

Scaling DevOps Deployments with AWS CodePipeline and Dynatrace AppMon

Followed by Live Q&A

- Part of Dynatrace Performance Clinic Series
- Get Dynatrace AppMon: <http://bit.ly/dtpersonal>
- Ask questions using the *Questions Tab* – or post on answers.dynatrace.com



Rob Jahn

Sr. Consultant @ CGI

@RobJahn45



Joe Sicree

Sr. Consultant @ CGI

@JoeSicree



Daniel Freij

Sol. Engineer @ Apica

@DanielFreij



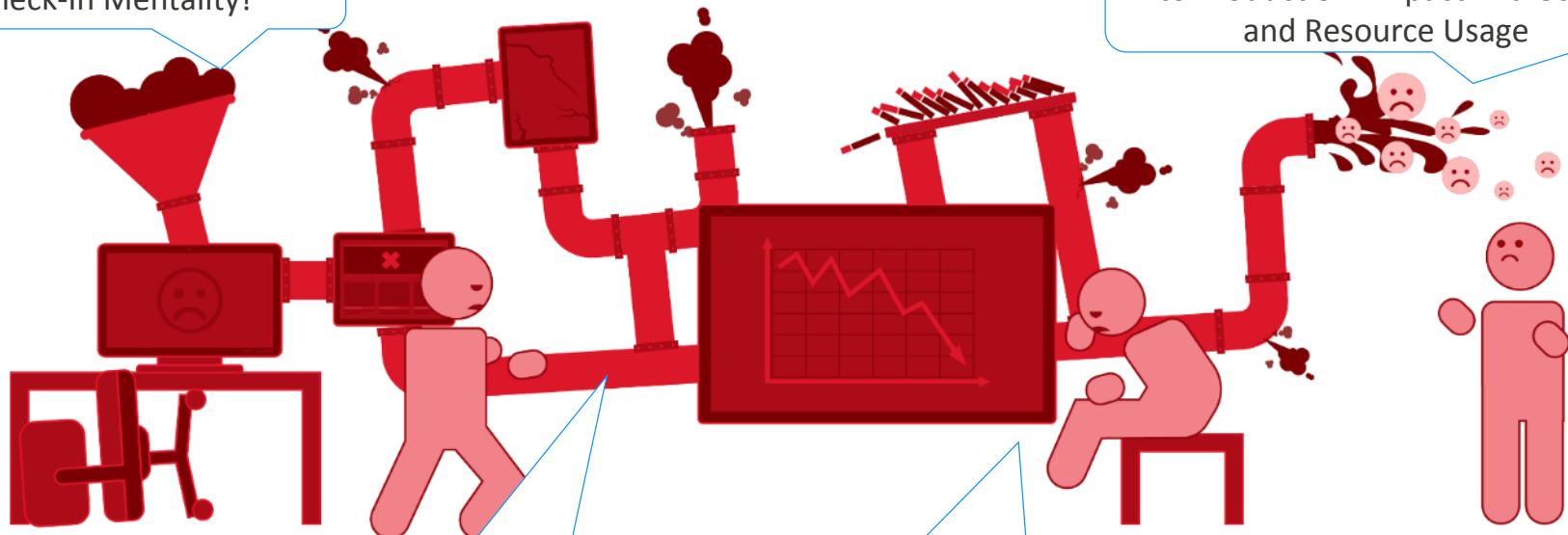
Andi Grabner

DevOps @ Dynatrace

@grabnerandi

AWS does NOT PREVENT your teams from pushing *bad code* through the pipeline *more frequently!*

Dev&Test: Trial & Error Check-In Mentality!

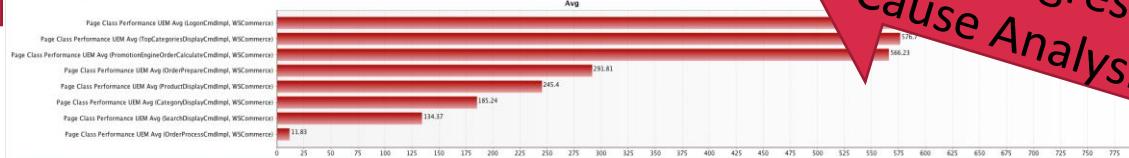
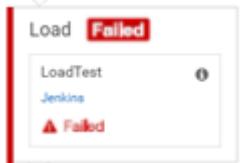
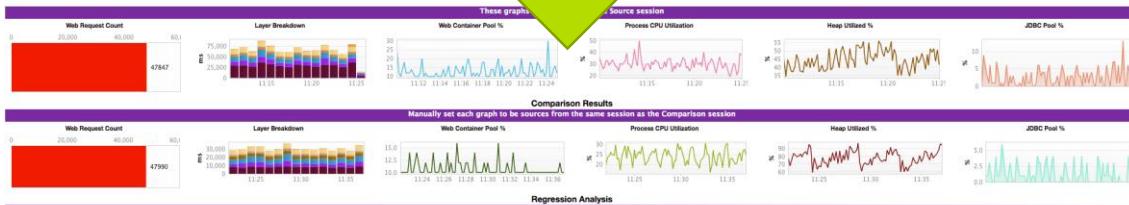
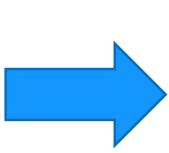
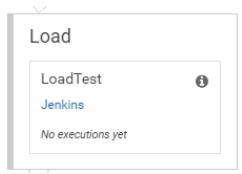


Ops/Biz: Bad Code Changes make it to Production. Impact End Users and Resource Usage

Test / CI: Basic Tests that only test high-level functionality!

Performance: Too many code commits leads to skipping performance testing to save time

Dynatrace: Automatic Performance Analysis across Load Tests



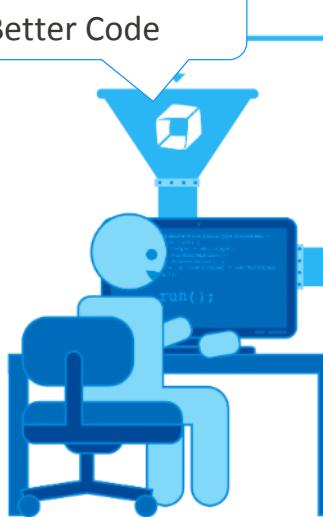
Auto Regressions and Root Cause Analysis across Tests

Scaling Continuous Innovation with Dynatrace on AWS:

Shift-Left Quality, Reduce Lead Time & Increase Flow



Dev&Test: Check-In Better Code



Test / CI: Stop 80% of Bad Builds Early

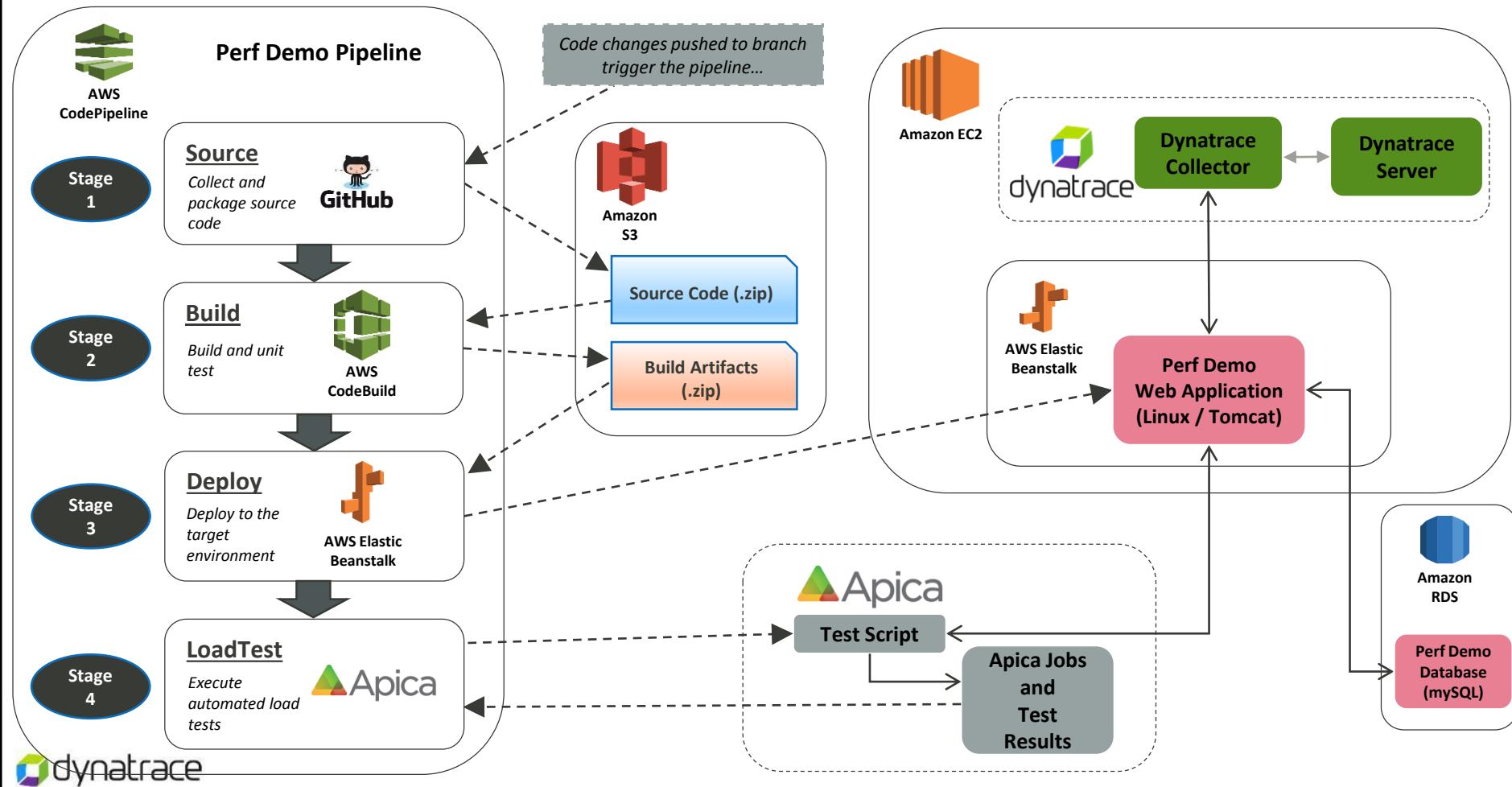


Ops/Biz: Monitor End User and Resource Health



Performance: Production Ready Checks on Good Builds Only!

Today's Goal: Full AWS CodePipeline Demo!



Agenda

- Introduction of Speakers
- What is AWS CodePipeline?
- How does Apica LoadTest and Dynatrace AppMon fit in?
- Live Demo: 3 Performance Engineering Use Cases
- Live Q&A



*"Helping companies with their end-user experience
focusing on availability and performance"*



SaaS

Hybrid

On-Premise



Founded in 2005

500+ Customers

Global Offices

Stockholm

London

New York

Santa Monica

 technology | Deloitte.TM
fast 50[™]
companies-to-watch

 dynatrace

CGI is a global information technology and business process services leader

World's 5th largest independent IT and BPS firm

High-end business and IT consulting

Client proximity
model complemented by unique global delivery network

Serving **5000+** clients from over **400 offices** around the world

End-to-end IT and business process services

100+ leading
IP-based solutions

71,000 professionals,
40 offices

Focused industry and domain **expertise**

36 years of successfully partnering with our clients

Revenue:
C\$10.4B *
Backlog:
C\$17.7B



CGI Performance Engineering Services



Software Performance Engineering

CGI's performance engineering services drive improvements in information technology and ensure scalability and stability for existing and planned systems, including improving overall quality, reliability, capacity, security, and performance.

Advisory Services

- Is my current team delivering the right services across the SDLC?
- Organizational and operational planning, strategy and design services for transforming internal capabilities.
- Establishing PE Center of Excellence (COE). Deliver for the enterprise or for a critical program.

Application Readiness Assessment and Performance Testing

- Is my application ready for increased volumes?
- Technically assess the application to understand the risks to performance, scalability and availability.
- Combination of architecture and code review to identify performance issues and risks along with comprehensive performance testing using CGI's proven PT methodology. End result is identification of issues and their remediation

Application Performance Management

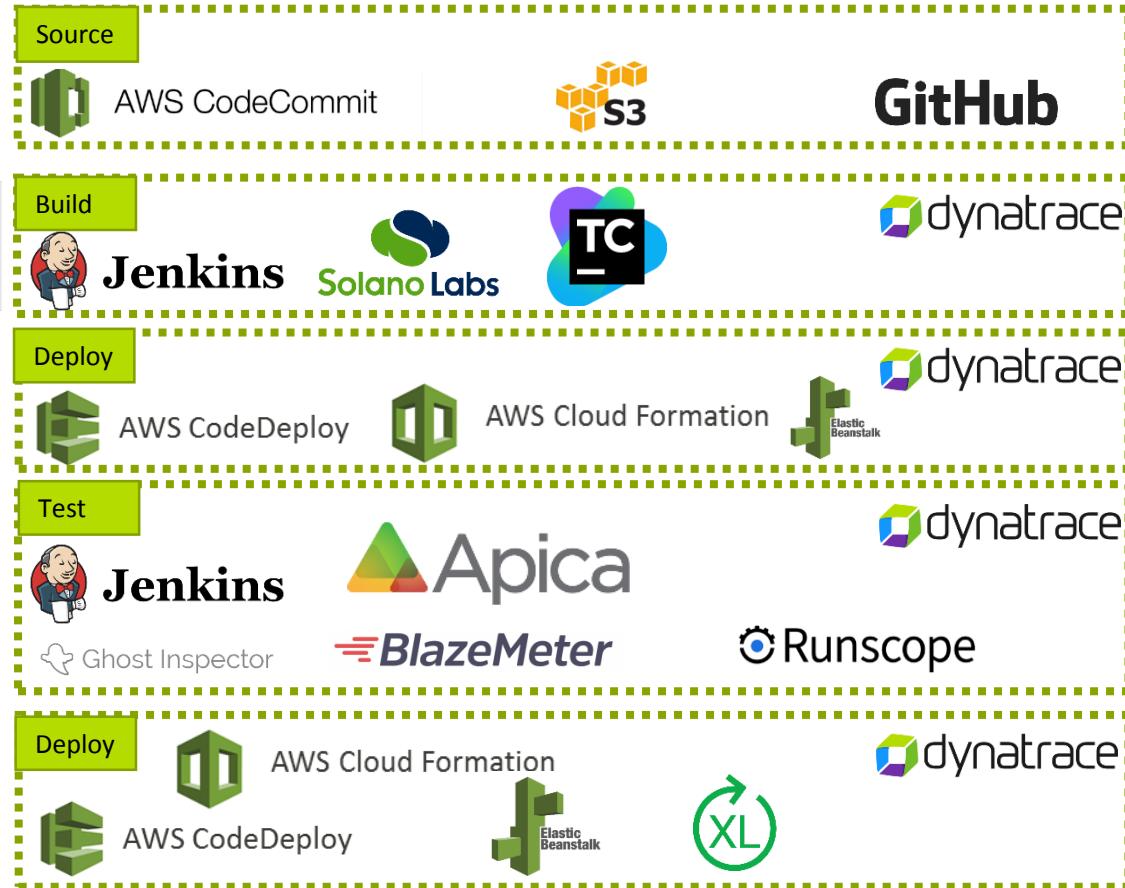
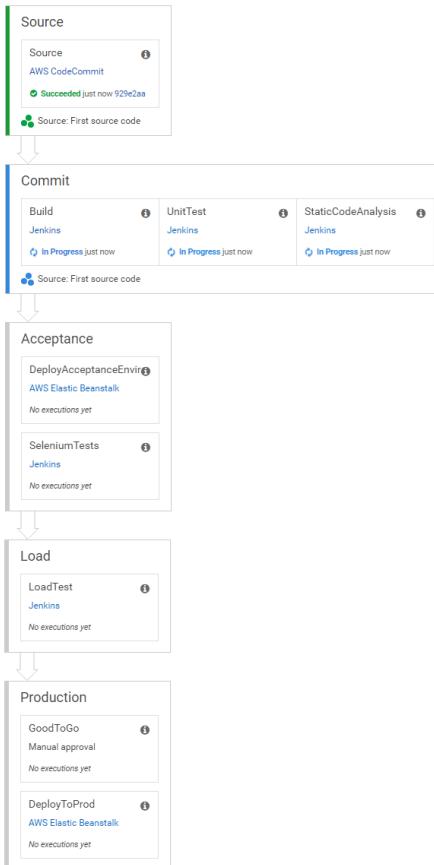
- Do you know your end user's experience?
- End to end proactive monitoring of critical business processes. Define and implement an APM strategy.
- Identify key operation use cases including key business transactions and business operational metrics.
- Build process for measurement, trending, and capacity management.
- Establish best practices via an APM reference architecture.

Production Performance Rescue

- Do you have a critical production issue resulting in repeated downtime and lost business?
- A proven approach to production performance and stability issues, reviewing the end to end transaction. Minimizing disruption and quickly restoring functionality and capability.

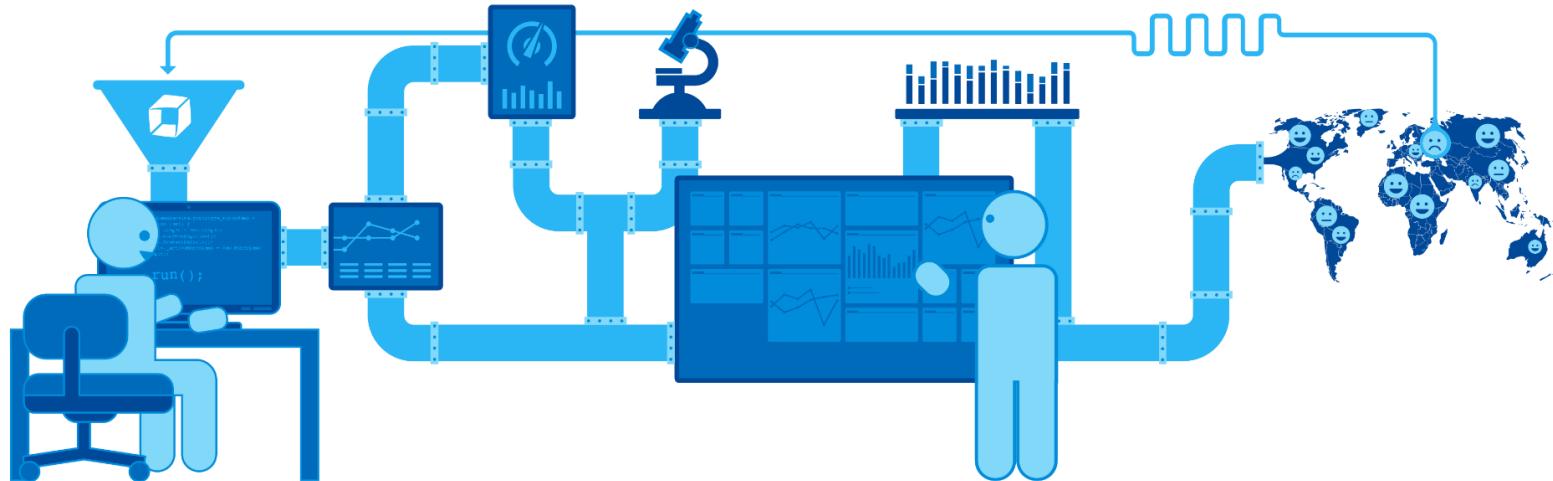


What is AWS CodePipeline? How does Apica
Load and Dynatrace fit in?



Make yourself familiar with ...

- AWS CodePipeline: <https://aws.amazon.com/codepipeline>
- Apica SaaS and Zebra Tester: <https://www.apicasystem.com/load-testing/>
- Dynatrace AppMon & UEM: <http://bit.ly/dttutorials>



DEMO TIME!



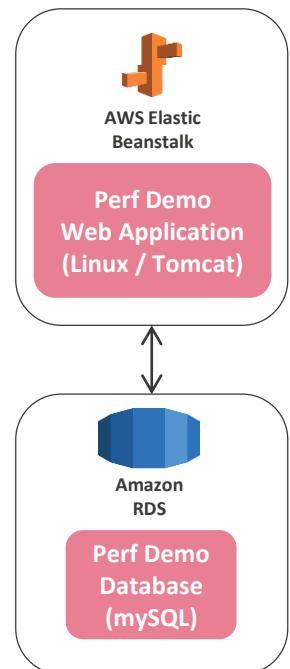
Demo Overview

- The following demo will walk us through a simple pipeline for a web application called Perf-Demo.
- The demo will focus on the following use cases:
 - Missed Service Level Agreements (SLA) for a mix of transactions
 - Memory leaks
 - Poor performance when accessing a database
- The demo will use a simple web application to demonstrate these use cases.



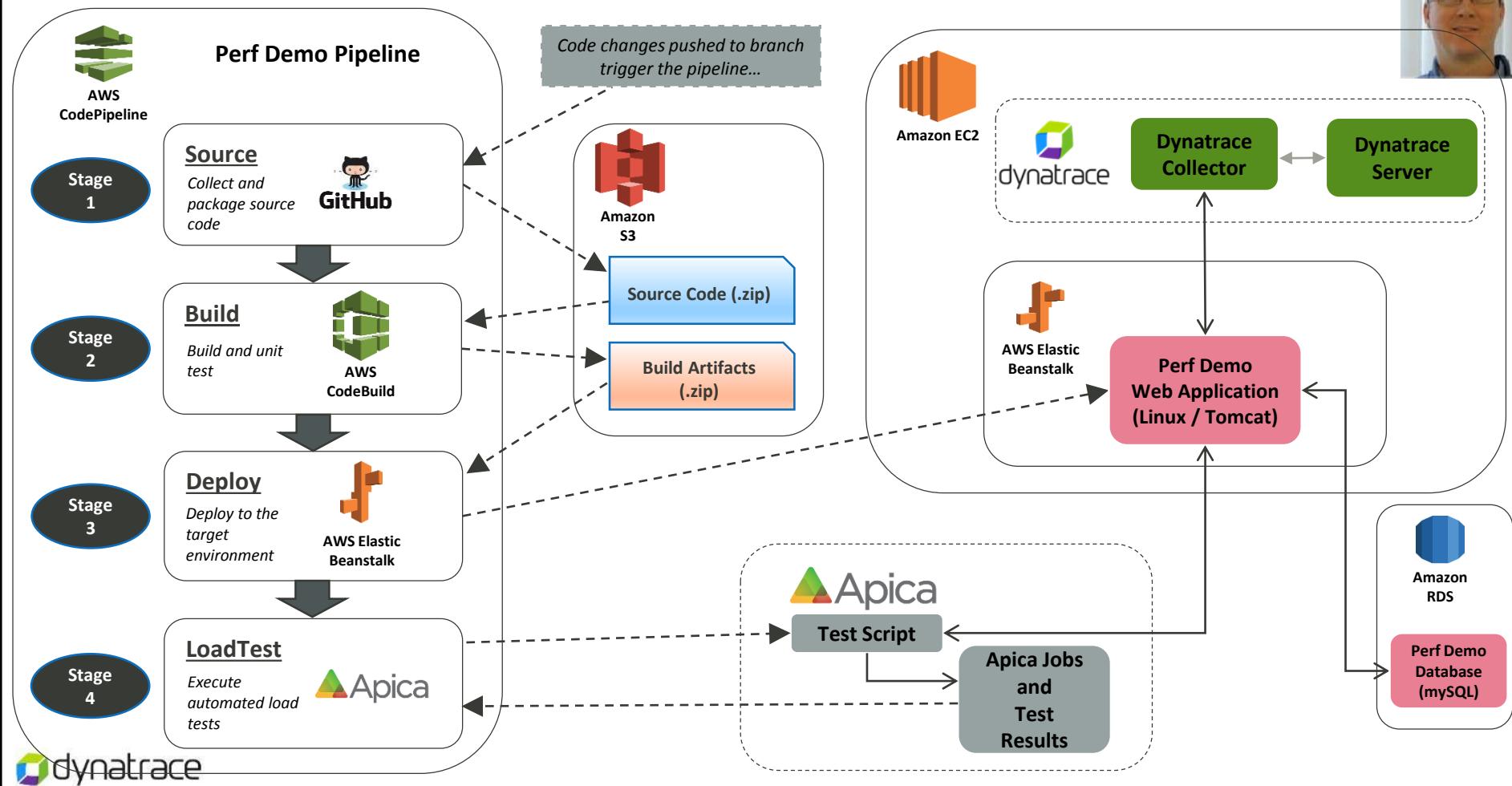
Demo Application

- Java Spring web application that provides a number of REST web services for use in performance testing.
- The web application provides the following services:
 - **calculateCompoundInterest** – Calculates compound interest for one or more accounts over a specified time period.
 - **addDataToMap** – Populates a hash map with data using a simple key
 - **getFinancialRecordsByDate** – Fetches financial data for one or more stocks on a specified date.



Get the code at <https://github.com/jsicree/perf-demo>

Demo AWS CodePipeline





Demo #1 – Slow Services

The Service

The computeCompoundInterest service computes the compound interest for a collection of accounts for a requested number of monthly or yearly intervals.

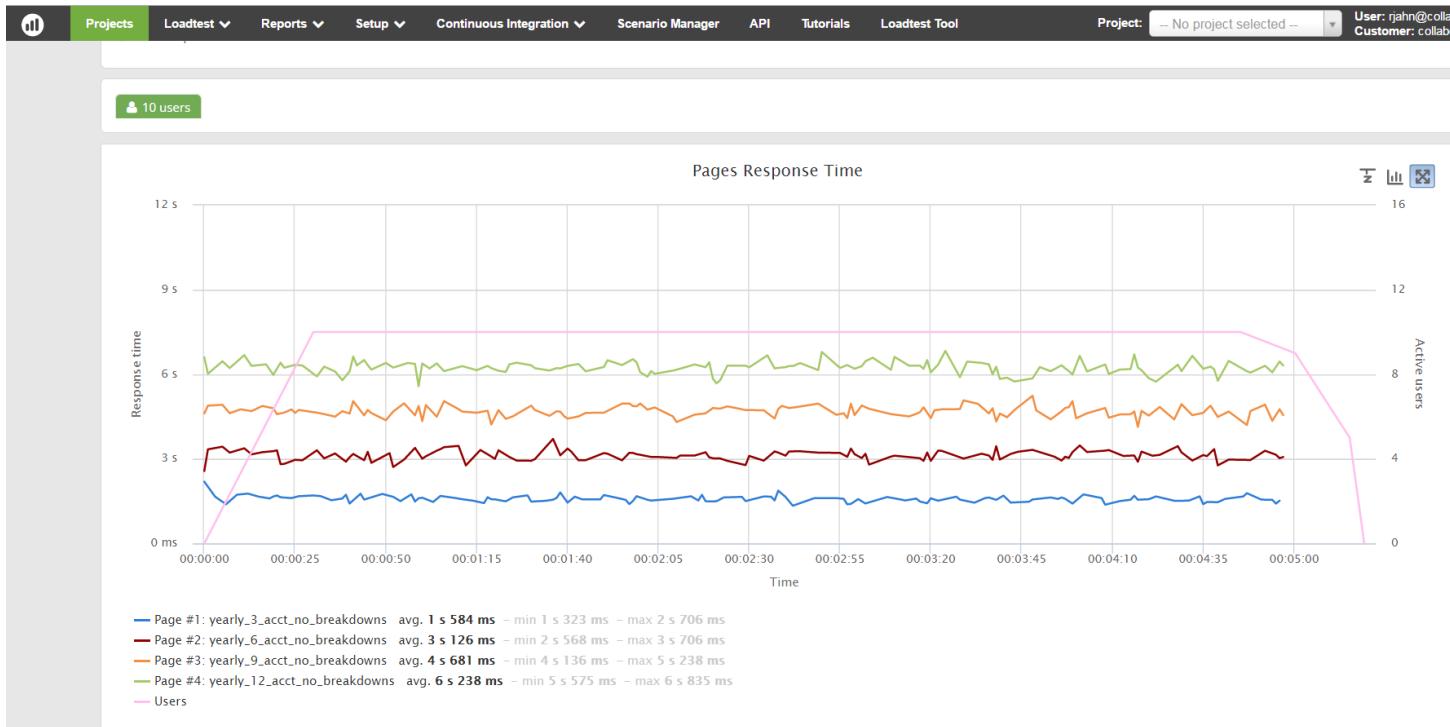
The Problem

The initial load test for the service using a mix of 3, 6, 9 and 12 accounts showed that the average response time was above the expected SLA for the requests with more accounts.



Demo #1 – Slow Services (V1 Test)

Exceeding SLA for Response Time





Demo #1 – Slow Services (cont.)

The Analysis

The underlying service was implemented synchronously. The more accounts included in the request, the longer the web service call took.

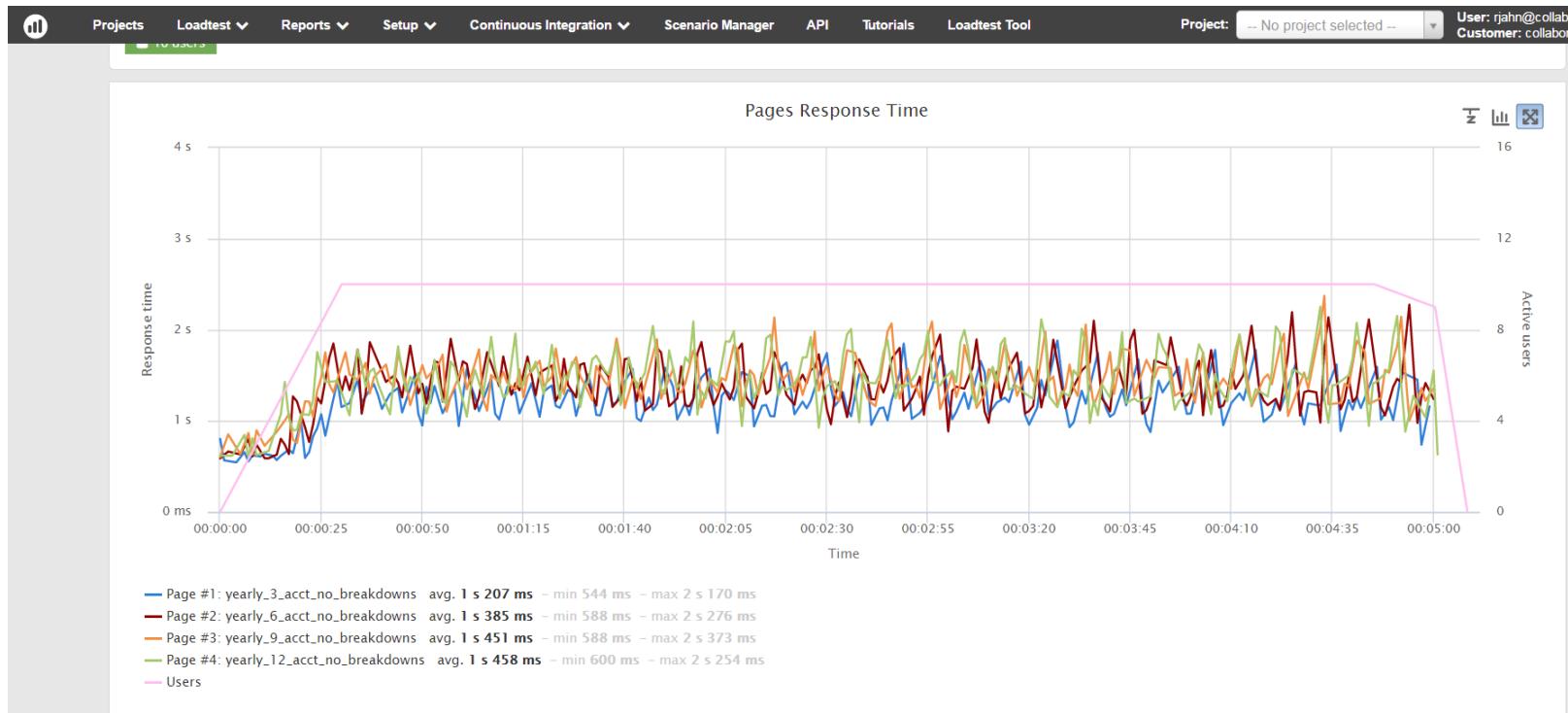
The Solution

The service was refactored to run asynchronously using threads.



Demo #1 – Slow Services (v2 Test)

All calls are under the 3 second SLA





Demo #2 – Memory Leak

The Service

The service addDataToMap adds a specified block of bytes to a hash map using a simple key.

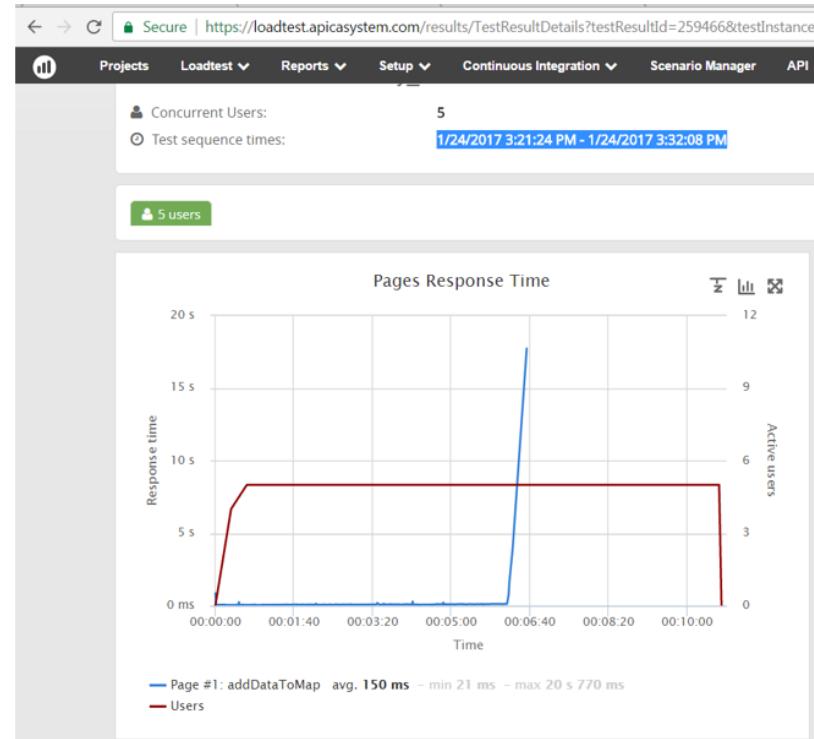
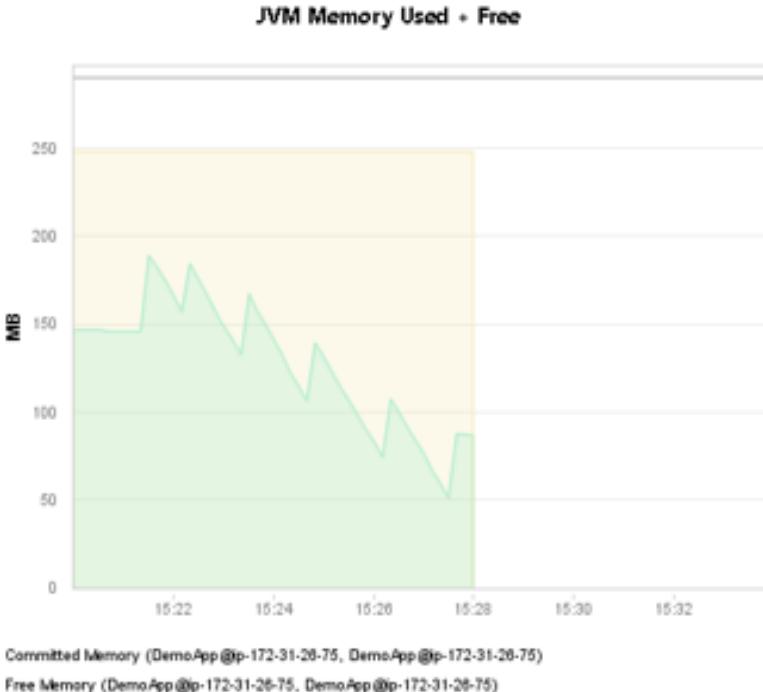
The Problem

The initial load test for the service showed that memory usage continuously increased, eventually causing an OutOfMemory error.

Demo #2 – Memory Leak (V1 Test)



Out of Memory caused server to crash. Timeout of request as a result





Demo #2 – Memory Leak (Cont.)

The Analysis

Service was implemented with a key class that did not implement an equals() method. As data was added to the map by each service call, values in the hash map were not being overwritten as intended.

Instead, the new values were being appended to the map. This resulted in a memory leak that eventually led to OutOfMemory errors.

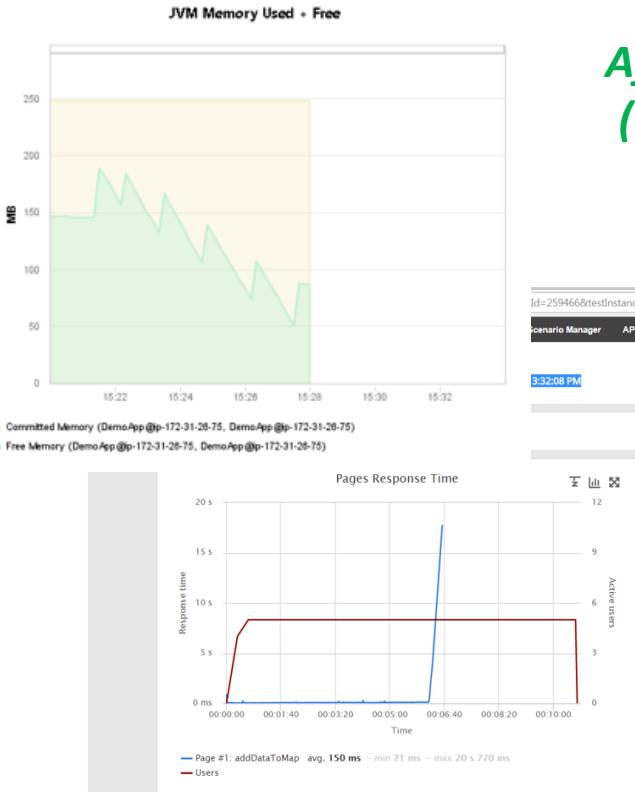
The Solution

The bad key class was refactored.

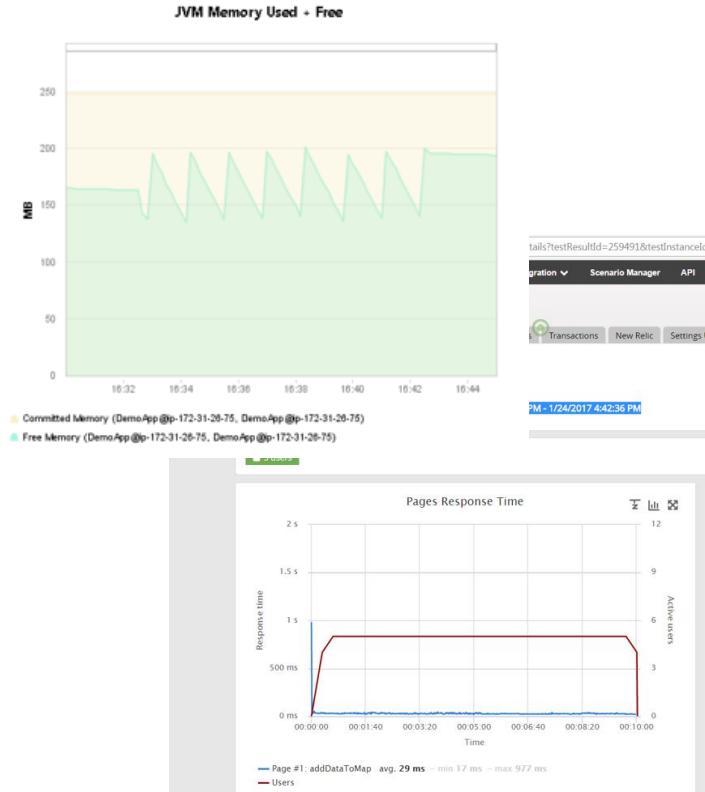
Demo #2 – Memory Leak (V2 Test)



**Before
(v1)**



**After
(v2)**





Demo #3 – Poor DB Performance

The Service

The service `getFinancialRecordsByDate` fetches financial data for a specified list of stock symbols for a specified date from database tables

The Problem

The initial load test for the service showed that there were a large number of database calls for each service call.

Demo #3 – Poor DB Performance (V1 Test)



Poor Response times and much of the time in the database

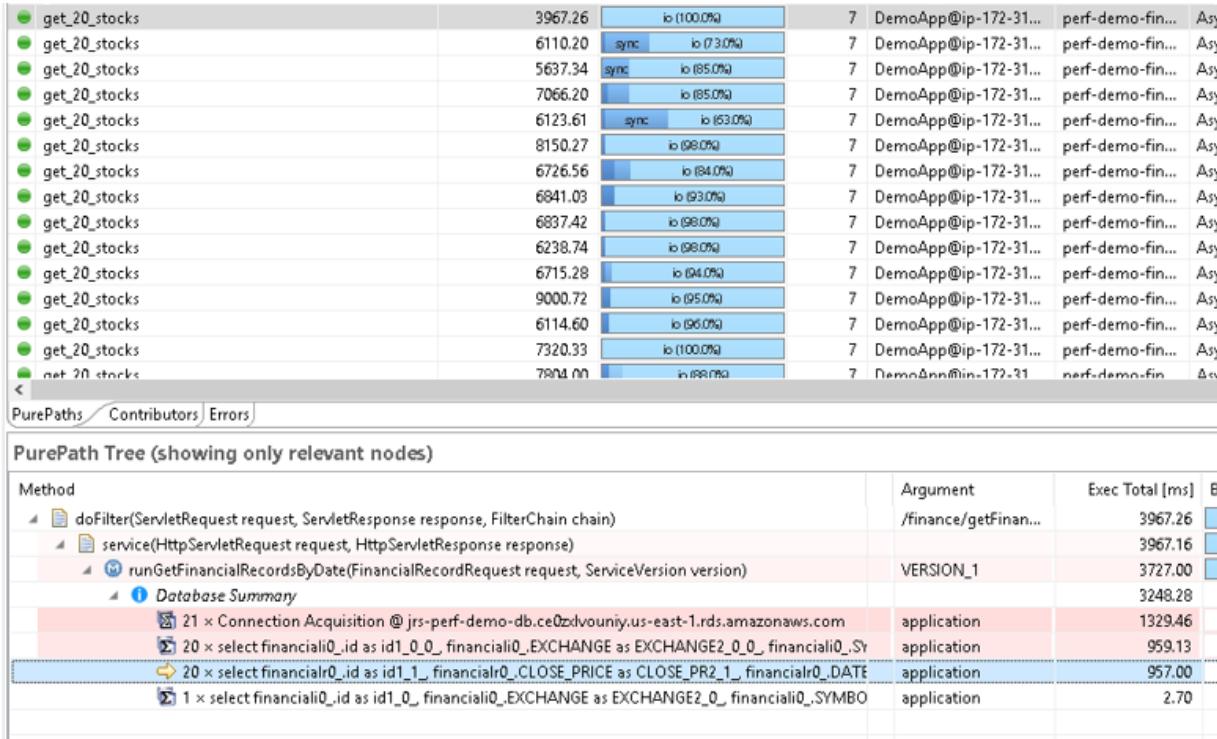
The screenshot displays two main sections of an APM tool:

- Real-time Monitoring (Left):** Shows a dual-axis chart for "Pages Response Time" over time (00:00:00 to 00:05:00). The left y-axis represents "Response time" from 0 ms to 40 s. The right y-axis represents "Active users" from 0 to 60. A blue line shows the average response time for Page #1: get_20_stocks, with metrics: avg. 8 s 55 ms, min 62 ms, max 22 s 212 ms. A red line shows the number of active users. A callout box indicates 25 concurrent users.
- Transaction Flow Analysis (Right):** Shows a "Topology" visualization mode with nodes for "Synthe...quest", "App", and "MySQL". The "App" node is highlighted with a green border and contains metrics: 1.21s, 25.32%, and 6211.5 per minute. An arrow points from the "App" node to the "MySQL" node, which is also highlighted with a green border and contains metrics: 3.56s and 74.68%. Top-level metrics shown are Total Transactions: 1,212 (151.5 per minute) and Failed Transactions: 0 (0%).



Demo #3 – Poor DB Performance (V1 Test)

41 SQL calls Per Service call.





Demo #3 – Poor DB Performance (cont)

The Analysis

Service suffered from an N+1 query problem where to fetch the data for N stocks, N+1 database queries were made.

The Solution

Refactored to use a SQL join clause to reduce the number of database queries to 1 per service call

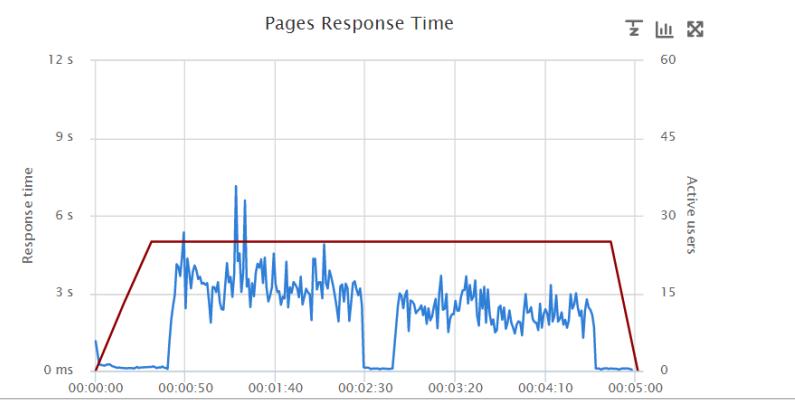
Demo #3 – Poor DB Performance (V2 Test)



Before – 10 sec



After – 3 sec – 20 less SQL calls



Method	Argument	Exec Total [ms]
doFilter(ServletRequest request, ServletResponse response, FilterChain chain)	/finance/getFinan...	3113.14
service(HttpServletRequest request, HttpServletResponse response)		3113.04
runGetFinancialRecordsByDate(FinancialRecordRequest request, ServiceVersion version)		2875.71
Database Summary		2875.71
21 x Connection Acquisition @jrs-perf-demo-db.ce0zdvouniy.us-east-1.rds.amazonaws.com		2872.68
20 x select financialr0_id as id1_0_, financialr0_EXCHANGE as EXCHANGE2_0_0_, financialr0_Sy		1797.81
20 x select financialr0_id as id1_1_, financialr0_CLOSE_PRICE as CLOSE_PR2_1_, financialr0_DATE		1072.89
1 x select financialr0_id as id1_0_, financialr0_EXCHANGE as EXCHANGE2_0_0_, financialr0_SYMBOL		1.97

Try it yourself!

Follow these simple steps

1. Setup AWS account for use of EBS, EC2, CodePipeline, CodeBuild:
<https://aws.amazon.com>
2. Setup Apica SaaS Trial Account: <https://www.apicasystem.com/>
3. Setup a Dynatrace AppMon Personal Licence: <http://bit.ly/dtpersonal>
4. Fork Demo App Code as to perform Builds and Run it – *Readme files have a lot of details* - <https://github.com/jsicree>

Contact reminders ...

- CGI – <http://www.cgi.com>
- Apica – <https://www.apicasystem.com>
- Dynatrace - <http://www.dynatrace.com>



Rob Jahn
Sr. Consultant @ CGI
@RobJahn45



Joe Sicree
Sr. Consultant @ CGI
@JoeSicree



Daniel Freij
Sol. Engineer @ Apica
@DanielFreij



Andi Grabner
DevOps @ Dynatrace
@grabnerandi

Event Reminders

- Dynatrace PERFORM – Feb 7th to 9th
 - <http://www.cgi.com/perform>
- YouTube Channel: <http://bit.ly/dttutorials>
- Education & Forums: <http://apmu.dynatrace.com> & <http://answers.dynatrace.com>
- Dynatrace AppMon Personal License: <http://bit.ly/dtpersonal>
- Share Your PurePath: <http://bit.ly/sharepurepath>
- PurePerformance Podcast: <http://bit.ly/pureperf>





Open Q&A

Use the Question Feature in GoTo Webinar
or post it later to
<http://answers.dynatrace.com>



dynatrace

Configuration Details

Source Stage in AWS CodePipeline

Commit Triggers
the Pipeline!

The screenshot shows the AWS CodePipeline console with the pipeline 'DempAppLoadTestBenchmark' being edited. The pipeline is structured into three stages:

- Source**: Triggered by a GitHub provider.
- Build**: Executed using AWS CodeBuild.
- Deploy**: Executed using AWS Elastic Beanstalk.

A large green arrow on the left points to the Source stage, with the text "Commit Triggers the Pipeline!" above it. The right side of the screen shows the "Edit action" configuration for the Source stage, specifically for the GitHub provider settings. It includes fields for Action name (Source), Source provider (GitHub), Connect to GitHub (Repository: robertjahn/perf-demo, Branch: master), and Output artifacts (Output artifact #1: AppSource). A "Required" button and an "Update" button are at the bottom of the configuration panel.

Build Stage in AWS CodePipeline

Trigger AWS CodeBuild

AWS CodePipeline | Edit: DempAppLoadTestBenchmark

Add or edit a stage in a pipeline or actions in a stage. [Learn more](#)

Cancel Delete Save pipeline changes

Source

GitHub

Stage

Build

AWS CodeBuild

Deploy

AWS Elastic Beanstalk

Stage

Edit action

Configure how your application is built.

Build actions

Action name* Build

Build provider* AWS CodeBuild

AWS CodeBuild

Solano CI

Configure your project

Select an existing build project

Create a new build project

Project name* DemoApp

View project details

Input artifacts

Choose one or more input artifacts for this action. The output of previous actions can be the input of this action. [Learn more](#)

* Required

Cancel Update

Show all

ec2-54-227-218-6...rdp

Dynatrace + Apic....pptx

Deploy Stage in AWS CodePipeline

Trigger AWS EBS Deployment

Secure | https://console.aws.amazon.com/codepipeline/home?region=us-east-1#/edit/DempAppLoadTestBenchmark

Cancel Delete Save pipeline changes

Source

Source GitHub

Stage

Build

Build AWS CodeBuild

Stage

Deploy

DeployEBS AWS Elastic Beanstalk

Stage

approve

Manual approval

Edit action

Configure where your application is deployed.

Deploy actions

Action name* DeployEBS

Deployment provider* AWS Elastic Beanstalk

AWS CloudFormation
AWS CodeDeploy
AWS Elastic Beanstalk
AWS OpsWorks

Choose one of your existing applications, or create a new one in AWS Elastic Beanstalk.

Application name* DemoApp

Choose one of your existing environments, or create a new one in AWS Elastic Beanstalk.

Environment name* DemoApp

Input artifacts

Choose one or more input artifacts for this action. The output of previous actions can be the input of this action. [Learn more](#)

Input artifacts #1 AppWar

* Required Cancel Update

Show all

ec2-54-227-218-6....rdp

Dynatrace + Apic...pptx

Test Stage in AWS CodePipeline

The screenshot shows the AWS CodePipeline console with a pipeline named "DempAppLoadTestBenchmark". The pipeline consists of four stages:

- Build**: AWS CodeBuild
- Stage**
- Deploy**: DeployEBS AWS Elastic Beanstalk
- Stage**

The second stage is expanded to show an **approve** step (Manual approval) and another Stage. The third stage is expanded to show a **Test** step.

The **Test** step contains an **Apica Load Test** action. A large green arrow points from the left towards this action. The **Edit action** dialog is open for this step, with the following details:

- Action category**: Test
- Test actions**: Choose from a list of test actions.
- Action name**: ApicaloadTest
- Test provider**: Apica Loadtest (selected from a dropdown menu)
- Apica Loadtest** (dropdown menu options):
 - Apica Loadtest
 - Add Jenkins
 - Apica Loadtest (selected)
 - BlazeMeter
 - Ghost Inspector UI Testing
 - HPE StormRunner Load
 - Jenkins (Version: 1)
 - Runscope API Monitoring
- Action configuration**:
 - LoadtestPresetName: DemoAppPreset - 15min 50vu
 - LoadtestScenarioFileName: DemoAppApicaFacade.class
 - LoadtestThresholds: failed_loops[gt;1]f.avg_resp_page[gt;5000]f
- Input artifacts**: Choose one or more input artifacts for this action. The output of previous actions can be used.

Trigger Apica Load Test

Apica Setup for Test Stage in AWS CodePipeline

The screenshot shows the Apica LoadTest Portal interface. The main title is "AWS CodePipeline Load Test Options". It asks to create a CodePipeline loadtest stage by defining a Settings Preset and Scenario.

Settings Preset: A dropdown menu shows "DemoAppPreset - 15min 50u". A green button "+ New Preset" is available.

Settings Preset Details:

- Users: 50
- Duration: 10 mins
- RampUpTime: 5 mins
- Location: New York

Email Preliminary Report To: robjhahnautomation@gmail.com

Scenario: A dropdown menu shows "DemoAppApicaFacade class". Buttons "+ New Scenario" and "+ Upload Scenario" are present.

Thresholds: A section for configuring failure thresholds.

IF Failed loops rate (%) > 1 THEN mark as FAILED

IF Average response time per page (ms) > 5000 THEN mark as FAILED

Buttons: "Cancel" and "Continue With These Settings"

Create advanced scenarios with Apica ZebraTester: A callout box with the following text:
Apica ZebraTester is the perfect complement to Apica LoadTest Portal. Quickly create complex scenarios in ZebraTester without any scripting knowledge and use them with Apica LoadTest Portals global load testing network. Manage your scenarios and projects and quickly create custom reports based on the results of your ZebraTester scenario.
Learn more [here](#) or download for free below.
Download ZebraTester for Free!

Download links for Windows, Windows (64-bit), Mac, and Linux (Ubuntu Preferred) are shown with cloud icons.

Help and Support: Links to LoadTest Portal User Guide, ZebraTester Tutorial Videos, and API Documentation.

Apica Logo: The Apica logo is at the bottom left.

Dynatrace Footer: The Dynatrace logo and the file name "ec2-54-227-218-6....rdp" are visible in the bottom left corner of the browser window.

EBS Configuration

- To get DT agent on EBS, need to use EB extensions as part of the deploy to install agent and to configure the JVM arguments.

Branch: master ▾ [perf-demo](#) / [demoweb](#) / [.ebextensions](#) /

This branch is 96 commits ahead, 8 commits behind jsicree:master.

 robertjahn committed on GitHub Update 04-tomcat.config

..

01_install_DT.config	Update 01_install_DT.config
02-move-war.config	Rename move-war.config to 02-move-wi
03-run-expand-war.config	Rename 03-run-expand-war to 03-run-e
04-tomcat.config	Update 04-tomcat.config

https://github.com/robertjahn/perf-demo/blob/master/demoweb/.ebextensions/01_install_DT.config

```
28 lines (25 sloc) | 863 Bytes

1 files:
2   "/tmp/dynatrace-agent-6.5.0.1289-unix.zip":
3     mode: "000755"
4     owner: root
5     group: root
6     source: https://s3.amazonaws.com/rjahn-temp-s3/dynatrace-agent-6.5.0.1289-unix.zip
7
8   "/opt/elasticbeanstalk/hooks/appdeploy/pre/99_install_dynatrace.sh":
9     mode: "000755"
10    owner: root
11    group: root
12    content: |
13      #!/bin/bash
14      if [ ! -d /opt/dynatrace-6.5 ]
15      then
16        #unzip_dynatrace
17        sudo unzip -o /tmp/dynatrace-agent-6.5.0.1289-unix.zip -d /tmp
18        #install_dynatrace in silent mode
19        cd /opt
20        sudo java -jar /tmp/dynatrace-agent-6.5.0.1289-unix.jar -y
21      fi
22
23 commands:
24   install_dynatrace:
25     command: /opt/elasticbeanstalk/hooks/appdeploy/pre/99_install_dynatrace.sh
26     cwd: /opt
27     test: "[ -d /opt/dynatrace-6.5 ]"
```

EBS Configuration

DT agent coming in from EBS

The screenshot shows the Dynatrace Agents Overview Dashboard for the server `ec2amaz-4b80fir`. The left sidebar navigation includes `Agents Overview`, `Databases`, `Deployment Health`, `Incidents`, `Infrastructure`, `License Overview`, `System Information`, `Tasks and Monitors`, and `System Profiles` (with `DemoApp` selected). The main panel displays the `Agents Overview` table with one entry:

Name	Host	Operating System	OS Arch	Connection State	Technology
<code>DemoApp@ip-172-31-18-124:27928</code>	<code>ip-172-31-18-124</code>	<code>Linux</code>	<code>x86_64</code>	<code>Connected (Hot S...)</code>	<code>Java</code>

A modal window is open over the dashboard, showing a GitHub commit details page for the file `04-tomcat.config` in the repository `perf-demo / demoweb / .ebextensions`. The commit information is:

Branch: master `perf-demo / demoweb / .ebextensions / 04-tomcat.config` Find file Copy path

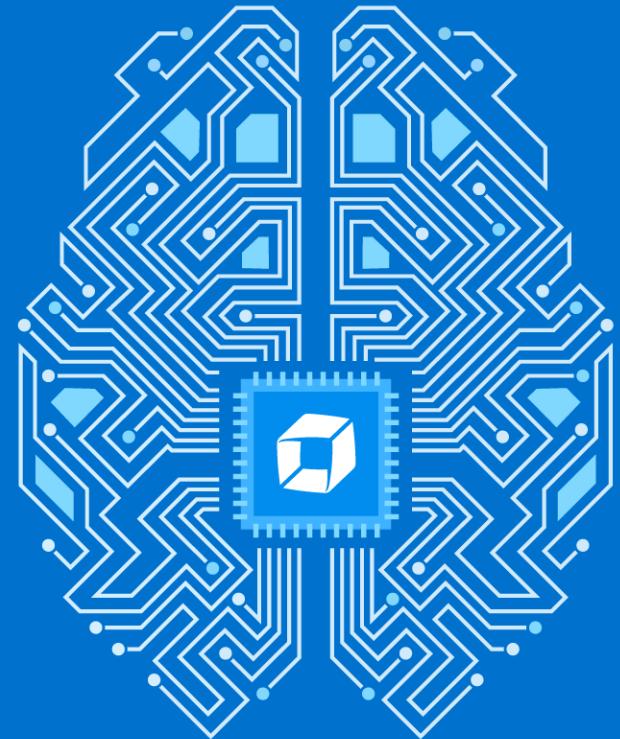
robertjahn Update 04-tomcat.config b4bab34 44 minutes ago

1 contributor

8 lines (6 sloc) 416 Bytes Raw Blame History

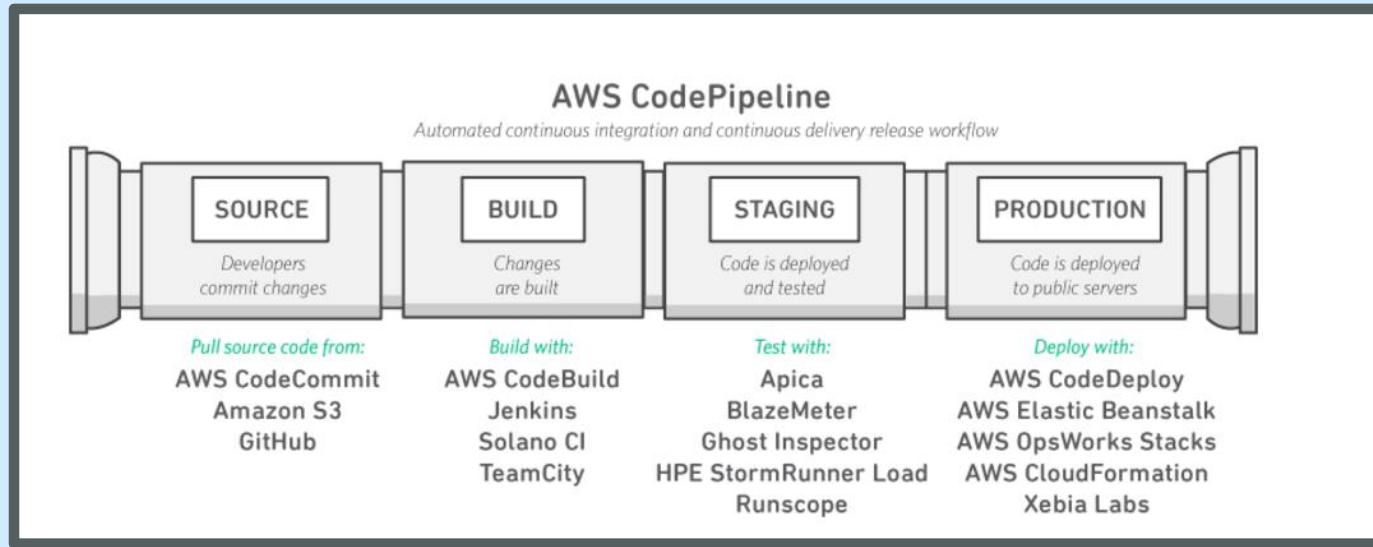
```
1 # the DT servername will need to be updated for your server
2 # if you use EC2 with no fixed domain then you also need to edit it after each stop/start of the instance
3
4 option_settings:
5   - namespace: aws:elasticbeanstalk:container:tomcat:jvmoptions
6     option_name: JVM Options
7     value: -agentpath:/opt/dynatrace-6.5/agent/lib64/libdtagent.so=name=DemoApp,collector=ec2-54-227-218-63.compute-1.amazonaws.com:9998
```

Backup



What is the AWS CodePipeline?

AWS CodePipeline is a [continuous integration](#) and [continuous delivery](#) service for fast and reliable application and infrastructure updates. CodePipeline builds, tests, and deploys your code every time there is a code change, based on the release process models you define.



3 Minute Explainer Video on <https://aws.amazon.com/codepipeline/>

Apica SaaS and Zebra Tester

Self Service - SaaS Model

Test like a pro with the most powerful SaaS platform, tools and expertise currently available.

Perform high capacity load testing and compare results in a pay-as-you-go model using the Apica LoadTest Portal.

Full API support and scheduling services provide an effective, streamlined platform for continuous testing.



- ✓ Know your limits; validate performance
- ✓ Scheduled tests/API support
- ✓ Unlimited testing for a fixed price
- ✓ Capacity-on-demand

[Learn More](#)

Apica ZebraTester: Powerful Load Testing Tool

Apica ZebraTester is a powerful load testing and script management software tool available inside Apica's LoadTest Portal.

Pinpoint exactly where and when poor performance starts using ZebraTester's highly accurate load test capabilities. No programming required.



- ✓ Powerful Testing tool
- ✓ Advanced scripting
- ✓ Unlimited testing for a fixed price
- ✓ Cloud scalability

[Learn More](#)

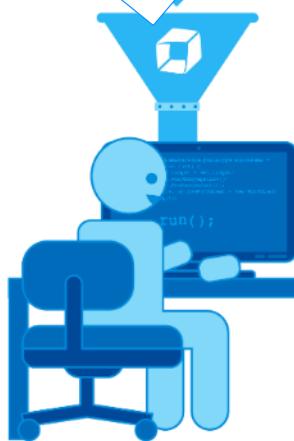
Learn more at <https://www.apicasystem.com/load-testing/>

Scaling Continuous Innovation with Dynatrace on AWS:

Shift-Left Quality, Reduce Lead Time & Increase Flow



Dev&Test: Check-In Better Code



Test / CI: Stop 80% of Bad Builds Early



Performance: Production Ready Checks on Good Builds Only!

Ops/Biz: Monitor End User and Resource Health



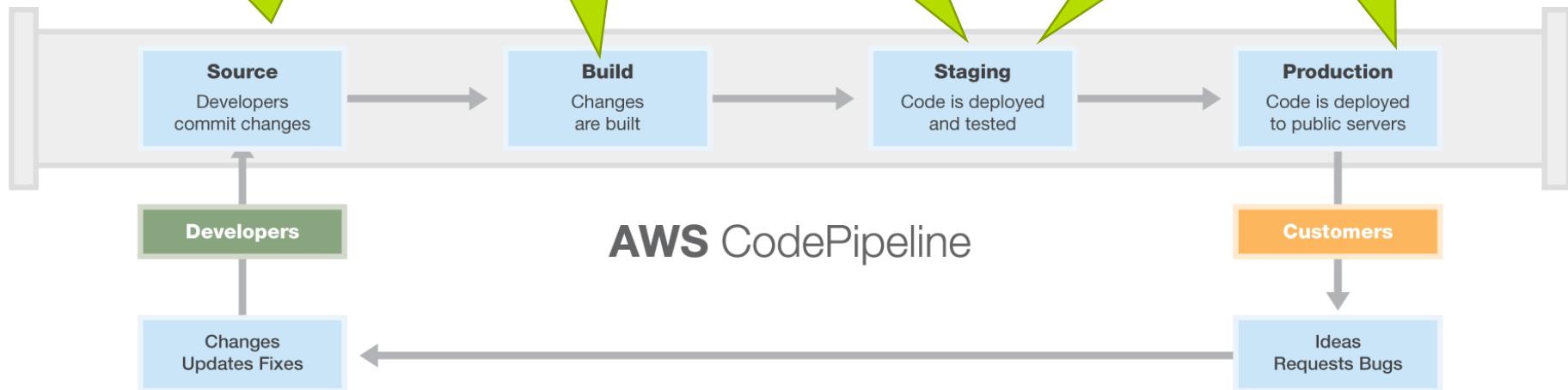
What is the AWS CodePipeline?

#1: Dev Checks in Code!
„A Commit“ to AWS CodeCommit, S3 or GitHub triggers the Pipeline. The committed code is the „Output Artifact“ for the next phase!

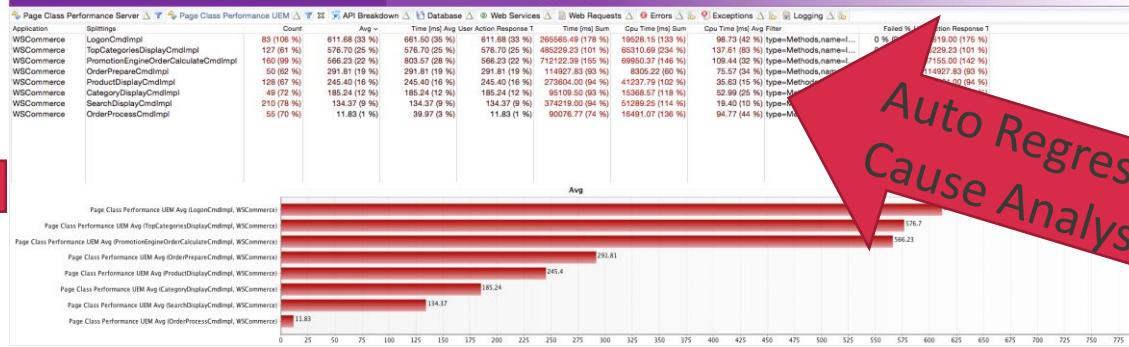
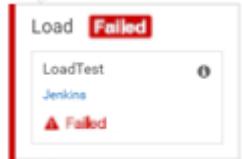
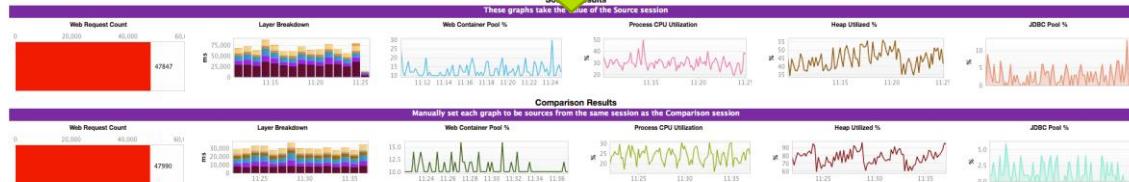
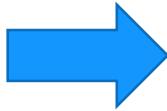
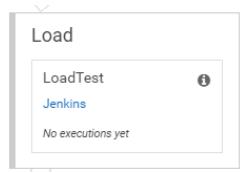
#2: Building!
Execute a Job in Jenkins or Solano CI. Result is another „Output Artifact“

#3: Testing
Execute Jenkins, Apica, Ghost Inspector, Runscope API to test your „Output Artifact“

#4: Deploy in Staging or Production
Deploy any Output Artifcat using AWS Cloud Formation, AWS CodeDeploy, AWS Elastics Beanstalk



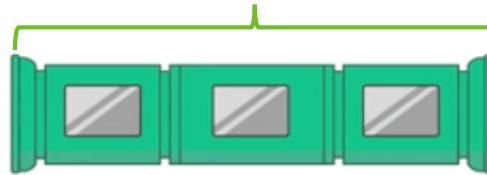
Dynatrace: Automatic *Performance Analysis Across Load Tests*



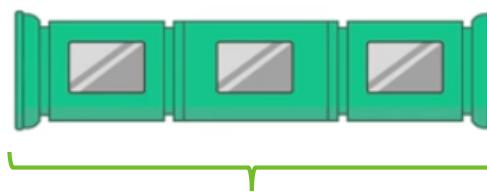
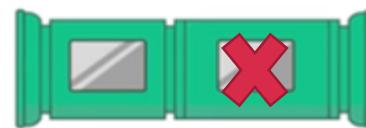
Auto Regressions and Root Cause Analysis across Tests

AWS + Dynatrace => *Faster Lead Times, Better Quality*

Optimal Lead Time with 1 Build in Pipeline



Happy Users



Keeping Optimal Lead Times with Many Builds

Get Started with 3 Simple Steps



Step #1: License

<http://bit.ly/dtpersonal>

Dynatrace Personal License

Get Dynatrace AppMon & UEM Lifetime Personal License for your local, web or mobile apps.

- Find memory leaks, CPU hotspots, architectural or deployment issues
- See every SQL Statement, Exception, Log message and HTTP Request detail
- Share the captured data with your colleagues or send it to Dynatrace and let us show you how to analyze the data! Share Your PurePath
- Integrate Dynatrace with JMeter, SOAPUI, KUnit, Selenium, WebDriver, Load Runner, Site or any other testing tool to see what your app is really doing
- Integrate Dynatrace with Jenkins, Bamboo, Team City or Team Foundation Server to stop bad builds automatically
- Supports your favorite technology stack including Java, .NET, PHP, Node.js, NGINX, Docker, Jenkins, iOS, Android, Apache, Chef, Puppet, Ansible, and many more.

During the first 30 days you can install Dynatrace Agents on up to 5 different machines to analyse application performance and scalability issues in your distributed applications. After the 30 days keep using it as the license automatically converts to a Personal License to be used on your local apps.

Watch our YouTube Tutorials to learn how best use it as a Developer, Tester, Architect or Ops: <http://bit.ly/dttutorials>

The form contains fields for Business Email (youremail@company.com), First Name, Last Name, Company (Your Company), Phone (55511234), Country (United States), State/Province (Massachusetts), and a checkbox for I accept personal license terms. There is also a link to Tell me more about application performance. A large green 'SUBMIT' button is at the bottom.

Step #2: Learn

<http://bit.ly/dttutorials>

The page shows a video thumbnail for 'What's Dynatrace AppMon?' followed by a list of 5 video thumbnails for 'Online Perf Clinic' sessions. The sessions are:

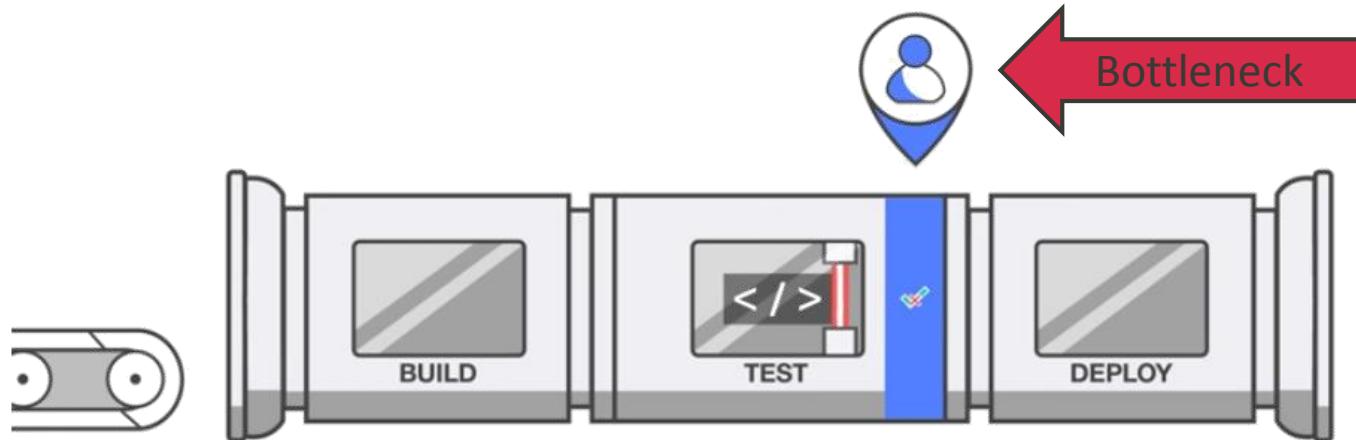
- Online Perf Clinic - What is Dynatrace AppMon & UEM and How to Get Started - August 2016
- Online Perf Clinic - System Profiles Explained, Agent Mappings and Troubleshooting - Oct 13th, 2014
- Online Perf Clinic - How Agents Work and How to Install Local and Remote Agents - Oct 27th, 2014
- Online Perf Clinic- Evaluating Dynatrace for PHP Applications
- Online Perf Clinic- Evaluating Dynatrace for .NET Applications
- Online Perf Clinic- Evaluating Dynatrace for Java Applications

Step #3: Expand

<https://github.com/Dynatrace>

The GitHub page for 'dynatrace' shows basic organization details: This organization, Search, Pull requests, Issues, and Gist. It features the Dynatrace logo, a brief description, and links to Waltham, Boston, their website, and the community page. The 'Repositories' tab is selected, showing a search bar and filters for Type: All and Language: All.

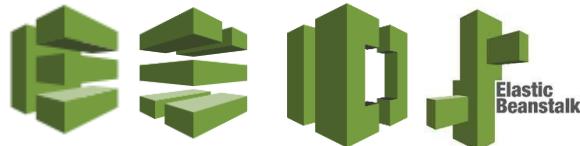
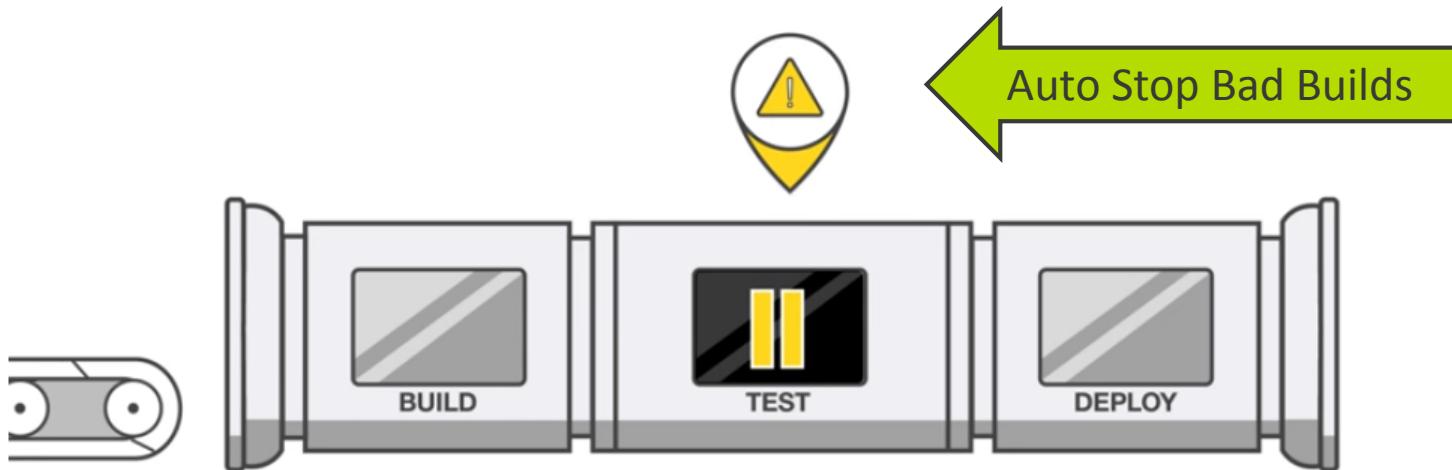
AWS DevOps Pipeline: Manual Approval and Regression Detection *doesn't scale*



AWS CodePipeline

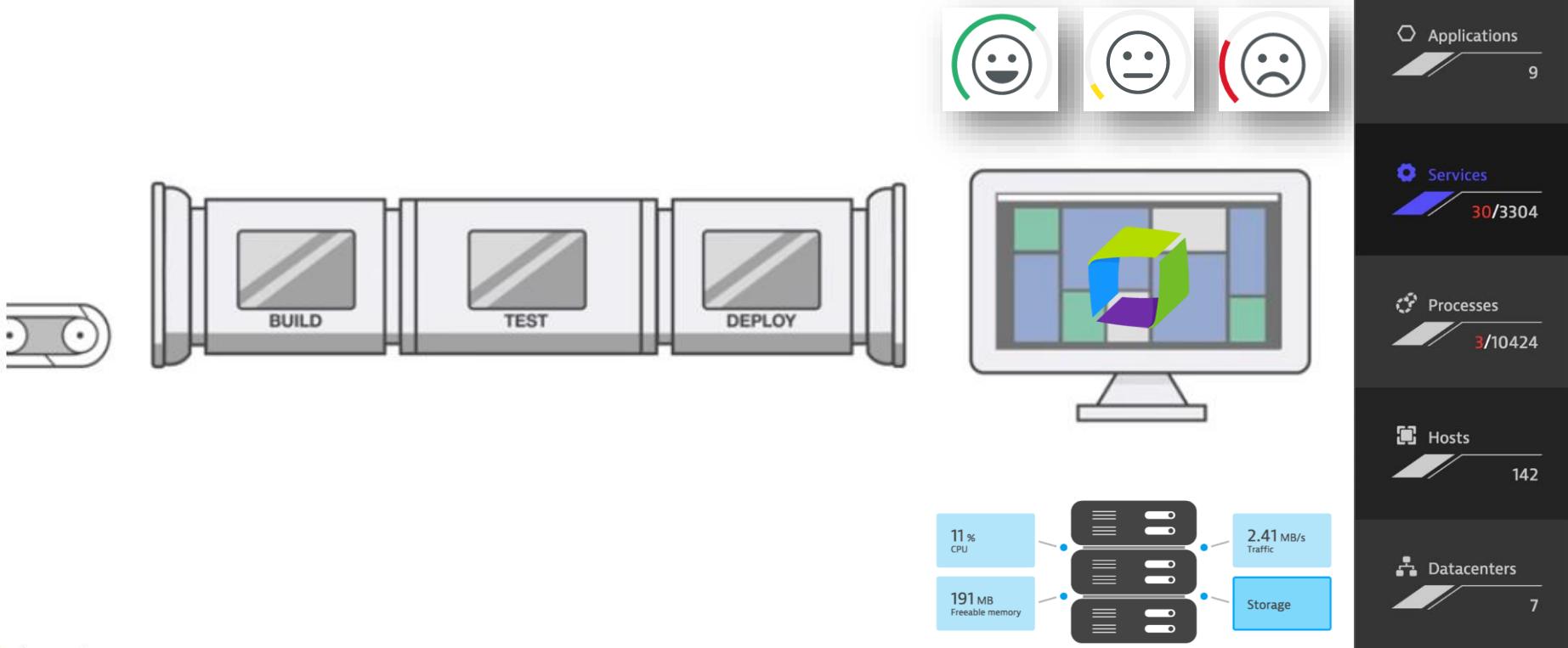


AWS + Dynatrace: Automatically Detect Architectural, Performance and Scalability Regressions



dynatrace

AWS + Dynatrace: Closing the Feedback Loop by automatically monitoring your Deployment





dynatrace