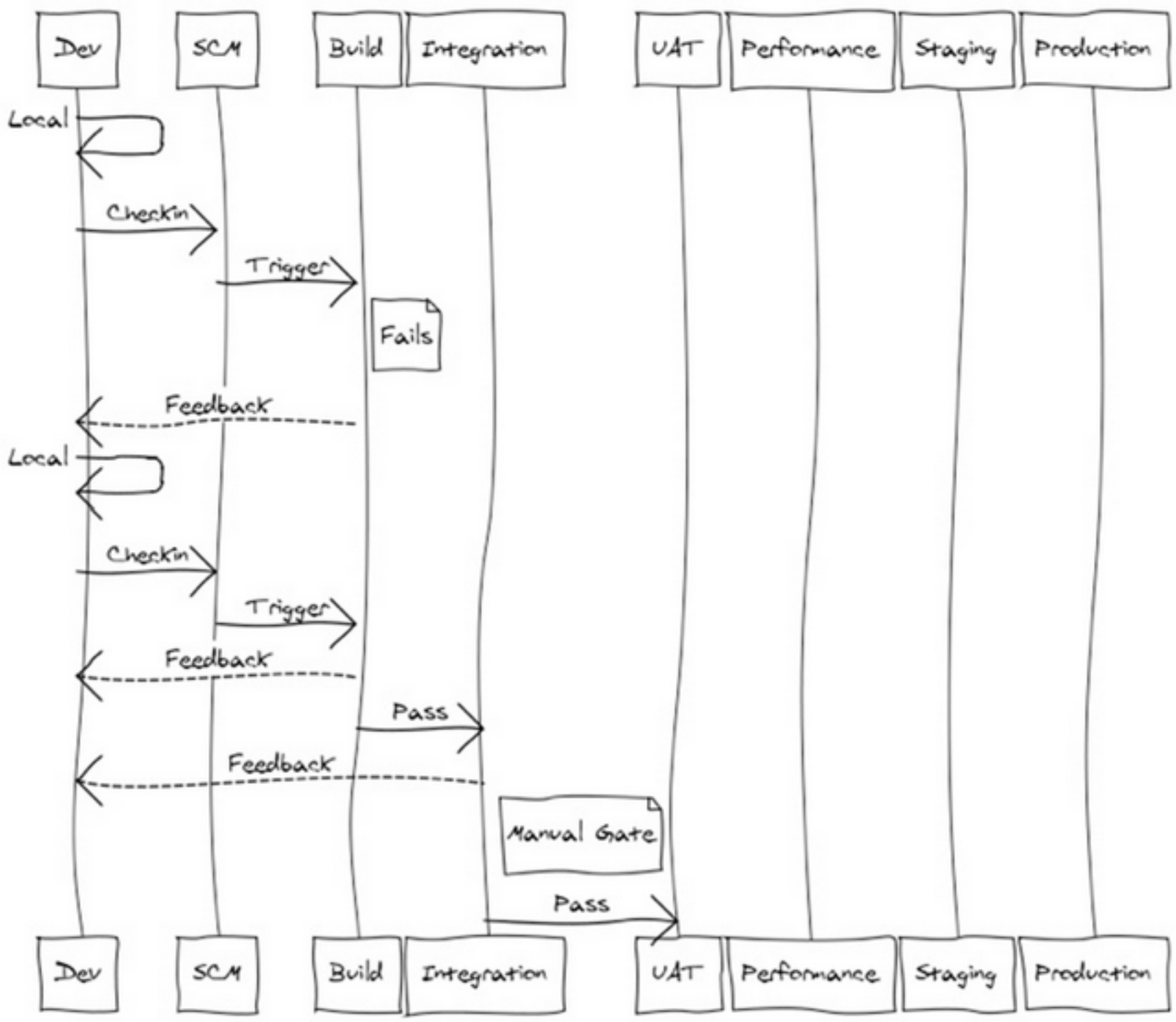
- Can not done in isolation
- Pick the right tool for the right job
- It’s not silver bullet


Build Pipeline

Traditional CI Flow



Build Pipeline Flow

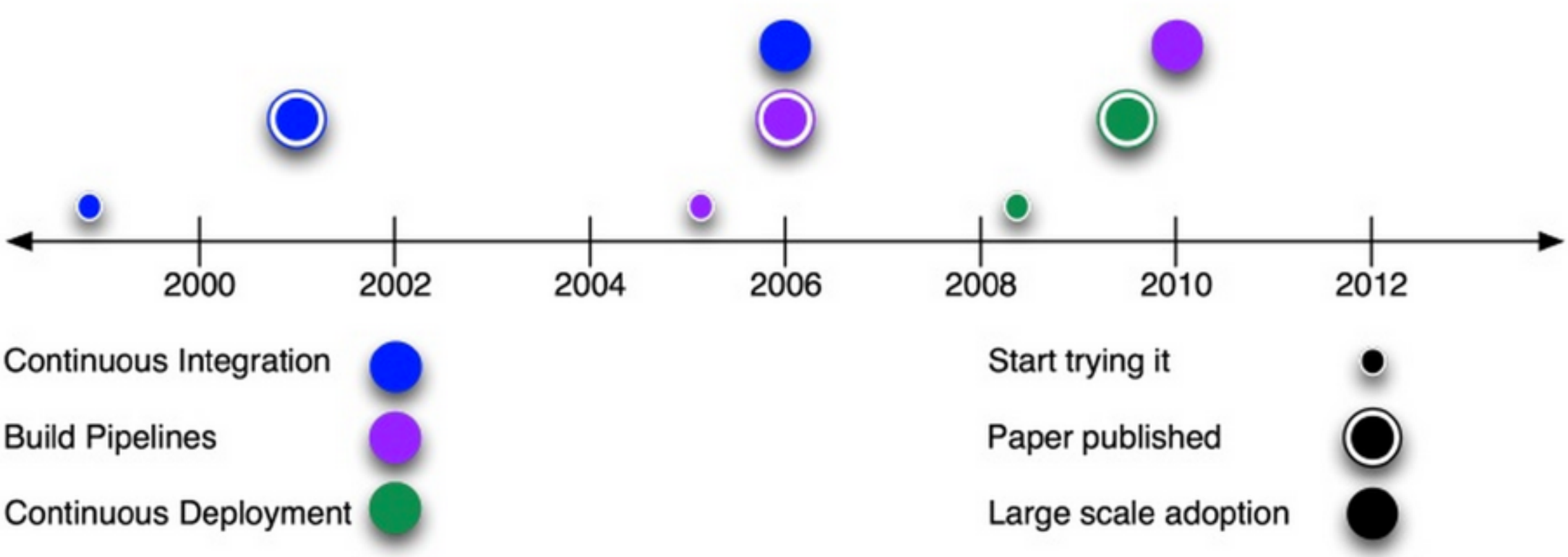


One Click Deploys

- Require the Build pipelines
- Reduce deployment time and risk
- Make go live a non-event

Continuous Deployment

History


2000
2002
2004
2006
2008
2010
2012
Continuous Integration
Build Pipelines
Continuous Deployment
Start trying it
Paper published
Large scale adoption

Summary

- CI and Build pipeline improve
  - Quality
  - Time to market
  - Confidence
- Continuous Deployment/Delivery is the next step
- Require Developers and System Administrator to work closely together

3 persons/group

Your

Build Pipeline

?

Build/Deployment Pipeline

1. Clean

2. Checkout

3. Download Library

4. Compile

5. Package

6. Setup DB

7. Run Test

8. Deploy

How to Self-Testing ?

Let’s Go !!!