Improving your Drupal Development workflow with Continuous Integration

Peter Drake
Sahana Murthy

Acquia®

DREAM IT.
DRUPAL IT.

Meet Us!!!

Peter Drake

- Cloud Software Engineer @Acquia
- Drupal Developer & sometimes core contributor

Sahana Murthy

- Developer Evangelist @Acquia
- Open Source Proponent
- @sahanatweets

Agenda

→ Standard Development Workflow

- What does it look like?
- What can go wrong and why?

→ Continuous Integration workflow

- What does it look like?
- Why is it better?
- Demo!

What do we mean by a "Standard Workflow"?

Acquia®

DREAM IT. **DRUPAL IT.**

Stage 1 – Hacking Code on Server

Pros	Cons
• Quickest and cheapest way to get started • New features- visible immediately • Bugs- easy to replicate • Feedback - instant • No time-sucking "release engineering" overhead	• The slightest mistake causes a WSOD • Encourages panic-driven engineering • Impossible to demo without a release • Impossible to do with a team • Basically, this never works beyond "install a new module"

Stage 2 – Develop locally, push to production

Pros	Cons
• Safer, and works w teams • Everyone uses a local copy of the prod database • Obvious bugs never make it to production • Normal software dev process possible • Version control to share/log changes	• Releases are slow + scary • Announce a "feature freeze" • Merge everyone's changes to release engineer's machine • Test, find bugs, point fingers, fix bugs • Copy code to prod servers • Manually upgrade prod • Find new bugs, scramble to fix or revert

Is the Standard Workflow ideal for today's Agile World?

Acquia®

DREAM IT. **DRUPAL IT.**

What could go wrong?

Acquia®

DREAM IT. **DRUPAL IT.**

Common Pitfalls

→ Deploying broken Code
- Developers overwriting each others work
- Partial changes
- In-development changes
- Untested changes

→ Deploying broken Config
- Missing libraries or dependencies
- Wrong Drupal config settings

→ Deploying a broken Database Schema
- Failure to write schema update hooks
- Failure to run update.php
- Manual schema changes

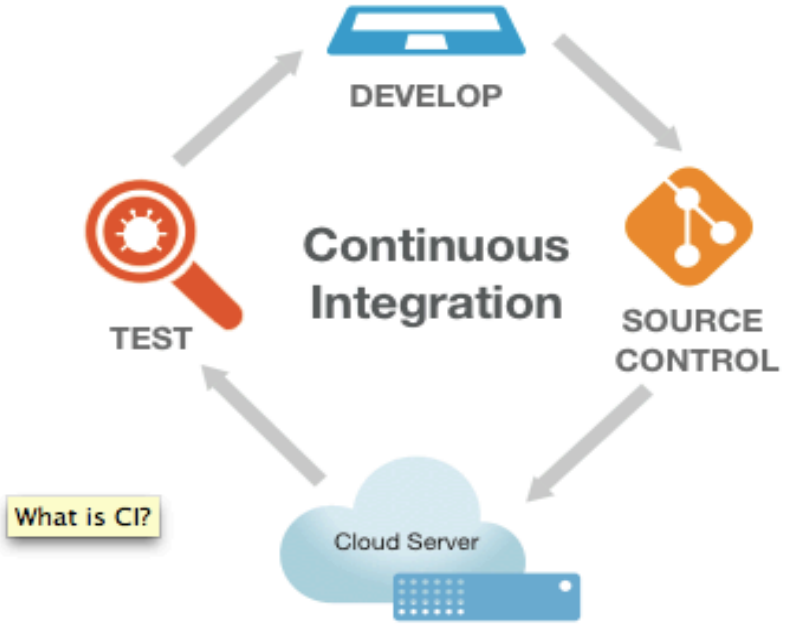
So…What is the RIGHT way???

Stage 3 – Continuous Integration

Pros	Pros!
• Source code repository • Step zero for good software engineering • It mostly doesn't matter which one you use • Version everything Code, tests, documentation, libraries, dependencies • Keep it simple • Main branch and tags Temporary, private feature branches Release branches only for major changes	• Integrate code constantly • Break big features into small steps • Automate the upgrade process • Implement tests as you go • Commit changes often • Update from main branch daily • Reduce integration conflicts, surface problems sooner

What is Continuous Integration?

- Continuous integration (CI) is software engineering's best practice of merging all developer working copies with a shared mainline several times a day.



Acquia

CI - Principles

- Use Version Control
- Integrate Code Constantly
- Test on Clone of Production
- Automate Testing
- Automate Deployment

Use Version Control

Principles

- Step “zero” for good software engineering
- Version everything
  - Code
  - Config
  - Database Schema
  - Automate Tests
  - Documentation
  - Depenencies

Resources

- A Successful Git Branching Model
- Pro Git Book

Recommendations

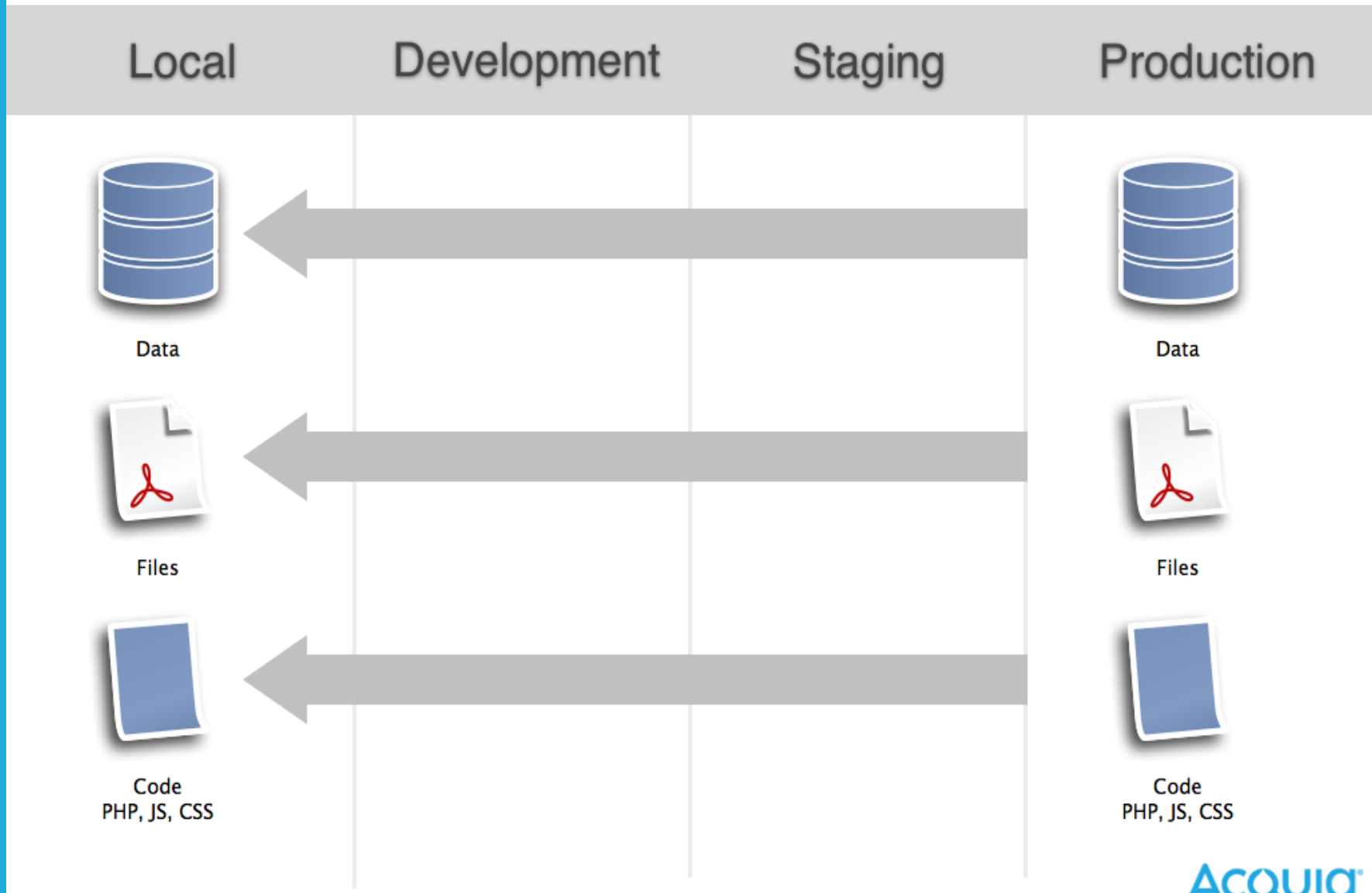
- Git is a distributed revision control and source code management (sCM) system with an emphasis on speed.
- Common Command:
  - pull: grab changes from repository.
  - add: Index the changes.
  - commit: track the changes.
  - push: send changes to repository.
  - status: list which files are staged, unstaged, or tracked.

What problems does Version Control solve for us?

Acquia®

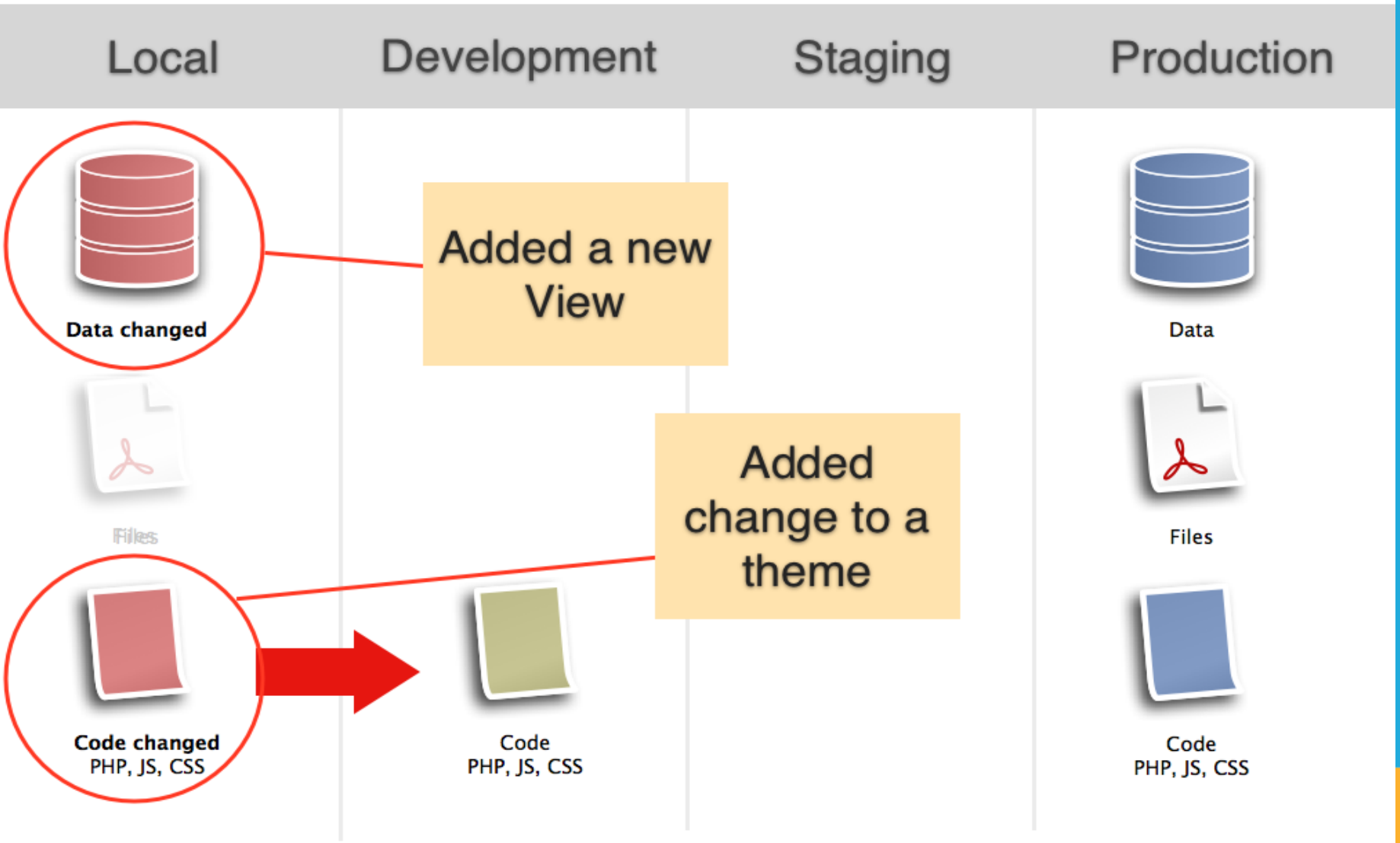
DREAM IT. **DRUPAL IT.**

The deployment problem

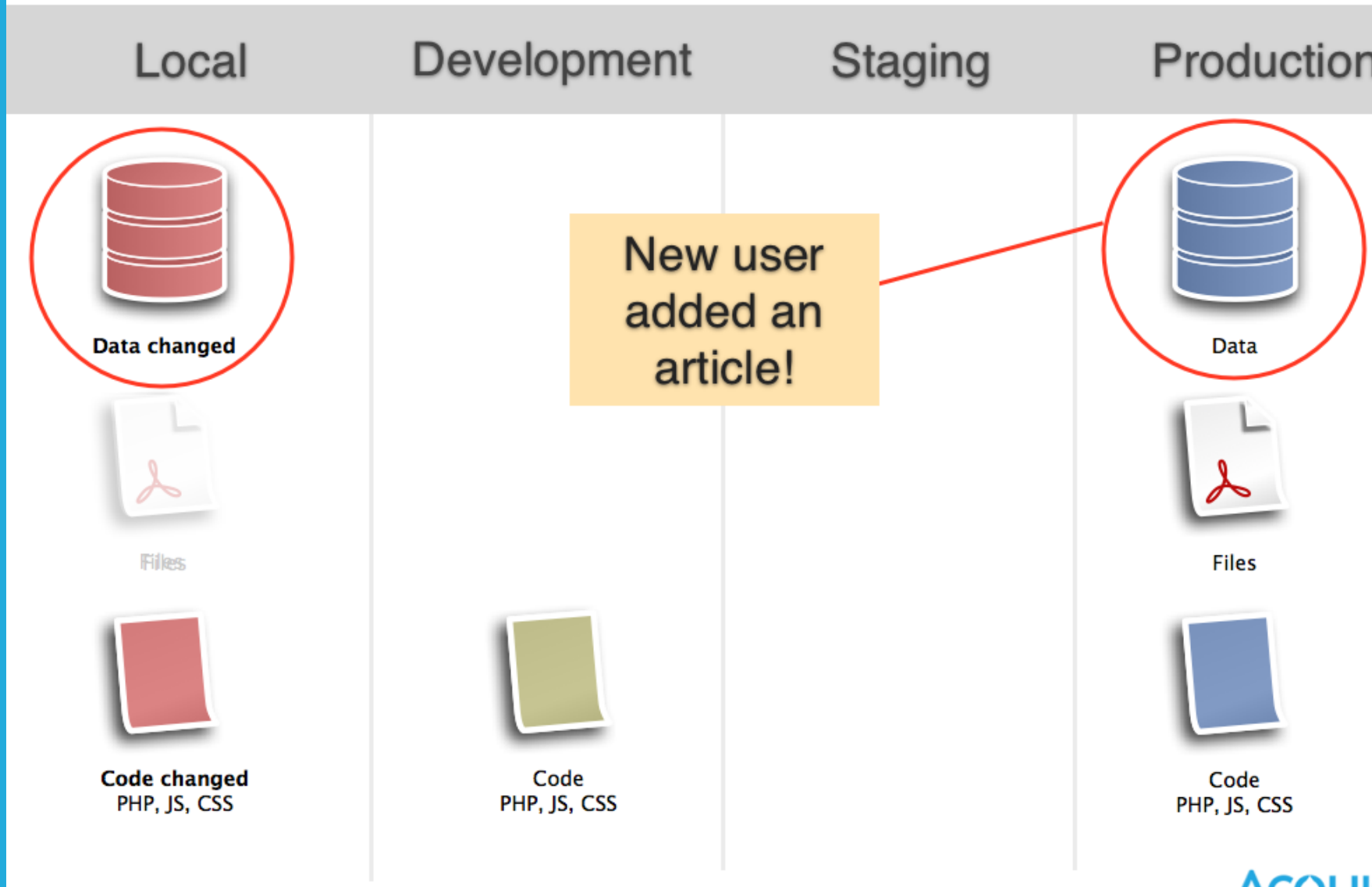


Acquia

With Version Control

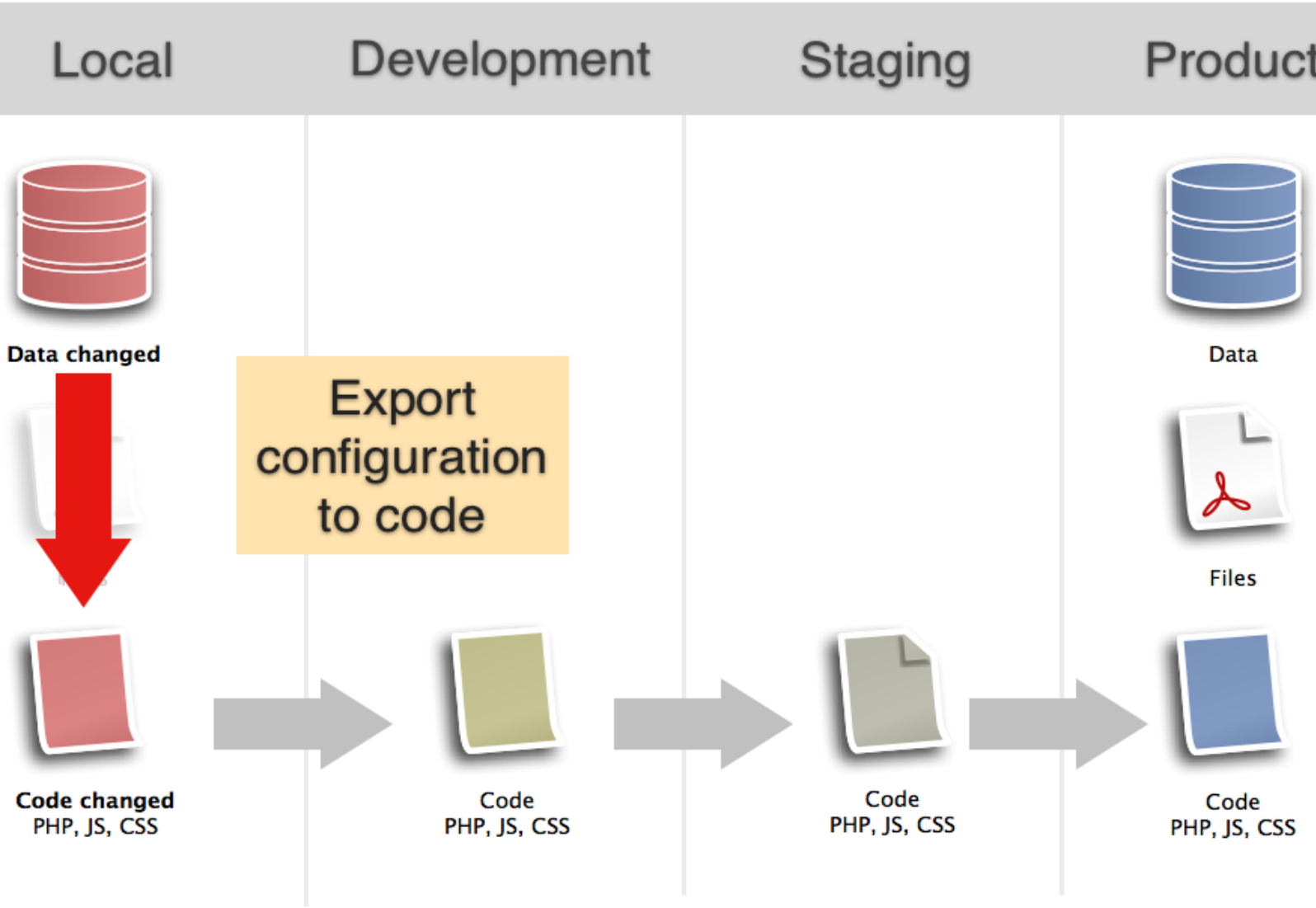


Source Code Conflicts

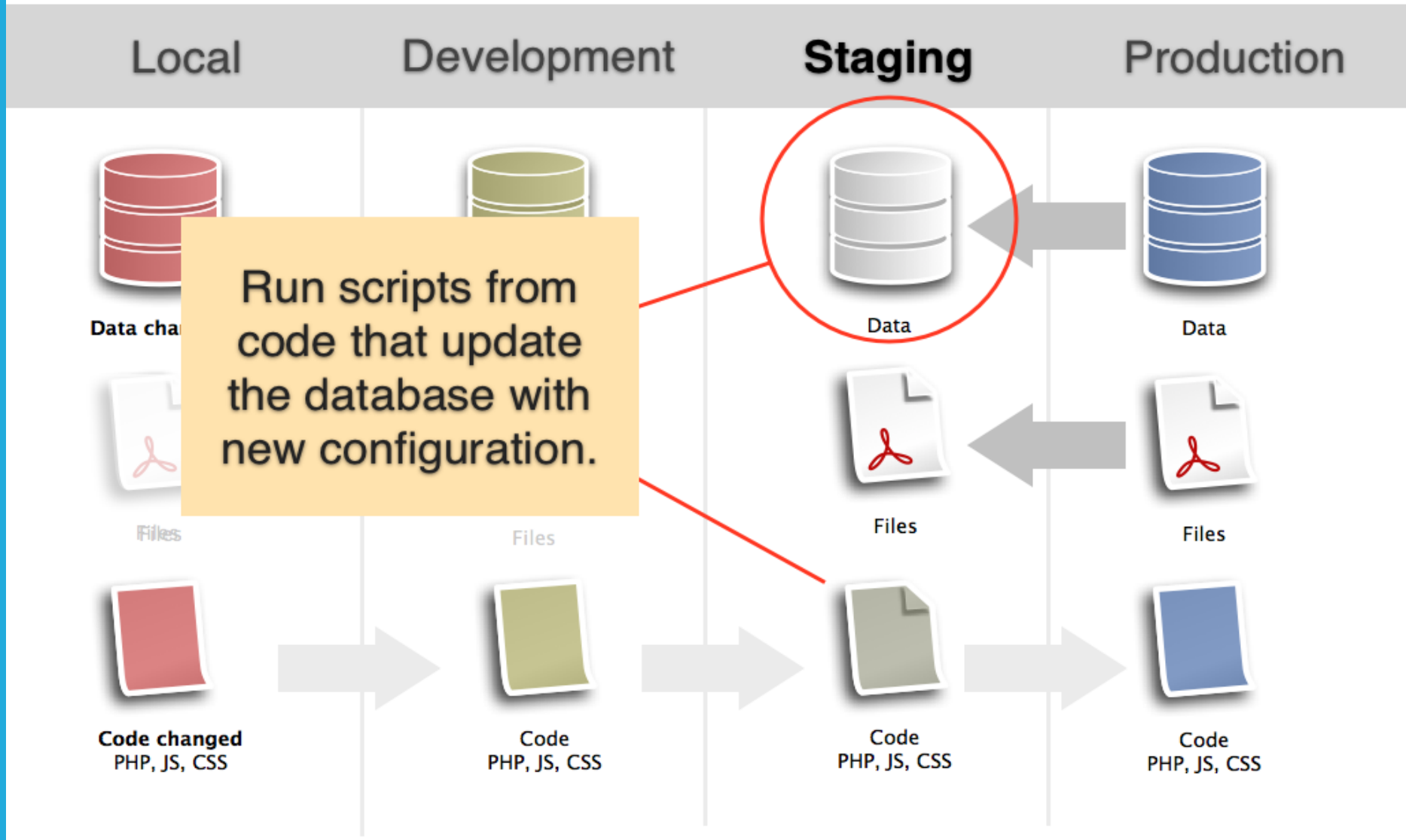


Acquia

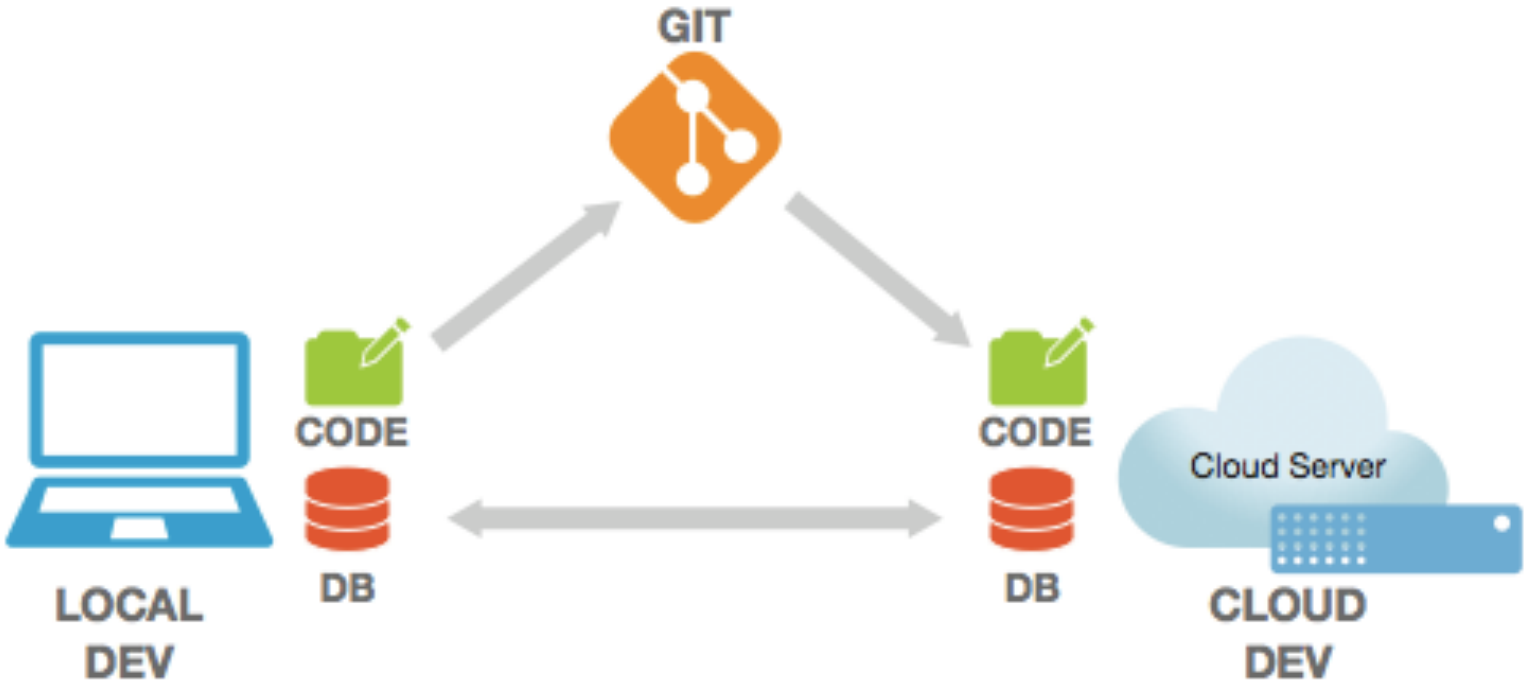
Configuration to code



Stage your testing



Version Control - Overview



Acquia

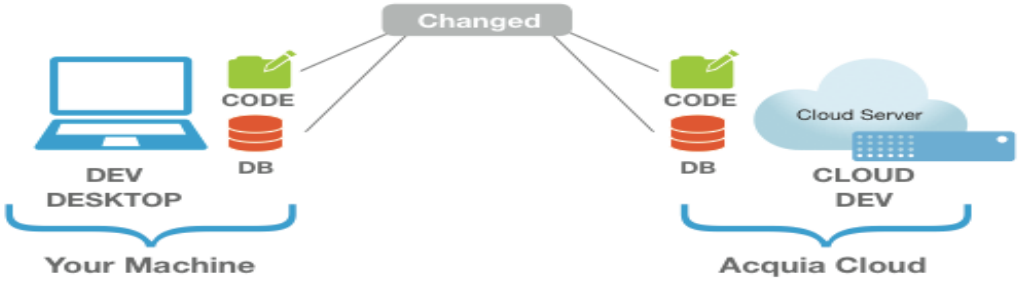
Integrate Code Constantly

Principles

- Break big features into small steps
- Commit changes often
- Automate the upgrade process
- Implement tests as you go
- Update from master branch daily

Results

- Identify conflicts early on
- Build quality is ensured
- Test coverage is consistent
- Development mistakes don't impact others' velocity



Acquia

Test on a Clone of Production

"Drupal depends heavily on its environment"

Principles

- Identical OS, Apache / MySQL / Varnish, PHP extensions, PEAR libraries, assorted packages
- Use a current (scrubbed) version of production database
- Unify and automate build of all environments

Results

- No more "It worked on my machine!" or "It worked on the testing database!"
- Manual server tweaks will not propagate to production

Automate Testing

"If it ain't broke, test it anyway."
"If it isn't tested, it doesn't work."

Principles

- Tests: build, unit, browser-based
- Run tests on every commit
- Don't merge changes until tests pass
- Announce when master breaks
- Deploy frequently to QA environment

Results

- Progress visibility for stakeholders
- Bugs discovered before deployment
- Consistent quality (QA)

Automate Deployment

"Drag-and-Drop is as automated as automated gets!!!"

Principles

- Deployment should be a push-button-and-relax operation
- Tag each release for future reference or rollback
- Snapshot databases for rollback
- Change database schema via code
- Update config via code
- No change is done until the upgrade is automated

Results

- Deployments are worry-free
- Eliminates human error
- Ensures stakeholder expectations are met
- Enables continuous deployment

In a Nutshell!!!

Three Phase Development

Development	Staging/QA	Production/Live
Developers collaborating and testing.	Test updates from development.	Live, stable version of site.
Code should be version controlled.	Should be exact same environment as production.	Tested.
Multiple dev environments (local and online).	Sync content and files from production.	This is where users login and add content (usually).

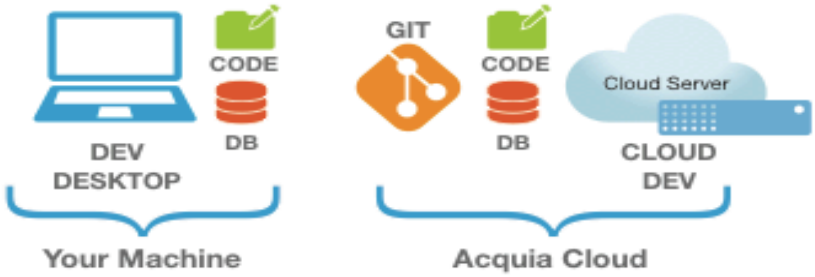
Lets watch CI in LIVE ACTION!

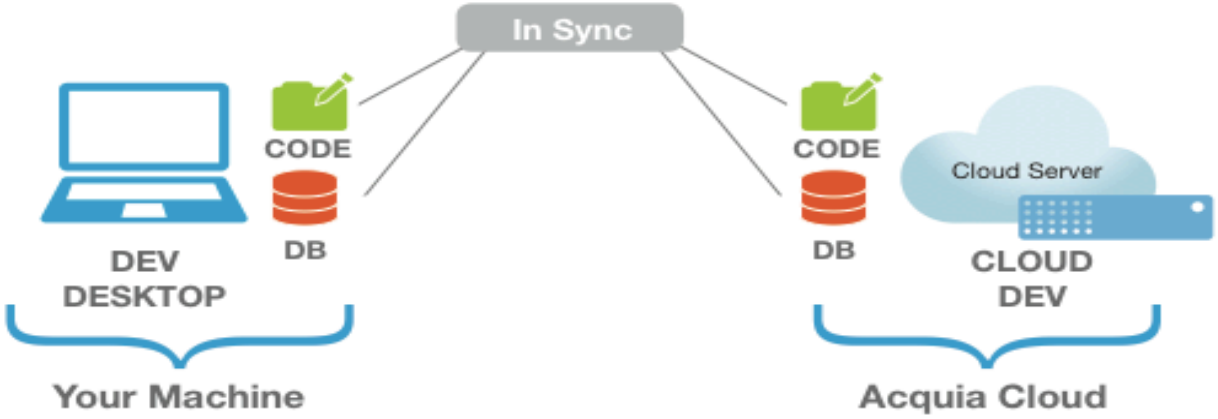
Acquia®

DREAM IT. **DRUPAL IT.**

How-To with Acquia Cloud:

- Start with your Drupal development on your local machine using Acquia Dev Desktop (Drupal tuned xAMP stack for developing locally).
- Commit to your source code management system.
- Build on Acquia Cloud Dev from SCM or push your code from Dev Desktop to Cloud Dev.
- Iterate multiple times, test, re-develop, and rebuild.





Acquia

Whats' next with CI and Drupal???

Acquia®

DREAM IT. **DRUPAL IT.**

Wait For It…

Drupal 8
WOOT WOOT!

Acquia®

DREAM IT. **DRUPAL IT.**