DevOps on AWS:

Deep Dive on Continuous Delivery and the AWS Developer Tools

Chris Munns, Business Development Manager - DevOps

August 11, 2016

https://secure.flickr.com/photos/mgifford/4525333972

Software moves faster today

Software creation and distribution is easier and faster than ever:

- Startups can now take on giants with little to no funding ahead of time
- Getting your software into the hands of millions is a download away
- Your ability to move fast is paramount to your ability to fight off disruption

The software delivery model has drastically changed

Old software delivery model

New software delivery model

What tools do you need to move fast?

Releasing software in this new software-driven world requires tools:

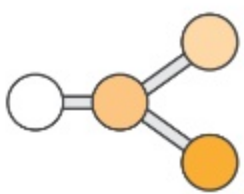
- To manage the flow of your software development release process
- To properly test and inspect your code for defects and potential issues
- To deploy your applications

First, we need to understand a little bit about software release processes

https://www.flickr.com/photos/jurvetson/5201796697/

Release processes have four major phases

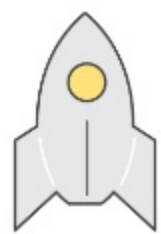
- Check in source code, such as .java files
- Peer review new code



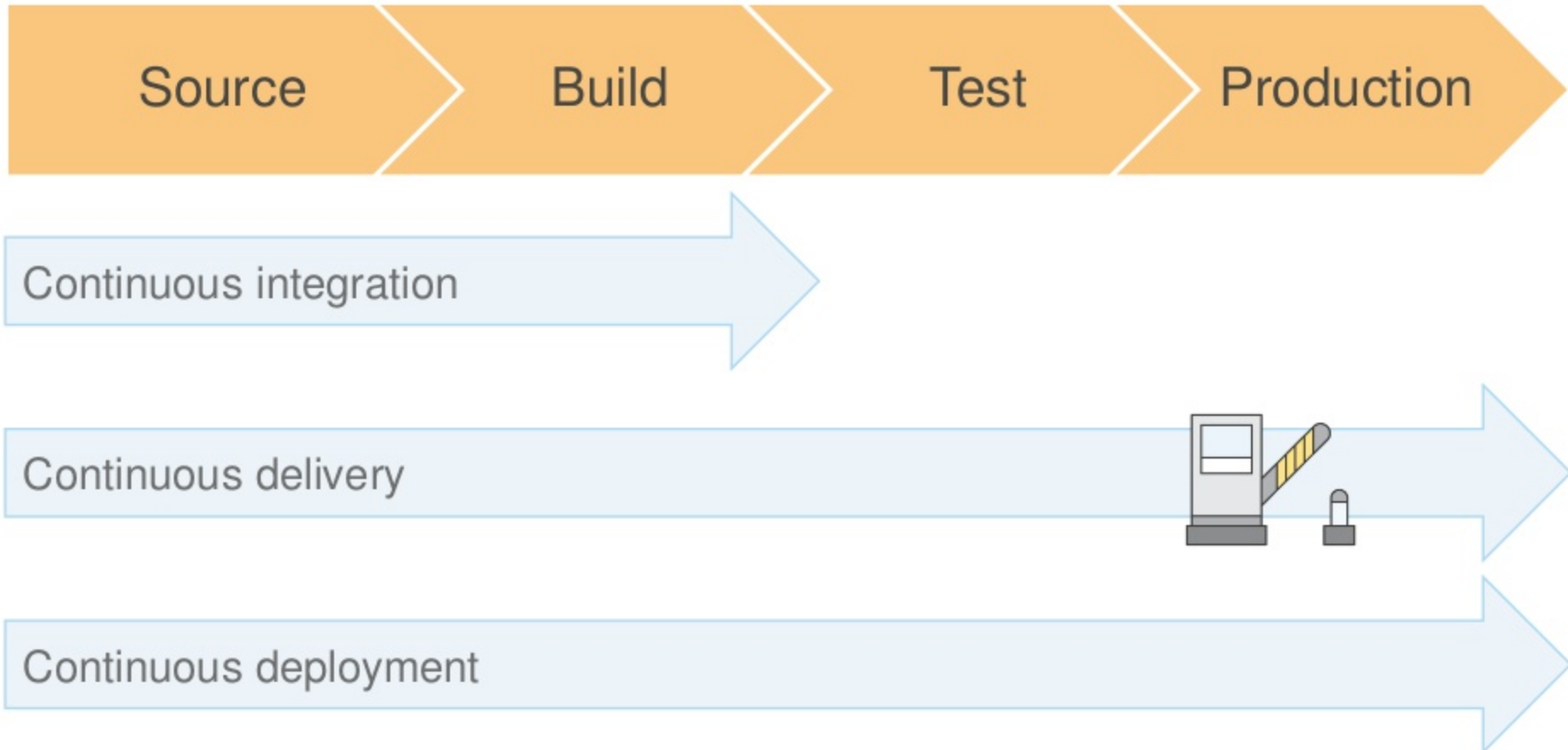
- Compile code
- Unit tests
- Style checkers
- Code metrics
- Create container images

- Integration tests with other systems
- Load testing
- UI tests
- Penetration testing

- Deployment to production environments



Release processes levels



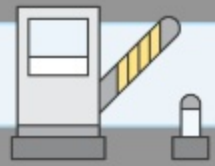
Release Processes levels

Source Build Test Production

Our focus today

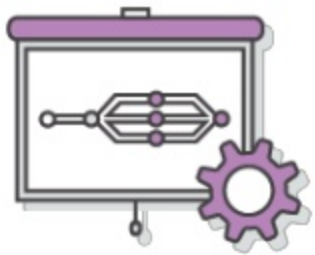
Continuous integration

Continuous delivery



Continuous deployment

Continuous delivery benefits



Automate the software release process

Improve developer productivity

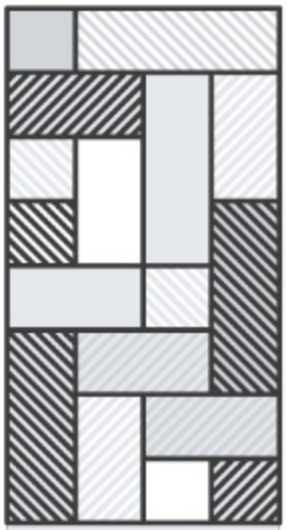
Find and address bugs quickly

Deliver updates faster


A look back at development at Amazon
https://secure.flickr.com/photos/pixelthing/15806918992/

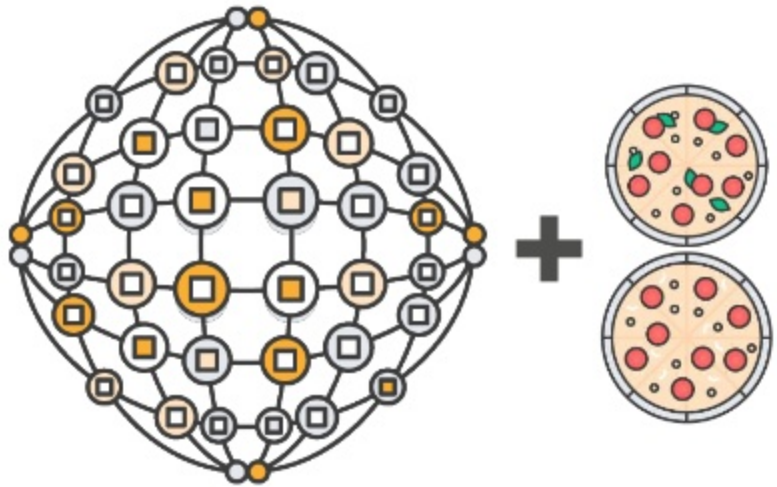
Development transformation at Amazon: 2001-2009

2001



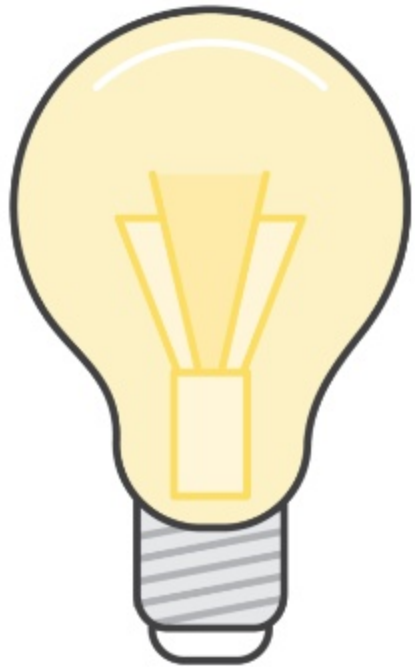
monolithic
application + teams

2009



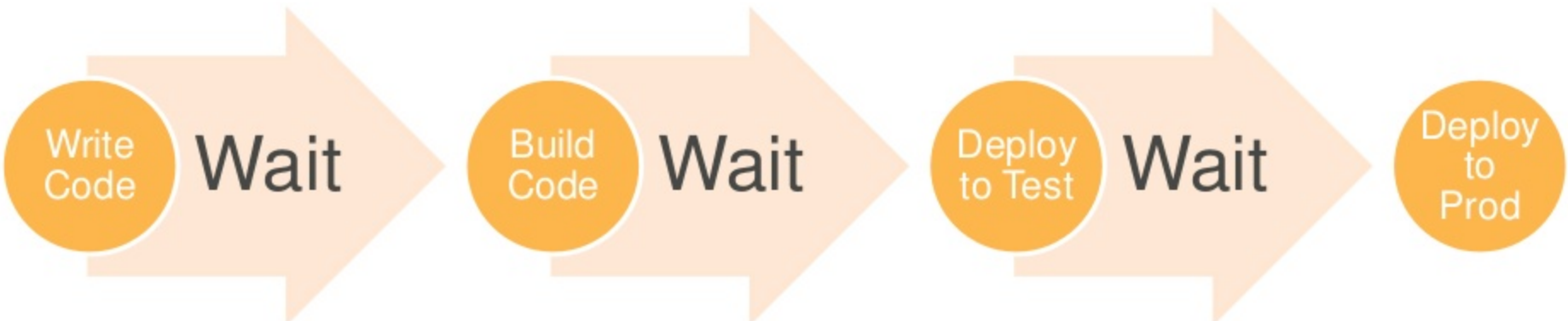
microservices + 2 pizza teams

Things went much better under this model and teams were developing features faster than ever, but we felt that we could still improve

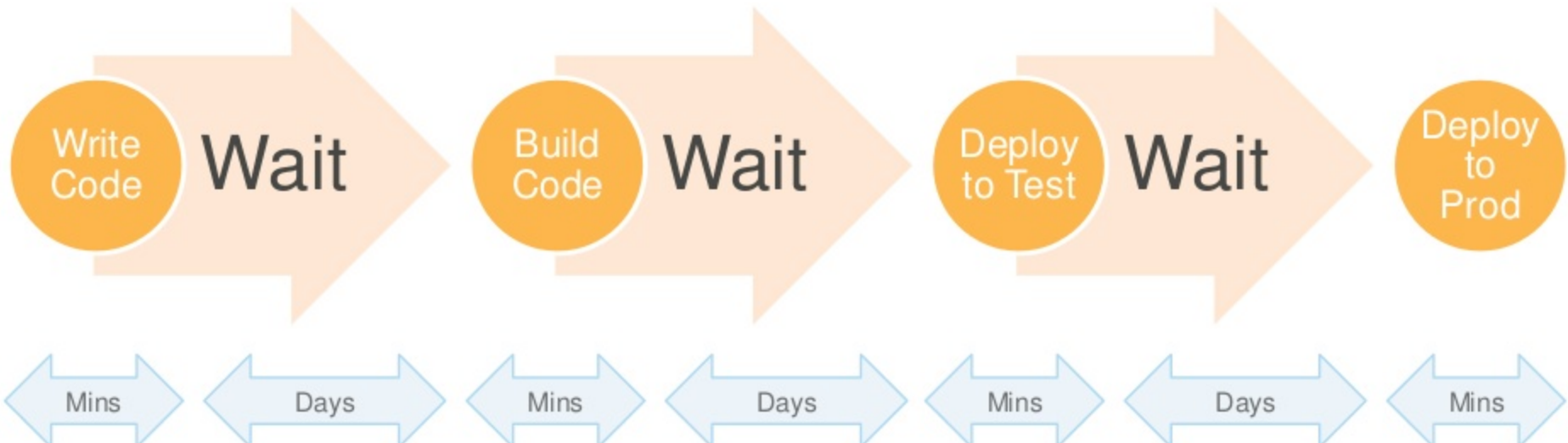


In 2009, we ran a study to find out where inefficiencies might still exist

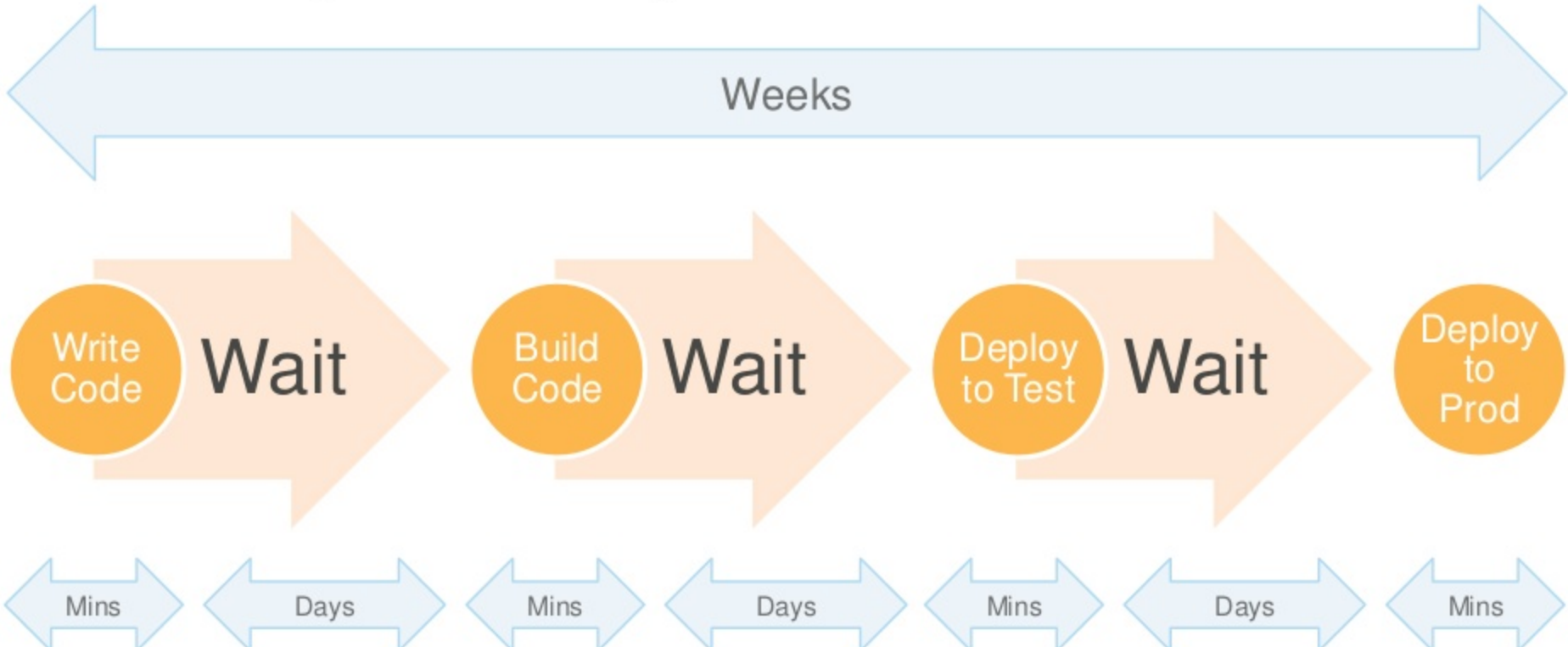
We were just waiting.



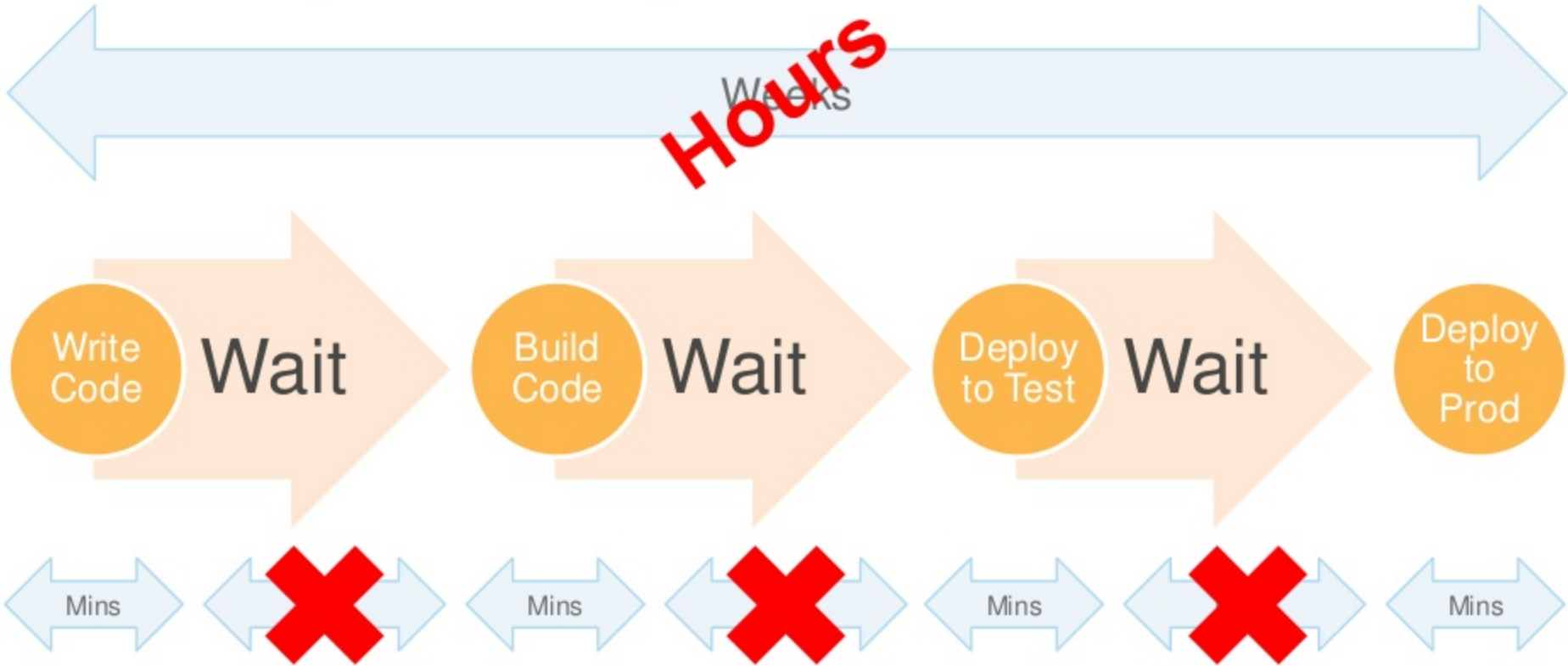
We were just waiting.



We were just waiting.



We were just waiting.




We built tools to automate our software release process
https://secure.flickr.com/photos/lindseygee/5894017854/

Automated actions and transitions from check-in to production

Development benefits:

- Faster
- Safer
- Consistent and standardized
- Visualization of the process

This has continued to work out really well:

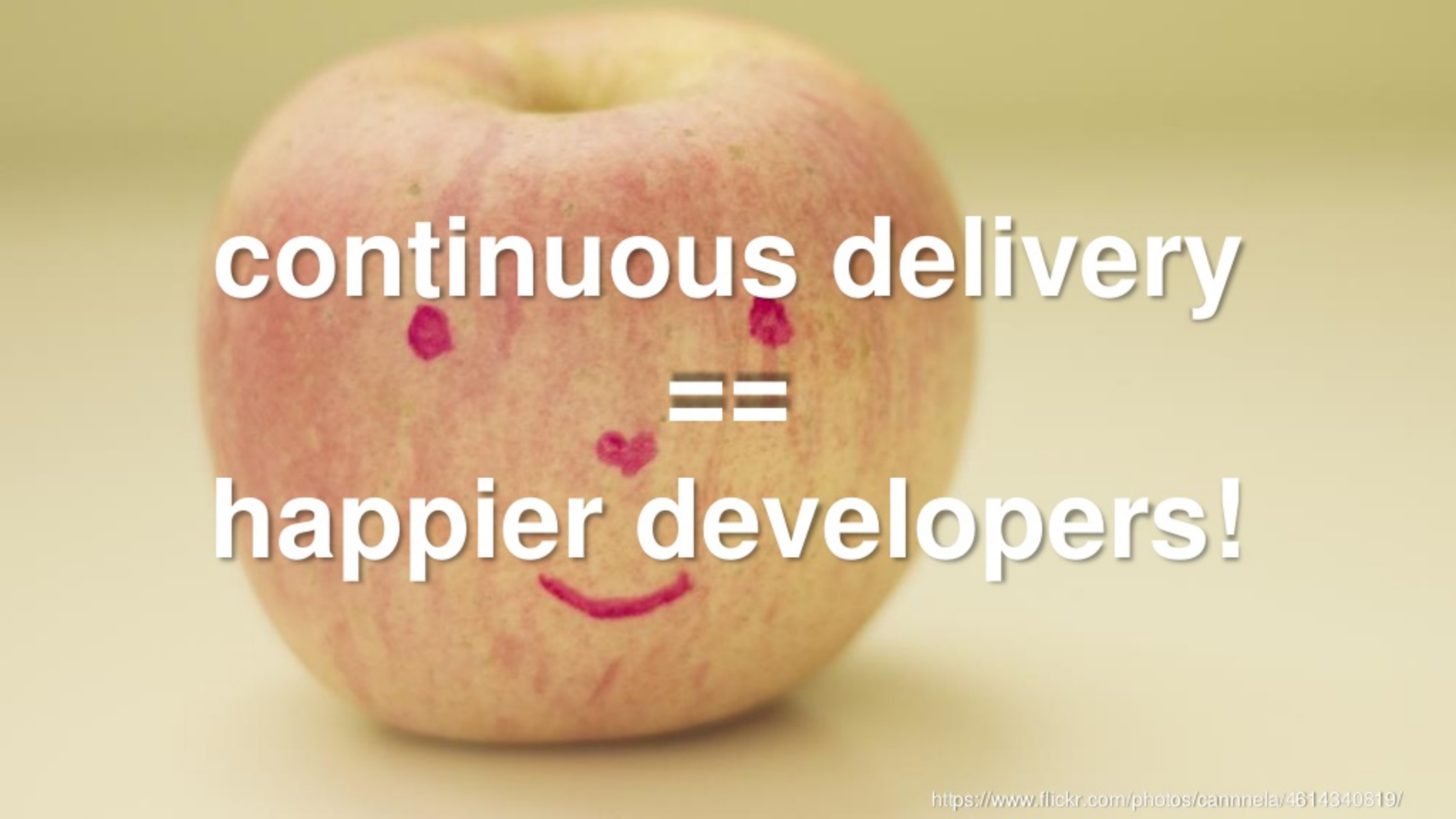
In 2014:

- Thousands of service teams across Amazon
- Building microservices
- Practicing continuous delivery
- Many environments (staging, beta, production)

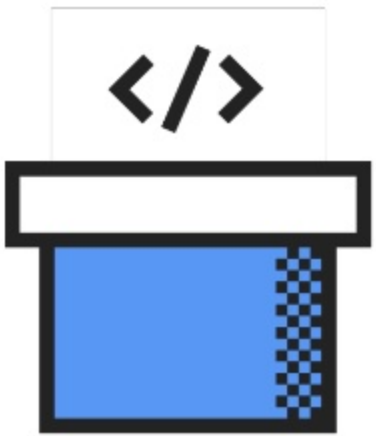
50 million deployments

We continue to survey our software developers every year and in 2014 results found only one development tool/service could be correlated statistically with happier developers:

Our pipelines service!


continuous delivery
==
happier developers!
https://www.flickr.com/photos/cannnela/4614340819/

AWS CodePipeline



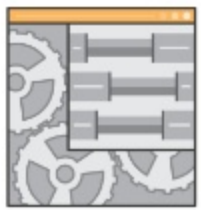
Continuous delivery service for fast and reliable application updates

Model and visualize your software release process

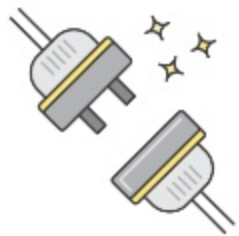
Builds, tests, and deploys your code every time there is a code change

Integrates with third-party tools and AWS

AWS CodePipeline benefits

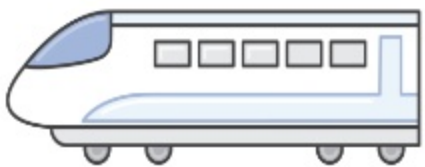


Configurable workflow



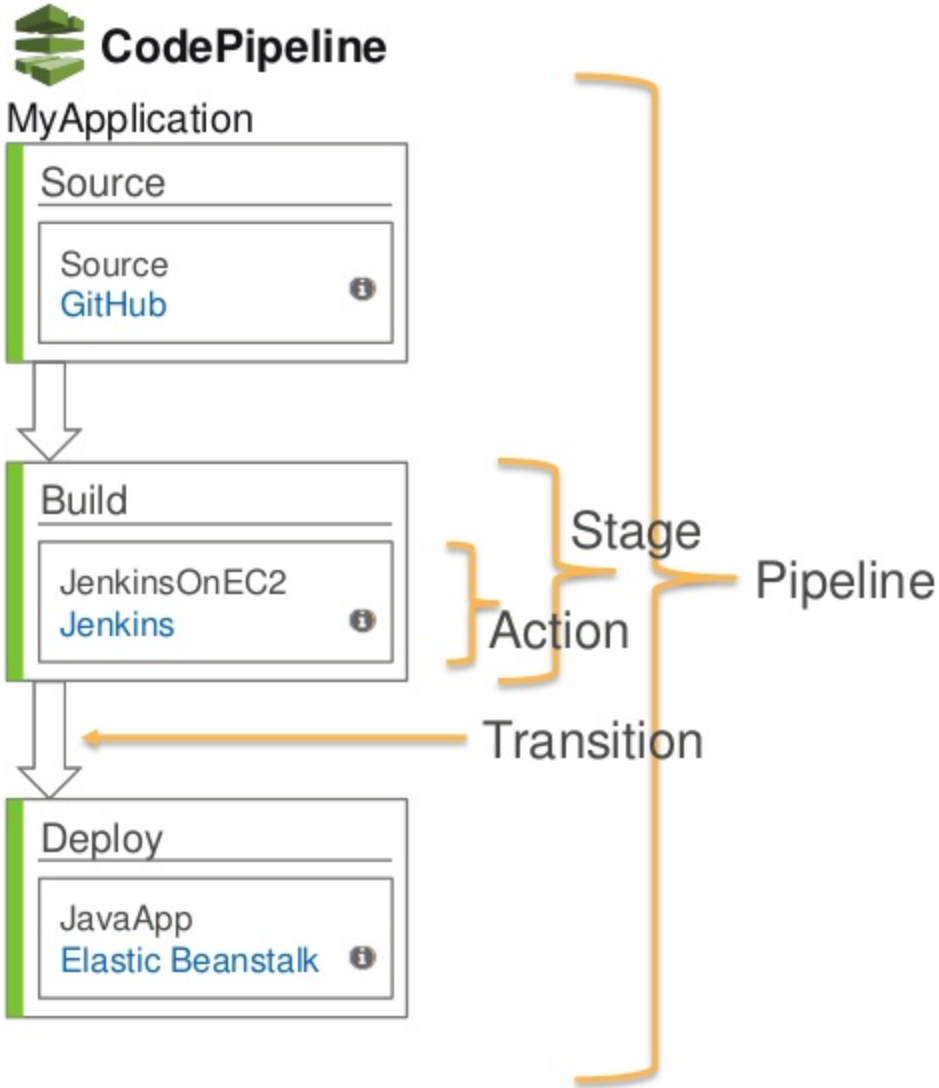
Easy to integrate

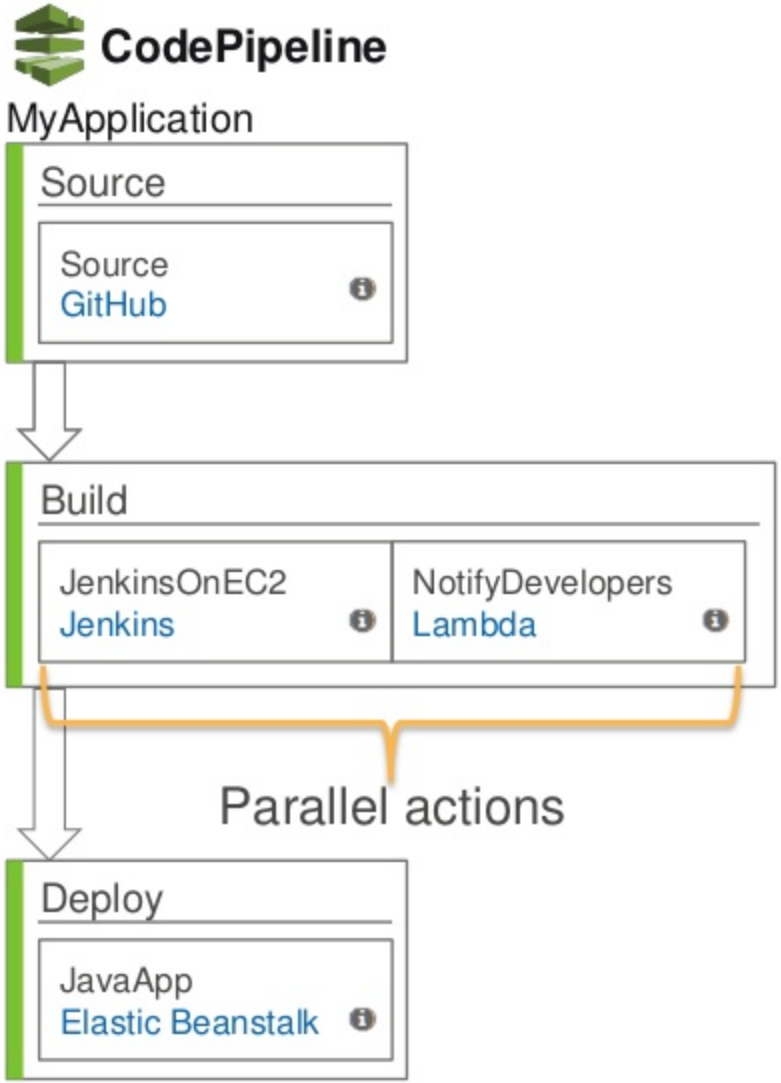
Improved quality

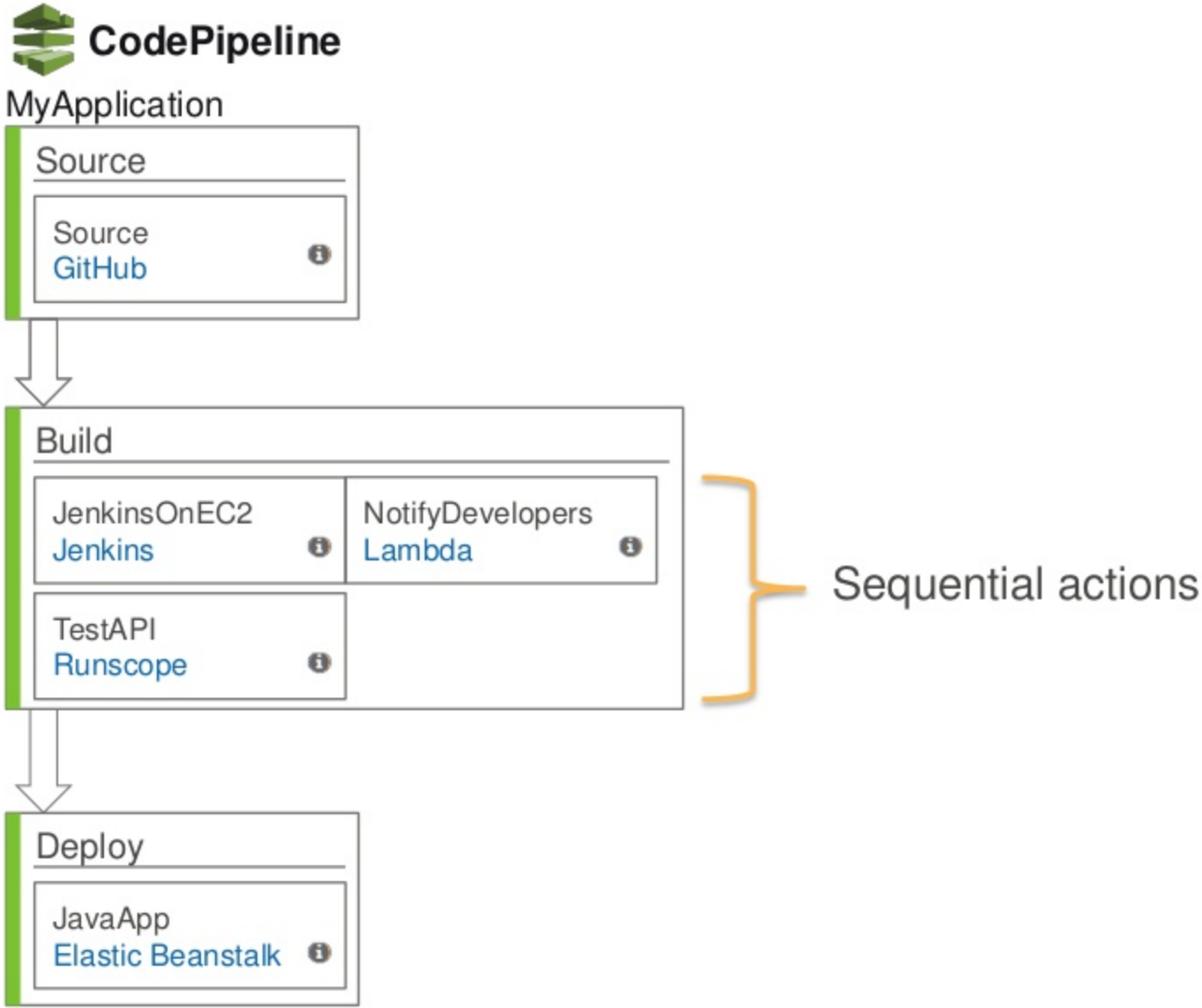


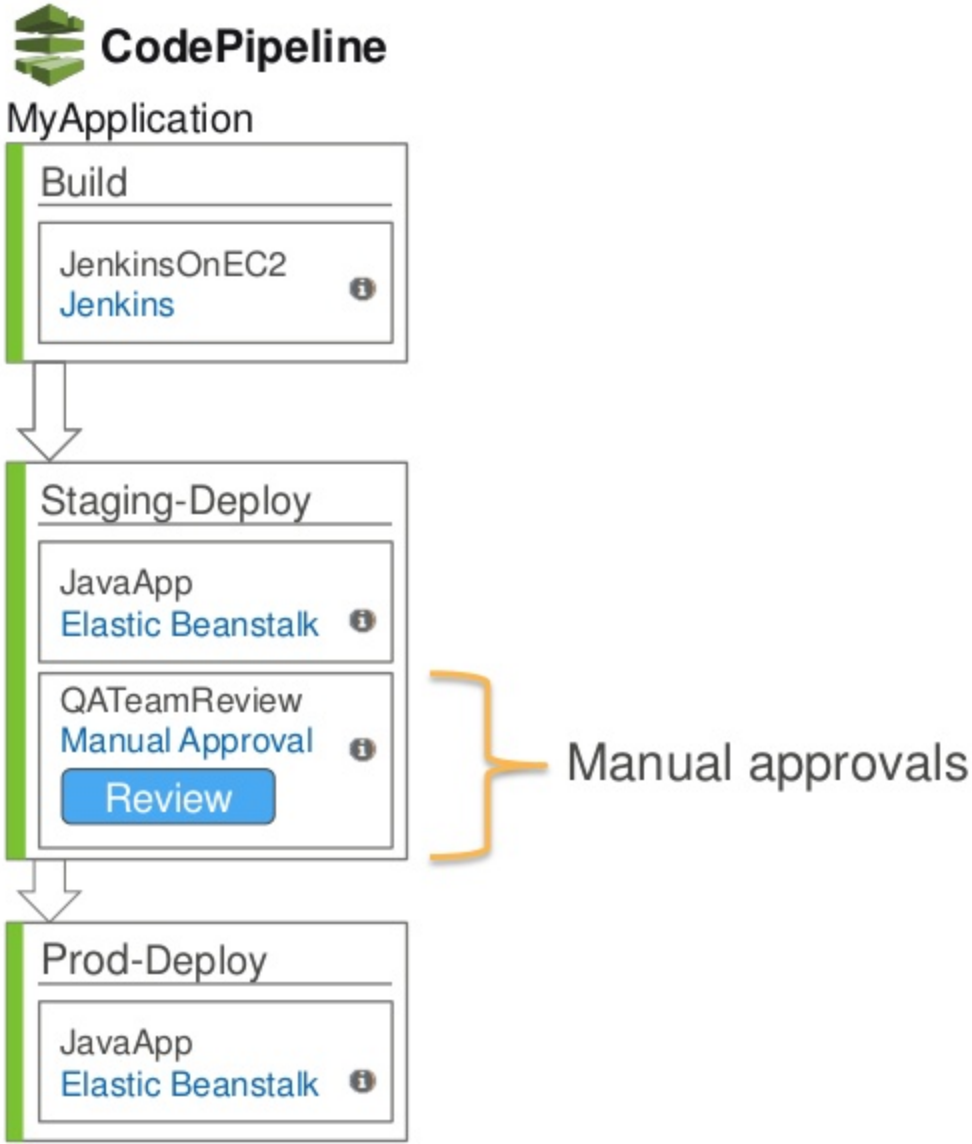
Rapid delivery

Get started fast

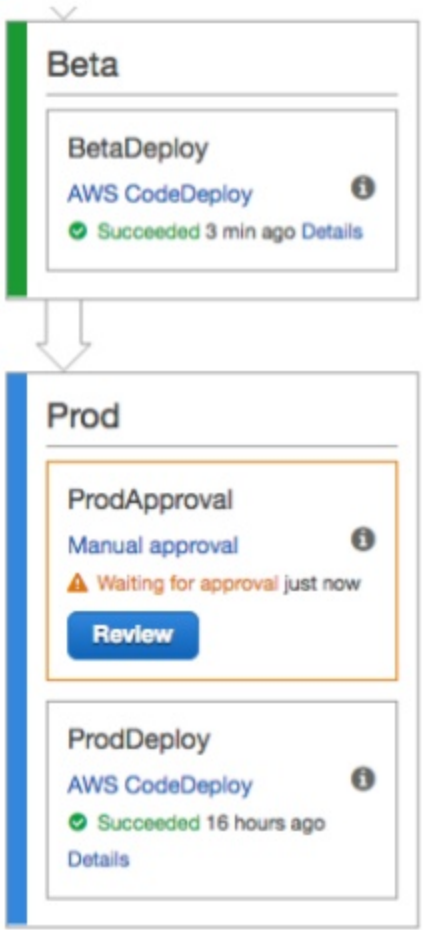

CodePipeline
MyApplication
Source
Source
GitHub
Build
JenkinsOnEC2
Jenkins
Deploy
JavaApp
Elastic Beanstalk
Stage
Action
Pipeline
Transition


CodePipeline
MyApplication
Source
Source
GitHub
Build
JenkinsOnEC2
Jenkins
NotifyDevelopers
Lambda
Parallel actions
Deploy
JavaApp
Elastic Beanstalk


CodePipeline
MyApplication
Source
Source
GitHub
Build
JenkinsOnEC2
Jenkins
NotifyDevelopers
Lambda
TestAPI
Runscope
Sequential actions
Deploy
JavaApp
Elastic Beanstalk

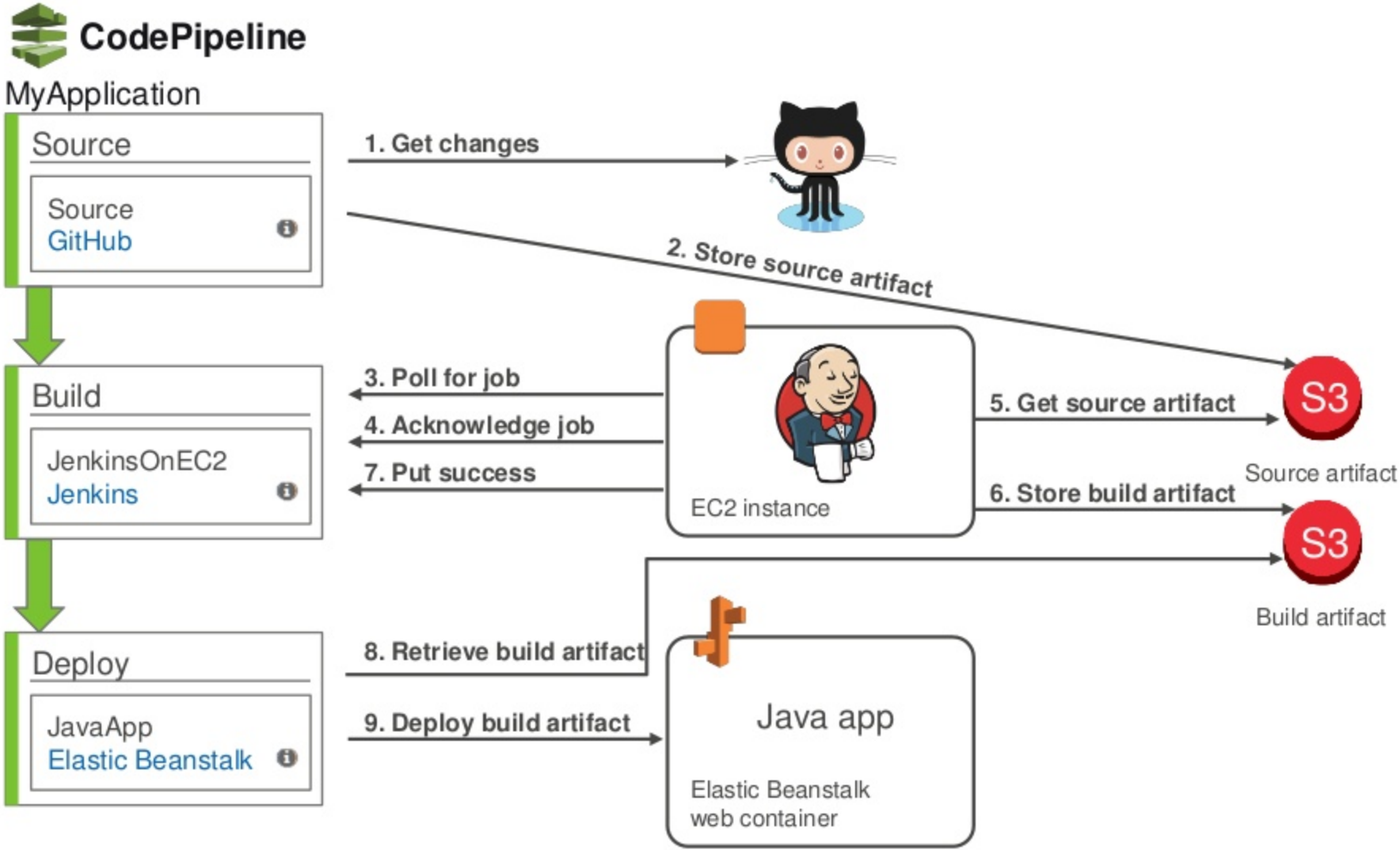

CodePipeline
MyApplication
Build
JenkinsOnEC2
Jenkins
Staging-Deploy
JavaApp
Elastic Beanstalk
QATeamReview
Manual Approval
Review
Manual approvals
Prod-Deploy
JavaApp
Elastic Beanstalk

Manual approvals – New!

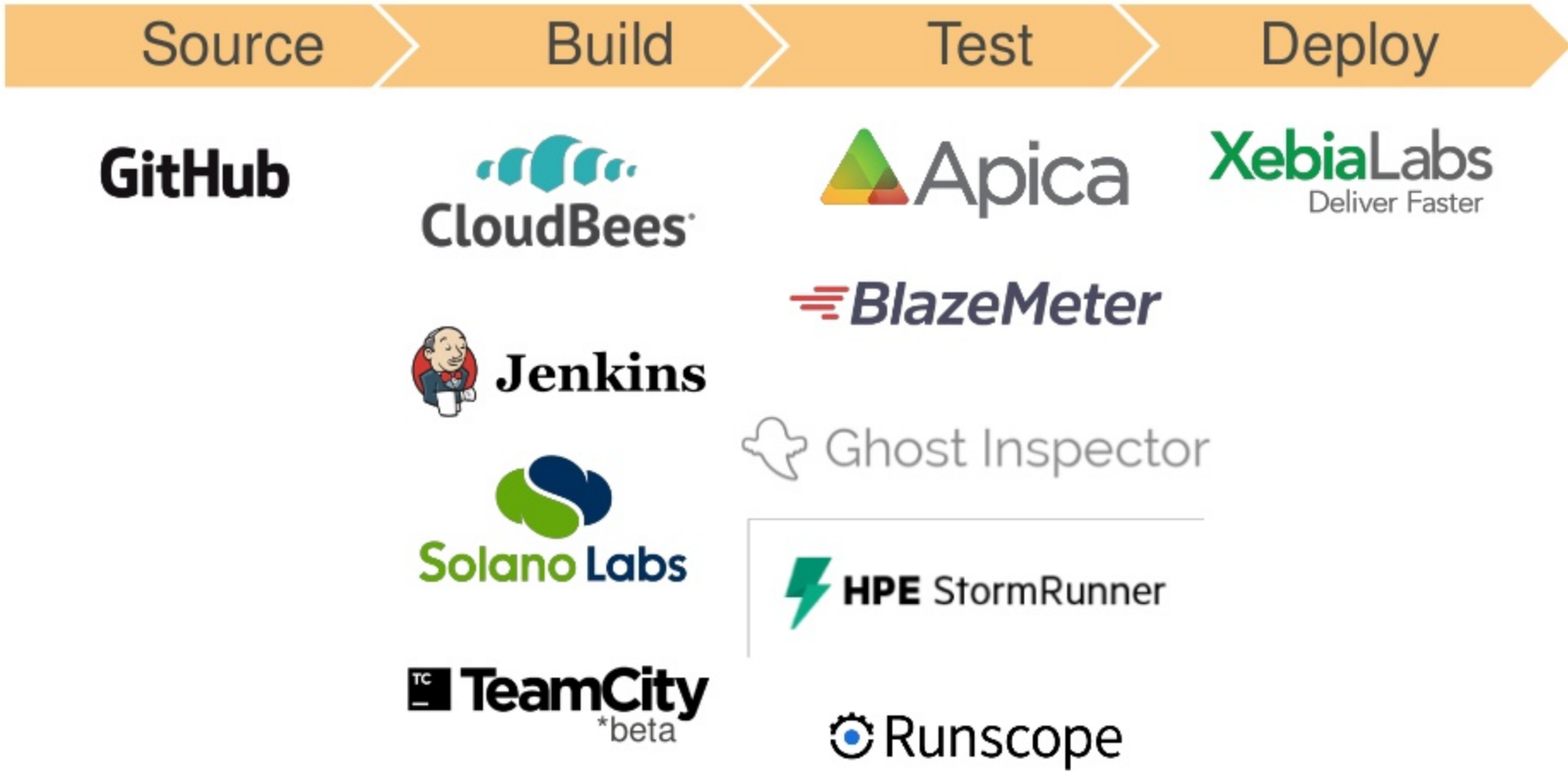


You can add a manual approval at the point where you want the pipeline to stop running until someone approves or rejects the revision in progress.

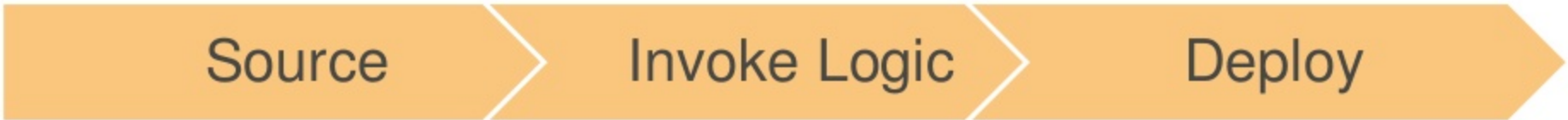
- Pipeline will stop executing when it has reached the point at which you set the approval action
- Pipeline execution resumes only when the action has been approved
- Approval action managed with AWS Identity and Access Management (IAM) permissions
- Notify approvers in several ways including email, SMS, webhooks, and more
- Useful for manual QA actions or as part of “Canary” deploy models


CodePipeline
MyApplication
Source
Source
GitHub
Build
JenkinsOnEC2
Jenkins
Deploy
JavaApp
Elastic Beanstalk
1. Get changes
2. Store source artifact
3. Poll for job
4. Acknowledge job
7. Put success
EC2 instance
5. Get source artifact
6. Store build artifact
S3
Source artifact
S3
Build artifact
8. Retrieve build artifact
9. Deploy build artifact
Java app
Elastic Beanstalk
web container

We have a strong partner list, and it's growing



AWS service integrations



Amazon S3

AWS CodeCommit

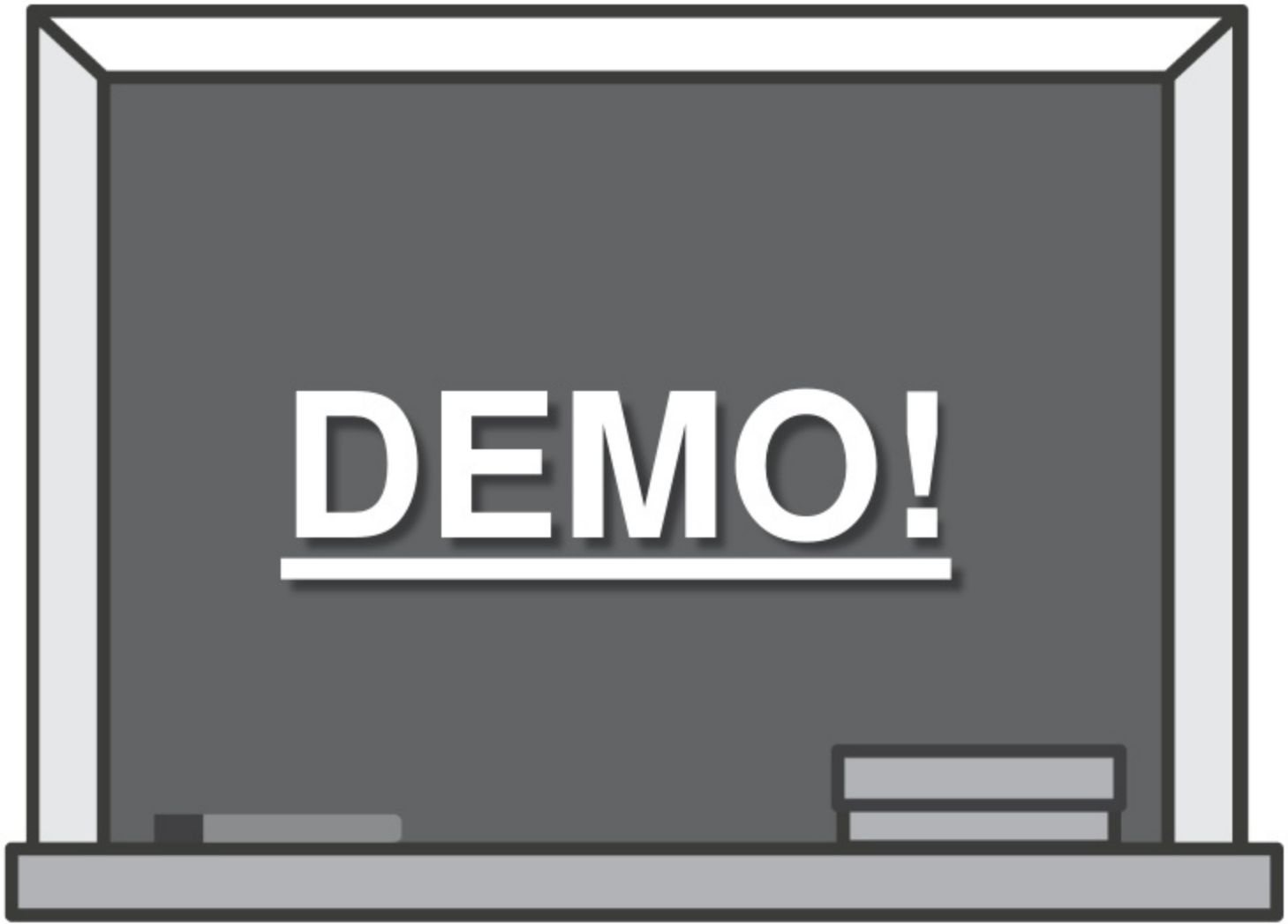
AWS Lambda

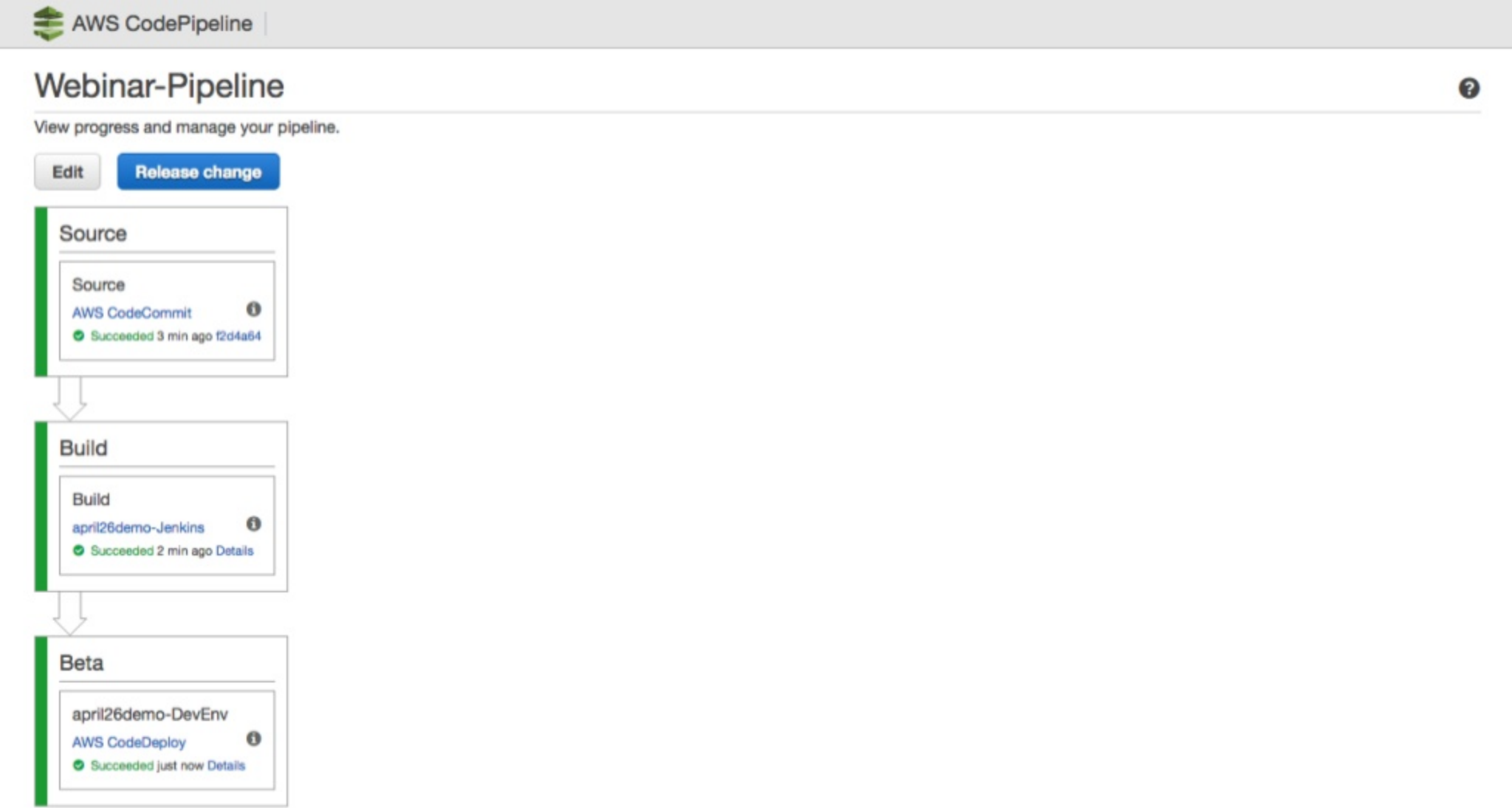
AWS CodeDeploy

AWS Elastic Beanstalk

AWS OpsWorks


Building your application development release pipeline
https://www.flickr.com/photos/seattlemunicipalarchives/12504672623/


DEMO!


AWS CodePipeline
Webinar-Pipeline
View progress and manage your pipeline.
Edit
Release change
Source
Source
AWS CodeCommit
Succeeded 3 min ago f2d4a64
Build
Build
april26demo-Jenkins
Succeeded 2 min ago Details
Beta
april26demo-DevEnv
AWS CodeDeploy
Succeeded just now Details

AWS CodePipeline

Webinar-Pipeline

View progress and manage your pipeline.

Edit | Release change

Source

Source

AWS CodeCommit

Succeeded 3 min ago f2d4a64

Build

Build

april26demo-Jenkins

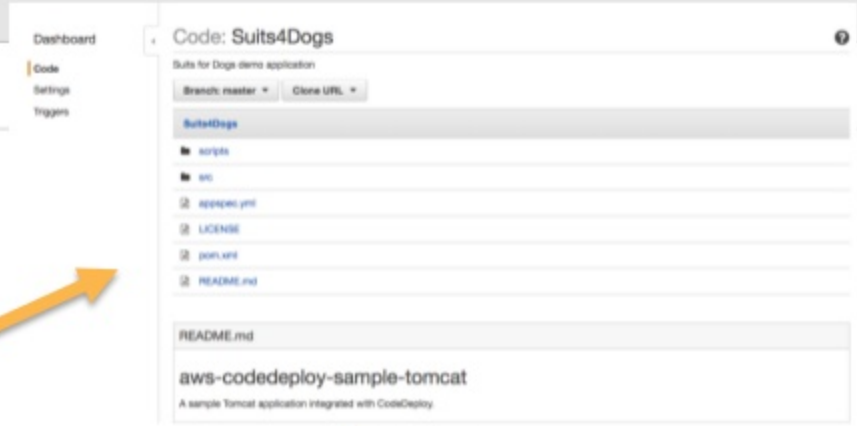
Succeeded 2 min ago Details

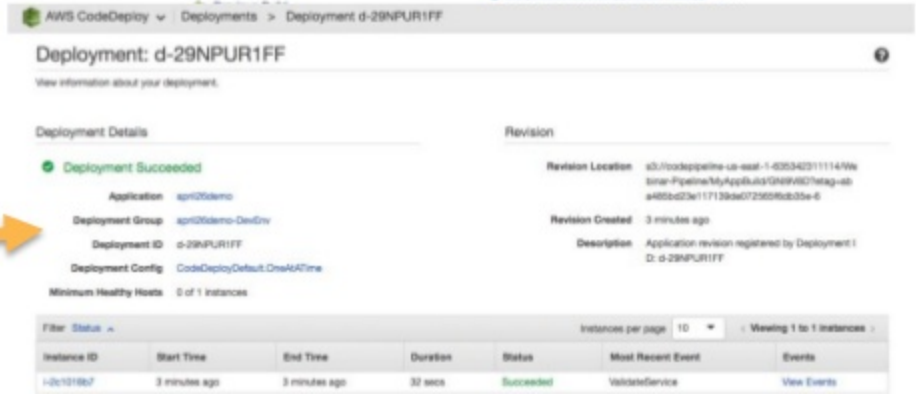
Beta

april26demo-DevEnv

AWS CodeDeploy

Succeeded just now Details





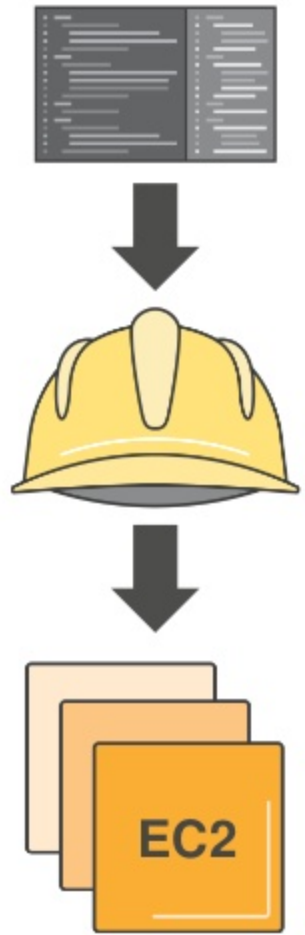

Build and test your application
https://secure.flickr.com/photos/spenceyc/7481166880

Building your code

"Building" code typically refers to languages that require compiled binaries:

- .NET languages: C#, F#, VB.NET, etc.
- Java and JVM languages: Java, Scala, JRuby
- Go
- iOS languages: Swift, Objective-C

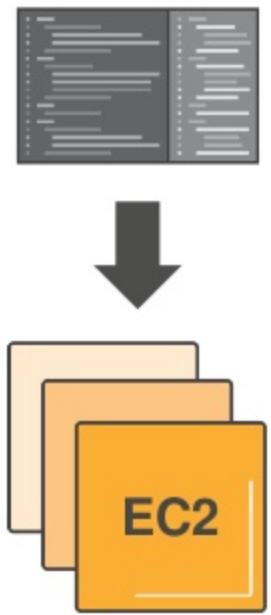
We also refer to the process of creating Docker container images as "building" the image.



No building required!

Many languages don't require building. These are considered interpreted languages:

- PHP
- Ruby
- Python
- Node.js



You can just deploy your code!

Testing your code

Testing is both a science and an art form!

Goals for testing your code:

- Want to confirm desired functionality
- Catch programming syntax errors
- Standardize code patterns and format
- Reduce bugs due to non-desired application usage and logic failures
- Make applications more secure


Deploying your applications
https://secure.flickr.com/photos/simononly/15386966677

AWS CodeDeploy

Automates code deployments to any instance

Handles the complexity of updating your applications

Avoid downtime during application deployment

Deploy to Amazon EC2 or on-premises servers, in any language and on any operating system

Integrates with third-party tools and AWS

appspec.yml example

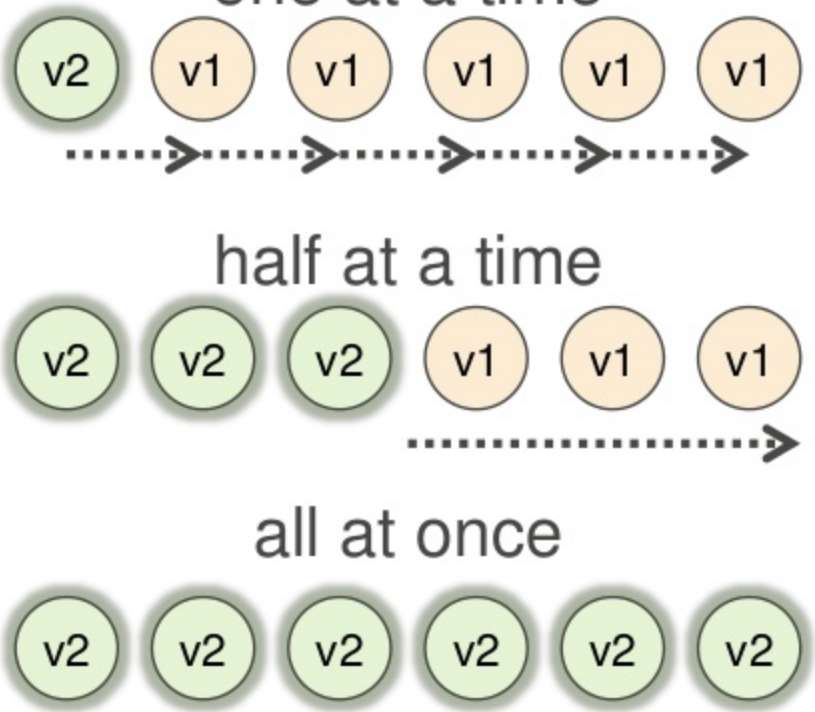
```yaml
version: 0.0
os: linux
files:
  - source: /
    destination: /var/www/html
permissions:
  - object: /var/www/html
    pattern: “*.html”
    owner: root
    group: root
    mode: 755
hooks:
  ApplicationStop:
    - location: scripts/deregister_from_elb.sh
  BeforeInstall:
    - location: scripts/install_dependencies.sh
  ApplicationStart:
    - location: scripts/start_httpd.sh
  ValidateService:
    - location: scripts/test_site.sh
    - location: scripts/register_with_elb.sh
```

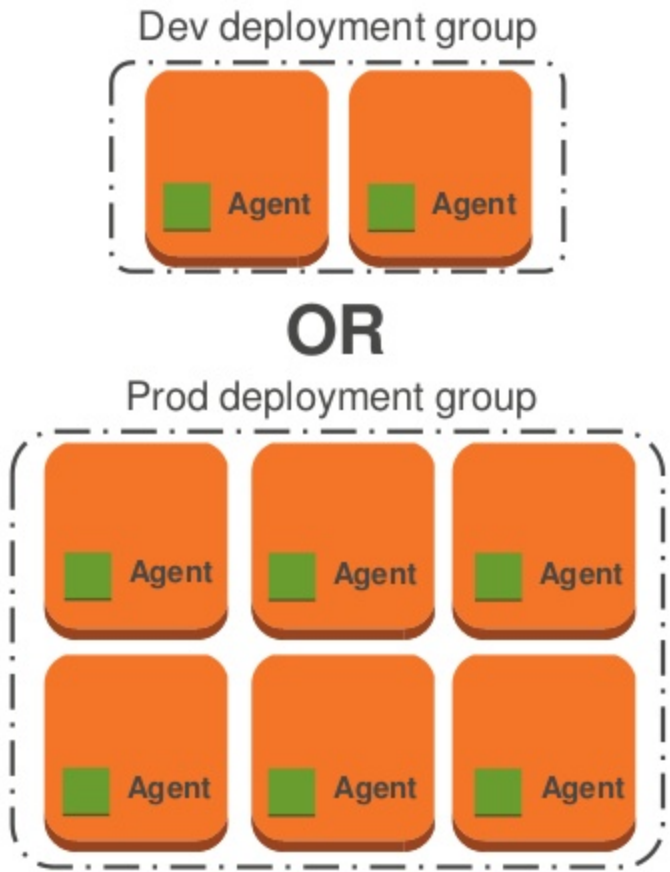
appspec.yml example

```
version: 0.0
os: linux
files:
  - source: /
    destination: /var/www/html
permissions:
  - object: /var/www/html
    pattern: "*.html"
    owner: root
    group: root
    mode: 755
hooks:
  ApplicationStop:
    - location: scripts/deregister_from_elb.sh
  BeforeInstall:
    - location: scripts/install_dependencies.sh
  ApplicationStart:
    - location: scripts/start_httpd.sh
  ValidateService:
    - location: scripts/test_site.sh
    - location: scripts/register_with_elb.sh
```

- Send application files to one directory and configuration files to another
- Set specific permissions on specific directories and files
- Remove/add instance to ELB
- Install dependency packages
- Start Apache
- Confirm successful deploy
- More!

Choose deployment speed and group





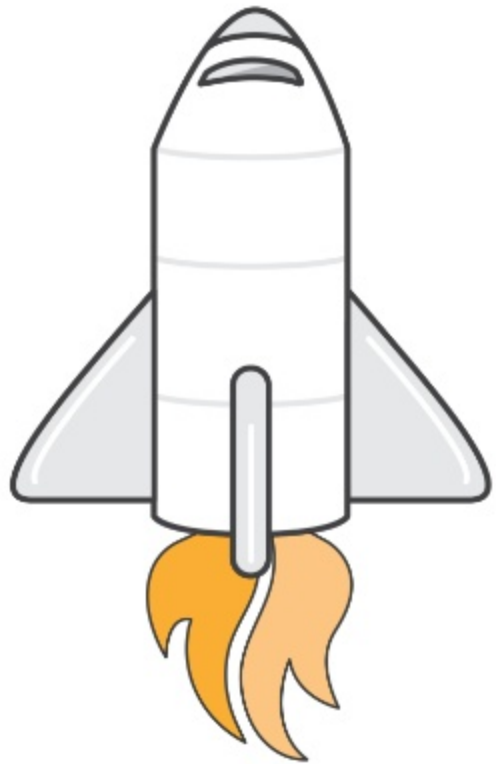

Launching to
production
https://www.flickr.com/photos/spacex/16510243060/

Launching to production

After you've built and tested your code and hopefully gone through a few pre-production deploys, its time for the real thing!

You'll want think about:

- Impact to customers
- Impact to infrastructure
- Impact to business

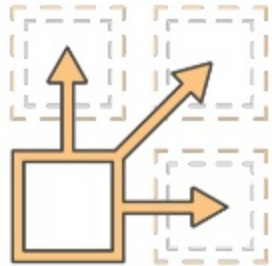


How can we track these and communicate deploys?

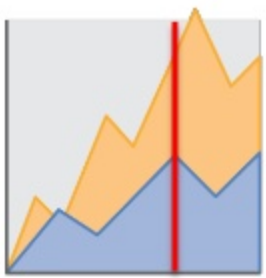
Extend AWS CodePipeline using custom actions

Mobile testing

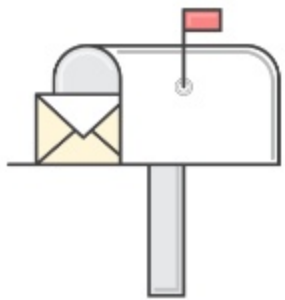
Update tickets



Provision resources

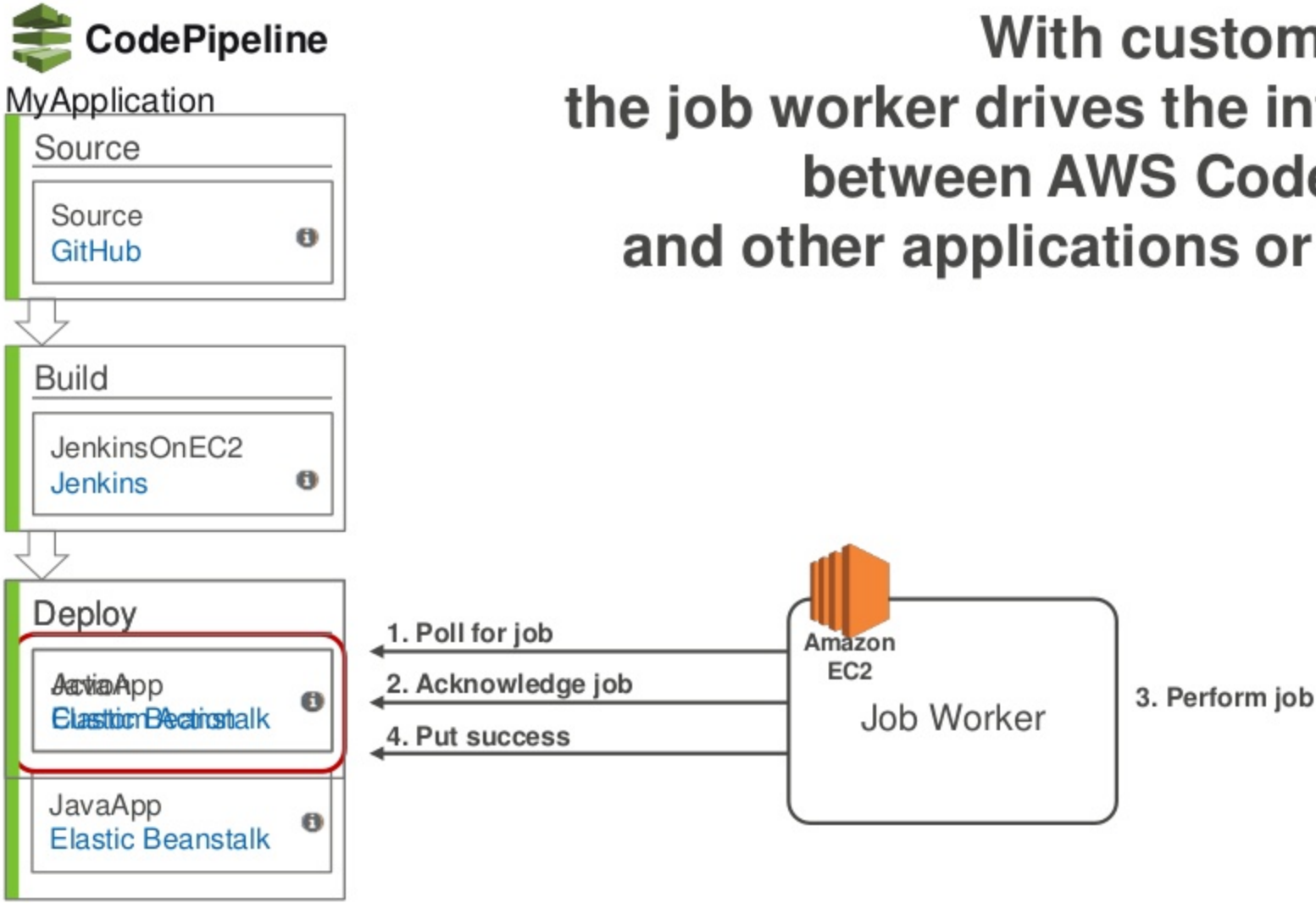


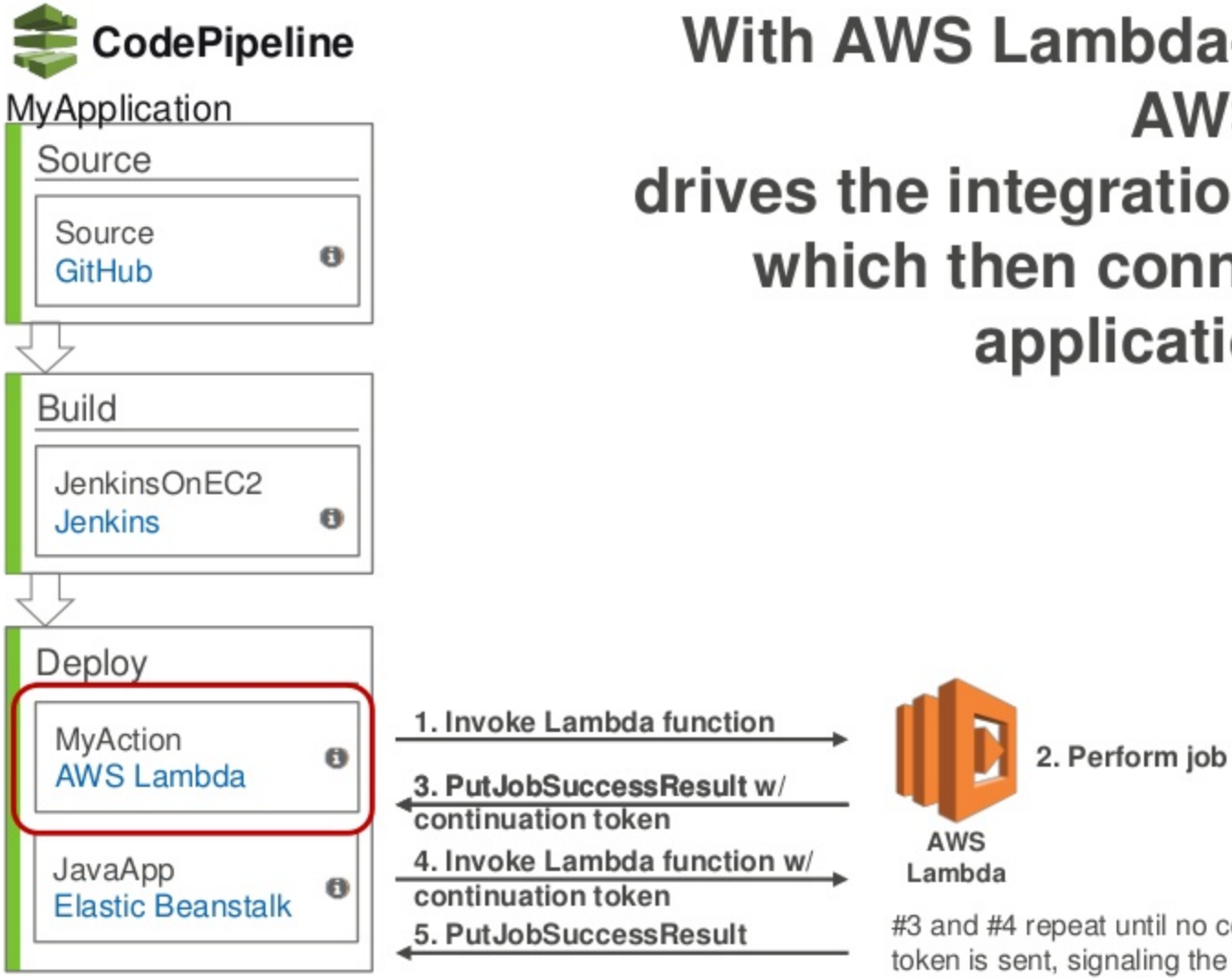
Update dashboards



Send notifications

Security scan


CodePipeline
MyApplication
Source
Source
GitHub
Build
JenkinsOnEC2
Jenkins
Deploy
JavaApp
Elastic Beanstalk
JavaApp
Elastic Beanstalk
1. Poll for job
2. Acknowledge job
4. Put success
Amazon
EC2
Job Worker
3. Perform job
With custom actions,
the job worker drives the interaction
between AWS CodePipeline
and other applications or services



With AWS Lambda-based actions, AWS CodePipeline drives the integration with Lambda, which then connects with other applications or services

What extension method should I use?

Lambda	Custom Action
Short-running tasks are easy to build	Can perform any type of workload
Long-running tasks need more work	Control over links displayed in console
Node.js, Python, and Java support	Any language support
Runs on AWS	Can run on-premises
No servers to provision or manage	Requires compute resources

FIN, ACK

We've seen a quick run through today of the benefits of continuous delivery on our software release process:

- Continuous integration (build/test) helps shrink our feedback loop greatly
- We can get our software out in front of our users much more rapidly
- By moving faster we can actually ensure better quality
- AWS CodePipeline allows for integration with almost any service or tool you can think of!
    - Plus visualization of what's going on!

Try it out today

Test out AWS CodePipeline and spin up a full continuous delivery pipeline using the Starter Kit

bit.ly/AWSCodeStarterKit

But wait, there's more!

Resources to learn more:

- Continuous integration: https://aws.amazon.com/devops/continuous-integration/
- Continuous delivery: https://aws.amazon.com/devops/continuous-delivery/

- AWS CodePipeline
  - https://aws.amazon.com/codepipeline/
  - https://aws.amazon.com/documentation/codepipeline/
- AWS CodeDeploy
  - https://aws.amazon.com/codedeploy/
  - https://aws.amazon.com/documentation/codedeploy/
  - https://github.com/awslabs/aws-codedeploy-samples
- AWS Code Services Starter Kit: http://bit.ly/AWSCodeStarterKit

Thank you!

amazon web services