Continuous Delivery - The Missing Parts

Paul Stack
http://twitter.com/stack72
mail: paul@paulstack.co.uk
About Me

- Infra Engineer for a semantic search based company
- Reformed ASP.NET / C# Developer
- DevOps Extremist
- Conference Junkie
Background to this talk
Continuous Delivery

.... is a set of practices and principles aimed at, building, testing and releasing software faster and more frequently.
8 principles of Continuous Delivery

- The process for releasing/deploying software MUST be repeatable and reliable.
- Automate everything!
- If somethings difficult or painful, do it more often.
- Keep everything in source control
- Done means “released”
- Build quality in!
- Everybody has responsibility for the release process
- Improve continuously
4 Practices of Continuous Delivery

- Build binaries only once
- Use precisely the same mechanism to deploy to every environment
- Smoke test your deployment
- If anything fails, stop the line!
Has anyone read this book?
So continuous delivery is only 5 years old??
Agile Manifesto

“Our highest priority is to satisfy the customer through early and continuous delivery of valuable software"
Common Misconceptions
1. Continuous Delivery is something only startups can achieve
2. Continuous Delivery only works for NodeJS / Ruby / Go developers
3. We can hire a consultant to help us implement ‘Continuous Delivery’
4. Right click and deploy in your IDE is continuous delivery
5. Continuous delivery is as simple as hooking github to our Azure / Heroku account.....
“Until your pretty code is in production, making money, or doing whatever it does, you’ve just wasted your time”

Chris Read
@cread
#LondonCI
But haven’t I just contradicted myself?
The traditional technical side of a company

- Developers
- QA
- SysAdmins
- Network
- Helpdesk
- InfoSec
- + lots more
IT Organization Model Diagram 5 - Detailed Functions
Prepared By: DP Harshman
Date: 10/28/09 Rev'd: 06/14/10
Purpose: to facilitate discussions

Notes:
1. Functions may be organized slightly differently than shown but must exist and must be managed.
2. IT HR, IT Finance, IT Legal departments/groups omitted for clarity.

© DP Harshman - www.fromtheranks.com
Aren’t we supposed to be one team?
“How long would it take your organisation to deploy a change that involved just one single line of code? Do you do this on a repeatable, reliable basis?”

Implementing Lean Software Development
Mary Poppendieck
Value Stream Map

- Issue Reported (15 minutes)
- Prioritization Meeting (1 hour)
- Developer Fixes Issue (2 hours)
- QA Tests Issue (2 hours)
- Release team creates hotfix and release (1 hour)
- Production Deployment (1 hour)

Wait (3.5 days avg)
Wait (4 days avg)
Wait (4 days avg)
Wait (4 days avg)
Wait (4 days avg)
The Value Stream Map for Ops is worse....
The rise of DevOps......
10 deploys per day
Dev & ops cooperation at Flickr

John Allspaw & Paul Hammond
Velocity 2009
October 30 - 31, 2009
Ghent, Belgium

Developers + System Administrators
for 2 days

= DevOpsDays
Development: I want change!

Wall of Confusion

Operations: I want stability!
Confuse of Dev or Ops? Simple rule: if you are praise for Web site success, you are Dev; if you are blame when Web site down, you are Ops.
WORKED FINE IN DEV
OPS PROBLEM NOW
“DevOps is not about learning each other's job, it's understanding the money is made when software is actually running in production.”

Dave Mangot
@davemangot
How does DevOps help with this type of thing?
‘The 3 ways’

• Systems thinking

• Amplify Feedback loops

• Culture of Continual Experimentation and learning
The first way...
Understanding Work..

- Business Projects (search, UI improvements)
- Internal Projects (architecture changes)
- Changes (deployments, schema updates)
- Unplanned work (downtime, investigations)
The second way...
We found that when we woke developers up at 2am, defects got fixed faster!

-Patrick Lightbody
CEO, Browsermob
The third way...
You can adopt DevOps through....

- Culture
- Automation
- Measurement
- Sharing
We are *all* part of the same team!
Metrics and Automation are key!
The technical side of the business post-DevOps....
Image by John Turner of PaddyPower.com
The ops side of Continuous Delivery
Configuration Management
Immutable Infrastructure
Trash Your Servers and Burn Your Code: Immutable Infrastructure and Disposable Components
Immutable Infrastructure
Disposable Infrastructure
Logging
Metrics
Monitoring
@@nagios_service { "$\{host\}  - Check LogStash Service":
  use => 'generic-service',
  host_name => $\{host\},
  service_description => 'Check LogStash Service',
  check_command => 'check_ssh_service!java!-a logstash',
  normal_check_interval => '5',
  retry_check_interval => '1',
}

@@nagios_service { "$\{host\}  - Check Redis-Server Service":
  use => 'generic-service',
  host_name => $\{host\},
  service_description => 'Check Redis-Server Service',
  check_command => 'check_ssh_service!redis-server',
  normal_check_interval => '5',
  retry_check_interval => '1',
}
nagios_service {"ES split brain":
  ensure        => present,
  use           => 'generic-service',
  host_name     => $vip_fqdn,
  service_description => 'ES Split Brain Check',
  check_command => "check_es_split_brain!${es_nodes}'",
  normal_check_interval => '5',
  retry_check_interval => '1'
}

nagios_service {"Check ES Nodes in Cluster":
  ensure        => present,
  use           => 'generic-service',
  host_name     => $vip_fqdn,
  service_description => 'ES Nodes in Cluster',
  check_command => "check_es_nodes!${num_es_nodes}'",
  normal_check_interval => '5',
  retry_check_interval => '1'
}
sensu_checks:
  - name: ram_check
    command: "{{ sensu_plugins_directory }}/check-ram.rb -w 50 -c 15"
    interval: 60
    handlers:
      - default
    subscribers:
      - common
  - name: check_cpu
    command: "{{ sensu_plugins_directory }}/check_cpu.sh -w 85 -c 95"
    interval: 60
    handlers:
      - default
    subscribers:
      - common
  - name: check_nginx_process
    command: "{{ sensu_plugins_directory }}/check-process.sh -p nginx"
    interval: 60
    handlers:
      - default
    subscribers:
      - web
  - name: check_sensu_server_process
    command: "{{ sensu_plugins_directory }}/check-process.sh -p sensu-server"
    interval: 60
    handlers:
      - default
    subscribers:
      - sensu
Orchestration
FREAK Vulnerability Test - https://freakattack.com/
in 18h39m

FREAK Vulnerability Test - https://freakattack.com/
in 18h29m

FREAK Vulnerability Test - https://freakattack.com/
in 18h44m

FREAK Vulnerability Test - https://freakattack.com/
in 18h59m

jet.it ➽ FREAK Vulnerability Test - https://freakattack.com/
in 19h14m

Poodle Vulnerability Test - http://www.troyhunt.com/2014/10/everything-you-need-to-know-about-

d ➽ Poodle Vulnerability Test - http://www.troyhunt.com/2014/10/everything-you-need-to-know-

st.it ➽ Poodle Vulnerability Test - http://www.troyhunt.com/2014/10/everything-you-need-

Ghost Vulnerability ➽ Test for the Ghost Vulnerability. This is essential an error in glibc library. (http://blogs.

Heartbleed Test ➽ Test to check for the vulnerability of our servers to the heartbleed security issue http://

Shellshock Vulnerability ➽ Test to make sure our hosts are not susceptible to ShellShock (http://www.

Data Center as Code?
resource "aws_subnet" "primary-public" {
  vpc_id = "${aws_vpc.default.id}"

  cidr_block = "${lookup(var.aws_primary_public_cidr, var.aws_region)}"
  availability_zone = "${lookup(var.aws_primary_az, var.aws_region)}"
  map_public_ip_on_launch = true

  tags {
    Name = "${concat("Public ", lookup(var.aws_primary_az, var.aws_region), " \\ DIGIT VPC")}" 
  }
}

resource "aws_subnet" "secondary-public" {
  vpc_id = "${aws_vpc.default.id}"

  cidr_block = "${lookup(var.aws_secondary_public_cidr, var.aws_region)}"
  availability_zone = "${lookup(var.aws_secondary_az, var.aws_region)}"
  map_public_ip_on_launch = true

  tags {
    Name = "${concat("Public ", lookup(var.aws_secondary_az, var.aws_region), " \\ DIGIT VPC")}" 
  }
}

resource "aws_subnet" "tertiary-public" {
  vpc_id = "${aws_vpc.default.id}"

  cidr_block = "${lookup(var.aws_tertiary_public_cidr, var.aws_region)}"
  availability_zone = "${lookup(var.aws_tertiary_az, var.aws_region)}"
  map_public_ip_on_launch = true

  tags {
    Name = "${concat("Public ", lookup(var.aws_tertiary_az, var.aws_region), " \\ DIGIT VPC")}" 
  }
}
resource "aws_route53_record" "private_uchiwa" {
    zone_id = "Z1IPZ887994N5C"
    name = "uchiwa-$var.aws_region"
    type = "CNAME"
    ttl = "60"
    records = ["${aws_elb.uchiwa_elb.dns_name}"
}

resource "aws_route53_record" "public_uchiwa" {
    zone_id = "Z91Y6CBGPDVBL"
    name = "uchiwa-$var.aws_region"
    type = "CNAME"
    ttl = "60"
    records = ["${aws_elb.uchiwa_elb.dns_name}"
}
Disaster Recovery?
The Continuous Delivery Maturity Model
## The Continuous Delivery Maturity Model

### Base
- Prioritized work
- Defined and documented process
- Frequent commits

### Beginner
- One backlog per team
- Share the pain
- Stable teams
- Adopt basic Agile methods
- Remove boundary dev & test

### Intermediate
- Extended team collaboration
- Component ownership
- Act on metrics
- Remove boundary dev & ops
- Common process for all changes
- Decentralize decisions

### Advanced
- Dedicated tools team
- Team responsible all the way to prod
- Deploy disconnected from Release
- Continuous improvement (Kaizen)

### Expert
- Cross functional teams
- No rollbacks (always roll forward)

### Design & Architecture
- Consolidated platform & technology
- Organize system into modules
- API management
- Library management
- Version control DB changes
- No (or minimal) branching
- Branch by abstraction
- Configuration as code
- Feature hiding
- Making components out of modules
- Full component based architecture
- Push business metrics
- Infrastructure as code
- Build bakery
- Zero touch continuous deployments

### Build & Deploy
- Versioned code base
- Scripted builds
- Basic scheduled builds (CI)
- Dedicated build server
- Documented manual deploy
- Some deployment scripts exists
- Polling builds
- Builds are stored
- Manual tag & versioning
- First step towards standardized deploys
- Auto triggered build (commit hooks)
- Automated tag & versioning
- Build once deploy anywhere
- Automated bulk of DB changes
- Basic pipeline with deploy to prod
- Scripted config changes (e.g. app server)
- Standard process for all environments
- Zero downtime deploys
- Multiple build machines
- Full automatic DB deploys
- Full automatic acceptance tests
- Automatic performance tests
- Automatic security tests
- Risk based manual testing
- Verify expected business value

### Test & Verification
- Automatic unit tests
- Separate test environment
- Automatic integration tests
- Automatic component tests (isolated)
- Some automatic acceptance tests
- Common information model
- Traceability built into pipeline
- Report history is available
- Graphing as a service
- Dynamic test coverage analysis
- Report trend analysis
- Dynamic graphing and dashboards
- Cross silo analysis

### Information & Reporting
- Baseline process metrics
- Manual reporting
- Measure the process
- Static code analysis
- Scheduled quality reports
- Graphing as a service
- Dynamic test coverage analysis
- Report trend analysis
- Dynamic graphing and dashboards
- Cross silo analysis
The Benefits of Continuous Delivery

- High Performing IT Teams are more Agile!
- High Performing IT Teams can recover faster!
- High Performing IT Teams can experiment more!

Continuous Delivery means better products for your customers!*

* Customers can be internal or external
Questions?